Disclaimer: All the material in this handout has been adopted from the text book "Digital Image Processing" by Gonzalez and Woods.

Unit III: Image Restoration and Morphological Processing.

Morphological Processing: Erosion, Dilation, Opening, Closing, Hit-or-Miss Transform, Boundary Detection, Hole filling, Connected components, thinning, thickening, skeletons, pruning.

Color Image Processing: Color Fundamentals, Color Models: RGB, CMY and CMYK, HIS, Conversion from RGB to HSI and vice versa.

Opening and Closing

As we have seen, dilation expands an image and erosion shrinks it. Opening generally smoothens the contour of an object, breaks narrow isthmuses, and eliminates thin protrusions.

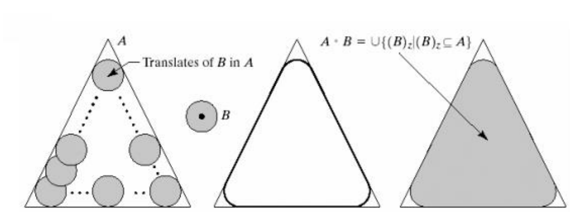
Closing also tends to smooth sections of contours but, as opposed to opening, it generally fuses narrow breaks and long thin gulfs, eliminates small holes, and fills gaps in the contour.

The opening of set A by structuring element B, denoted

$$A \circ B = (A \Theta B) \oplus B$$

The closing of set A by structuring element B, denoted

$$A \bullet B = (A \oplus B) \Theta B$$



a b c d

FIGURE 9.8 (a) Structuring element B "rolling" along the inner boundary of A (the dot indicates the origin of B). (c) The heavy line is the outer boundary of the opening. (d) Complete opening (shaded).

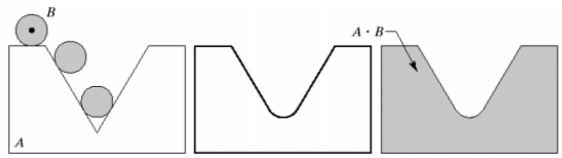
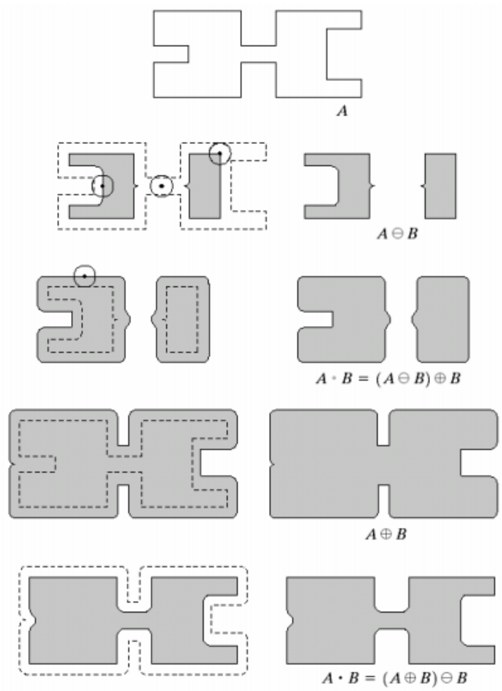


FIGURE 9.9 (a) Structuring element B "rolling" on the outer boundary of set A. (b) Heavy line is the outer boundary of the closing. (c) Complete closing (shaded).



a
b c
d e
f g
h i

FIGURE 9.10 Morphological opening and closing. The structuring element is the small circle shown in various positions in (b). The dark dot is the center of the structuring element.

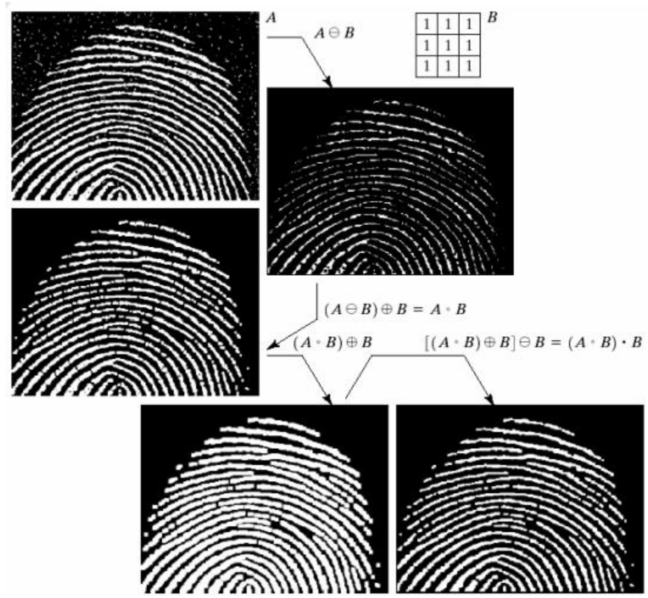


FIGURE 9.11
(a) Noisy image.
(c) Eroded image.
(d) Opening of *A*.
(d) Dilation of the opening.
(e) Closing of the opening. (Original image for this example courtesy of the National Institute of Standards and Technology.)

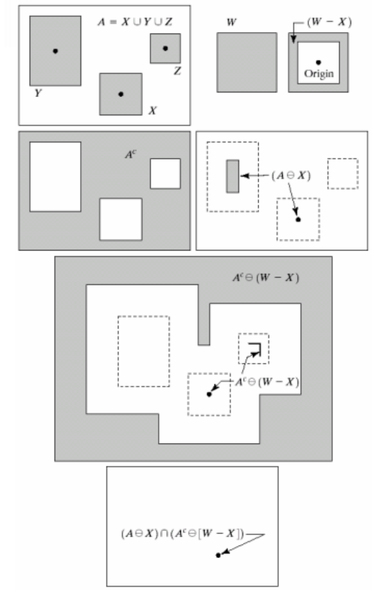
In figure 9.11 the background noise was completely eliminated in the erosion stage of opening because in the case all noise components are physically smaller than the structuring element. We note in Fig 9.11(d) that the net effect of opening was to eliminate virtually all noise components in both the background and the fingerprint itself. However, new gaps between the fingerprint ridges were created. To counter this undesirable effect, we perform a dilation on the opening.

