Region Growing 5

5.1 REGIONS

Chapter 4 concentrated on the linear features (discontinuities of image gray level) that often correspond to object boundaries, interesting surface detail, and so on. The "dual" problem to finding edges around regions of differing gray level is to find the regions themselves. The goal of region growing is to use image characteristics to map individual pixels in an input image to sets of pixels called *regions.* An image region might correspond to a world object or a meaningful part of one.

Of course, very simple procedures will derive a boundary from a connected region of pixels, and conversely can fill a boundary to obtain a region. There are several reasons why both region growing and line finding survive as basic segmentation techniques despite their redundant-seeming nature. Although perfect regions and boundaries are interconvertible, the processing to find them initially differs in character and applicability; besides, perfect edges or regions are not always required for an application. Region-finding and line-finding techniques can cooperate to produce a more reliable segmentation.

The geometric characteristics of regions depend on the domain. Usually, they are considered to be connected two-dimensional areas. Whether regions can be disconnected, non-simply connected (have holes), should have smooth boundaries, and so forth depends on the region-growing technique and the goals of the work. Ultimately, it is often the segmentation goal to partition the entire image into quasi-disjoint regions. That is, regions have no two-dimensional overlaps, and no pixel belongs to the interior of more than one region. However, there is no single definition of region—they may be allowed to overlap, the whole image may not be partitioned, and so forth.

Our discussion of region growers will begin with the most simple kinds and progress to the more complex. The most primitive region growers use only aggregates of properties of local groups of pixels to determine regions. More sophisti-

cated techniques "grow" regions by *merging* more primitive regions. To do this in a structured way requires sophisticated representations of the regions and boundaries. Also, the merging *decisions* can be complex, and can depend on descriptions of the boundary structure separating regions in addition to the region semantics. A good survey of early techniques is [Zucker 1976].

The techniques we consider are:

1. *Local techniques.* Pixels are placed in a region on the basis of their properties or the properties of their close neighbors.
2. *Global techniques.* Pixels are grouped into regions on the basis of the properties of large numbers of pixels distributed throughout the image.
3. *Splitting and merging techniques.* The foregoing techniques are related to individual pixels or sets of pixels. State space techniques merge or split regions using graph structures to represent the regions and boundaries. Both local and global merging and splitting criteria can be used.

The effectiveness of region growing algorithms depends heavily on the application area and input image. If the image is sufficiently simple, say a dark blob on a light background, simple local techniques can be surprisingly effective. However, on very difficult scenes, such as outdoor scenes, even the most sophisticated techniques still may not produce a satisfactory segmentation. In this event, region growing is sometimes used conservatively to preprocess the image for more knowledgeable processes [Hanson and Riseman 1978].

In discussing the specific algorithms, the following definitions will be helpful. Regions R_k are considered to be sets of points with the following properties:

$\mathbf{x}_i$ in a region R is *connected* to $\mathbf{x}_j$ iff there is a sequence $\{\mathbf{x}_i, \ldots, \mathbf{x}_j\}$ such that $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ are connected and all the points are in R. (5.1)

R is a *connected region* if the set of points $\mathbf{x}$ in R has the property that every pair of points is connected. (5.2)

$$I, \text{ the entire image} = \bigcup_{k=1}^{m} R_k \tag{5.3}$$

$$R_i \cap R_j = \phi, \qquad i \neq j \tag{5.4}$$

A set of regions satisfying (5.2) through (5.4) is known as a *partition.* In segmentation algorithms, each region often is a unique, homogeneous area. That is, for some Boolean function $H(R)$ that measures region homogeneity,

$$H(R_k) = \text{true for all } k \tag{5.5}$$

$$H(R_i \cup R_j) = \text{false for } i \neq j \tag{5.6}$$

Note that R_i does not have to be connected. A weaker but still useful criterion is that neighboring regions not be homogeneous.

The counterpart to the edge tracker for binary images is the blob-coloring algorithm. Given a binary image containing four-connected blobs of 1's on a background of 0's, the objective is to "color each blob"; that is, assign each blob a different label. To do this, scan the image from left to right and top to bottom with a special L-shaped template shown in Fig. 5.1. The coloring algorithm is as follows.

Algorithm 5.1: Blob Coloring

Let the initial color, $k = 1$. Scan the image from left to right and top to bottom.

If $f(\mathbf{x}_C) = 0$ then continue
else
begin

if $(f(\mathbf{x}_U) = 1$ and $f(\mathbf{x}_L) = 0)$
then color $(\mathbf{x}_C)$:= color $(\mathbf{x}_U)$

if $(f(\mathbf{x}_L) = 1$ and $f(\mathbf{x}_U) = 0)$
then color $(\mathbf{x}_C)$:= color $(\mathbf{x}_L)$

if $(f(\mathbf{x}_L) = 1$ and $f(\mathbf{x}_U) = 1)$
then begin
color $(\mathbf{x}_C)$:= color $(\mathbf{x}_L)$
color $(\mathbf{x}_L)$ is equivalent to color $(\mathbf{x}_U)$
end

comment: two colors are equivalent.

if $(f(\mathbf{x}_L) = 0$ and $f(\mathbf{x}_U) = 0)$
then color $(\mathbf{x}_L)$:= k; k := $k + 1$

comment: new color

end

After one complete scan of the image the color equivalences can be used to assure that each object has only one color. This binary image algorithm can be used as a simple region-grower for gray-level images with the following modifications. If in a

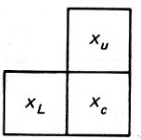


Fig. 5.1 L-shaped template for blob coloring.

gray-level image $f(\mathbf{x}_C)$ is approximately equal to $f(\mathbf{x}_U)$, assign $\mathbf{x}_C$ to the same region (blob) as $\mathbf{x}_U$. This is equivalent to the condition $f(\mathbf{x}_C) = f(\mathbf{x}_U) = 1$ in Algorithm 5.1. The modifications to the steps in the algorithm are straightforward.

