B. LOW-LEVEL VISION

UNDERLYING some approaches to computational vision and to machine vision are basic tasks of breaking up an image into component regions. This *segmentation* problem must be tackled before determining 3-D surface characteristics or recognizing objects in the scene. A large variety of methods have been invented and studied for this initial analysis task. The next subsection gives an overview of this subfield, expanding upon the description of *region analysis* in Article XIII.C5 in Volume III.

B1. Segmentation Techniques

IN TRADITIONAL APPROACHES to computer vision, the pixels of an image are grouped into regions in a process called *segmentation*, and this is done prior to any attempt to interpret the regions as objects in the scene. A perceived advantage of computing a segmentation is that one could, relatively easily, achieve a relatively concise representation of the image's essential pictorial aspects, and that this would permit the semantic phase of the analysis to be accomplished painlessly. Except in certain artificial environments, segmentation has proven to be difficult in itself, and it seems that semantic considerations are often needed at the segmentation level. Nonetheless, various segmentation methods make up an important part of the arsenal of techniques that can be employed in computer vision, and they provide a good starting point for a tutorial overview of developments in vision.

What should a good image segmentation be? Although this depends largely on the application, it can be answered in an application-independent way to a certain extent. Let us attempt to do so.

Regions of an image segmentation should be homogeneous—uniform with respect to some characteristic such as gray tone or texture. Region interiors should usually be simple and without many small holes. Adjacent regions of a segmentation should have significantly different values with respect to the characteristic on which they are uniform. Boundaries of each segment should be simple, not ragged, and must be spatially accurate.

Achieving all these desired properties is difficult because strictly uniform and homogeneous regions are typically full of small holes and have ragged boundaries. Insisting that adjacent regions have large differences in values can cause regions that ought to be kept separated to merge and thus the intervening boundaries to be lost.

Just as there is no generally accepted theory of clustering in statistics, there is no well-accepted theory of image segmentation. Image segmentation techniques tend to be ad hoc. They differ in the ways in which they emphasize one or more of the desired properties and in the ways in which they balance and compromise one desired property against another.

Image segmentation techniques can be classified into one of the following groups:

1. Measurement-space–guided spatial clustering
2. Single-linkage region-growing schemes
3. Hybrid-linkage region-growing schemes
4. Centroid-linkage region-growing schemes
5. Spatial clustering schemes
6. Split-and-merge schemes

As this brief typology suggests, image segmentation can be viewed as a clustering process. The difference between image segmentation and clustering is in grouping. In clustering, the grouping is done in measurement space (e.g., the space of gray values rather than the space of pixel coordinate pairs). In image segmentation, the grouping is done on the spatial domain of the image, and there is an interplay in the clustering between the (possibly overlapping) groups in measurement space and the mutually exclusive groups of the image segmentation.

The single-linkage region-growing schemes are the simplest and most prone to the unwanted region-merge errors. The hybrid-linkage and centroid-linkage region-growing schemes are better in this regard. The split-and-merge technique is not as subject to the unwanted region-merge error. However, it suffers from large memory usage and excessively blocky region boundaries. The measurement-space–guided spatial clustering tends to avoid both the region-merge errors and the blocky boundary problems because of its primary reliance on measurement space. But the regions produced are not smoothly bounded, and they often have holes, giving the effect of salt-and-pepper noise. The spatial clustering schemes may be better in this regard, but they have not been tested well enough. The hybrid-linkage schemes appear to offer the best compromise between having smooth boundaries and few unwanted region merges.

The remainder of this section describes the main ideas behind the major image segmentation techniques. Additional image segmentation surveys can be found in Zucker (1976), Riseman and Arbib (1977), Kanade (1980), and Fu and Mui (1981), and Haralick and Shapiro (1985).

Measurement-space–Guided Spatial Clustering

This technique for image segmentation uses the measurement-space clustering process to define a partition in measurement space (e.g., the space of pixel gray values of the image). Then each pixel is assigned the label of the cell in the measurement-space partition to which it belongs. The image segments are defined as the connected components of the pixels having the same label.

The accuracy of image segmentation using the measurement-space clustering process depends directly on how well the objects of interest on the image separate into distinct measurement-space clusters. Typically the process works well in situations where there are a few kinds of distinct objects having widely different gray-tone intensities (or gray-tone intensity vectors, for multiband images) and these objects appear on a nearly uniform background.

Clustering procedures that use the pixel as a unit and compare each pixel value with every other pixel value can require excessively large computation times because of the large number of pixels in an image. Iterative partition-rearrangement schemes such as ISODATA have to go through the image data set many times and if done without sampling can also take excessive computation time. Histogram-mode seeking, because it requires only one pass through the data, probably involves the least computation time of the measurement-space clustering techniques, and it is the one we discuss here.

Histogram-mode seeking is a measurement-space clustering process in which it is assumed that homogeneous objects on the image manifest themselves as the clusters in measurement space. Image segmentation is accomplished by mapping the clusters back to the image domain where the maximal connected components of the mapped back clusters constitute the image segments. For single-band images, calculation of this histogram in an array is direct. The measurement-space clustering can be accomplished by determining the valleys in this histogram and declaring the clusters to be the interval of values between valleys. A pixel whose value is in the ith interval is labeled with index i and the segment it belongs to is one of the connected components of all pixels whose label is i.

Ohlander et al. (1975) refines the clustering idea in a recursive way. He begins by defining a mask selecting all pixels on the image. Given any mask, a histogram of the masked image is computed. Measurement-space clustering enables the separation of one mode of the histogram set from another mode. Pixels on the image are then identified with the cluster to which they belong. If there is only one measurement-space cluster, the mask is terminated. If multiple clusters are present, the process is repeated for each connected component (region) associated with each cluster. Note that one cluster may produce more than one connected component. During successive iterations, the next mask in the stack selects pixels in the histogram-computation process. Clustering is repeated for each new mask until the stack is empty. The process is illustrated in Figure B–1.

Single-linkage Region Growing

Single-linkage region growing schemes regard each pixel as a node in a graph. Neighboring pixels whose properties are "similar enough" are joined by an arc. The image segments are maximal sets of pixels all belonging to the same connected component. Single-linkage image-segmentation schemes are attractive for their simplicity. They do, however, have a problem with chaining, because it takes only one arc leaking from one region to a neighboring one to cause the regions to merge.

The simplest single-linkage scheme defines "similar enough" by pixel difference. Two neighboring pixels are similar enough if the absolute value of the difference between their gray-tone intensity values is small enough. Bryant (1979) defines "similar enough" by normalizing the difference by the quantity $\sqrt{2}$ times the root-mean-square value of neighboring pixel differences taken over the entire image.

