Segmentation

1. Edge detection + Hough transform

2. Histogram + thresholding

3. Region growing – seeds, merge and split

4. Watershed

Distance Measures

Definition 1 *a distance metric satisfies*

(a) $D(p,q) \geq 0 \qquad D(p,q) = 0 \quad iff\ p = q$

(b) $D(p,q) = D(q,p)$

(c) $D(p,z) \leq D(p,q) + D(q,z)$

examples

1. Euclidean distance

   If $p = (x,y)$ and $q = (s,t)$ then

   $$D(p,q) = \sqrt{(x-s)^2 + (y-t)^2}$$

   advantage: isotropic

   disadvantage: non-integer, non-local

   Going from pixel p to z to q does not give the same distance as going from p to q

2. D_4 or city-block distance

   $$D(p,q) = |x-s| + |y-t|$$

3. D_8 or chessboard distance

   $$D(p,q) = \max\left(|x-s|, |y-t|\right)$$

   Note: D_4 and D_8 are path independent that involve only pixels.

4. other masks: e.g. nearest 4 neighbors have distance 3 and corners have distance 4 one can normalize at the end if necessary approximation to Euclidean with integers

Definition 2 *Chamfering: find distance of pixels to an image subset*

The distance is 0 for pixels in the subset, small for pixels near the subset.
and large for pixels far from the subset.
Note: The results changes a binary image to a gray level image

Chamfer algorithm:

1. set $D = 0$ for pixels in the subset S and $D = \infty$ for pixels not in S
2. pass through the image from top to bottom and left to right (i.e. points denoted by AL)

   set
   $$F(p) = \min_{q \in AL} [F(p), D(p,q) + F(q)]$$
3. pass through the image from bottom to top and right to left (i.e. points denoted by BR)

   set
   $$F(p) = \min_{q \in BR} [F(p), D(p,q) + F(q)]$$
4. F contains the chamfer of the subset S.

AL	AL	BR
AL	AL&BR	BR
AL	BR	BR

We can define erosion by a ball of radius n as

$$\varepsilon(X) = \{x \in X \,|\, D(x) > n\}$$

and then thresholding

Definition 3 *The Hausdorff distance between two sets X and Y is the minimum of the radius λ of disks B_λ such that X dilated by B_λ contains Y, and Y dilated by B_λ contains X.*

$$d_H(X,Y) = \min \lambda \,|\, X \subseteq \delta_{B_\lambda}(Y), \quad Y \subseteq \delta_{B_\lambda}(X)$$

Caution: The Hausdorff function is very sensitive to noise.

Definition 4 *B is a maximal disk in X if there are no other disks that contain B*

Theorem 5 *B is a maximal disk in X iff B is tangent to at least 2 distinct points on the boundary.*

Definition 6 *The skeleton SK(X) of a set X is defined by the centers of the maximal centers*

$$x \in SK(x) \ \ \textit{iff}$$
$$\exists y_1, y_2 \in \partial X \,|\, y_1 \neq y_2 \ \textit{and}\ d(x, \partial X) = d(x, y_1) = d(x, y_2)$$

18.7.5 The Distance Transformation

Another related operation that can be performed on binary images is the distance transformation. It results, however, not in another binary image, but in a gray-level image. The gray level at each pixel is the distance from that pixel to the nearest background pixel.

An approximate distance transformation can be computed by an erosion-like operation wherein, on each pass, pixels are labeled with the iteration number rather than being eliminated from the object. The so-called *chamfer algorithm* computes a distance transformation in only two passes over the image [51,52].

Figure 18–26 illustrates the concept of the two-pass distance transformation in one dimension. Figure 18–26(a) is a one-dimensional binary image containing an object denoted by 1's on a background of 0's. Figure 18–26(b) is the result of the first (forward) pass, which is conducted from left to right. At each pixel, background points are left as zeros, but interior points are replaced with a count of how many steps have been taken since the last zero was encountered. In Figure 18–26(c), we see the result of the second (backward) pass, which is conducted from right to left. In this pass, each pixel is replaced with the minimum of (a) what it was or (b) the number of steps taken since a zero was last encountered. The result is an image in which gray level reflects distance to the nearest boundary.

In the two-dimensional distance transformation, a mask resembling a convolution kernel (see Figure 18–27) is passed over the image in a process reminiscent of the convolution operation. (Recall Sec. 9.3.4.) As with the one-dimensional distance transformation,

0	0	0	0	1	1	1	1	1	0	0	0	0	(a)
0	0	0	0	1	2	3	4	5	0	0	0	0	(b)
0	0	0	0	1	2	3	2	1	0	0	0	0	(c)

Figure 18–26 One-dimensional distance transformation: (a) binary image; (b) result of first (L→R) pass; (c) result of second (R→L) pass

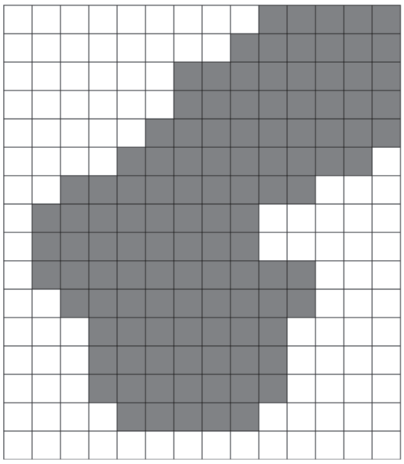
Distance Transforms

Let P be a binary picture (defined on grid $\mathbb{G}$) in which

$$\langle P \rangle = \{p : p \in \mathbb{G} \wedge P(p) = 1\} = P^{-1}(1), \text{ and}$$

$$\langle \overline{P} \rangle = \{p : p \in \mathbb{G} \wedge P(p) = 0\} = P^{-1}(0)$$

are proper subsets of $\mathbb{G}$. For any grid metric d_α, the d_α *distance transform* of P associates with every pixel p of $\langle P \rangle$ the d_α distance from p to $\langle \overline{P} \rangle$.