5.3 GLOBAL TECHNIQUES: REGION GROWING VIA THRESHOLDING

This approach assumes an object-background image and picks a threshold that divides the image pixels into either object or background:

$\mathbf{x}$ is part of the Object iff $f(\mathbf{x}) > T$
Otherwise it is part of the Background

The best way to pick the threshold T is to search the histogram of gray levels, assuming it is bimodal, and find the minimum separating the two peaks, as in Fig. 5.2. Finding the right valley between the peaks of a histogram can be difficult when the histogram is not a smooth function. Smoothing the histogram can help but does not guarantee that the correct minimum can be found. An elegant method for treating bimodal images assumes that the histogram is the sum of two composite normal functions and determines the valley location from the normal parameters [Chow and Kaneko 1972].

The single-threshold method is useful in simple situations, but primitive. For example, the region pixels may not be connected, and further processing such as that described in Chapter 2 may be necessary to smooth region boundaries and remove noise. A common problem with this technique occurs when the image has a background of varying gray level, or when collections we would like to call regions vary smoothly in gray level by more than the threshold. Two modifications of the threshold approach to ameliorate the difficulty are: (1) high-pass filter the image to deemphasize the low-frequency background variation and then try the original technique; and (2) use a spatially varying threshold method such as that of [Chow and Kaneko 1972].

The Chow-Kaneko technique divides the image up into rectangular subimages and computes a threshold for each subimage. A subimage can fail to have a threshold if its gray-level histogram is not bimodal. Such subimages receive inter-

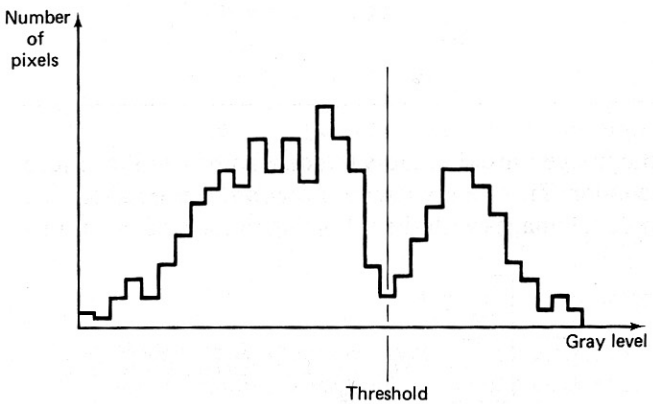


Fig. 5.2 Threshold determination from gray-level histogram.

polated thresholds from neighboring subimages that are bimodal, and finally the entire picture is thresholded by using the separate thresholds for each subimage.

5.3.1 Thresholding in Multidimensional Space

An interesting variation to the basic thresholding paradigm uses color images; the basic digital picture function is vector-valued with red, blue, and green components. This vector is augmented with possibly nonlinear combinations of these values so that the augmented picture vector has a number of components. The idea is to re-represent the color solid redundantly and hope to find color parameters for which thresholding does the desired segmentation. One implementation of this idea used the red, green, and blue color components; the intensity, saturation, and hue components; and the N.T.S.C. Y, I, Q components (Chapter 2) [Ohlander et al. 1979].

The idea of thresholding the components of a picture vector is used in a primitive form for multispectral LANDSAT imagery [Robertson et al. 1973]. The novel extension in this algorithm is the recursive application of this technique to nonrectangular subregions.

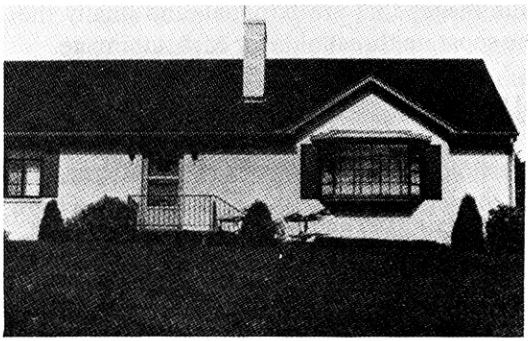
The region partitioning is then as follows:

Algorithm 5.2: Region Growing via Recursive Splitting

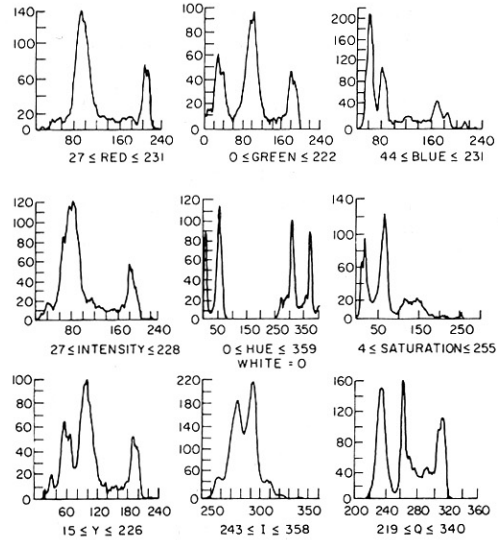
1. Consider the entire image as a region and compute histograms for each of the picture vector components.
2. Apply a peak-finding test to each histogram. If at least one component passes the test, pick the component with the most significant peak and determine two thresholds, one either side of the peak (Fig. 5.3). Use these thresholds to divide the region into subregions.
3. Each subregion may have a "noisy" boundary, so the binary representation of the image achieved by thresholding is smoothed so that only a single connected subregion remains. For binary smoothing see ch. 8 and [Rosenfeld and Kak 1976].
4. Repeat steps 1 through 3 for each subregion until no new subregions are created (no histograms have significant peaks).