The Hit-or-Miss Transformation

The morphological hit-or-miss transform is a basic tool for shape detection. The objective is to find the location of one of the shapes, say, X. $A \Theta X$ may be viewed geometrically as the set of all locations of the origin of X at which X found a match (hit) in A. B denotes the set composed of X and its background, the match (or set of matches) of B in A, denoted $A \circledast B$.

$$A \circledast B = (A \Theta X) \cap \left[A^c \Theta \left(W - X \right) \right]$$

$$A \circledast B = (A \Theta B_1) - \left(A \oplus \hat{B}_2 \right) \qquad ($$



a b
c d
e
f

FIGURE 9.12 (a) Set A. (b) A window, W, and the local background of X with respect to W, $(W - X)$. (c) Complement of A. (d) Erosion of A by X. (e) Erosion of A^c by $(W - X)$. (f) Intersection of (d) and (e), showing the location of the origin of X, as desired.

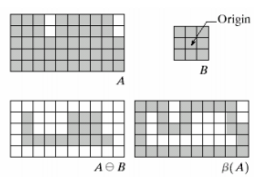
The reason for using a structuring element B1 associated with objects and an element B2 associated with the background is based on assumed definition that two or more objects are distinct only if they form disjoint (disconnected) sets.

Boundary Extraction

$$\beta(A) = A - (A \Theta B)$$

a b
c d

FIGURE 9.13 (a) Set A. (b) Structuring element B. (c) A eroded by B. (d) Boundary, given by the set difference between A and its erosion.

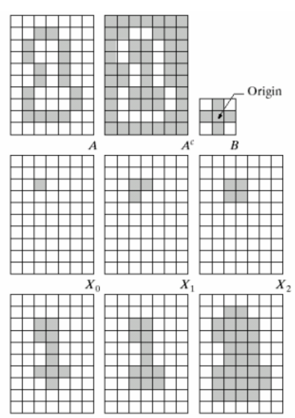


Region/Hole Filling

$$X_K = (X_{K-1} \oplus B) \cap A^c \quad k = 1,2,3....$$

a b c
d e f
g h i

FIGURE 9.15 Region filling. (a) Set A. (b) Complement of A. (c) Structuring element B. (d) Initial point inside the boundary. (e)–(h) Various steps of Eq. (9.5-2). (i) Final result [union of (a) and (h)].



Extraction of Connected Components

$$X_k = (X_{k-1} \oplus B) \cap A \quad k = 1, 2, 3$$

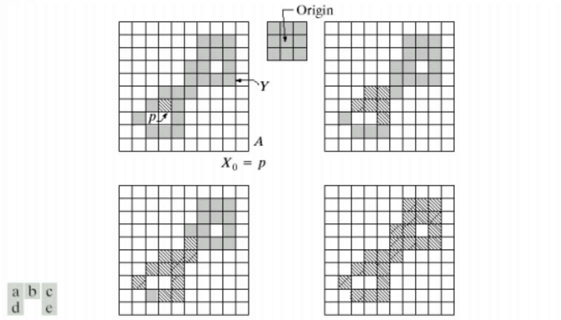


FIGURE 9.17 (a) Set A showing initial point p (all shaded points are valued 1, but are shown different from p to indicate that they have not yet been found by the algorithm). (b) Structuring element. (c) Result of first iterative step. (d) Result of second step. (e) Final result.

Thinning

$$A \otimes B = A - (A \circledast B)$$

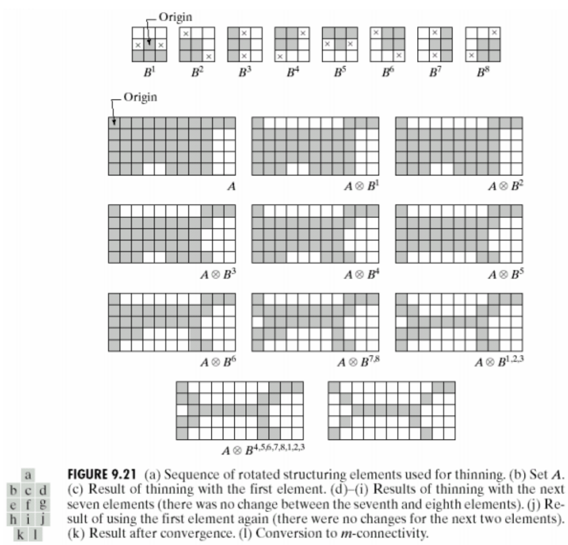


FIGURE 9.21 (a) Sequence of rotated structuring elements used for thinning. (b) Set A. (c) Result of thinning with the first element. (d)–(i) Results of thinning with the next seven elements (there was no change between the seventh and eighth elements). (j) Result of using the first element again (there were no changes for the next two elements). (k) Result after convergence. (l) Conversion to m-connectivity.

Thickening

$$A \odot B = A \cup (A \circledast B)$$

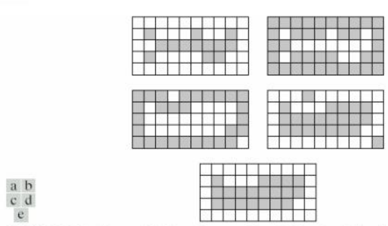


FIGURE 9.22 (a) Set A. (b) Complement of A. (c) Result of thinning the complement of A. (d) Thickened set obtained by complementing (c). (e) Final result, with no disconnected points.