									1	1	1	1	1
								1	2	2	2	2	1
						1	1	2	3	3	3	2	1
						1	2	3	4	4	3	2	1
					1	2	3	4	3	3	2	2	1
				1	2	3	4	3	2	2	1	1	
		1	1	2	3	4	3	2	1	1			
	1	2	2	3	4	3	2	1					
	1	2	3	4	4	3	2	1					
	1	2	3	4	5	4	3	2	1	1			
		1	2	3	4	5	4	3	2	1			
			1	2	3	4	3	2	1				
			1	2	3	3	3	2	1				
			1	2	2	2	2	2	1				
				1	1	1	1	1					

									1	1	1	1	1
								1	1	2	2	2	1
						1	1	1	2	2	3	2	1
						1	2	2	2	3	2	2	1
					1	1	2	3	2	2	2	1	1
				1	1	2	2	2	2	1	1	1	
		1	1	1	2	2	2	1	1	1			
	1	1	2	2	2	3	2	1					
	1	2	2	3	3	3	2	1					
	1	1	2	2	3	3	2	1	1	1			
		1	1	2	3	3	2	2	1	1			
			1	2	3	3	3	2	1				
			1	2	2	3	2	2	1				
			1	1	2	2	2	1	1				
				1	1	1	1	1					

Left: Picture. Center: d_4 distance transform. Right: d_8 distance transform. – Distance transforms are frequently used when analyzing regions or patterns in pictures.

We assume that pixels of the background component (i.e., containing all pixels outside of the rectangular region $\mathbb{G}$) all have value 0.

Two-Pass Algorithm

The d_4 or d_8 distance transform of P (and others – see later) can be computed by performing a series of local operations while scanning $\mathbb{G}$ twice.

(A *local operation* gives each pixel p a new value that depends only on the old values of the neighbors of p.)

For any $p \in \mathbb{G}$ let $B(p)$ ("before") be the set of pixels (4- or 8-) adjacent to p that precede p when $\mathbb{G}$ is scanned in standard order:

> if p has coordinates (x, y), B contains $(x, y+1)$ and $(x-1, y)$, and if we use 8-adjacency it also contains $(x-1, y+1)$ and $(x+1, y+1)$.

Let $A(p)$ ("after") be the remaining (4- or 8-) neighbors of p.

First Scan

$$f_1(p) = \begin{cases} 0 & \text{if } p \in \langle \overline{P} \rangle \\ \min\{f_1(q) + 1 : q \in B(p)\} & \text{if } p \in \langle P \rangle \end{cases}$$

Compute $f_1(p)$ for all $p \in \mathbb{G}$ in a single standard scan of $\mathbb{G}$; for each p, f_1 has already been computed for all of the qs in $B(p)$

(If p is on the top row or in the left column of $\mathbb{G}$, some of these qs are outside $\mathbb{G}$ with $f_1 = 0$.)

									1	1	1	1	1
								1	2	2	2	2	2
						1	1	2	3	3	3	3	3
						1	2	3	4	4	4	4	4
					1	2	3	4	5	5	5	5	5
				1	2	3	4	5	6	6	6	6	
		1	1	2	3	4	5	6	7	7			
	1	2	2	3	4	5	6	7					
	1	2	3	4	5	6	7	8					
	1	2	3	4	5	6	7	8	1	1			
		1	2	3	4	5	6	7	2	2			
			1	2	3	4	5	6	3				
			1	2	3	4	5	6	4				
			1	2	3	4	5	6	5				
				1	2	3	4	5					

									1	1	1	1	1
								1	1	2	2	2	1
						1	1	1	2	2	3	2	1
						1	2	2	2	3	3	2	1
					1	1	2	3	3	3	3	2	1
				1	1	2	2	3	4	4	3	2	
		1	1	1	2	2	3	3	4	4			
	1	1	2	2	2	3	3	4					
	1	2	2	3	3	3	4	1					
	1	2	3	3	4	4	2	1	1	1			
		1	2	3	4	3	2	2	2	1			
			1	2	3	3	3	3	2				
			1	2	3	4	4	3	1				
			1	2	3	4	4	2	1				
				1	3	4	3	2					

After the first scan. Left: d_4 transform. Right: d_8 transform.

Second Scan

$$f_2(p) = \min \{ f_1(p), f_2(q) + 1 : \; q \in A(p) \} \; .$$

Compute $f_2(p)$ for all $p \in \mathbb{G}$ in a single reverse standard (i.e., right-to-left, bottom-to-top) scan of $\mathbb{G}$ (note: each p, f_2 has already been computed for all of the qs in $A(p)$ or is known because they are outside of $\mathbb{G}$).

Theorem 1 *$f_2(p) = d(p, \langle \overline{P} \rangle)$ for all $p \in \mathbb{G}$ where $d = d_4$ for the 4-adjacency version of the algorithm and $d = d_8$ for the 8-adjacency version.*

Thus, the obtained values after the second scan are as shown on page 1.

2D Approximations to Euclidean Metric

Set of points within a given d_4 or d_8 distance from a given point is a square (and not a digitized disk).

These distances depend on direction; their "disks" are not good approximations to Euclidean disks.

If we restrict d_4 and d_8 to $\mathbb{Z}^2$ the set of grid points q such that $d_4(p,q) \le k$ is a diagonally oriented square (a *diamond*) of odd diagonal length $2k+1$ centered at p, and the set of grid points q such that $d_8(p,q) \le k$ is an upright square of odd side length $2k+1$ centered at p.

Example of a better approximation of the Euclidean metric:

$$d(p,q) = \max\{d_8(p,q),\ \tfrac{2}{3} \cdot d_4(p,q)\}$$

The set of grid points such that $d(p,q) \le k$ is the intersection of an upright square of side length k with a diamond of diagonal length $3k/2$; this intersection is an upright octagon.

Best approximation of the Euclidean metric:

$\lceil d_e \rceil$ is the integer-valued metric that best approximates d_e.

"Incremental" algorithms for distance computation on a grid normally use local neighborhoods; this makes it easy to compute metrics such as d_4, or d_8 or octagonal metrics, but not (in the same straightforward way) $\lceil d_e \rceil$.

2D Chamfer Metrics

A general method of defining approximations to Euclidean distance: count moves in different directions (e.g., isothetic moves, diagonal moves) and use different weights for these moves (e.g., 1 and $\sqrt{2}$).