A refinement of step 2 of this scheme is to create histograms in higher-dimensional space [Hanson and Riseman 1978]. Multiple regions are often in the same histogram peak when a single measurement is used. The advantage of the multimeasurement histograms is that these different regions are often separated into individual peaks, and hence the segmentation is improved. Figure 5.4 shows some results using a three-dimensional RGB color space.

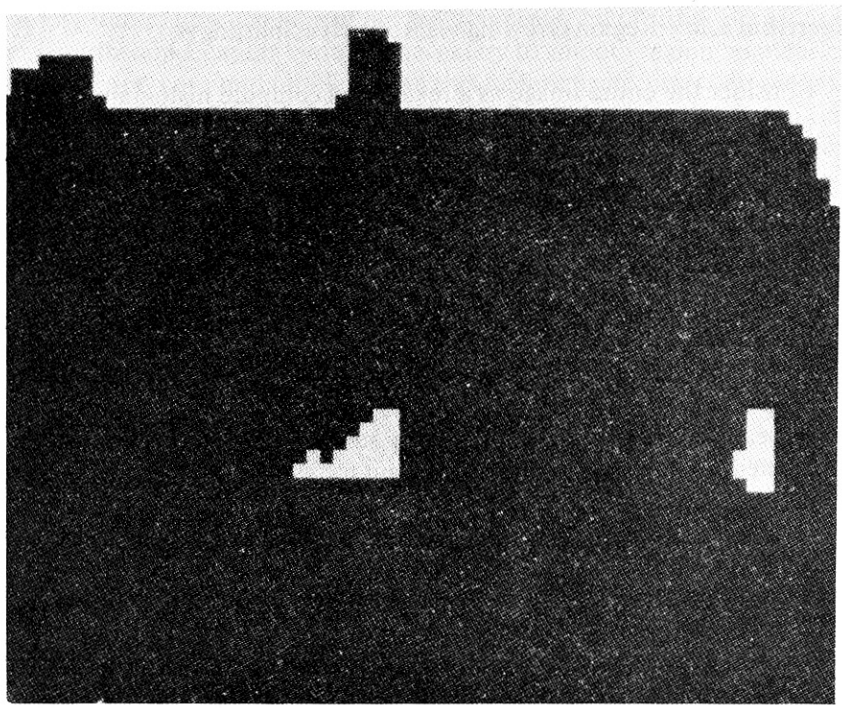
The figure shows the clear separation of peaks in the three-dimensional histogram that is not evident in either of the one-dimensional histograms. How many



(a)

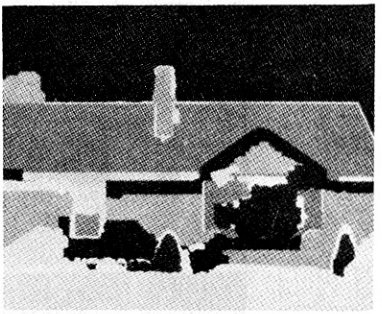


(b)



(c)

Fig. 5.3 Peak detection and threshold determination. (a) Original image. (b) Histograms. (c) Image segments resulting from first histogram peak.



(d)

Fig. 5.3 (d) Final segments.

dimensions should be used? Obviously, there is a trade-off here: As the dimensionality becomes larger, the discrimination improves, but the histograms are more expensive to compute and noise effects may be more pronounced.

5.3.2 Hierarchical Refinement

This technique uses a pyramidal image representation (Section 3.7) [Harlow and Eisenbeis 1973]. Region growing is applied to a coarse resolution image. When the algorithm has terminated at one resolution level, the pixels near the boundaries of regions are disassociated with their regions. The region-growing process is then repeated for just these pixels at a higher-resolution level. Figure 5.5 shows this structure.

5.4 SPLITTING AND MERGING

Given a set of regions R_k, $k = 1,\ldots,m$, a low-level segmentation might require the basic properties described in Section 5.1 to hold. The important properties from the standpoint of segmentation are Eqs. (5.5) and (5.6).

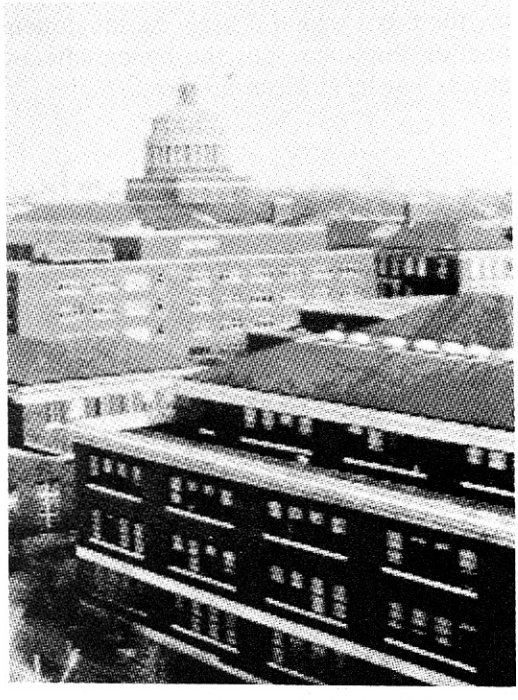
If Eq. (5.5) is not satisfied for some k, it means that that region is inhomogeneous and should be split into subregions. If Eq. (5.6) is not satisfied for some i and j, then regions i and j are collectively homogeneous and should be merged into a single region.