Image Skeletons

$$S(A)=\bigcup_{k=0}^{k} S_k(A)$$
$$S_k(A)=(A\Theta kB)-(A\Theta kB)\circ B$$
$$(A\Theta kB)=(.....(A\Theta B)\Theta B)\Theta.....)\Theta B$$
$$K=\max\left\{k \,\middle|\, (A\Theta kB)\neq\varnothing\right\}$$

Image A can be reconstructed from these subsets by using the equation

$$A=\bigcup_{k=0}^{k}(S_k(A)\oplus kB)$$

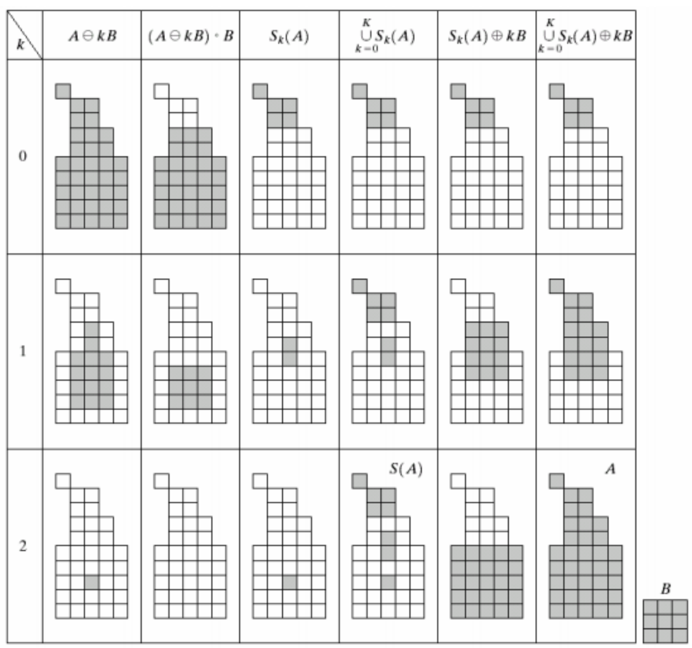
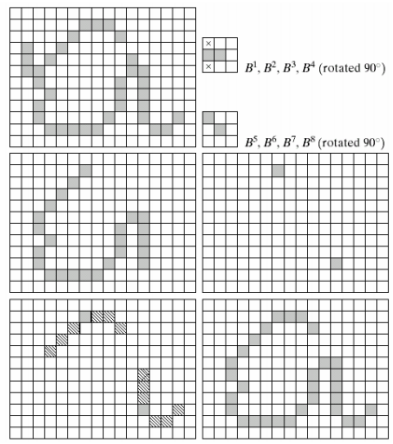


FIGURE 9.24 Implementation of Eqs. (9.5-11) through (9.5-15). The original set is at the top left, and its morphological skeleton is at the bottom of the fourth column. The reconstructed set is at the bottom of the sixth column.

Pruning

Pruning methods are an essential complement to thinning and skeletonizing algorithms because these procedures tend to leave parasitic components that need to be “cleaned up”.



a b
c
d e
f g

FIGURE 9.25 (a) Original image. (b) and (c) Structuring elements used for deleting end points. (d) Result of three cycles of thinning. (e) End points of (d). (f) Dilation of end points conditioned on (a). (g) Pruned image.

TABLE 9.2 Summary of morphological operations and their properties.

Operation	Equation	Comments (The Roman numerals refer to the structuring elements shown in Fig. 9.26).
Translation	$(A)_z = \{w \mid w = a + z, \quad \text{for } a \in A\}$	Translates the origin of A to point z.
Reflection	$\hat{B} = \{w \mid w = -b, \quad \text{for } b \in B\}$	Reflects all elements of B about the origin of this set.
Complement	$A^c = \{w \mid w \notin A\}$	Set of points not in A.
Difference	$A - B = \{w \mid w \in A, w \notin B\} = A \cap B^c$	Set of points that belong to A but not to B.
Dilation	$A \oplus B = \{z \mid (\hat{B})_z \cap A \neq \varnothing\}$	"Expands" the boundary of A. (I)
Erosion	$A \ominus B = \{z \mid (B)_z \subseteq A\}$	"Contracts" the boundary of A. (I)
Opening	$A \circ B = (A \ominus B) \oplus B$	Smoothes contours, breaks narrow isthmuses, and eliminates small islands and sharp peaks. (I)
Closing	$A \bullet B = (A \oplus B) \ominus B$	Smoothes contours, fuses narrow breaks and long thin gulfs, and eliminates small holes. (I)
Hit-or-miss transform	$A \circledast B = (A \ominus B_1) \cap (A^c \ominus B_2) = (A \ominus B_1) - (A \oplus \hat{B}_2)$	The set of points (coordinates) at which, simultaneously, B_1 found a match ("hit") in A and B_2 found a match in A^c.
Boundary extraction	$\beta(A) = A - (A \ominus B)$	Set of points on the boundary of set A. (I)
Region filling	$X_k = (X_{k-1} \oplus B) \cap A^c$; $X_0 = p$ and $k = 1, 2, 3, \ldots$	Fills a region in A, given a point p in the region. (II)
Connected components	$X_k = (X_{k-1} \oplus B) \cap A$; $X_0 = p$ and $k = 1, 2, 3, \ldots$	Finds a connected component Y in A, given a point p in Y. (I)
Convex hull	$X_k^i = (X_{k-1}^i \circledast B^i) \cup A$; $i = 1, 2, 3, 4$; $k = 1, 2, 3, \ldots$; $X_0^i = A$; and $D^i = X_{\text{conv}}^i$.	Finds the convex hull $C(A)$ of set A, where "conv" indicates convergence in the sense that $X_k^i = X_{k-1}^i$. (III)

TABLE 9.2 Summary of morphological results and their properties. *(continued)*

Unit IV: Edge Detection and Segmentation.