Let $p, q \in \mathbb{Z}^2$ and ρ a sequence of king's moves from p to q.

m = number of isothetic moves; n = number of diagonal moves

$$l_{a,b}(\rho) = ma + nb$$

$$d_{a,b}(p,q) = \min_\rho l_{a,b}(\rho)$$

$d_{a,b}$ is the (a, b) *chamfer distance* (or *weighted distance*) from p to q (G. Borgefors, 1984).

Theorem 2 *If* $0 < a \leq b \leq 2a$ *(called the* Montanari condition*), then the* (a, b) *chamfer distance* $d_{a,b}$ *is a metric.*

The chamfer distance $d_{1,b}$ that best approximates d_e has $b = (1/\sqrt{2}) + \sqrt{\sqrt{2} - 1} \approx 1{\cdot}351$; for this $d_{1,b}$ we have a maximum error

$$|d_e - d_{1,b}| \leq ((1/\sqrt{2}) - \sqrt{\sqrt{2} - 1})k \approx 0{\cdot}06k$$

on an $(k+1) \times (k+1)$ grid $\mathbb{G}$. This optimal b is close to $4/3$; we therefore get a good approximation to $3d_e$ by using $a = 3$ and $b = 4$.

there are two passes. The forward pass moves from left to right, working down the image from the top, while the backward pass moves from right to left, working up the image from the bottom. At each position, a set of two-term sums is formed by adding each element in the mask to the underlying pixel value. Where the mask is blank, nothing is done. The pixel under the center of the mask is replaced by the minimum of the sums.

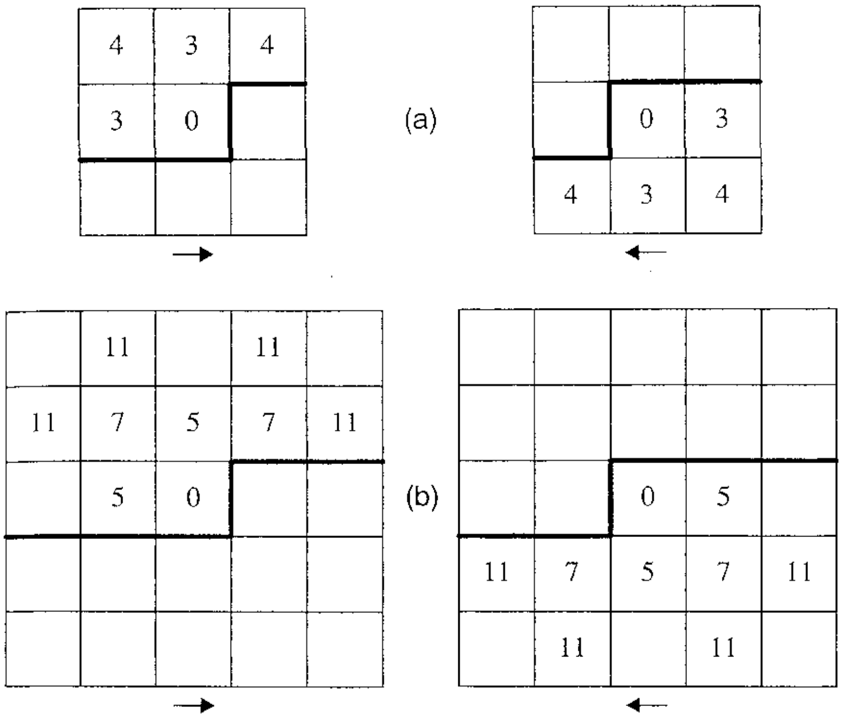


Figure 18–27 Mask pairs for two-dimensional distance transformation using the chamfer algorithm: (a) 3×3, (b) 5×5

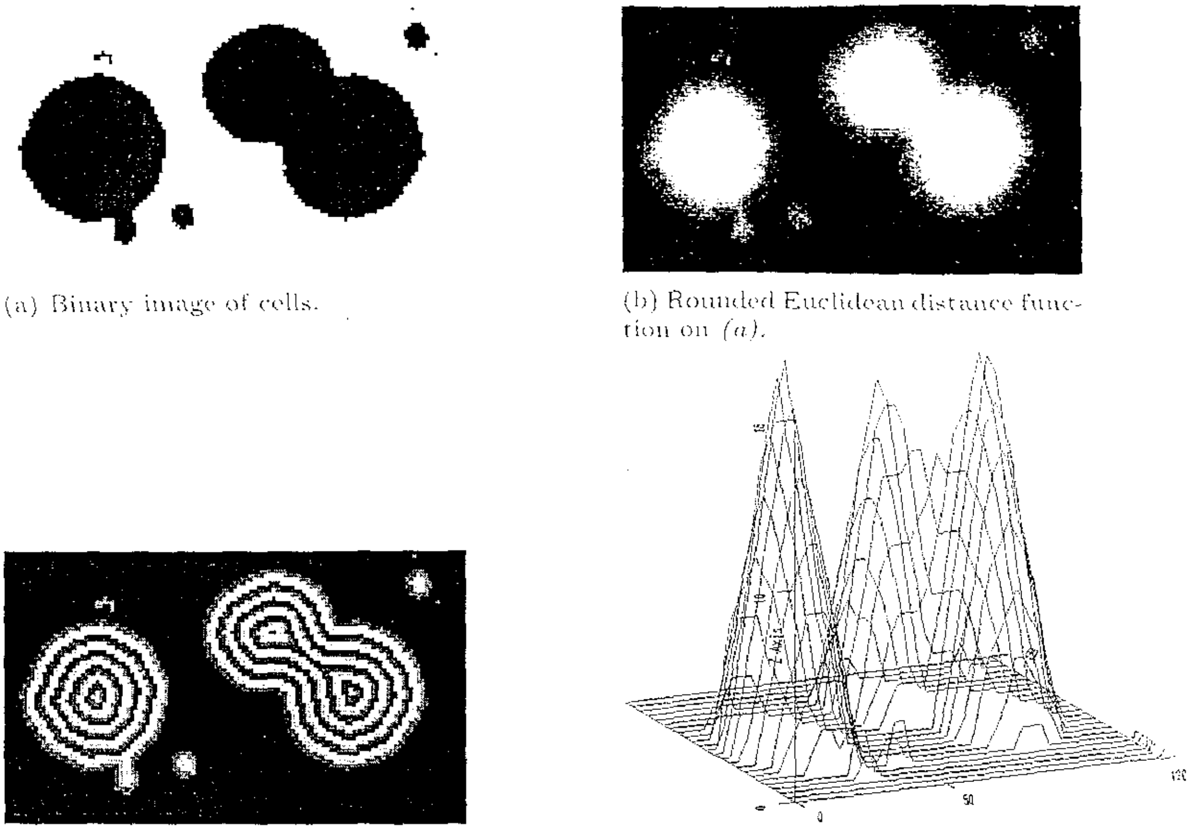
The three-by-three masks in Figure 18–27 yield a distance image in which the gray levels are three times the Euclidian distance to the boundary. The maximum deviation from true Euclidian distance is 8 percent. The five-by-five masks yield a distance image that is scaled up by a factor of five, and their maximum error is only 2 percent [52].