In our previous discussions we used

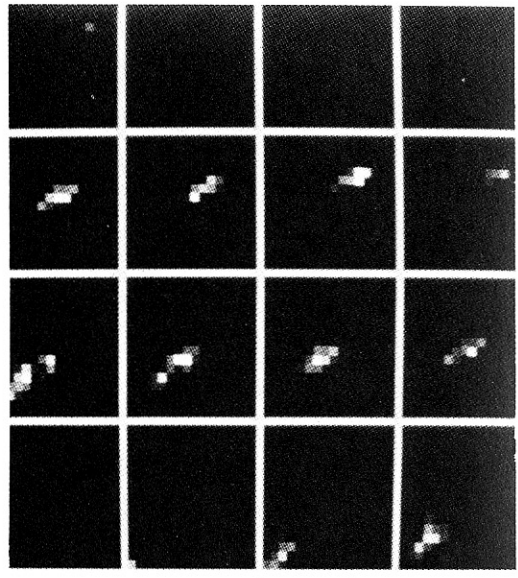
$$H(R) = \begin{cases} \text{true} & \text{if all neighboring pairs of points} \\ & \text{in } R \text{ are such that } f(\mathbf{x}) - f(\mathbf{y}) < T \\ \text{false} & \text{otherwise} \end{cases} \tag{5.7}$$

and

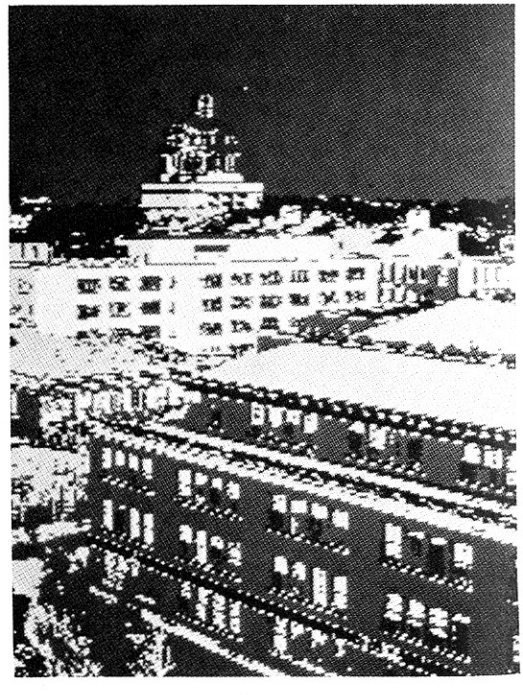
$$H(R) = \begin{cases} \text{true} & \text{if the points in } R \text{ pass a} \\ & \text{bimodality or peak test} \\ \text{false} & \text{otherwise} \end{cases} \tag{5.8}$$



(a)



(b)



(c)

Fig. 5.4 Multi-dimensional histograms in segmentation. (a) Image. (b) RGB histogram showing successive planes through a 16 × 16 × 16 color space. (c) Segments. (See color inserts.)

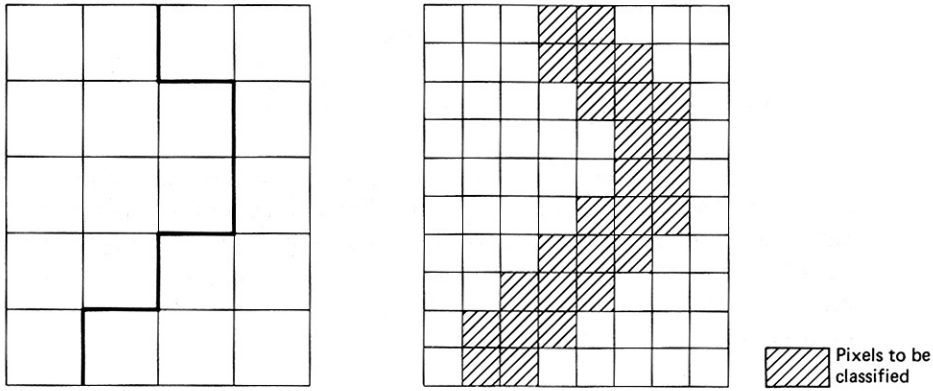


Fig. 5.5 Hierarchical region refinement.

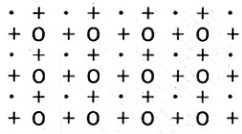
A way of working toward the satisfaction of these homogeneity criteria is the split-and-merge algorithm [Horowitz and Pavlidis 1974]. To use the algorithm it is necessary to organize the image pixels into a pyramidal grid structure of regions. In this grid structure, regions are organized into groups of four. Any region can be split into four subregions (except a region consisting of only one pixel), and the appropriate groups of four can be merged into a single larger region. This structure is incorporated into the following region-growing algorithm.

Algorithm 5.3: Region Growing via Split and Merge [Horowitz and Pavlidis 1974]

1. Pick any grid structure, and homogeneity property H. If for any region R in that structure, $H(R)$ = false, split that region into four subregions. If for any four appropriate regions $R_{k1}, \ldots, R_{k4}$, $H(R_{k1} \cup R_{k2} \cup R_{k3} \cup R_{k4})$ = true, merge them into a single region. When no regions can be further split or merged, stop.
2. If there are any neighboring regions R_i and R_j (perhaps of different sizes) such that $H(R_i \cup R_j)$ = true, merge these regions.