Edge detection: Basic Formulation: Detecting Points and Lines, Edge Models; Gradient and its Properties; Gradient Operators: Roberts, Prewitt, Sobel; Canny Edge Detector; Thresholding: Basic Global Thresholding, Basic Adaptive Thresholding.

Region based segmentation: Basic Formulation, Region growing, Region splitting and Merging; Segmentation by morphological watersheds: Basic concepts, Dam construction, Watershed Algorithm.

Edge Detection Foundation

Segmentation subdivides an image into its constituent regions or objects. The level to which the subdivision is carried depends on the problem being solved. Image segmentation algorithms generally are based on one of two basic properties of intensity values: discontinuity and similarity.

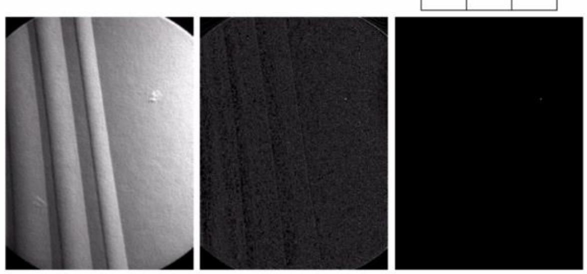
In the intensity, such as approach in an image is to partition an image based on abrupt changes in intensity, such as edges in an image. The principal approaches in the second category are based on partitioning an image into regions that are similar according to a set of pre-defined criteria. Thresholding, region growing, and region splitting and merging are examples of methods in this category.

Point Detection

Using the mask shown in Fig 10.2(a), we say that a point has been detected at the location on which the mask is centered if |R| ≥ T. The idea is that an isolated point will be quite different from its surroundings, and thus be easily detectable by this type of mask. The most common way to look for discontinuities is to convolve a mask on the image.

$$R = w_1 z_1 + w_2 z_2 + \cdots + w_9 z_9 = \sum_{i=1}^{9} w_i z_i$$

−1	−1	−1
−1	8	−1
−1	−1	−1



a
b c d

FIGURE 10.2 (a) Point detection mask. (b) X-ray image of a turbine blade with a porosity. (c) Result of point detection. (d) Result of using Eq. (10.1-2). (Original image courtesy of X-TEK Systems Ltd.)

Line Detection

Consider the masks shown in Fig 10.3. If the first mask were moved around an image, it would respond more strongly to lines (one pixel thick) oriented horizontally.

FIGURE 10.3 Line masks.

Horizontal

−1	−1	−1
2	2	2
−1	−1	−1

+45°

−1	−1	2
−1	2	−1
2	−1	−1

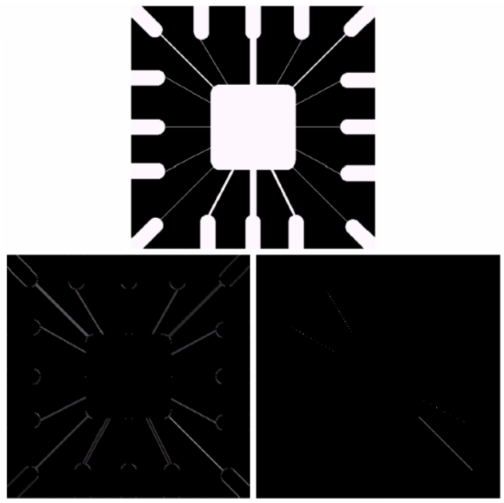
Vertical

−1	2	−1
−1	2	−1
−1	2	−1

−45°

2	−1	−1
−1	2	−1
−1	−1	2

Fig 10.4(a) shows a digitized (binary) portion of a wire-bond mask for an electronic circuit. Suppose that we are interested in finding all the lines that are one pixel thick and are oriented at -45°. For the purpose, we use last mask shown in Fig. 10.3.

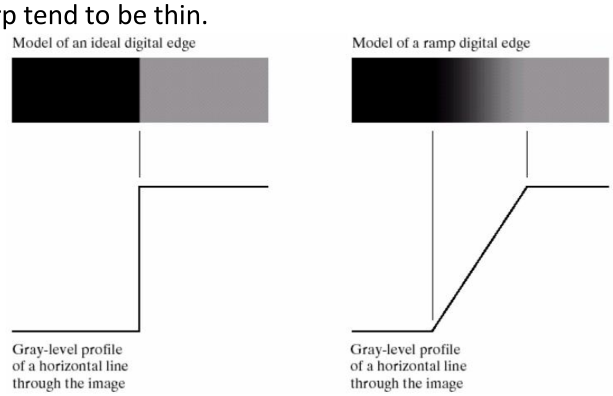


a
b c
FIGURE 10.4 Illustration of line detection. (a) Binary wire-bond mask. (b) Absolute value of result after processing with −45° line detector. (c) Result of thresholding image (b).

In order to determine which line best fit the mask, we simply threshold this image. The result of using a threshold equal to the maximum value in the image is shown in Fig 10.4(c).

Edge Detection

Edge detection is by far the most common approach for detecting meaning discontinuities in grey level. Intuitively, an edge is a set of connected pixels that lie on the boundary between two regions. A reasonable definition of "edge" requires the ability to measure grey-level transitions in a meaningful way. Intuitively, an ideal edge has the properties has the properties of the model shown in Fig 10.5(a). Edges are more closely modeled as having a "ramplike" profiles, such as the one shown in Fig 10.5(b). Blurred edges tend to be thick and sharp tend to be thin.