For pixels having vector values, the obvious generalization is to use a vector norm of the pixel-difference vector. Instead of using a Euclidean distance, Asano and Yokoya (1981) suggest that two pixels be joined together if the absolute value of their difference is small enough compared to the average absolute value of the center pixel minus neighbor pixel for each of the neighborhoods to which the pixels belong. The ease with which unwanted region chaining can occur with this technique limits its potential on complex or noisy data.

Hybrid-linkage Region Growing

Hybrid single-linkage techniques are more powerful than the simple single-linkage technique. The hybrid techniques seek to assign a property vector to each pixel where the property vector depends on the neighborhood of the pixel. Pixels that are similar are so because their neighborhoods in some special sense are similar. Similarity is thus established as a function of neighboring pixel values, and this makes the technique better behaved on noisy data.

One hybrid single-linkage scheme relies on an edge operator to establish whether two pixels are joined with an arc. Here an edge operator is applied to the image, labeling each pixel as edge or nonedge. Neighboring pixels, neither of which are edges, are joined by an arc. The initial segments are the connected components of the nonedge labeled pixels. The edge pixels can either be left as edges and be considered as background or they can be assigned to the spatially nearest region having a

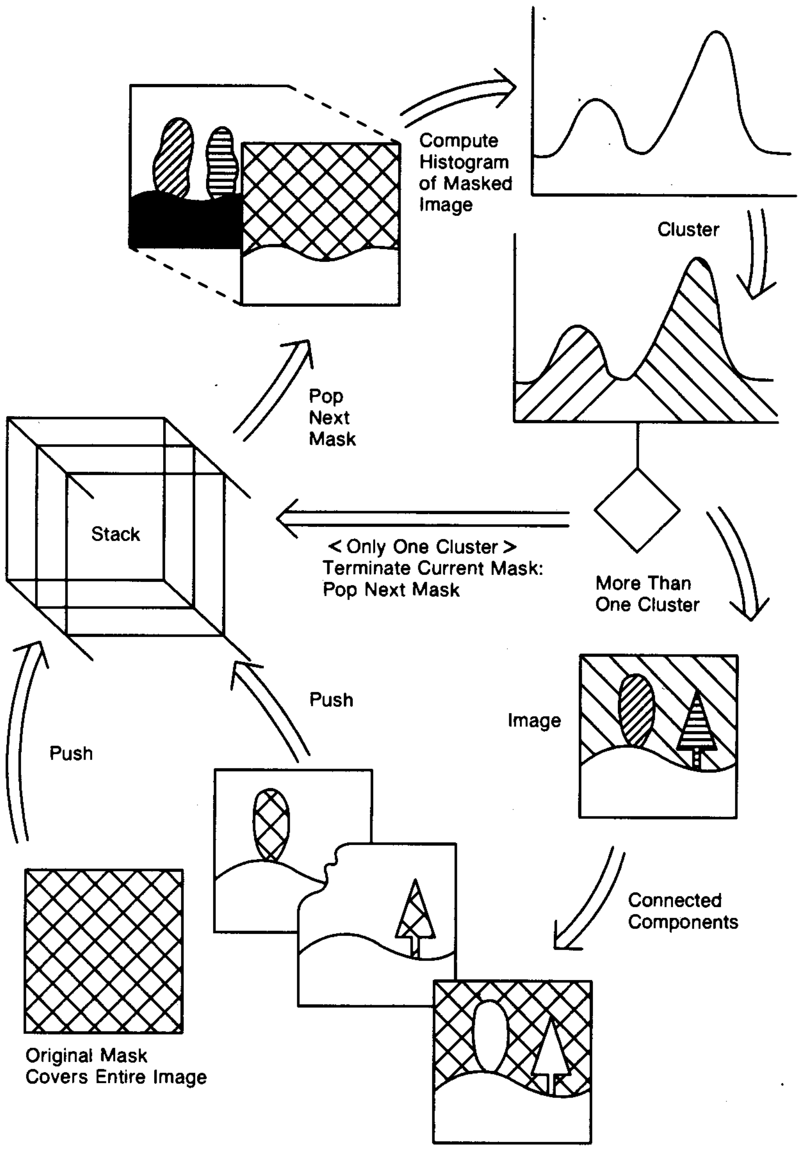

Figure B–1. The recursive histogram spatial clustering method of Ohlander.

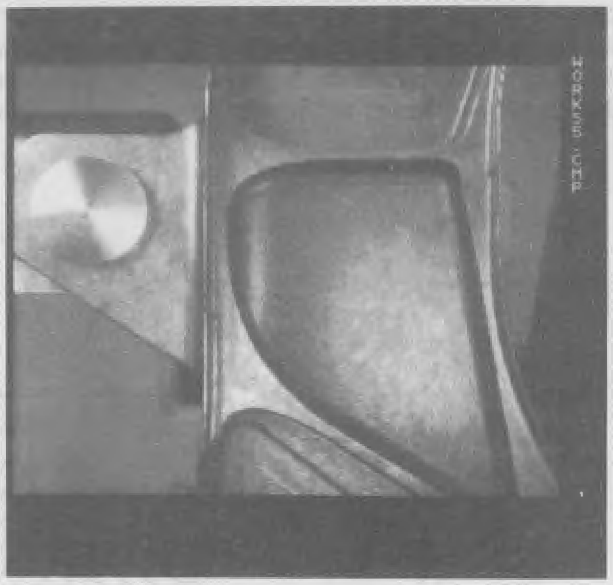

Figure B–2. Image of a bulkhead of an F-15 aircraft.

label. Successful use of this technique may require closing edge gaps before performing the region growing.

Figure B–2 illustrates an image of a section of an F-15 aircraft bulkhead. Figure B–3 illustrates a second directional derivative zero-crossing operator applied to the image of Figure B–2. Figure B–4 shows the segmentation that results from connecting the non-edge pixels. The method is thus a hybrid-linkage region-growing scheme in which any pair of neighboring pixels, neither of which are edge pixels, can link together. The resulting segmentation consists of the connected components of the nonedge pixels and where each edge pixel is assigned to its nearest connected component.