5.4.1 State-Space Approach to Region Growing

The "classical" state-space approach of artificial intelligence [Nilsson 1971, 1980] was first applied to region growing in [Brice and Fennema 1970] and significantly extended in [Feldman and Yakimovsky 1974]. This approach regards the initial two-dimensional image as a discrete state, where every sample point is a separate region. Changes of state occur when a boundary between regions is either removed or inserted. The problem then becomes one of searching allowable changes in state to find the best partition.



· Unassigned
+ Edge data
O Grey level data

Fig. 5.6 Grid structure for region representation [Brice and Fennema 1970].

An important part of the state-space approach is the use of data structures to allow regions and boundaries to be manipulated as units. This moves away from earlier techniques, which labeled each individual pixel according to its region. The high-level data structures do away with this expensive practice by representing regions with their boundaries and then keeping track of what happens to these boundaries during split-and merge-operations.

5.4.2 Low-level Boundary Data Structures

A useful representation for boundaries allows the splitting and merging of regions to proceed in a simple manner [Brice and Fennema 1970]. This representation introduces the notion of a supergrid S to the image grid G. These grids are shown in Fig. 5.6, where · and + correspond to supergrid and O to the subgrid. The representation is assumed to be four-connected (i.e., $\mathbf{x}1$ is a neighbor of $\mathbf{x}2$ if $\|\mathbf{x}1 - \mathbf{x}2\| \leqslant 1$).

With this notation boundaries of regions are directed crack edges (see Sec. 3.1) at the points marked +. That is, if point $\mathbf{x}_k$ is a neighbor of $\mathbf{x}_j$ and $\mathbf{x}_k$ is in a different region than $\mathbf{x}_j$, insert two edges for the boundaries of the regions containing $\mathbf{x}_j$ and $\mathbf{x}_k$ at the point + separating them, such that each edge traverses its associated region in a counterclockwise sense. This makes merge operations very simple: To merge regions R_k and R_l, remove edges of the opposite sense from the boundary as shown in Fig. 5.7a. Similarly, to split a region along a line, insert edges of the opposite sense in nearby points, as shown in Fig. 5.7b.

The method of [Brice and Fennema 1970] uses three criteria for merging regions, reflecting a transition from local measurements to global measurements. These criteria use measures of boundary strength s_{ij} and w_{ij} defined as

$$s_{ij} = |f(\mathbf{x}_i) - f(\mathbf{x}_j)| \tag{5.9}$$

$$w_k = \begin{cases} 1 & \text{if } s_k < T_1 \\ 0 & \text{otherwise} \end{cases} \tag{5.10}$$

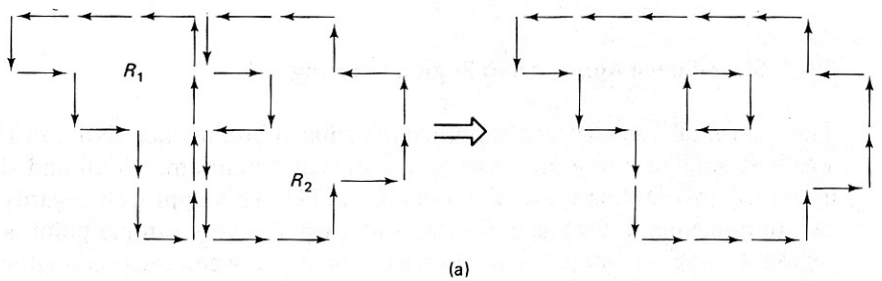


Fig. 5.7 Region operations on the grid structure of Fig. 5.6.

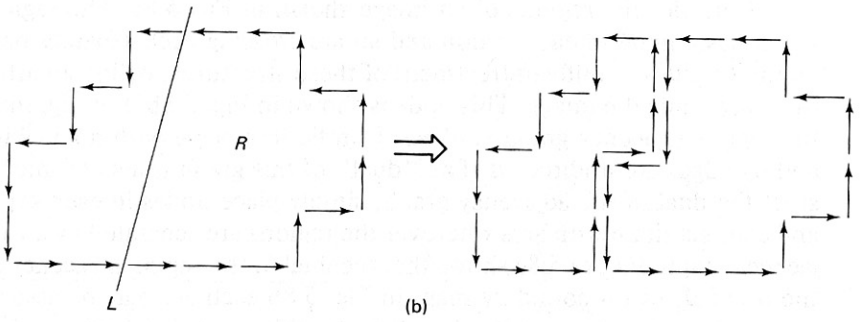


Fig. 5.7 (cont.)

where $\mathbf{x}_i$ and $\mathbf{x}_j$ are assumed to be on either side of a crack edge (Chapter 3). The three criteria are applied sequentially in the following algorithm:

Algorithm 5.4: Region Growing via Boundary Melting (T_k, $k = 1, 2, 3$ are preset thresholds)

1. For all neighboring pairs of points, remove the boundary between $\mathbf{x}_i$ and $\mathbf{x}_j$ if $i \neq j$ and $w_{ij} = 1$. When no more boundaries can be removed, go to step 2.
2. Remove the boundary between R_i and R_j if

$$\frac{W}{\min [p_i, p_j]} \geqslant T_2 \tag{5.11}$$

   where W is the sum of the w_{ij} on the common boundary between R_i and R_j, that have perimeters p_i and p_j respectively. When no more boundaries can be removed, go to step 3.
3. Remove the boundary between R_i and R_j if

$$W \geqslant T_3 \tag{5.12}$$

5.4.3 Graph-Oriented Region Structures

The Brice–Fennema data structure stores boundaries explicitly but does not provide for explicit representation of regions. This is a drawback when regions must be referred to as units. An adjunct scheme of region representation can be developed using graph theory. This scheme represents both regions and their boundaries explicitly, and this facilitates the storing and indexing of their semantic properties.

