

# Image Processing

## Lecture 14: Feature Extraction (Ch11 Feature Extraction)

Zhiguo Zhang

[zhiguo Zhang@hit.edu.cn](mailto:zhiguo Zhang@hit.edu.cn)



# Review of Last Lecture

---

- In the last lecture we learnt:
  - Thresholding
  - Region based segmentation
  - Morphological watersheds approach

# Contents of This Lecture

---

- Background of feature extraction
- Boundary feature descriptor
- Region feature descriptor
- Other descriptors

# Background

---

- After an image has been segmented into regions or their boundaries, the resulting sets of segmented pixels usually have to be converted into a form suitable for further computer processing.
- Typically, the step after segmentation is **feature extraction**, which consists of **feature detection** and **feature description**.

# Background

---

- Although there is no universally accepted, formal definition of what constitutes an *image feature*, there is little argument that, intuitively, we generally think of a feature as a distinctive attribute or description of “something” we want to *label* or *differentiate*.
- The “something” of interest refers either to individual image objects, or even to entire images or sets of images.
- Thus, we think of features as attributes that are going to help us assign unique labels to objects in an image or, more generally, are going to be of value in differentiating between entire images or families of images.

# Background

---

- *Feature detection* refers to finding the features in an image, region, or boundary.
- *Feature description* assigns quantitative attributes to the detected features.
- For example, we might *detect* corners in a region boundary, and *describe* those corners by their orientation and location, both of which are quantitative attributes.

# Background

---

- Feature processing methods discussed in this chapter are subdivided into three principal categories, depending on whether they are applicable to
  - boundaries
  - regions
  - whole images
- Some features are applicable to more than one category.

# Sensitivity

---

- Feature selected as descriptors should be **as insensitive as possible** to variations in
  - size
  - translation
  - rotation
- Following descriptors satisfy one or more of these properties.



# Boundary Processing

---

- Segmentation techniques yield raw data in the form of pixels along a boundary or pixels contained in a region.
- These data sometimes are used directly to obtain descriptors.
- It is a standard practice to use schemes that compact the segmented data into representations that facilitate the computation of descriptors.

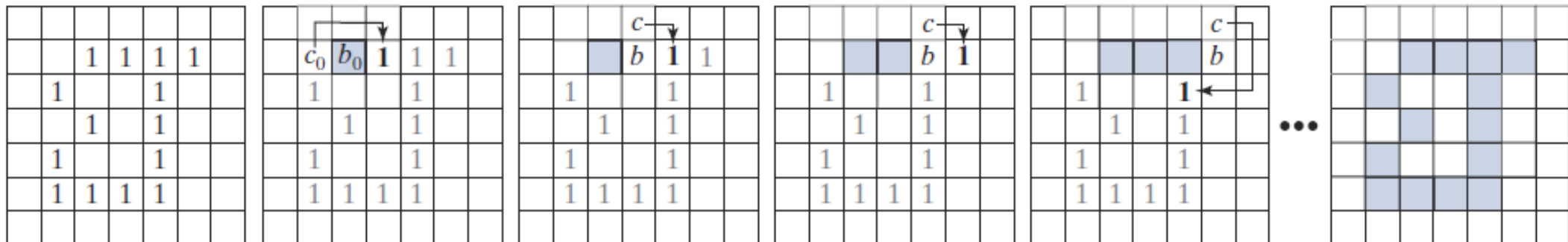
# Boundary Processing

---

- Several algorithms discussed in this lecture require that the points in the boundary of a region be ordered in a clockwise or counterclockwise direction.
- Consequently, we begin our discussion by introducing a boundary-following algorithm whose output is an ordered sequence of points.
- We assume (1) that we are working with binary images in which object and background points are labeled 1 and 0, respectively; and (2) that images are padded with a border of 0's to eliminate the possibility of an object merging with the image border.