Centroid-linkage Region Growing

In centroid-linking region growing, in contrast with single-linkage region growing, pairs of neighboring pixels are not compared for similarity. Rather, the image is scanned in some predetermined manner such as left to right or top to bottom. A pixel's value is compared to the mean of an already existing but not necessarily completed neighboring segment. If its value and the segment's mean value are close enough, the pixel is added to the segment and the segment's mean is updated. If more than one region is close enough, it is added to the closest region. However, if the means of the two competing regions are close enough, the two

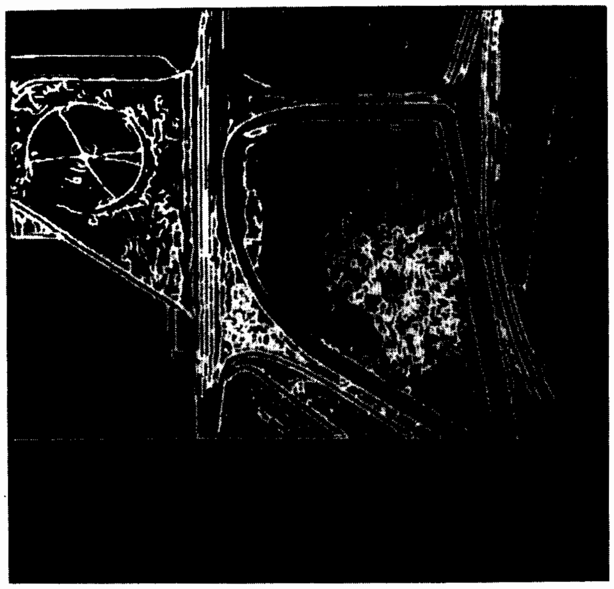

Figure B–3. Directional derivative zero-crossing operator applied to the F-15 image.

regions are merged and the pixel is added to the merged region. If no neighboring region has its mean close enough, a new segment is established having the given pixel's value as its first member. The scan geometry for the centroid-linkage region-growing scheme is shown in Figure B–5.

Keeping track of the means and scatters for all region as they are being determined does not require large amounts of memory space. There cannot be more regions active at one time than the number of pixels in a row of the image. Hence a hash table mechanism with the space of a small multiple of the number of pixels in a row can work well.

One way of performing the region growing is by the use of the *T*-test. Let *R* be a segment of *N* pixels neighboring a pixel with gray-tone intensity *y*. Define the mean $\overline{X}$ and scatter S^2 by

$$X = \frac{1}{N} \sum_{(r,c)\epsilon R} I(r, c) \tag{1}$$

and

$$S^2 = \sum_{(r,c)\epsilon R} (I(r, c) - X)^2 \tag{2}$$

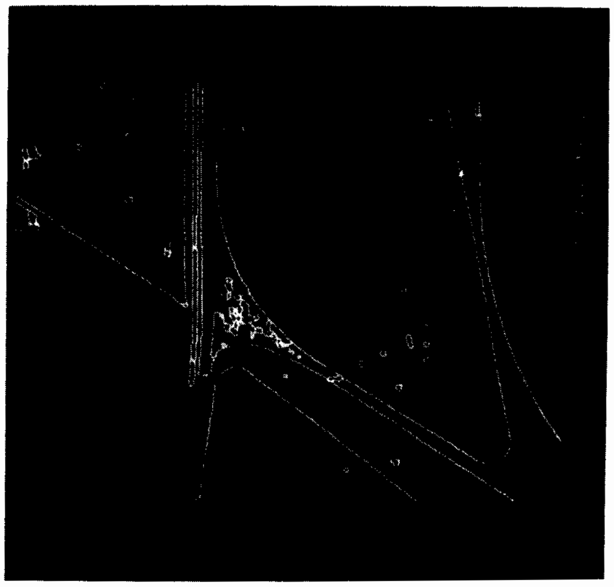

Figure B–4. Segmentation of the F-15 image.

Under the assumption that all the pixels in R and the test pixel y are independent and identically distributed normals, the statistic

$$T = \left[\frac{(N-1)N}{(N+1)}(y - \overline{X})^2/S^2\right]^{1/2} \tag{3}$$

has a T_{N-1} distribution. If T is small enough, y is added to region R and the mean and scatter are updated using y. The new mean and scatter are given by

$$\overline{X}_{\text{new}} \leftarrow (N\overline{X}_{\text{old}} + y)/(N+1) \tag{4}$$

and

$$S^2_{\text{new}} \leftarrow S^2_{\text{old}} + (y - \overline{X})^2 + N(\overline{X}_{\text{new}} - \overline{X}_{\text{old}})^2 \tag{5}$$

If T is too high, the value y is not likely to have arisen from the population of pixels in R. If y is different from all of its neighboring regions, it begins its own region. A slightly stricter linking criterion can require that not only must y be close enough to the mean of the neighboring regions, but also that a neighboring pixel in that region must have a close enough value to y. This combines a centroid linkage and single linkage criterion.

The Levine and Shaheen scheme (1981) is similar. The difference is that Levine and Shaheen attempt to keep regions more homogeneous

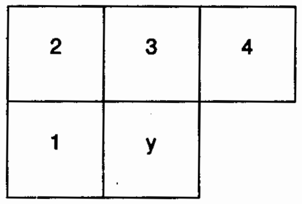

Figure B–5. Region-growing geometry for the one-pass scan, left-right, top-bottom region growing.

and try to keep the region scatter from getting too high. They do this by requiring the differences to be more significant before a merge takes place if the region scatter is high. For a user-specified value θ, they define a test statistic T where

$$T = |y - X_{\text{new}}| - (1 - S/\overline{X}_{\text{new}})\theta \qquad (6)$$

If $T < 0$ for the neighboring region R in which $|y - \overline{X}|$ is the smallest, y is added to R. If $T > 0$ for the neighboring region in which $|y - \overline{X}|$ is the smallest, y begins a new region.

Figure B–6 illustrates the application of the centroid-linkage region-growing technique to the bulkhead image. This application uses two successive scans of the image. The first is a left-right top-down scan, and the second is a right-left bottom-top scan.

Hybrid-linkage Combination Techniques

The centroid-linkage and the hybrid-linkage methods can be combined in a way that takes advantage of their relative strengths. The strength of the single-linkage method is that boundaries are placed in a spatially accurate way. Its weakness is that edge gaps result in excessive merging. The strength of the centroid-linkage method is its ability to place boundaries in weak-gradient areas. It can do this because it does not depend on a large difference between the pixel and its neighbor to declare a boundary. It depends instead on a large difference between the pixel and the mean of the neighboring region to declare a boundary.