The scheme is based on a special graph called the *region adjacency graph*, and its "dual graph." In the region adjacency graph, nodes are regions and arcs exist between neighboring regions. This scheme is useful as a way of keeping track of regions, even when they are inscribed on arbitrary nonplanar surfaces (Chapter 9).

Consider the regions of an image shown in Fig. 5.8a. The region adjacency graph has a node in each region and an arc crossing each separate boundary segment. To allow a uniform treatment of these structures, define an artificial region that surrounds the image. This node is shown in Fig. 5.8b. For regions on a plane, the region adjacency graph is *planar* (can lie in a plane with no arcs intersecting) and its edges are undirected. The "dual" of this graph is also of interest. To constuct the dual of the adjacency graph, simply place nodes in each separate region and connect them with arcs wherever the regions are separated by an arc in the adjacency graph. Figure 5.8c shows that the dual of the region adjacency graph is like the original region boundary map; in Fig. 5.8b each arc may be associated with a specific boundary segment and each node with a junction between three or more boundary segments. By maintaining both the region adjacency graph and its dual, one can merge regions using the following algorithm:

Algorithm 5.5: Merging Using the Region-Adjacency Graph and Its Dual

Task: Merge neighboring regions R_i and R_j.

Phase 1. Update the region-adjacency graph.

1. Place edges between R_i and all neighboring regions of R_j (excluding, of course, R_i) that do not already have edges between themselves and R_i.
2. Delete R_j and all its associated edges.

Phase 2. Take care of the dual.

1. Delete the edges in the dual corresponding to the borders between R_i and R_j.
2. For each of the nodes associated with these edges:
   (a) if the resultant degree of the node is less than or equal to 2, delete the node and join the two dangling edges into a single edge.
   (b) otherwise, update the labels of the edges that were associated with j to reflect the new region label i.

Figure 5.9 shows these operations.

5.5 INCORPORATION OF SEMANTICS

Up to this point in our treatment of region growers, domain-dependent "semantics" has not explicitly appeared. In other words, region-merging decisions were based on raw image data and rather weak heuristics of general applicability about the likely shape of boundaries. As in early processing, the use of domain-dependent knowledge can affect region finding. Possible interpretations of regions can affect the splitting and merging process. For example, in an outdoor scene possible region interpretations might be sky, grass, or car. This kind of knowledge is quite separate from but related to measurable region properties such as intensity

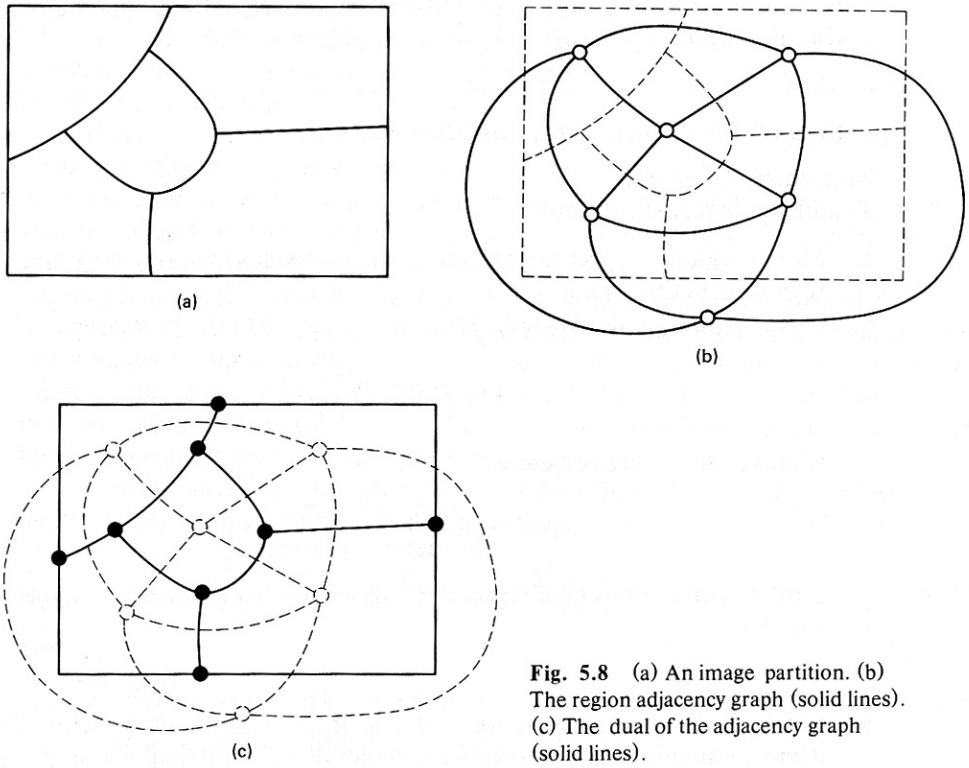


Fig. 5.8 (a) An image partition. (b) The region adjacency graph (solid lines). (c) The dual of the adjacency graph (solid lines).

and hue. An example shows how semantic labels for regions can guide the merging process. This approach was originally developed in [Feldman and Yakimovsky 1974]. it has found application in several complex vision systems [Barrow and Tenenbaum 1977; Hanson and Riseman 1978].