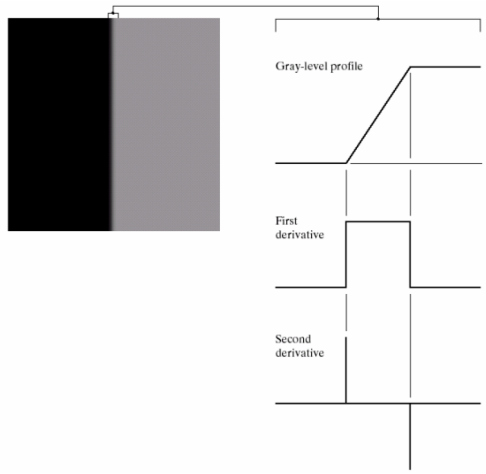
a b
FIGURE 10.5 (a) Model of an ideal digital edge. (b) Model of a ramp edge. The slope of the ramp is proportional to the degree of blurring in the edge.

We conclude from these observations that the magnitude of the first derivative can be used to detect the presence of an edge at a point in an image. Similarly, the sign of the second derivation can be used to determine whether an edge pixel lies on the dark or light side of an edge. Two additional properties of the second derivative around an edge:

– (1) It produces two value for every edge in an image

– (2) an imaging straight line joining the extreme positive and negative values of the second derivative would cross zero near the midpoint of the edge.

The zero-crossing property of the second derivative is quite useful for locating the centres of thick edges.



a b
FIGURE 10.6 (a) Two regions separated by a vertical edge. (b) Detail near the edge, showing a gray-level profile, and the first and second derivatives of the profile.

Gradient operators

The gradient of an image f(x,y) at location (x,y) is defined as the vector $\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$

$\nabla f = mag(\nabla f) = \left[G_x^2 + G_y^2\right]^{1/2}$

Let α(x,y) represent the direction angle of the vector ∇f at (x,y). Then, from vector analysis, $\alpha(x,y) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$ where the angle is measured with respect to the x-axis. The direction of an edge at (x,y) is perpendicular to the direction of the gradient vector at that point. One of the simplest ways to implement a first-order partial derivative at point z_5 is to use the following **Roberts cross-gradient operators**:

$$G_x = (z_9 - z_5)$$
$$G_y = (z_8 - z_6)$$

An approach using masks of size 3x3 is given by

$$G_x = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$$
$$G_y = (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)$$

called the *Prewitt operators*

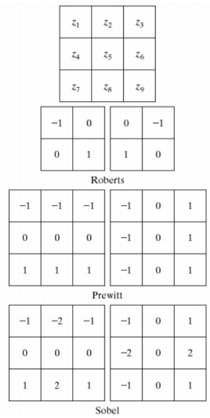
A slight variation of these two equations uses a weight of 2 in the centre coefficient:

$$G_x = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$
$$G_y = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

A weight value of 2 is used to achieve some smoothing by giving more importance to the centre point, called the Sobel operators. The Prewitt and Sobel operators are among the most used in practice for computing digital gradients. Prewitt masks are simpler to implement than the Sobel masks, but the later have slightly superior noise-suppression characteristics, an important issue when dealing with derivatives.

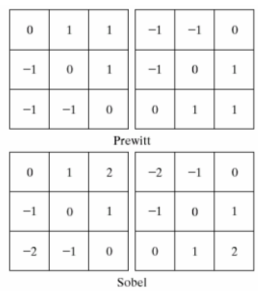
a
b c
d e
f g

FIGURE 10.8 A 3 × 3 region of an image (the z's are gray-level values) and various masks used to compute the gradient at point labeled z_5.



An approach used frequently is to approximate the gradient by absolute values: $\nabla f \approx |G_x| + |G_y|$

The two additional Prewitt and Sobel masks for detecting discontinuities in the diagonal directions are shown in Fig. 10.9.

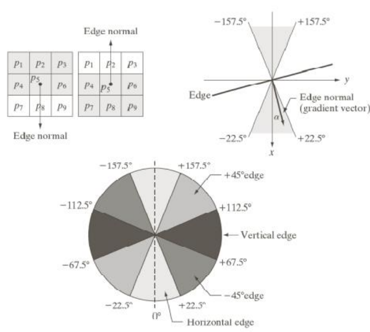


a b
c d

FIGURE 10.9 Prewitt and Sobel masks for detecting diagonal edges.

Canny Edge Detector

1. Low error rate
2. Edge point should be well localized
3. Single edge point response



Algorithm

1. Smooth image with a Gaussian - optimizes the trade-off between noise filtering and edge localization
2. Compute the Gradient magnitude using approximations of partial derivatives

- At each point convolve with

$$G_x = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \quad G_y = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}$$

- magnitude and orientation of the Gradient are computed as

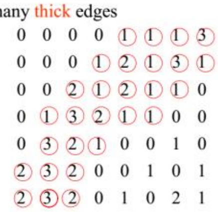
$$M[i,j] = \sqrt{P[i,j]^2 + Q[i,j]^2}$$

$$\theta[i,j] = \tan^{-1}(Q[i,j], P[i,j])$$

- Avoid floating point arithmetic for fast computation

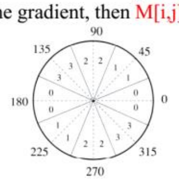
3. Thin edges by applying non-maxima suppression to the gradient magnitude

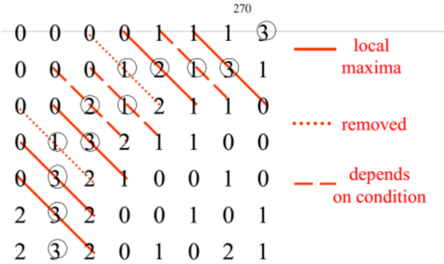
- Thin edges by keeping large values of Gradient
  - not always at the location of an edge
  - there are many thick edges

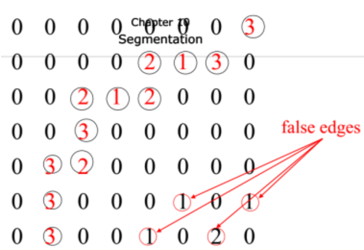


- Thin the broad ridges in M[i,j] into ridges that are only one pixel wide
- Find local maxima in M[i,j] by suppressing all values along the line of the Gradient that are not peak values of the ridge