The distance transform is useful, for example, in segmenting clusters of objects that are in contact. Each object in the cluster produces a local maximum (located roughly at its center) in the distance image. The watershed algorithm (decreasing from an initially high threshold) can then segment the distance image into the individual component objects, as shown in Figure 18–28. Using the watershed algorithm on the distance transformed image (Figure 18–28(b)) effectively breaks apart circular objects that are touching (Figure 18–28(c)).

18.7.6 Boundary Curvature Analysis

The curvature at a point on a curve is defined as the rate of change of the tangent angle at that point, as one traverses the curve. The curvature of an object's boundary is positive in regions where the object is convex and negative where it is concave.

In Figure 18–29, for example, a plot of the curvature of the boundary shown reveals two sharp negative peaks corresponding to the two concavities. If the objects are expected to be convex, this signals a segmentation error. A cutting line, drawn between the two points a and b, separates the two objects. Thus, the boundary curvature function can assist in the automatic detection and correction of segmentation errors.



(a) Binary image of cells.

(b) Rounded Euclidean distance function on *(a)*.

(c) Distance function modulo 4.

(d) Topographic representation of *(b)*.

Fig. 2.32. Distance function on a binary image of cells. Note that the high values of the distance function correspond to the centre of the cells.

pixels. Once the two scans have been performed, the input binary image f holds the distance function:

Forward scan of all pixels $p \in \mathcal{D}_f$
    if $f(p) = 1$
        $f(p) \leftarrow 1 + \min\{f(q) \mid q \in N_G^{\leftarrow}(p)\}$;
Backward scan of all pixels $p \in \mathcal{D}_f$

3D Chamfer Metrics

$d_{a,b,c}$ where a, b, and c correspond to moves in which

only one coordinate changes (isothetic moves),

two coordinates change,

and all three coordinates change,

and we can obtain good approximations to Euclidean distance by appropriately choosing a, b, and c.

Generalization

Generalized chamfer distances can be defined using additional types of moves that are not necessarily moves between 8-neighbors or 26-neighbors.

Conclusion

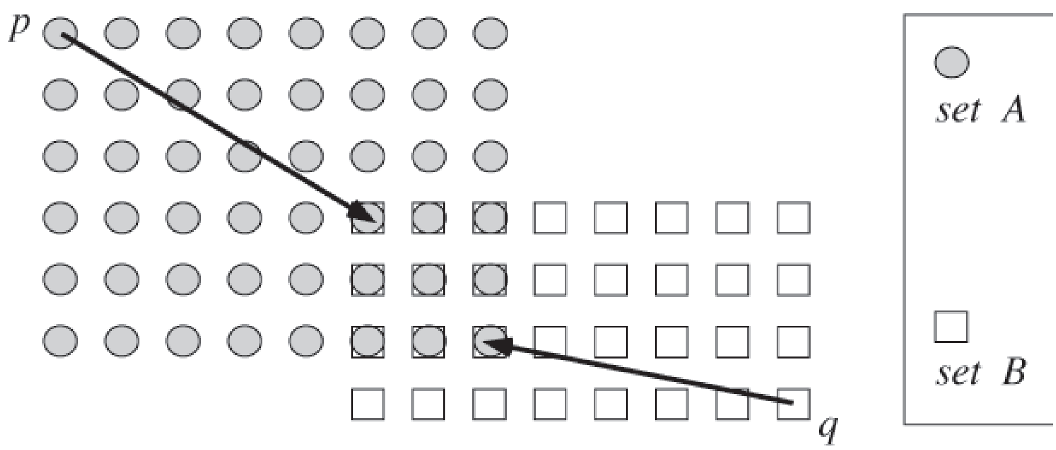
The two-scan algorithm can be used for any chamfer-metric distance transform, and this allows to "approximate the Euclidean case".

Hausdorff Distance Between Sets

Any metric d on a set S can be extended to a *Hausdorff metric* on the family of all nonempty subsets A, B of S:

$$d(A,B) = \max\left\{\max_{p\in A}\ \min_{q\in B}\ d(p,q), \max_{p\in B}\ \min_{q\in A}\ d(p,q)\right\}$$

(Actually, in general we should use $\sup$ and $\inf$ in this definition; but in our context we can stay with $\max$ and $\min$.)



The Hausdorff distance between A and B is defined by the maximum of the two indicated distances.

Hausdorff Metric in Steps

In general: *closest distance* from $p \in S$ to $T \subseteq S$

$$d(p, T) = \inf_{q \in T} d(p, q)$$

Let $A, B \subset S$, $h_p(B) = d(p, B)$ for all $p \in A$, $h_p(A) = d(p, A)$ for all $p \in B$, and