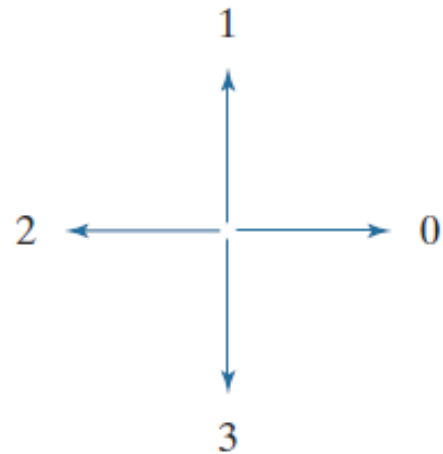
# Boundary Processing

- The following algorithm traces the boundary of a 1-valued region,  $R$ , in a binary image.
  - Let the starting point,  $b_0$ , be the *uppermost-leftmost* point in the boundary that is labeled 1. Denote by  $c_0$  the *west* neighbor of  $b_0$ . Examine the 8-neighbors of  $b_0$ , starting at  $c_0$  and proceeding in a clockwise direction. Let  $b_1$  denote the *first* neighbor encountered whose value is 1, and let  $c_1$  be the (background) point immediately preceding  $b_1$  in the sequence. Store the locations of  $b_0$  for use in Step 5.
  - Let  $b = b_0$  and  $c = c_0$ .
  - Let the 8-neighbors of  $b$ , starting at  $c$  and proceeding in a clockwise direction, be denoted by  $n_1, n_2, \dots, n_8$ . Find the first neighbor labeled 1 and denote it by  $n_k$ .
  - Let  $b = n_k$  and  $c = n_{k-1}$ .
  - Repeat Steps 3 and 4 until  $b = b_0$ . The sequence of  $b$  points found when the algorithm stops is the set of ordered boundary points.

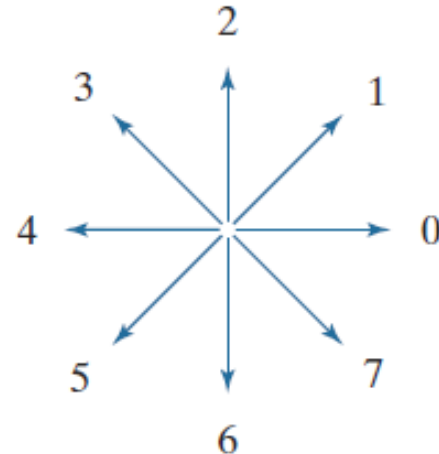


# Chain Codes

- Chain codes are used to represent a boundary by a connected sequence of straight line segments of specified length and direction.
- Typically, a chain code representation (a.k.a. *Freeman chain code*) is based on 4- or 8-connectivity of the segments. The direction of each segment is coded by using a numbering scheme.



4-directional chain code



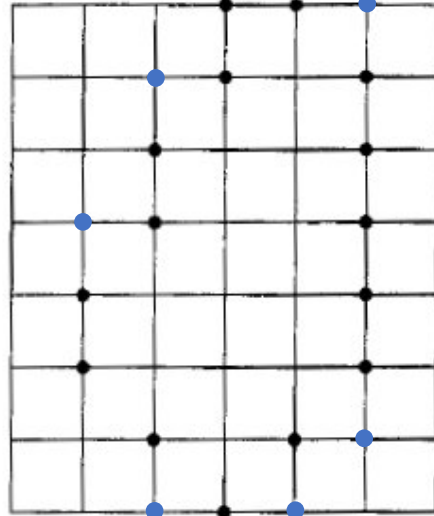
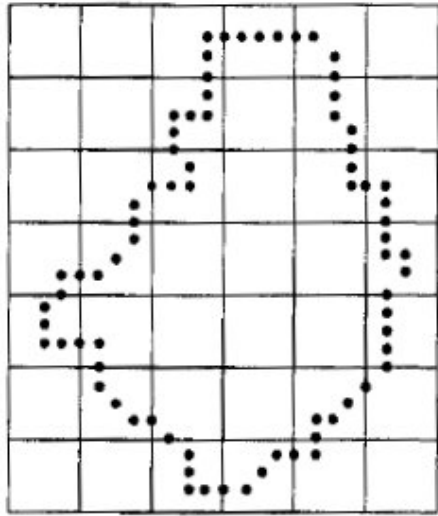
8-directional chain code

# Chain Codes

---

- But it is unacceptable in practice because
  - the resulting chain of codes tends to be quite long,
  - it is very sensitive to noise.
- Circumvent the problems by
  - Resample the boundary by selecting a larger grid spacing.
  - Then, as the boundary is traversed, a boundary point is assigned to each node of the large grid, depending on the proximity of the original boundary to that node.