- Reduce angle of Gradient θ[i,j] to one of the 4 sectors
- Check the 3x3 region of each M[i,j]
- If the value at the center is not greater than the 2 values along the gradient, then M[i,j] is set to 0





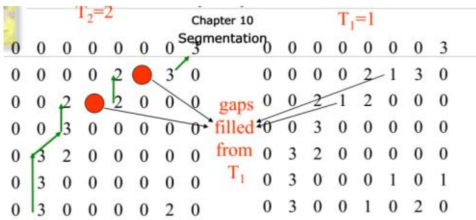


The suppressed magnitude image will contain many false edges caused by noise or fine texture

4. Detect edges by double thresholding

- Reduce number of false edges by applying a threshold T
  - all values below T are changed to 0
  - selecting a good values for T is difficult
  - some false edges will remain if T is too low
  - some edges will disappear if T is too high
  - some edges will disappear due to softening of the edge contrast by shadows

- Apply two thresholds in the suppressed image
  - $T_2 = 2T_2$
  - two images in the output
  - the image from T_2 contains fewer edges but has gaps in the contours
  - the image from T_1 has many false edges
  - combine the results from T_1 and T_2
  - link the edges of T_2 into contours until we reach a gap
  - link the edge from T_2 with edge pixels from a T_1 contour until a T_2 edge is found again



- A T_2 contour has pixels along the green arrows
- Linking: search in a 3x3 of each pixel and connect the pixel at the center with the one having greater value
- Search in the direction of the edge (direction of Gradient)

Basic Global Thresholding

One obvious way to extract the objects from the background is to select a threshold T that separates these modes. Then any point (x,y) for which f(x,y)>T is called an object point; otherwise, the point is called a background point.

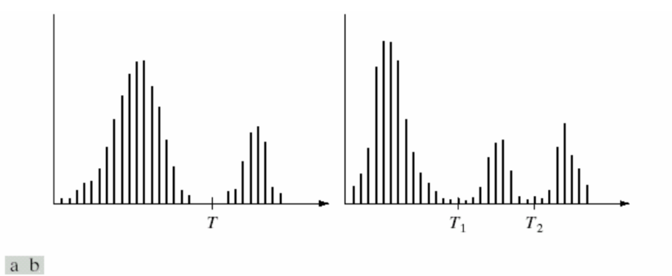


FIGURE 10.26 (a) Gray-level histograms that can be partitioned by (a) a single threshold, and (b) multiple thresholds.

Thresholding may be viewed as an operation that involves tests against a function T of the form where f(x,y) is the gray level of point (x,y) and p(x,y) denotes some local property of this point. When T depends only on f(x,y) (that is, only on graylevel values) the threshold is called global. If T depends on both f(x,y) and p(x,y), the threshold is called local. If, in addition, T depends on the spatial coordinates x and y, the threshold is called dynamic or adaptive.

The following algorithm can be used to obtain T automatically:

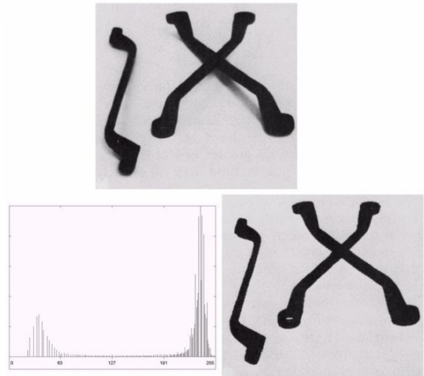
– 1.Select an initial estimate for T.

– 2.Segment the image using T. This will produce two groups of pixels: G1 consisting of all pixels with gray level > T and G2 consisting of pixels with values ≦ T.

– 3.Compute the average gray level value µ1 and µ2 for the pixels in regions G1 and G2.

– 4.Compute a new threshold value: $T = \frac{1}{2}(\mu_1 + \mu_2)$

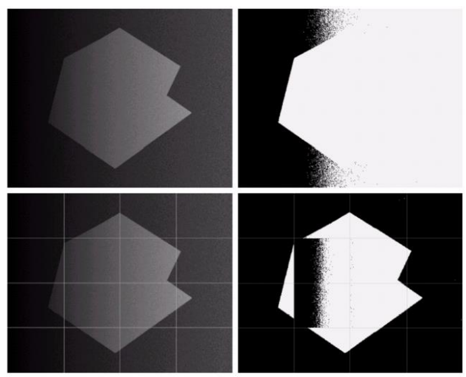
– 5.Repeat step 2 though 4 until the difference in T in successive iterations is smaller than a predefined parameter T_0.



a
b c
FIGURE 10.28 (a) Original image. (b) Image histogram. (c) Result of global thresholding with *T* midway between the maximum and minimum gray levels.

Basic Adaptive Thresholding

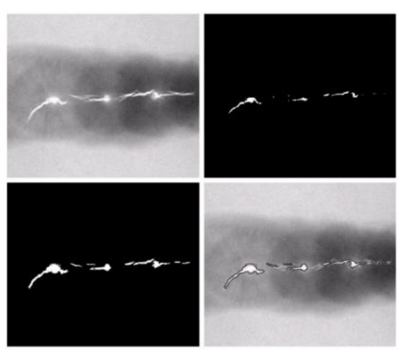
Divide the original image into subimages and then utilize a different threshold to segment each subimage. The key issues in this approach are how to subdivide the image and how to estimate the threshold for each resulting subimage.



a b
c d
FIGURE 10.30 (a) Original image. (b) Result of global thresholding. (c) Image subdivided into individual subimages. (d) Result of adaptive thresholding.