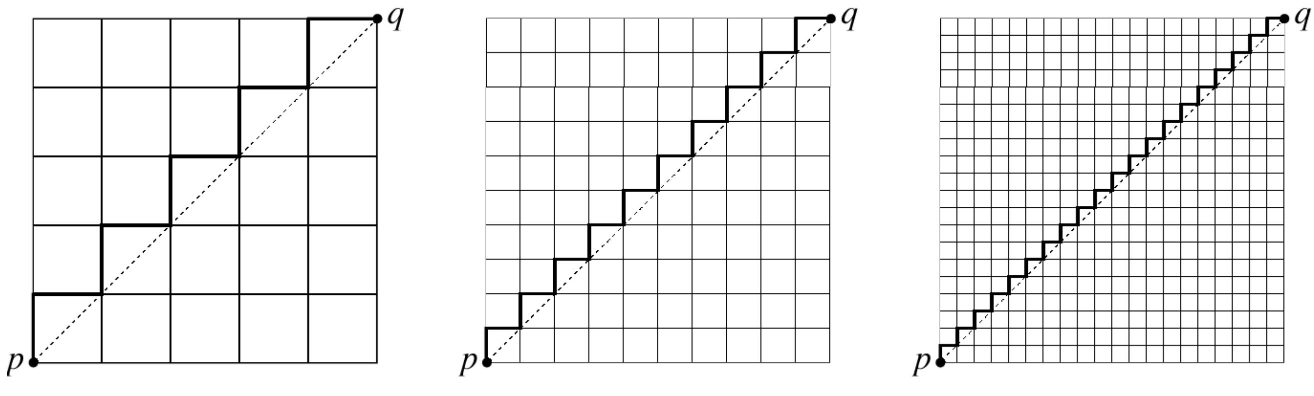
$$h_A(B) = \sup_{p \in A} h_p(B); \quad h_B(A) = \sup_{p \in B} h_p(A)$$

Then we have the Hausdorff metric

$$d(A, B) = \max\{h_A(B), h_B(A)\}$$

Figure on page 7: Assume $d = d_e$. From $h_A(B) = h_p(B) = \sqrt{34}$ and $h_B(A) = h_q(A) = \sqrt{26}$ we have $d(A, B) = h_A(B)$.

Convergence in Hausdorff Metric



The Hausdorff distance between diagonal pq (set A) and "staircase arc" (set B) converges to zero as the grid resolution increases (but the length of the staircase does not converges to the length of the diagonal).

Definition Based on Dilation

An alternative way of defining the Hausdorff metric uses the *ε-neighborhood* of a set:

$$U_\varepsilon(A) = \{q : \; q \in S \; \wedge \; h_q(A) < \varepsilon\}$$

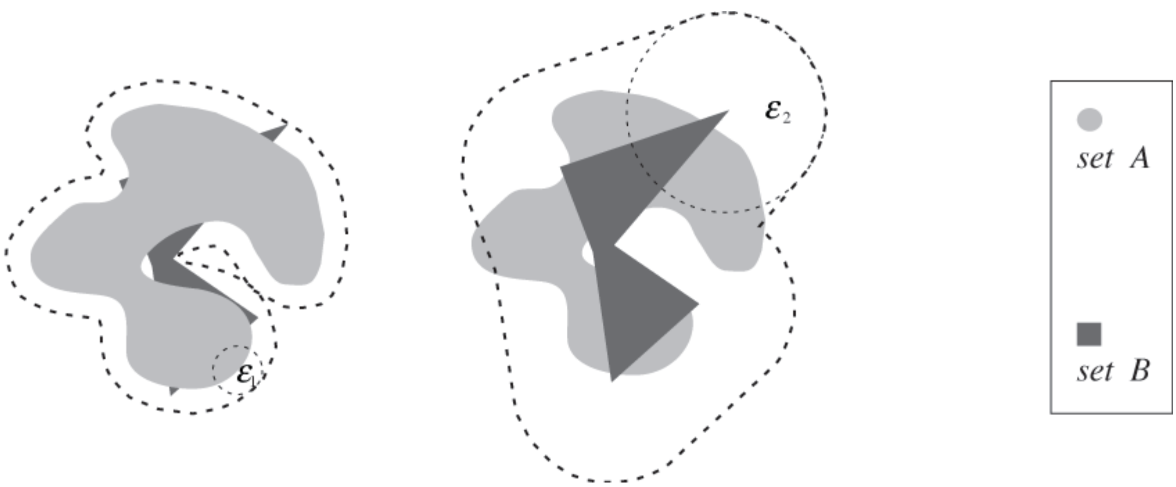
where h_q is defined by a metric d on S, $\varepsilon > 0$, and $A \subset S$.

Let A and B be nonempty subsets of S:

$$h_A(B) = \inf\{\varepsilon : \; A \subseteq U_\varepsilon(B)\}$$

(analogously for $h_B(A)$) and

$$d(A, B) = \max\{h_A(B), h_B(A)\}$$



Left: B (a simple polygon) is completely contained in $U_{\varepsilon_1}(A)$. Right: A is not completely contained in $U_{\varepsilon_2}(B)$, showing that $d(A, B) > \varepsilon_2$.

An Algorithm for Hausdorff Distance

Assume two finite sets $A, B \subset \mathbb{G}_{m,n}$ of grid points.

For any $S \subset \mathbb{G}_{m,n}$, the *distance field* $F(S)$ is an array of size $m \times n$ such that $F(S)(p) = h_p(S)$; in particular, $F(S)(p) = 0$ iff $p \in S$.

Note: the distance field can be calculated (in $\mathcal{O}(mn)$ computation steps) by a two-scan distance transform, assuming that d is a chamfer metric.

1. Calculate a distance field $F(A)$ in an array of size $m \times n$.
2. Calculate a distance field $F(B)$ in an array of size $m \times n$.
3. Let a be the maximum value in $F(A)$ at all positions belonging to B.
4. Let b be the maximum value in $F(B)$ at all positions belonging to A.
5. $H(A, B) = \max\{a, b\}$.

Algorithm for calculating the Hausdorff distance between two subsets A and B of an $m \times n$ grid.

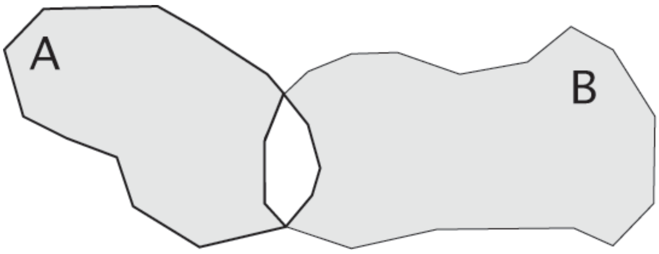
Metrics by Symmetric Difference

Hausdorff metrics are based on maximum distances between sets; a single point (an "outlier") in a set can strongly influence these distances.

Distances between sets defined by set-theoretic differences are less sensitive to single points.