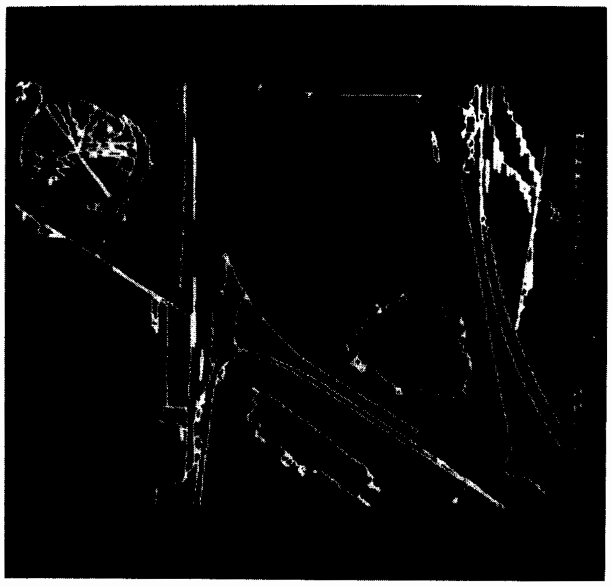

Figure B–6. The two-pass top-down centroid segmentation of the bulkhead image.

The combined centroid-hybrid linkage technique does the obvious thing. Centroid linkage is only done for nonedge pixels; that is, region growing is not permitted across edge pixels. Saying it another way, edge pixels are not permitted to be assigned to any region and cannot link to any region. Thus, if the parameters of centroid linkage were set so that any difference, however large, between pixel value and region mean was considered small enough to permit merging, the two-pass hybrid combination technique would produce a connected components of the nonedge pixels. As the difference criterion is made more strict, the centroid linkage produces boundaries in addition to those produced by the edges. Figure B–7 illustrates the application of the hybrid-linkage technique to the bulkhead image.

Split-and-Merge

The split-and-merge method for segmentation begins with the entire image as the initial segment. Then it successively splits each of its current segments into quarters if the segment is not homogeneous enough. Homogeneity can be easily established by determining if the difference between the largest and smallest gray-tone intensities is small enough. Algorithms of this type were first suggested by Robertson (1973)

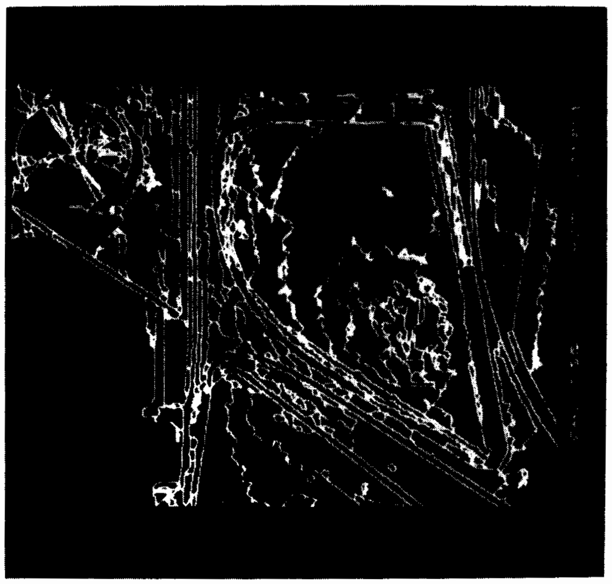

Figure B–7. Segmentation using the one-pass combined centroid and hybrid linkage method.

and Klinger (1973). Kettig and Landgrebe (1975) try to split all nonuniform 2×2 neighborhoods before beginning the region merging. Fukada (1980) suggests successively splitting a region into quarters until the sample variance is small enough. The efficiency of the split-and-merge method can be increased by arbitrarily partitioning the image into square regions of a user-selected size and then splitting these further if they are not homogeneous.

Because segments are successively divided into quarters, the boundaries produced by the split technique tend to be squarish and slightly artificial. Sometimes adjacent quarters coming from adjacent split segments need to be joined rather than remain separate. Horowitz and Pavlidis (1976) suggest a split-and-merge strategy to take care of this problem. Muerle and Allen (1968) suggest merging a pair of adjacent regions if their gray-tone intensity distributions are similar enough. They recommend the Kolmogorov-Smirnov test.

Chen and Pavlidis (1980) suggest using statistical tests for uniformity rather than a simple examination of the difference between the largest and smallest gray-tone intensities in the region under consideration for splitting. The uniformity test requires that there be no significant difference between the mean of the region and each of its quarters.

The Chen and Pavlidis tests assume that the variances are equal and known.

Let each quarter have K pixels, X_{ij} be the jth pixel in the ith region, X_i be the mean of the ith quarter, and $X_{..}$ be the grand mean of all the pixels in the four quarters. Then, for a region to be considered homogeneous, Chen and Pavlidis require that

$$|X_i - X_{..}| \leq \epsilon, \qquad i = 1, 2, 3, 4 \tag{7}$$

We give here the F-test for testing the hypothesis that the mean and variances of the quarters are identical. The value of variance is not assumed known. If we assume that the regions are independent and identically distributed normals, the optimal test is given by the statistic F, which is defined by

$$F = \frac{K \sum_{i=1}^{4}(X_{i.} - X_{..})^2/3}{\sum_{i=1}^{4} \sum_{k=1}^{K}(X_{ik} - X_{i.})^2/4(K-1)} \tag{8}$$

It has a $F_{3,4(K-1)}$ distribution. If F is too high, the region is declared not uniform.

The data structures required to do a split-and-merge on images larger than 512×512 are very large. Execution of the algorithm on virtual-memory computers results in so much paging that the dominant activity may be paging rather than segmentation. Browning and Tanimoto (1982) describe a split-and-merge scheme where the split-and-merge is first accomplished on mutually exclusive subimage blocks and the resulting segments are then merged between adjacent blocks to take care of the artificial block boundaries.

B2. Edges

IF AN IMAGE is successfully segmented into regions, the contours of the regions are available for shape analysis. However, it is sometimes more expedient to compute the contours directly from the image, rather than to go through one of the previously described segmentation processes. To compute contours directly from the image, "edge detection" must be performed. This subsection discusses the important characteristics of edges. Edge detection continues to be a subject of intense research. Elementary methods for edge detection, including the Roberts cross operator and the Sobel operator, are described in Article XIII.C4, Vol. III.

The Difficulties of Finding the Contours of Objects in an Image

What is an edge in a digital image? The first intuitive notion is that a digital edge occurs on the boundary between two pixels when the respective brightness values of the two pixels are significantly different. "Significantly different" may depend on the distribution of brightness values around each of the pixels.