Region Growing

Region growing is a procedure that groups pixels or sub-regions into larger regions based on predefined criteria. The basic approach is to start with a set of seed" points and from these grow regions by appending to each seed those neighbouring pixels that have properties similar to the seed (such as specific ranges of grey level or color). The selection of similarity criteria depends not only on the problem under consideration, but also on the type of image data available. Another problem in region growing is the formulation of a stopping rule.



a b
c d
FIGURE 10.40 (a) Image showing defective welds. (b) Seed points. (c) Result of region growing. (d) Boundaries of segmented defective welds (in black). (Original image courtesy of X-TEK Systems, Ltd.).

Region Splitting and Merging

Let R represent the entire image region and select a predicate P. One approach for segmenting R is to subdivide it successively into smaller and smaller quadrant regions so that, for any region R_i, $P(R_i) = \text{TRUE}$. We start with the entire region. If $P(R) = \text{FALSE}$, we divide the image into quadrants. If P is FALSE for any quadrant, we subdivide that quadrant into subquadrants, and so on. This particular splitting technique has a convenient representation in the form of a so-called *quadtree* (that is, a tree in which nodes have exactly four descendants), as illustrated in Fig. 10.42. Note that the root of the tree corresponds to the entire image and that each node corresponds to a subdivision. In this case, only R_4 was subdivided further.

a b

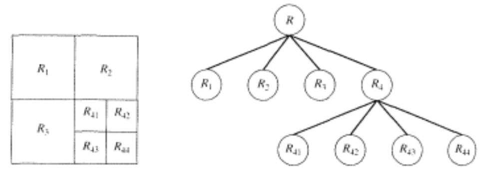


FIGURE 10.42 (a) Partitioned image. (b) Corresponding quadtree.

If only splitting were used, the final partition likely would contain adjacent regions with identical properties. This drawback may be remedied by allowing merging, as well as splitting. Satisfying the constraints of Section 10.4.1 requires merging only adjacent regions whose combined pixels satisfy the predicate P. That is, two adjacent regions R_j and R_k are merged only if $P(R_j \cup R_k) = \text{TRUE}$.

The preceding discussion may be summarized by the following procedure, in which, at any step we

1. Split into four disjoint quadrants any region R_i for which $P(R_i) = \text{FALSE}$.
2. Merge any adjacent regions R_j and R_k for which $P(R_j \cup R_k) = \text{TRUE}$.
3. Stop when no further merging or splitting is possible.

Segmentation by morphological watersheds

The concept of watersheds is based on visualizing an image in three dimensions: two spatial coordinates versus gray levels. In such a "topographic" interpretation, we consider three types of points: (a) points belonging to a regional minimum; (b) points at which a drop of water, if placed at the location of any of those points, would fall with certainty to a single minimum; and (c) points at which water would be equally likely to fall to more than one such minimum. For a particular regional minimum, the set of points satisfying condition (b) is called the *catchment basin* or *watershed* of that minimum. The points satisfying condition (c) form crest lines on the topographic surface and are termed *divide lines* or *watershed lines*.

The principal objective of segmentation algorithms based on these concepts is to find the watershed lines. The basic idea is simple: Suppose that a hole is punched in each regional minimum and that the entire topography is flooded from below by letting water rise through the holes at a uniform rate. When the rising water in distinct catchment basins is about to merge, a dam is built to prevent the merging. The flooding will eventually reach a stage when only the tops of the dams are visible above the water line. These dam boundaries correspond to the divide lines of the watersheds. Therefore, they are the (continuous) boundaries extracted by a watershed segmentation algorithm.

These ideas can be explained further with the aid of Fig. 10.44. Figure 10.44(a) shows a simple gray-scale image and Fig. 10.44(b) is a topographic view, in which the height of the "mountains" is proportional to gray-level values in the input image. For ease of interpretation, the backsides of structures are shaded. This is not to be confused with gray-level values; only the general topography of the three-dimensional representation is of interest. In order to prevent the rising water from spilling out through the edges of the structure, we imagine the perimeter of the entire topography (image) being enclosed by dams of height greater than the highest possible mountain, whose value is determined by the highest possible gray-level value in the input image.

Suppose that a hole is punched in each regional minimum [shown as dark areas in Fig. 10.44(b)] and that the entire topography is flooded from below

a b
c d

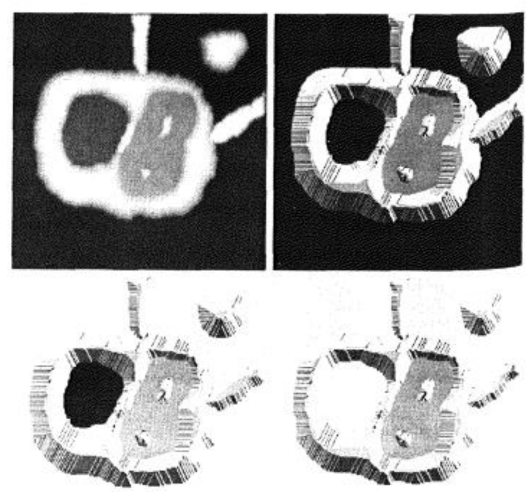
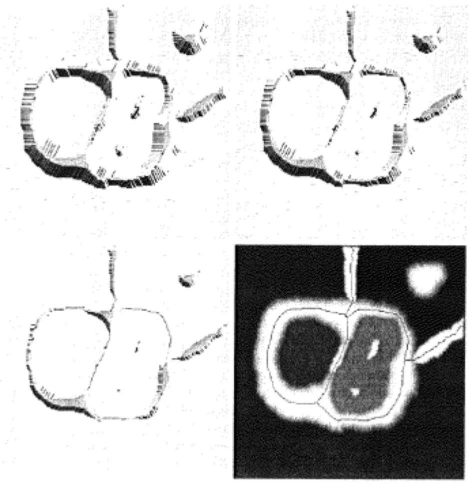


FIGURE 10.44 (a) Original image. (b) Topographic view. (c)–(d) Two stages of flooding.