Symmetric difference between two subsets A, B of a set S:

$$A \Delta B = (A \setminus B) \cup (B \setminus A)$$



The symmetric difference is shaded.

$$d_{\mathrm{sym}}(A,B) = \mathrm{card}(A \Delta B) \text{ and } d'_{\mathrm{sym}}(A,B) = \frac{\mathrm{card}(A \Delta B)}{\mathrm{card}(A \cup B) + 1}$$

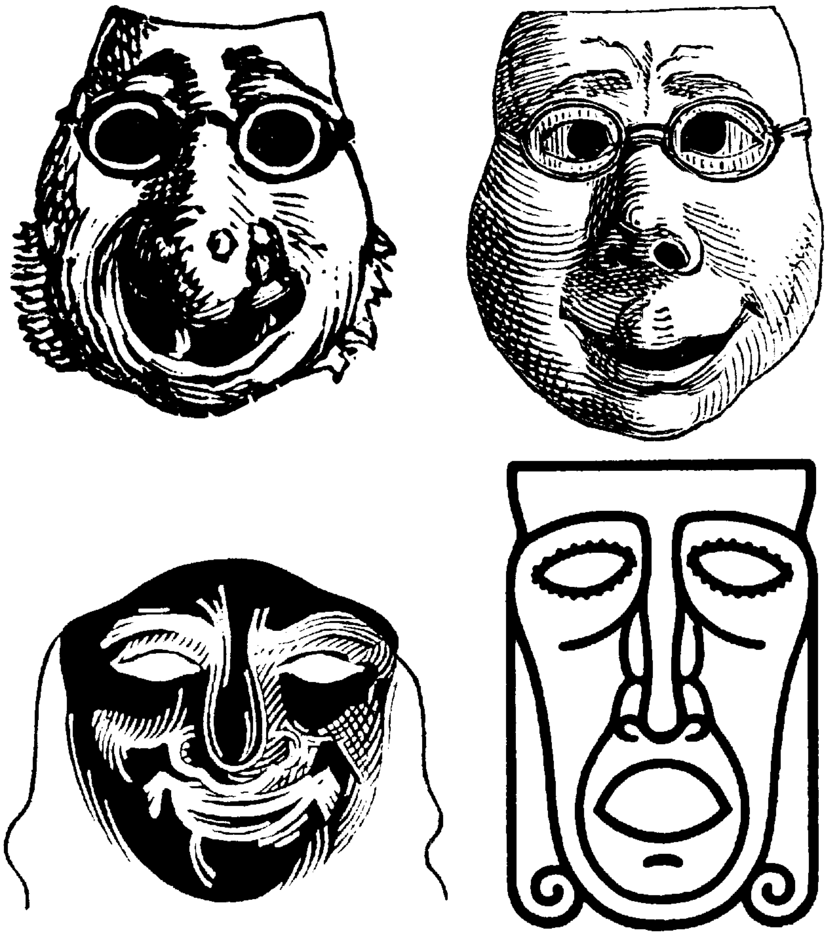
Theorem 3 *d_{sym} and d'_{sym} are metrics on the family of all finite subsets of S.*

Here, we only have to calculate cardinalities $\mathrm{card}(M)$ of sets M of pixels or voxels.

Coursework

Related material in textbook: Sections 3.4.2, 3.1.8, 3.2.3 (you may skip d_h), and 3.2.5. Do Exercises 11 and 17 on page 114.

A.8. [4 marks] Use a set of at least four different binary pictures of similar objects. Implement a program which measures the similarity of these based on calculated distances.



For example, at first center all objects at their origin (possibly align them also in orientation) and then compare these objects based on a chosen metric.

18.3.7 The Watershed Algorithm

A relative of adaptive thresholding is the watershed algorithm. Figure 18–8 illustrates how this approach works. We assume that the objects in the figure are of low gray level, on a high-gray-level background. The figure shows the gray levels along one scan line that cuts through two objects that are close together.

The image is initially thresholded at a low gray level, one that segments the image into the proper number of objects, but with boundaries that are too small. Then the threshold is raised gradually, one gray level at a time. The objects' boundaries will expand as the threshold increases. When they touch, however, the objects are not allowed to merge. Thus, these points of first contact become the final boundaries between adjacent objects. The process is terminated before the threshold reaches the gray level of the background—that is, at the point when the boundaries of well-isolated objects are properly set.

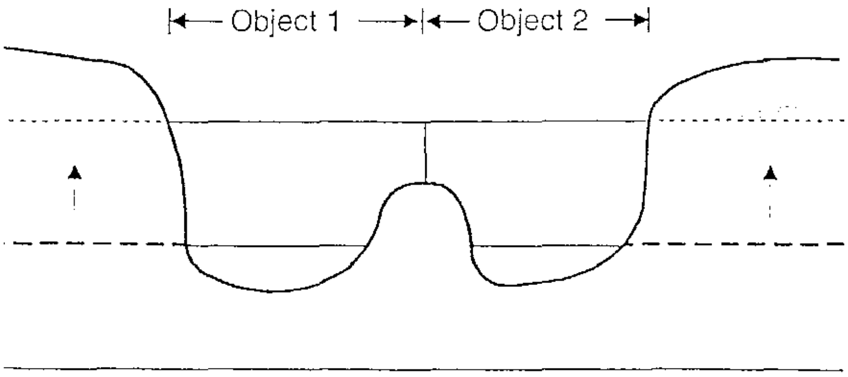


Figure 18–8 The watershed algorithm

Rather than simply thresholding the image at the optimum gray level, then, the watershed approach begins with a threshold that is too low, but that properly isolates the individual objects. Then as the threshold is gradually raised to the optimum level, merging of objects is not allowed. This can solve the problem posed by objects that are too close together for global thresholding to work. The final segmentation will be correct (i.e., there will be one boundary per actual object in the image) if and only if the segmentation at the initial threshold is correct.

Both the initial and final threshold gray levels must be well chosen. If the initial threshold is too low, then low-contrast objects will be missed at first and then merged with nearby objects as the threshold increases. If the initial threshold is too high, objects will be merged from the start. The final threshold value determines how well the final boundaries fit the objects. The threshold selection methods discussed in this chapter can be useful in setting these two values.