We often point to a region on an image and say this region is *brighter* than its surrounding area, meaning that the mean of the brightness values of pixels inside the region is greater than the mean of the brightness values outside the region. Having noticed this, we would then say that an *edge* exists between each pair of neighboring pixels where one pixel is inside the region and the other is outside the region. Such edges are referred to as *step edges*.

Step edges are not the only kind of edge. If we scan through a region left to right observing the brightness values steadily increasing, and then after a certain point we observe that the brightness values are steadily decreasing, we are likely to say that there is an edge at the point of change from increasing to decreasing brightness values. Such edges are called *roof edges*.

Thus, in general, an *edge* is a place in an image where there appears to be a jump in brightness value or a jump in brightness value derivative.

In some sense, this summary statement about edges is quite revealing because in a discrete array of brightness values there are jumps (in the literal sense) between neighboring brightness values if the brightness values are different, even if only slightly different. Perhaps more to the heart of the matter, there exists no definition of derivative for a discrete array of brightness values. The only way to interpret jumps in value and jumps in derivatives when referring to a discrete array of values is to assume that the discrete array of values comes about as some kind of sampling of a real-valued function defined on a bounded and connected subset of the real plane R^2. The jumps in value and jumps in derivative really must refer to points of discontinuity of f and to points of discontinuity in the partial derivatives of f.

Edge finders should then regard the digital picture function as a sampling of the underlying function f, where some kind of random noise has been added to the true function values. To do this, the edge finder must assume some kind of parametric form for the underlying function f, use the sampled brightness values of the digital picture function to estimate the parameters, and finally make decisions regarding the locations of discontinuities of the underlying function and its partial derivatives based on the estimated values of the parameters.

Of course, it is impossible to determine the true locations of discontinuities in value or derivatives based on samplings of the functions. The locations are estimated by function approximation. The location of the estimated discontinuity will be where the first derivative has a relative maximum. This is where the second derivative will have a negatively shaped zero-crossing if the edge is being crossed from low value to high value. Sharp discontinuities will reveal themselves in high values for estimates of first partial derivatives. Sharp discontinuities in derivative will reveal themselves in high values for estimates of the second partial derivatives. This means that the best we can do is to assume that the first and second derivatives of any possible underlying image function have known bounds. Therefore any estimated first- or second-order partials that exceed these known bounds must be due to discontinuities in value of the underlying function. The location of the estimated discontinuity in derivative will be where the second derivative has a relative extremum and this will be where the third derivative has an appropriately shaped zero-crossing.

Recent Developments

Marr and Hildreth (1980) used for the second derivative the isotropic Laplacian. Haralick (1984) and Canny (1986) used, for the second derivative, the second directional derivative taken in a direction that extremizes the first directional derivative. The implementation of each of these zero-crossing edge operators is quite different.

Since the differentiation of a sampled signal is, properly speaking, an ill-posed problem, it has been proposed that edge detection be performed by first filtering the image (or "regularizing" it) and then differentiating it. A mathematical problem is *well-posed* in the sense of Hadamard, provided its solution exists, is unique, and depends continuously on the given data. Regularization refers to the transformation of an ill-posed problem into a well-posed one. Standard methods of regularization have been developed—see, for example, Tikhonov and Arsenin (1977)—and applied in edge detection. Details may be found in Torre and Poggio (1986). A good overview of edge detection, including a discussion of regularization, may be found in Hildreth (1987).

B3. Stereo

Overview

The objective in many computer vision problems is to reconstruct a three-dimensional surface representation of a scene from the image information output by cameras. Video cameras provide only 2-D images, and stereo methods must be used to obtain depth information. The use of two (or more) images of the same scene, taken from different positions, can permit the determination of depth using *parallax*—the analysis of each triangle formed by some notable surface point in the scene and the two camera viewpoints. With two such images, the method of depth determination is called *binocular stereo.* With three, it is *trinocular stereo.* With more, it is sometimes called *multiple-image stereo.* For an introduction to binocular stereo, see Article XIII.D3, or see Barnard and Fischler (1987). When the scene is static but a sequence of images is taken from a moving viewpoint, *motion stereo* may be used to establish 3-D information.

The usual sequence of steps needed in binocular stereo is as follows:

1. Input images either from two cameras or from one camera at two different times and positions.
2. Determine camera parameters—position, orientation, focal length, and so on.
3. Detect/select feature points in the images that are candidates for matching (e.g., edge points).
4. Match feature points by constructing a correspondence between feature points of the two images.
5. Compute depth values at the locations of the matched feature points.
6. Interpolate depth values at all or many of the points in the image that are not locations of matched feature points.

Feature Point Detection/Selection

With a simple camera geometry we may assume that the two images of a point in the scene have a positional disparity along the x-axis of the image but not along the y-axis. To determine this disparity, using feature-based or edge-based stereo, the points must be detected in each image and then put into correspondence. Generally speaking, only certain points in the image are capable of being matched directly; these are prominent locations in the image that are easily distinguished from neighboring points. In most cases the feature points can be obtained using edge-detection methods.

A popular method for finding feature points for stereo matching requires that the Laplacian operator be applied to the image (see Volume 3, p. 211–212). Then the zero-crossing contours of the resulting image are identified. The points on the zero-crossing contours are taken as the feature points. Since the digital images have a limited number of scan lines, the number of zero-crossing points is generally manageable.

Because the disparities occur in the x direction, it is usually sufficient to perform the differentiation (or apply the Laplacian) in one dimension, along each scan line of the image. This is computationally inexpensive in comparison with two-dimensional Laplacians.

If general camera geometries are used, the feature points must be distinguishable in both the x and y directions. Although the detection of these points is therefore more computationally expensive, the resulting number of points is usually less than for one-dimensional analysis, and this can speed up the matching process. Scene points that generate good feature points with distinction in both dimensions are corners (vertices) of polyhedra and bright spots and corners of 2-D patterns painted on the surfaces of objects in the scene.

It is also possible to match areas rather than features. In area-based matching, correspondences are typically established using cross-correlation. This tends to be computationally more expensive and also less accurate than feature-based or edge-based matching. However, area-based stereo can be more robust in cases of noisy images or images with poorly defined edges.

Matching. Although matching for stereo is similar in spirit to model matching for object recognition, it is also somewhat different. In the case of horizontally constrained displacement, we have a collection of one-dimensional matching problems, one for each scan line. We can expect the disparity function along the scan line to exhibit some coherence as we move to each successive scan line, as well as along the line. Therefore the solutions to each 1-D matching problem are not completely independent.