# Chain Codes



a b  
c d

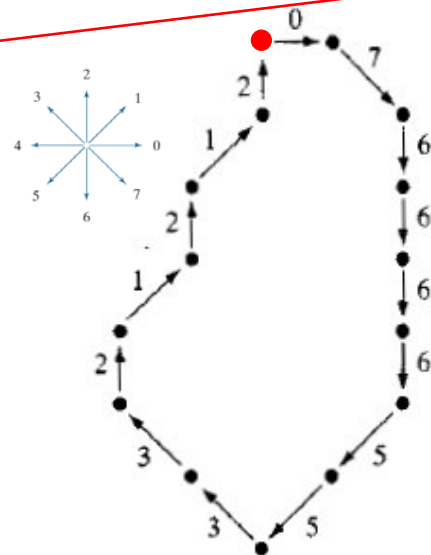
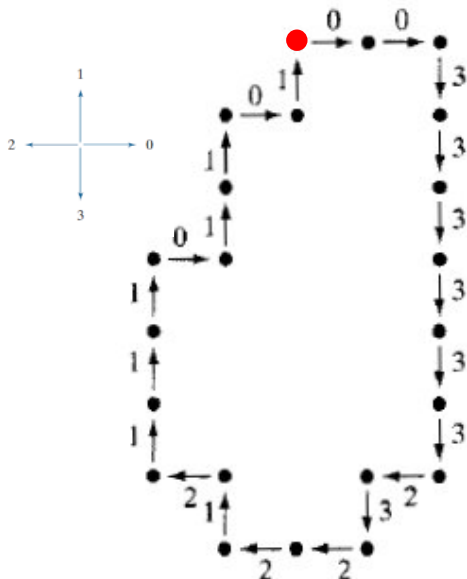
(a) Digital boundary with resampling grid superimposed.

(b) Result of resampling (black dots for 8-directional chain code, both black and blue dots for 4-directional chain code).

(c) 4-directional chain code.

(d) 8-directional chain code.

003333323221211101101

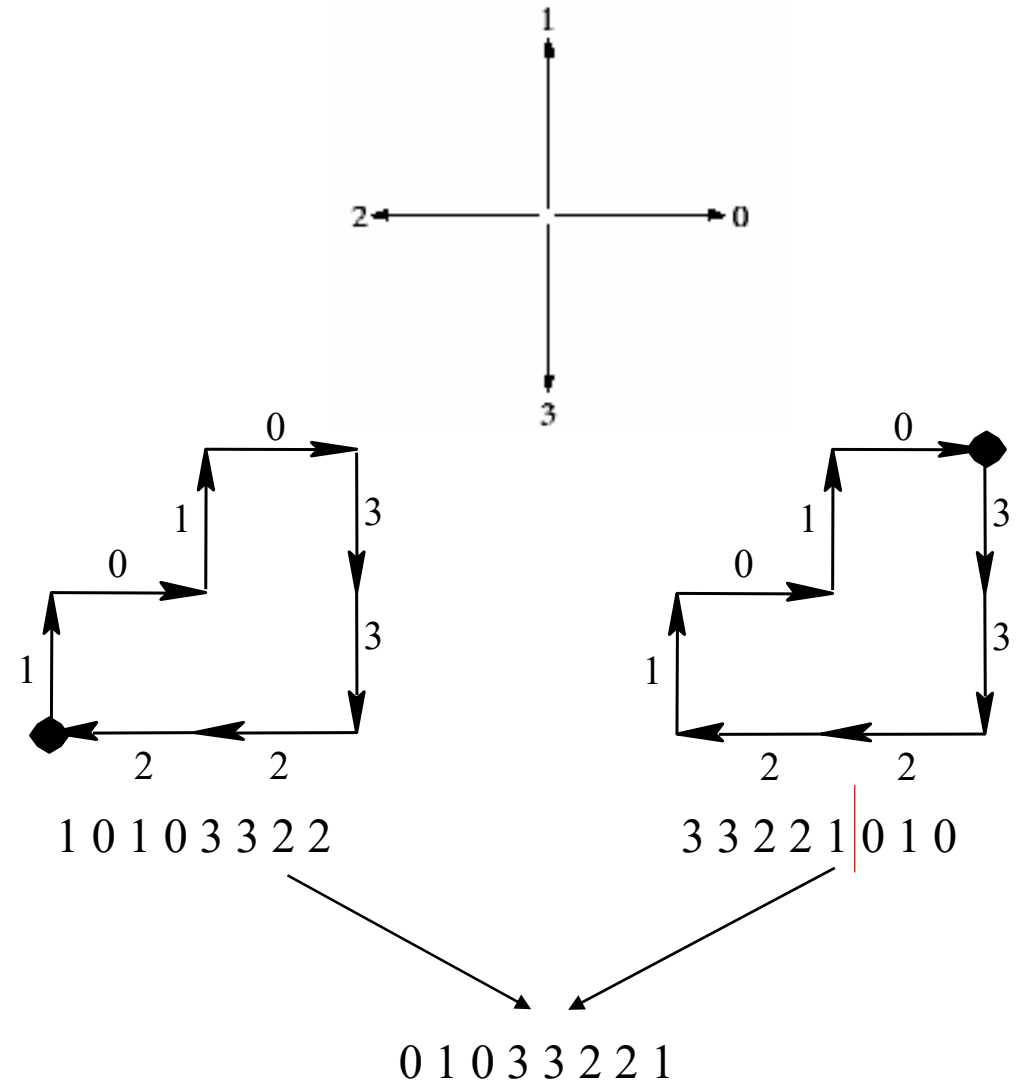


07666553321212

- However, different grids can generate different chain codes.
- The accuracy of the resulting code representation depends on the spacing of the sampling grid.