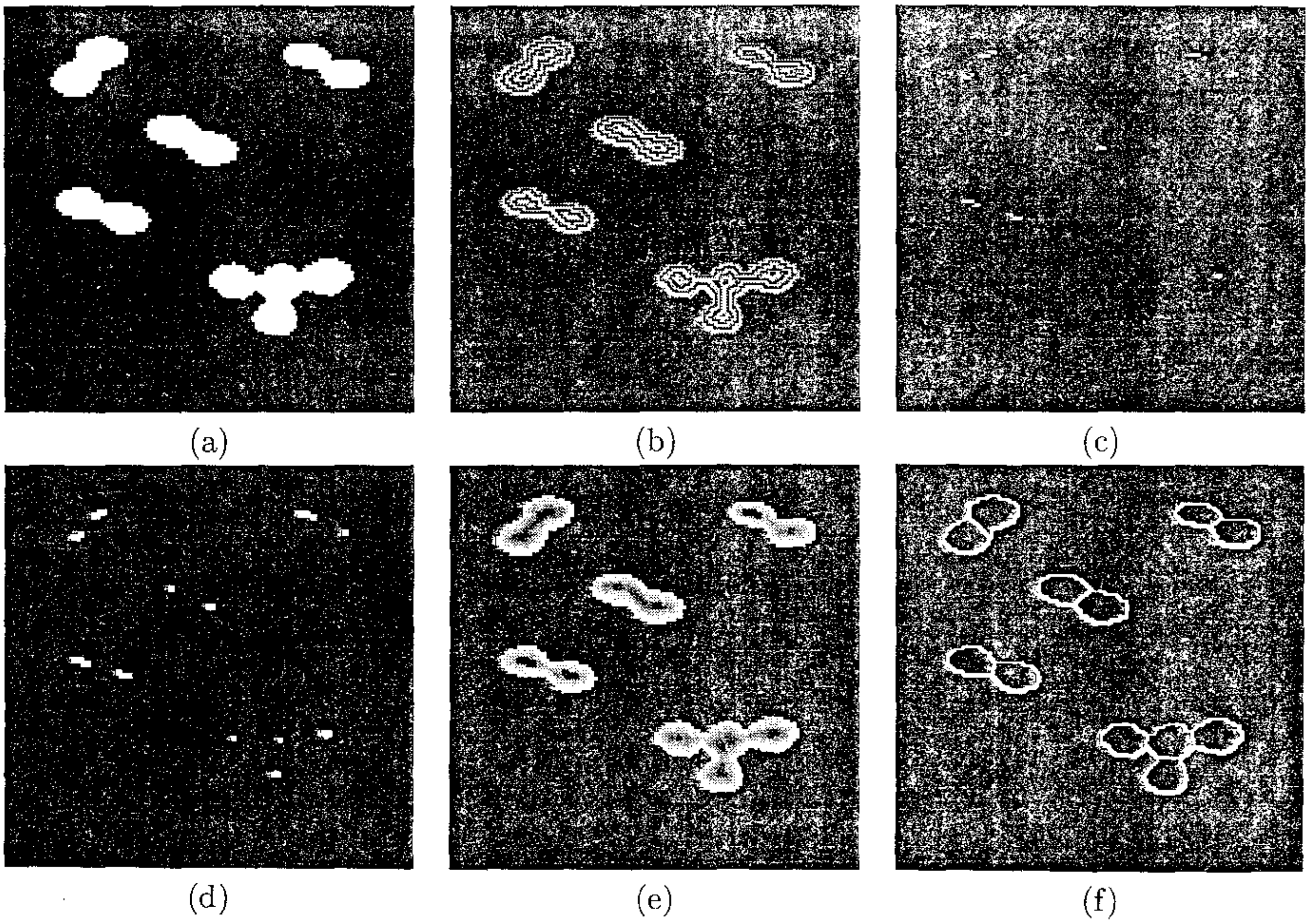


Figure 11.47: *Particle segmentation by watersheds: (a) original binary image; (b) distance function visualized using contours; (c) regional maxima of the distance function used as particle markers; (d) dilated markers; (e) inverse of the distance function with the markers superimposed; (f) resulting contours of particles obtained by watershed segmentation. Courtesy P. Kodl, Rockwell Automation Research Center, Prague, Czech Republic.*

11.7.3 Gray-scale segmentation, watersheds

The markers and watersheds method can also be applied to gray-scale segmentation. Watersheds are also used as crest-line extractors in gray-scale images. The contour of a region in a gray-level image corresponds to points in the image where gray-levels change most quickly—this is analogous to edge-based segmentation considered in Chapter 5. The watershed transformation is applied to the gradient magnitude image in this case (see Figure 11.48). There is a simple approximation to the gradient image used in mathematical morphology called Beucher's gradient [Serra 82], calculated as the algebraic difference of unit-size dilation and unit-size erosion of the input image X.

$$\text{grad}(X) = (X \oplus B) - (X \ominus B) \tag{11.74}$$

The main problem with segmentation via gradient images without markers is **oversegmentation**, meaning that the image is partitioned into too many regions (Figure 11.47c). Some techniques to limit oversegmentation in watershed segmentation are given in [Vincent