by letting water rise through the holes at a uniform rate. Figure 10.44(c) shows the first stage of flooding, where the "water," shown in light gray, has covered only areas that correspond to the very dark background in the image. In Figs. 10.44(d) and (e) we see that the water now has risen into the first and second catchment basins, respectively. As the water continues to rise, it will eventually overflow from one catchment basin into another. The first indication of this is shown in 10.44(f). Here, water from the left basin actually overflowed into the basin on the right and a short "dam" (consisting of single pixels) was built to prevent water from merging at that level of flooding (the details of dam building are discussed in the following section). The effect is more pronounced as water continues to rise, as shown in Fig. 10.44(g). This figure shows a longer dam between the two catchment basins and another dam in the top part of the right basin. The latter dam was built to prevent merging of water from that basin with water from areas corresponding to the background. This process is continued until the maximum level of flooding (corresponding to the highest gray-level value in the image) is reached. The final dams correspond to the watershed lines, which are the desired segmentation result. The result for this example is shown in Fig. 10.44(h) as a dark, one-pixel-thick path superimposed on the original image. Note the important property that the watershed lines form a connected path, thus giving continuous boundaries between regions.

One of the principal applications of watershed segmentation is in the extraction of nearly uniform (bloblike) objects from the background. Regions characterized by small variations in gray levels have small gradient values. Thus, in practice, we often see watershed segmentation applied to the gradient of an image, rather than to the image itself. In this formulation, the regional minima of catchment basins correlate nicely with the small value of the gradient corresponding to the objects of interest.



e f
g h

FIGURE 10.44 *(Continued)* (e) Result of further flooding. (f) Beginning of merging of water from two catchment basins (a short dam was built between them). (g) Longer dams. (h) Final watershed (segmentation) lines. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

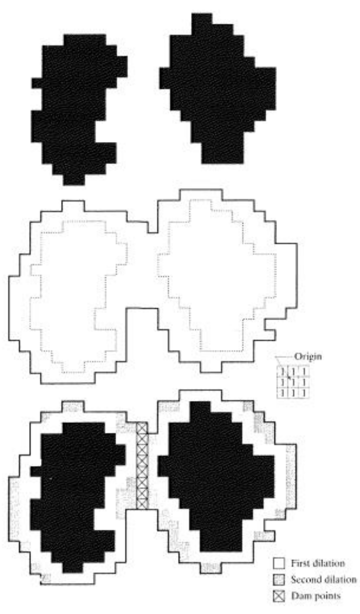
10.5.2 Dam Construction

Before proceeding, let us consider how to construct the dams or watershed lines required by watershed segmentation algorithms. Dam construction is based on binary images, which are members of 2-D integer space Z^2 (see Section 2.4.2). The simplest way to construct dams separating sets of binary points is to use morphological dilation (see Section 9.2.1).

The basics of how to construct dams using dilation are illustrated in Fig. 10.45. Figure 10.45(a) shows portions of two catchment basins at flooding step $n - 1$ and Fig. 10.45(b) shows the result at the next flooding step, n. The water has spilled from one basin to the other and, therefore, a dam must be built to keep this from happening. In order to be consistent with notation to be introduced shortly, let M_1 and M_2 denote the sets of coordinates of points in two regional minima. Then let the set of coordinates of points in the *catchment basin* associated with these two minima at stage $n - 1$ of flooding be denoted by $C_{n-1}(M_1)$ and $C_{n-1}(M_2)$, respectively. These are the two black regions shown in Fig. 10.45(a).

Let the union of these two sets be denoted by $C[n - 1]$. There are two connected components in Fig. 10.45(a) (see Section 2.5.2 regarding connected components) and only one connected component in Fig. 10.45(b). This connected component encompasses the earlier two components, shown dashed. The fact that two connected components have become a *single* component indicates that water between the two catchment basins has merged at flooding step n. Let this connected component be denoted q. Note that the two components from step $n - 1$ can be extracted from q by performing the simple AND operation $q \cap C[n - 1]$. We note also that all points belonging to an individual catchment basin form a single connected component.

Suppose that each of the connected components in Fig. 10.45(a) is dilated by the structuring element shown in Fig. 10.45(c), subject to two conditions: (1) The dilation has to be constrained to q (this means that the center of the structuring element can be located only at points in q during dilation), and (2) the dilation cannot be performed on points that would cause the sets being dilated to merge (become a single connected component). Figure 10.45(d) shows that a first dilation pass (in light gray) expanded the boundary of each original connected component. Note that condition (1) was satisfied by every point during dilation, and condition (2) did not apply to any point during the dilation process; thus the boundary of each region was expanded uniformly.



a
b c
d

FIGURE 10.45 (a) Two partially flooded catchment basins at stage $n - 1$ of flooding. (b) Flooding at stage n, showing that water has spilled between basins (for clarity, water is shown in white rather than black). (c) Structuring element used for dilation. (d) Result of dilation and dam construction.

In the second dilation (shown in medium gray), several points failed condition (1) while meeting condition (2), resulting in the broken perimeter shown in the figure. It also is evident that the only points in q that satisfy the two conditions under consideration describe the one-pixel-thick connected path shown crossed-hatched in Fig. 10.45(d). This path constitutes the desired separating dam at stage n of flooding. Construction of the dam at this level of flooding is completed by setting all the points in the path just determined to a value greater than the maximum gray-level value of the image. The height of all dams is generally set at 1 plus the maximum allowed value in the image. This will prevent water from crossing over the part of the completed dam as the level of flooding is increased. It is important to note that dams built by this procedure, which are the desired segmentation boundaries, are connected components. In other words, this method eliminates the problems of broken segmentation lines.

Although the procedure just described is based on a simple example, the method used for more complex situations is exactly the same, including the use of the 3×3 symmetric structuring element shown in Fig. 10.45(c).