Some of the approaches to matching are as follows:

1. Coarse-to-fine (see Marr and Poggio, 1977; and Grimson, 1985)
2. Dynamic programming (see Baker and Binford, 1981)
3. Energy minimization (see Direct Matching with Simulated Annealing, described below)
4. Ad hoc correspondence building

Interpolating Depth Values. The problem of obtaining a full set of depth values from the sparse set obtained from feature-based stereo can be solved with interpolation. However, this interpolation should satisfy both smoothness on surfaces and maintain sudden depth changes at surface boundaries. In the case of natural terrain, quadratic surface fitting may be appropriate (see Smith, 1984). For rapid interpolation subject to smoothness constraints, multigrid methods may be used (see Section D).

Direct Matching with Simulated Annealing. A method of matching a stereo pair of images using simulated annealing has been proposed by Barnard (1987). This is an area-based rather than a feature-based approach. An energy measure E is to be minimized through the adjustment of disparity values $D_{i,j}$:

$$I = \sum_{i,j} (|\Delta I_{ij}| + \lambda|\nabla D_{ij}|)$$

where $\Delta I_{ij} = \mathrm{I}_L(i,j) - \mathrm{I}_R(i,j + D_{ij})$; I_L and I_R are the left- and right-image intensity values; and D_{ij} is the disparity value for location (i,j). This measures the difference in intensity between each two matched points as well as the unsmoothness of the disparity function. If both of these terms are zero, the two images match perfectly, except for a translation, and the scene must be flat.

Starting from an initial high-energy state, the disparity values are adjusted stochastically according to the Metropolis algorithm (see page 576) or with an alternative method proposed by Barnard.

Nonbinocular Methods. *Trinocular stereo* employs three images of a scene to obtain 3-D surface data. The third image, taken from a viewpoint not colinear with the other two, greatly reduces the number of incorrect matches and it can increase the accuracy of the resulting depth information. A method that permits the three cameras to be in arbitrary positions is described by Ayache and Lustman (1987). One that requires the viewpoints to form a right triangle is given by Ohta et al. (1986). Others are given by Yachida et al. (1986), Ito and Ishii (1986), and Pietikainen and Harwood (1986). The number of viewpoints need not be limited to three. Multiple-image stereo allows additional improvements in accuracy at the expense of higher computational cost (see Yachida, 1985).

In addition to binocular, trinocular, and multiple-image stereo, surface orientation may be computed using two images from the same viewpoint, but taken under illumination by a light source in two different positions. This method is called *photometric stereo* and is described briefly in the Overview to Chapter XIII in Volume III of the *Handbook*. The change in shading at a surface point from one image to the other gives an indication of the surface gradient at that point. Such methods are described in Woodham (1980).

B4. Mathematical Morphology for Image Analysis

A CLASS OF TECHNIQUES called *mathematical morphology* has found a variety of applications in industrial machine vision. This section presents the primary operations of mathematical morphology: dilation, erosion, opening, and closing. In addition to their definitions, some properties of these operations are also given.

The mathematical morphology approach to the processing of digital images is based on shape. Appropriately used, these techniques can simplify image data, preserving essential shape characteristics and eliminating irrelevancies. Since the identification of objects, features, and manufacturing defects depend closely on shape, this approach is natural for such tasks.

Although the techniques are being used in the industrial world, the basis and theory of binary morphology are not covered in many texts or monographs. Exceptions are the highly mathematical books by Matheron (1975) and Serra (1982).

The language of mathematical morphology is that of set theory. Sets in mathematical morphology represent the shapes that are manifested on binary or gray-tone images. The set of all the black pixels in a black and white image (a binary image) constitutes a complete description of the binary image. Sets in two-dimensional Euclidean space are represented by foreground regions in binary images. Sets in three-dimensional Euclidean space may actually represent time-varying binary imagery or static gray-scale imagery as well as binary solids. Sets in higher dimensional spaces may incorporate additional image information such as color, or multiple perspective imagery. Mathematical morphology transformations apply to sets of any dimensions, including those in Euclidean N-space and its discrete or digitized equivalents, the set of N-tuples of integers, Z^N. For the sake of simplicity we will refer to either of these sets as E^N.

Those points in a set being morphologically transformed are considered as the *selected set* of points, and those in the complement set are considered as not selected. Hence, morphology from this point of view is *binary* morphology. We begin our discussion with the morphological operation of dilation.

Dilation

Dilation is a morphological transformation that combines two sets using vector addition of set elements. If A and B are sets in N-space (E^N) with elements a and b, respectively, $a = (a_1, ..., a_N)$ and $b = (b_1, ..., b_N)$ being N-tuples of element coordinates, then the dilation of A by B is the set of all possible vector sums of pairs of elements, one coming from A and one coming from B. Denoting dilation by $\oplus$,

$$A \oplus B = \{c \in E^N \mid c = a + b \quad \text{for some } a \in A \text{ and } b \in B\}$$

Dilation as a set theoretic operation was proposed by Minkowski (1903) to characterize integral measures of certain open (sparse) sets. Dilation as an image-processing operation was employed by several early investigators in image processing as smoothing operations: Unger (1958), Golay (1969), and Preston (1961, 1973). Dilation as an image operator for shape extraction and estimation of image parameters was explored by Matheron (1975) and Serra (1972).

Mathematically the roles of the sets A and B are symmetric; the dilation operation is commutative because addition is commutative. Hence $A \oplus B = B \oplus A$. In practice, A and B are handled quite differently. The first operand is considered to be the image undergoing analysis, whereas the second operand, referred to as the *structuring element*, is thought of as constituting a single shape parameter of the dilation transformation.

Dilation of a set by a structuring element in the shape of a disk results in an isotropic swelling or expansion of the set. (Approximating the disk by a small square, 3×3, the expansion can be implemented as a neighborhood operation on a mesh architecture or pipelined image-processing architecture.) Some sample dilation transformations are illustrated in Figures B–8 and B–9. In Figure B–8, the upper left is the input image consisting of a cross. The lower right shows an octagonal structuring element. The upper right shows the input image dilated by the octagonal structuring element. In Figure B–9, the upper left contains the input image consisting of two objects. The upper right shows the input image dilated by the structuring element $\{(0, 0), (14, 0)\}$. The lower left shows the input image dilated by the structuring element $\{(0, 0), (0, 14)\}$. The lower right shows the input image dilated by the structuring element $\{(0, 0), (14, 0), (0, 14)\}$. This example illustrates that dilation can be viewed as the replication of a pattern. In actual use, the replicated copies of the pattern usually overlap, as in Figure B–8.