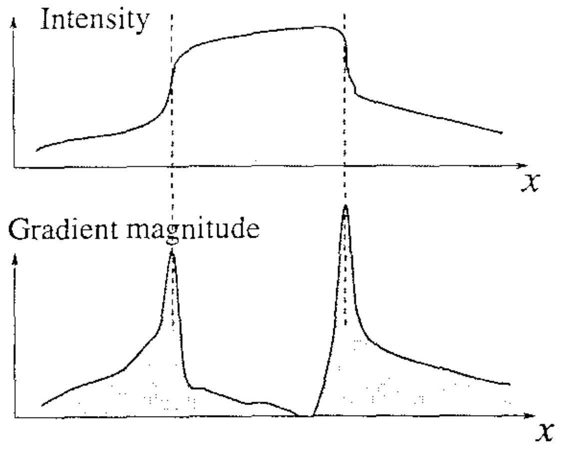


Figure 11.48: *Segmentation in gray-scale images using gradient magnitude.*

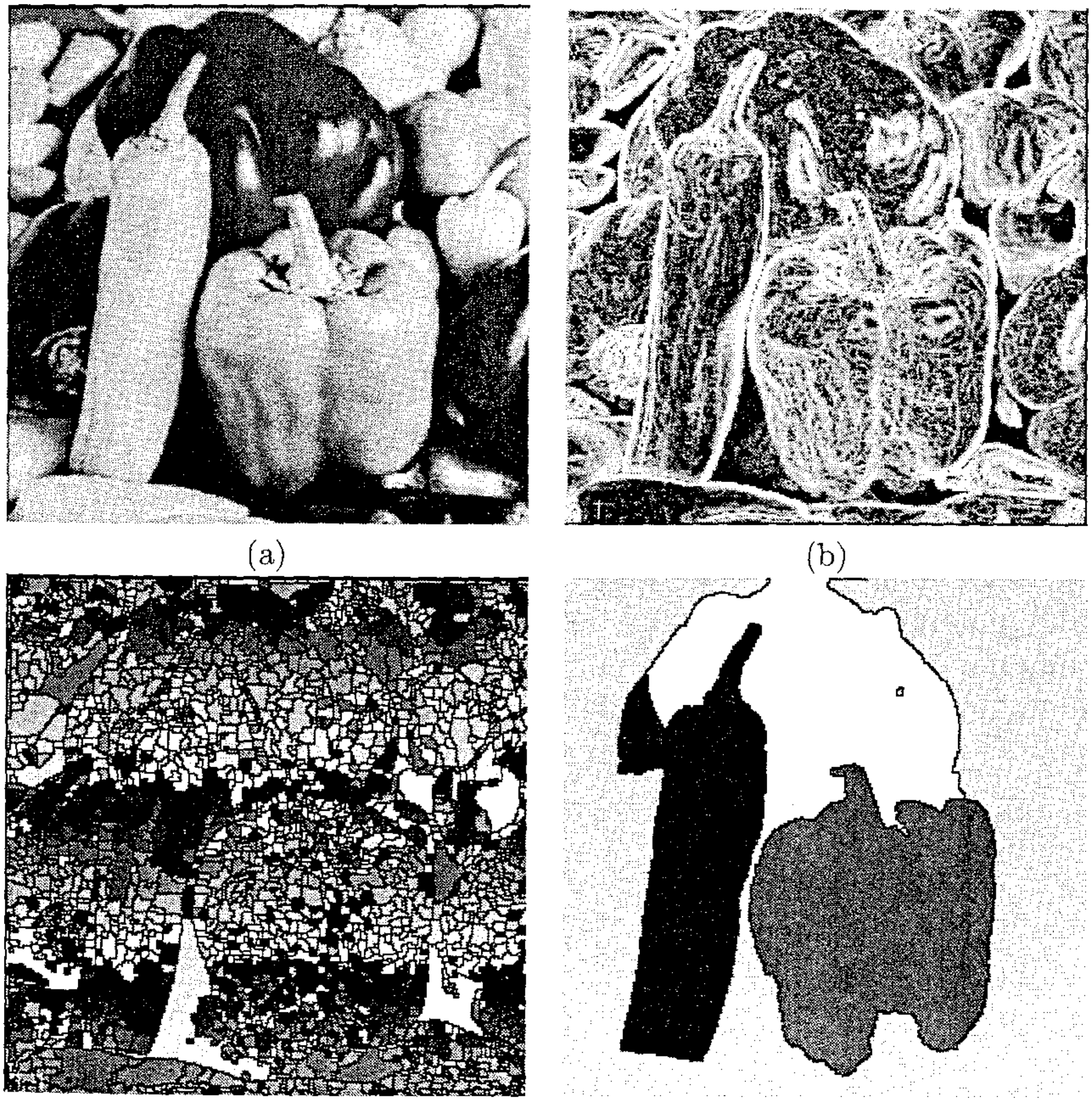
93]. The watershed segmentation methods with markers do not suffer from oversegmentation, of course.

An example from opthalmology will illustrate the application of watershed segmentation. The input image shows a microscopic picture of part of a human retina, Figure 11.49a—the task is to segment individual cells on the retina. The markers/watershed paradigm was followed, with markers being found using a carefully tuned Gaussian filter (see Figure 11.49b). The final result with the outlined contours of the cells is in Figure 11.49c.

11.8 Summary

- **Mathematical morphology**
  - Mathematical morphology stresses the role of **shape** in image pre-processing, segmentation, and object description. It constitutes a set of tools that have a solid mathematical background and lead to fast algorithms. The basic entity is a **point set**. Morphology operates using transformations that are described using operators in a relatively simple **non-linear algebra**. Mathematical morphology constitutes a counterpart to traditional signal processing based on linear operators (such as convolution).
  - Mathematical morphology is usually divided into **binary mathematical morphology** which operates on binary images (2D point sets), and **gray-level mathematical morphology** which acts on gray-level images (3D point sets).
- **Morphological operations**
  - In images, morphological operations are **relations of two sets**. One is an image and the second a small probe, called a **structuring element**, that systematically traverses the image; its relation to the image in each position is stored in the output image.
  - Fundamental operations of mathematical morphology are **dilation** and **erosion**. Dilation expands an object to the closest pixels of the neighborhood. Erosion

segmentation. An approach introduced in [Liow and Pavlidis 88, Pavlidis and Liow 90] solve several quadtree-related region growing problems and incorporates two post-processing step: First, boundary elimination removes some borders between adjacent regions according to thei contrast properties and direction changes along the border, taking resulting topology int consideration. Second, contours from the previous step are modified to be located precisel on appropriate image edges. Post-processing contour relaxation is suggested in [Aach e al. 89]. A combination of independent region growing and edge-based detected borders i described in [Koivunen and Pietikainen 90]. Other approaches combining region growing anc edge detection can be found in [Venturi et al. 92, Manos et al. 93, Gambotto 93, Wu 93, Chu and Aggarwal 93, Vlachos and Constantinides 93, Lerner et al. 94, Falah et al. 94, Gever and Smeulders 97].



(a) (b) (c) (d)

Figure 5.51: *Watershed segmentation: (a) original; (b) gradient image,* 3×3 *Sobel edge detection, histogram equalized; (c) raw watershed segmentation; (d) watershed segmentation using region markers to control oversegmentation. Courtesy W. Higgins, Penn State University.*