Early steps in the Feldman–Yakimovsky region grower used essentially the same steps as Brice–Fennema. Once regions attain significant size, semantic cri-

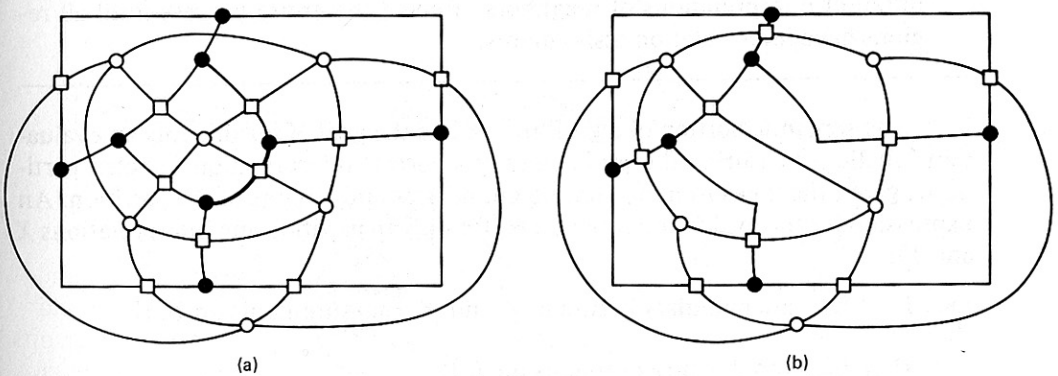


Fig. 5.9 Merging operations using the region adjacency graph and its dual. (a) Before merging regions separated by dark boundary line. (b) After merging.

teria are used. The region growing consists of four steps, as summed up in the following algorithm:

Algorithm 5.6 Semantic Region Growing

Nonsemantic Criteria
T_1 and T_2 are preset thresholds

1. Merge regions i, j as long as they have one weak separating edge until no two regions pass this test.
2. Merge regions i, j where $S(i, j) \leqslant T_2$ where

$$S(i, j) = \frac{c_1 + \alpha_{ij}}{c_2 + \alpha_{ij}}$$

where c_1 and c_2 are constants,

$$\alpha ij = \frac{(\text{area}_i)^{1/2} + (\text{area}_j)^{1/2}}{\text{perimeter}_i \cdot \text{perimeter}_j}$$

until no two regions pass this test. (This is a similar criterion to Algorithm 5.4, step 2.)

Semantic Criteria

3. Let B_{ij} be the boundary between R_i and R_j. Evaluate each B_{ij} with a Bayesian decision function that measures the (conditional) probability that B_{ij} separates two regions R_i and R_j of the *same interpretation.* Merge R_i and R_j if this conditional probability is less than some threshold. Repeat step 3 until no regions pass the threshold test.
4. Evaluate the interpretation of each region R_i with a Bayesian decision function that measures the (conditional) probability that an interpretation is the correct one for that region. Assign the interpretation to the region with the highest confidence of correct interpretation. Update the conditional probabilities for different interpretations of neighbors. Repeat the entire process until all regions have interpretation assignments.

The semantic portion of algorithm 5.6 had the goal of maximizing an evaluation function measuring the probability of a correct interpretation (labeled partition), given the measurements on the boundaries and regions of the partition. An expression for the evaluation function is (for a given partition and interpretations X and Y):

$$\max_{X, Y} \prod_{i, j} \{P[B_{ij} \text{ is a boundary between } X \text{ and } Y \mid \text{measurements on } B_{ij}]\}$$
$$\times \prod_{i} \{P[R_i \text{ is an } X \mid \text{measurements on } R_i]\}$$
$$\times \prod_{i} \{P[R_j \text{ is an } Y \mid \text{measurements on } R_j]\}$$

where P stands for probability and Π is the product operator.

How are these terms to be computed? Ideally, each conditional probability function should be known to a reasonable degree of accuracy; then the terms can be obtained by lookup.

However, the straightforward computation and representation of the conditional probability functions requires a massive amount of work and storage. An approximation used in [Feldman and Yakimovsky 1974] is to quantize the measurements and represent them in terms of a classification tree. The conditional probabilities can then be computed from data at the leaves of the tree. Figure 5.10 shows a hypothetical tree for the region measurements of intensity and hue, and interpretations ROAD, SKY, and CAR. Figure 5.11 shows the equivalent tree for two boundary measurements m and n and the same interpretations. These two figures indicate that $P[R_i \text{ is a CAR} \mid 0 \leqslant i < I, 0 \leqslant h < H_1] =$, and $P[B_{ij}$ divides two car regions $\mid M_k \leqslant m < M_{k+1}, N_l < n \leqslant N_{l+1} =$. These trees were created by laborious trials with correct segmentations of test images.