Since addition is associative, the dilation of an image A by a structuring element D, which is itself a dilation $D = B \oplus C$, can be computed as

$$A \oplus D = A \oplus (B \oplus C) = (A \oplus B) \oplus C$$

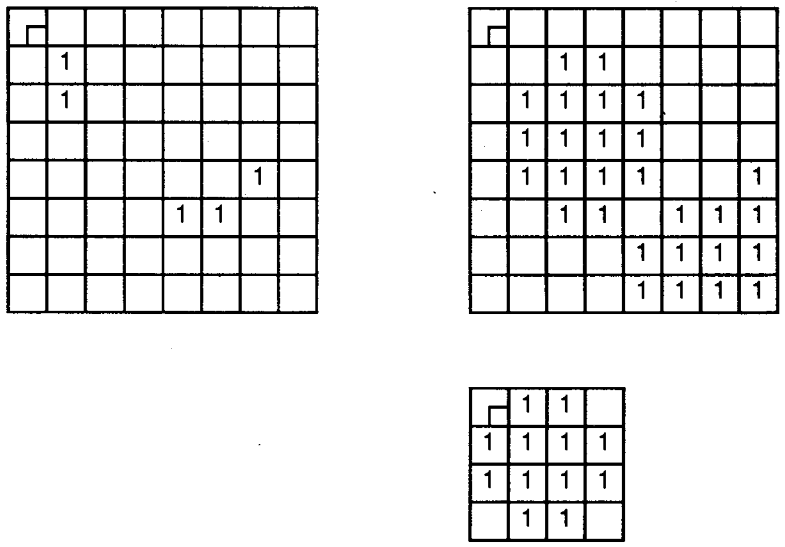

Figure B–8. Dilation by an octagonal structuring element.

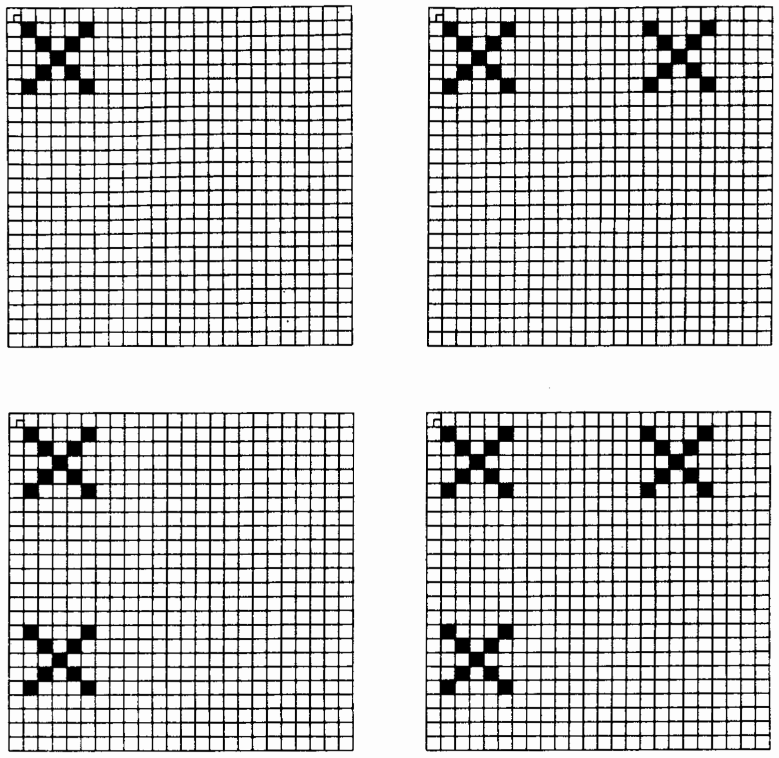

Figure B–9. Dilation with an additional structuring element.

That is, dilation is associative. The form $(A \oplus B) \oplus C$ gives a considerable savings in number of operations to be performed when A is the image and $B \oplus C$ is the structuring element. The savings come about because a brute force dilation by $B \oplus C$ might take as many as N^2 operations, whereas first dilating A by B and then dilating the result by C could take as few as $2N$ operations, where N is the number of elements in B and in C.

The dilation of A by B can be computed as the union of translations of A by the elements of B. That is,

$$A \oplus \mathrm{B} = \bigcup_{b \in B} (A)_b$$

Erosion

Erosion is the morphological dual to dilation. It is normally used to eliminate small protrusions on a shape or islands in an image. It can widen cracks and holes. Erosion combines two sets using vector subtraction of set elements. If A and B are sets in Euclidean N-space, the *erosion* of A by B is the set of all elements x for which $x + b \in A$ for every $b \in B$.

Let us denote the erosion of A by B as $A \ominus B$. Erosion is thus defined by

$$A \ominus B = \{x \in E^N \mid x + b \in A \quad \text{for every } b \in B\}$$

The utility of the erosion transformation is better appreciated when the erosion is expressed in a different form (that given by Matheron, 1975). The erosion of an image A by a structuring element B is the set of all elements x of E^N for which B translated to x is contained in A.

$$A \ominus B = \{x \in E^N \mid (B)_x \subseteq A\}$$

Erosion is illustrated in Figure B–10. The upper left shows the input image consisting of two blobs. The upper right shows the input image eroded by the structuring element

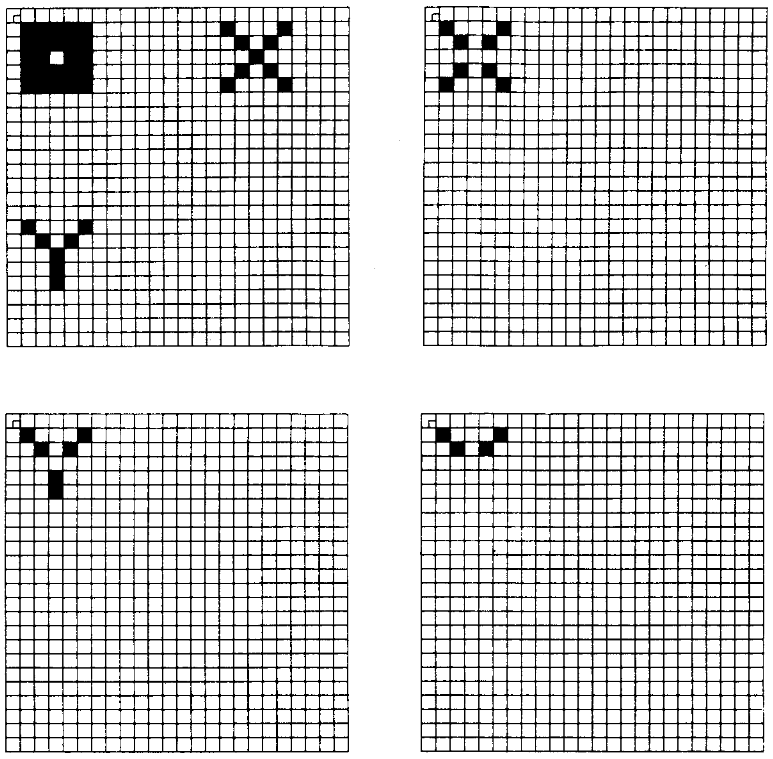

Figure B–10. Erosion of an image of two blobs.

$$\{(0, 0), (-14, 0)\}$$

The lower left shows the input image eroded by the structuring element

$$\{(0, 0), (0, -14)\}$$

The lower right shows the input image eroded by the structuring element

$$\{(0, 0), (0, -14), (-14, 0)\}$$

Openings and Closings

In practice, dilations and erosions are usually employed in pairs, either dilation of an image followed by the erosion of the dilated result or image erosion followed by dilation. In either case, the result of iteratively applied dilations and erosions is an elimination of specific image detail smaller than the structuring element without the global geometric distortion of unsuppressed features. The *opening* of image B by structuring element K is denoted by $B \circ K$ and is defined as $B \circ K = (B \ominus K) \oplus K$. The *closing* of image B by structuring element K is denoted by $B \bullet K$ and is defined by $B \bullet K = (B \oplus K) \ominus K$.

For example, opening an image with a disk-shaped structuring element smooths the contour, breaks narrow isthmuses, and eliminates small islands and sharp peaks or capes. Closing an image with a disk-structuring element smooths the contours, fuses narrow breaks and long thin gulfs, eliminates small holes, and fills gaps on the contours.

Of particular significance is the fact that image transformations employing iteratively applied dilations and erosions are idempotent, that is, their reapplication effects no further changes to the previously transformed result. The practical importance of idempotent transformations is that they comprise complete and closed stages of image analysis algorithms because shapes can be naturally described in terms of under what structuring elements they can be opened or can be closed and yet remain the same.

If B is unchanged by opening it with K, we say that B is open with respect to K, whereas if B is unchanged by closing it with K, then B is closed with respect to K.

Sets that can be expressed as some set dilated by K are necessarily open under K.

$$A \oplus K = (A \oplus K) \circ K$$

Similarly, images that have been eroded by K are necessarily closed under K.

$$A \ominus K = (A \ominus K) \bullet K$$

From these two facts, the idempotency of opening and closing follows. Openings and closings have other properties. For example, it follows immediately from the increasing property of dilation and the increasing property of erosion that both opening and closing are increasing.

There is a nice geometric characterization to the opening operation. This characterization justifies why mathematical morphology provides material for extracting shape information from image data. The opening of A by B is the union of all translations of B that are contained in A.

Discussion

Dilation, erosion, opening, and closing can be used as the basis of image algebras. These algebras allow the definition of shape transformations that are customized for particular applications. A sequence of these operations, with suitable structuring elements, can be used to identify gear teeth in images of gears, or holes of particular sizes in images of machine parts. These techniques have been successfully applied to the problem of visually detecting shorts and open circuits in the wiring of printed circuit boards. This is illustrated schematically in Figure B–11.

Opening removes small protrusions, isthmuses and islands. Closing removes small cracks, bays, and holes. Taking the exclusive-OR of the resulting image with the original gives an image in which only potential defects remain. The original binary image is shown in the upper left. The result after erosion is in the upper center. After dilating that image, the result in the upper right is obtained. A second step of dilation takes us to the result in the lower left, and then another erosion takes us to the lower center. Exclusive-ORing this with the original produces the image of the isolated defects, shown in the lower right.

These operations can be efficiently computed with appropriate hardware. An entire session of the 1985 IEEE Computer Society Workshop on Computer Architecture for Pattern Analysis and Image Database Management was devoted to computer architecture specialized to perform morphological operations. Papers included those by McCubbrey and Lougheed (1985), Wilson (1985), Kimmel, Jaffe, Manderville, and Lavin (1985), Leonard (1985), Pratt (1985), and Haralick (1985). Gerritsen and Verbeek (1984) show how convolution followed by a table lookup operation can accomplish binary morphology operations.

Mathematical morphology is being extended to encompass more and more general classes of operators. Gray-scale extensions have been studied. Efforts have been made to cast morphology operations into a digital signal processing framework. A tutorial article presenting many more of the details of mathematical morphology is the paper by Haralick, Sternberg, and Zhuang (1987).

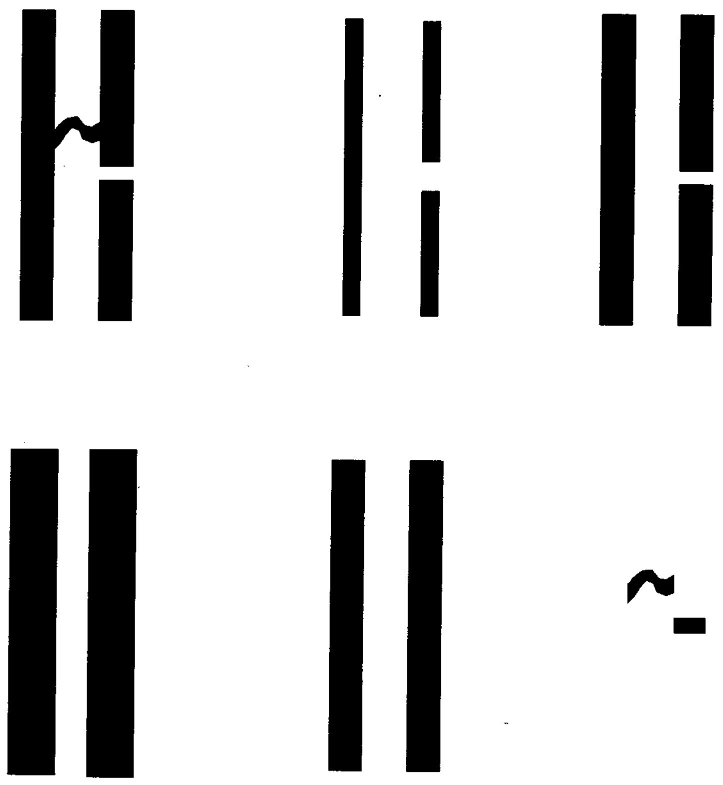
Figure B–11. Application of opening and closing to PC board inspection.