Now, finally, consider again step 3 of Algorithm 5.6. The probability that a boundary B_{ij} between regions R_i and R_j is false is given by

$$P_{\text{false}} = \frac{P_f}{P_t + P_f} \tag{5.13}$$

where

$$P_f = \sum \{P[B_{ij} \text{ is between two subregions } X \mid B_{ij}\text{'s measurements}]\} \times \{P[R_i \text{ is } X \mid \text{meas}]\} \times \{P[R_j \text{ is } X \mid \text{meas}]\} \tag{5.14a}$$

$$P_t = \sum_{x,y} \{P[B_{ij} \text{ is between } X \text{ and } Y \mid \text{meas}]\} \times \{P[R_i \text{ is } X \mid \text{meas}]\} \times \{P[R_j \text{ is } Y \mid \text{meas}]\} \tag{5.14b}$$

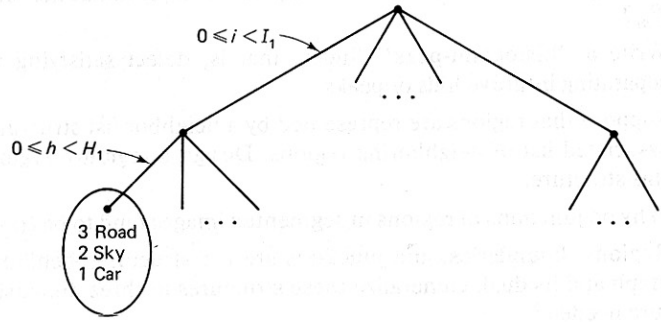


Fig. 5.10 Hypothetical classification tree for region measurements showing a particular branch for specific ranges of intensity and hue.

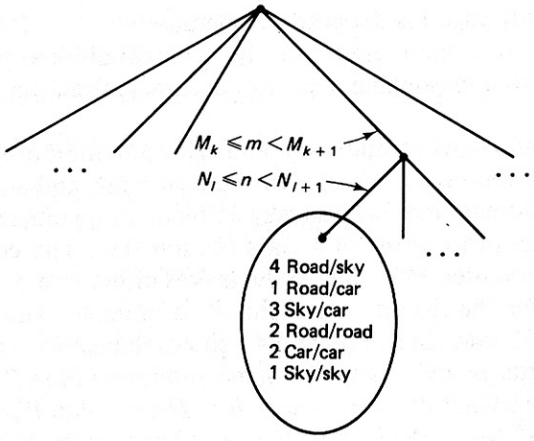


Fig. 5.11 Hypothetical classification tree for boundary measurements showing a specific branch for specific ranges of two measurements m and n.

And for step 4 of the algorithm,

$$\text{Confidence}_i = \frac{P[R_i \text{ is } X1 \mid \text{meas}]}{P[R_i \text{ is } X2 \mid \text{meas}]} \tag{5.15}$$

where $X1$, $X2$ are the first and second most likely interpretations, respectively. After the region is assigned interpretation $X1$, the neighbors are updated using

$$P[R_i \text{ is } X \mid \text{meas}] := Prob\ [Rj \text{ is } X \mid \text{meas}] \times P[B_{ij} \text{ is between } X \text{ and } X1 \mid \text{meas}] \tag{5.16}$$

EXERCISES

5.1 In Algorithm 5.1, show how one can handle the case where colors are equivalent. Do you need more than one pass over the image?

5.2 Show for the heuristic of Eq. (5.11) that

(a) $IT_2 \geqslant WT_2 > P_j$

(b) $P_m < P_i + I(1/T_2 - 2)$

where P_m is the perimeter of $R_i \bigcup R_j$, I is the perimeter common to both i and j and $P_m = \min(P_i, P_j)$. What does part (b) imply about the relation between T_2 and P_m?

5.3 Write a "histogram-peak" finder; that is, detect satisfying valleys in histograms separating intuitive hills or peaks.

5.4 Suppose that regions are represented by a neighbor list structure. Each region has an associated list of neighboring regions. Design a region-merging algorithm based on this structure.

5.5 Why do junctions of regions in segmented images tend to be trihedral?

5.6 Regions, boundaries, and junctions are the structures behind the region-adjacency graph and its dual. Generalize these structures to three dimensions. Is another structure needed?

5.7 Generalize the graph of Figure 5.8 to three dimensions and develop the merging algorithm analogous to Algorithm 5.5. (Hint: see Exercise 5.6.)

REFERENCES

BARROW, H. G. and J. M. TENENBAUM. "Experiments in model-driven scene segmentation." *Artificial Intelligence 8*, 3, June 1977, 241–274.

BRICE, C. and C. FENNEMA. "Scene analysis using regions." *Artificial Intelligence 1*, 3, Fall 1970, 205–226.

CHOW, C. K. and T. KANEKO. "Automatic boundary detection of the left ventricle from cineangiograms." *Computers and Biomedical Research 5*, 4, August 1972, 388–410.

FELDMAN, J. A. and Y. YAKIMOVSKY. "Decision theory and artificial intelligence: I. A semantics-based region analyzer." *Artificial Intelligence 5*, 4, 1974, 349–371.

HANSON, A. R. and E. M. RISEMAN. "Segmentation of natural scenes." In *CVS*, 1978.

HARLOW, C. A. and S. A. EISENBEIS. "The analysis of radiographic images." *IEEE Trans. Computers 22*, 1973, 678–688.

HOROWITZ, S. L. and T. PAVLIDIS. "Picture segmentation by a directed split-and-merge procedure." *Proc.*, 2nd IJCPR, August 1974, 424–433.

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

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

OHLANDER, R., K. PRICE, and D. R. REDDY. "Picture segmentation using a recursive region splitting method." *CGIP 8*, 3, December 1979.

ROBERTSON, T. V., P. H. SWAIN, and K. S. FU. "Multispectral image partitioning." TR-EE 73-26 (LARS Information Note 071373), School of Electrical Engineering, Purdue Univ., August 1973.

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

ZUCKER, S. W. "Region growing: Childhood and adolescence." *CGIP 5*, 3, September 1976, 382–399.