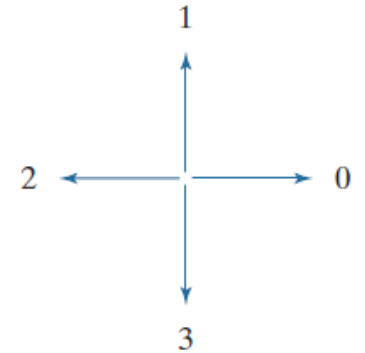
# Normalized Chain Codes

- The chain code of a boundary is dependent on the starting point.
- However, the code can be normalized with respect to the starting point by a straightforward procedure.
- We simply treat the chain code as a circular sequence of direction numbers and redefine the starting point so that the resulting sequence of numbers forms an integer of **minimum magnitude**.



# Normalized Chain Codes

- We can normalize also for rotation by using the *first difference* of the chain code instead of the code itself.
  - This difference is obtained by counting **the number of direction changes (in a counterclockwise direction)** that separate two adjacent elements of the code.
  - If we treat the code as a circular sequence to normalize it with respect to the starting point, then the first element of the difference is computed by using the transition between the last and first components of the chain.
- For instance, the first difference of the 4-directional chain code 10103322 is 33133030.





# Normalized Chain Codes

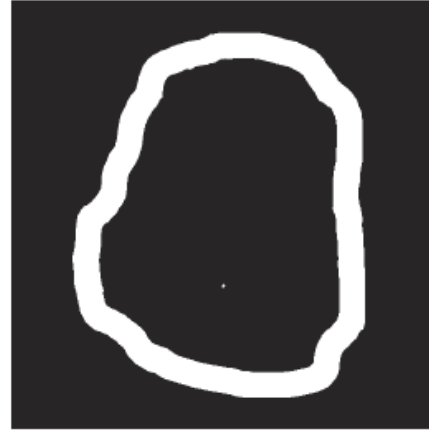
(a) Noisy image of size  $570 \times 570$  pixels.



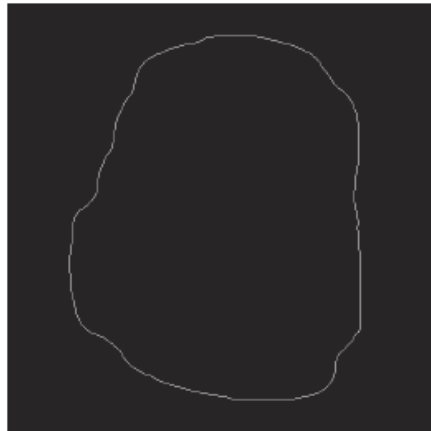
(b) Image smoothed with a  $9 \times 9$  box kernel.



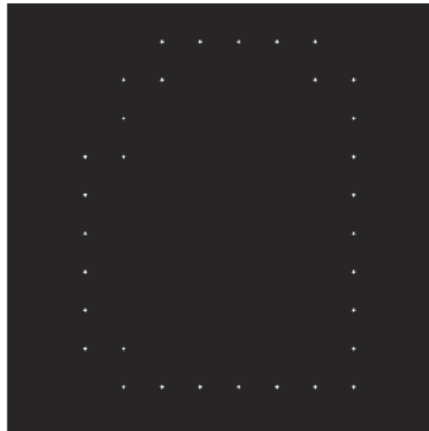
(c) Smoothed image, thresholded using Otsu's method.



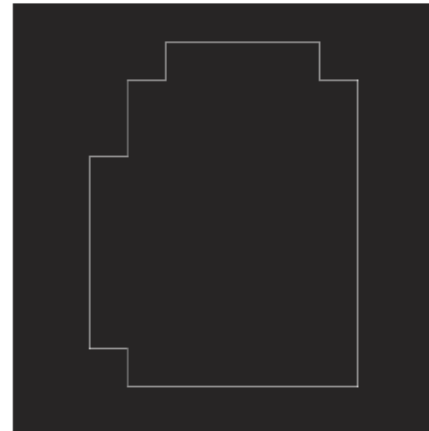
(d) Longest outer boundary of (c).



(e) Subsampled boundary (the points are shown enlarged for clarity).



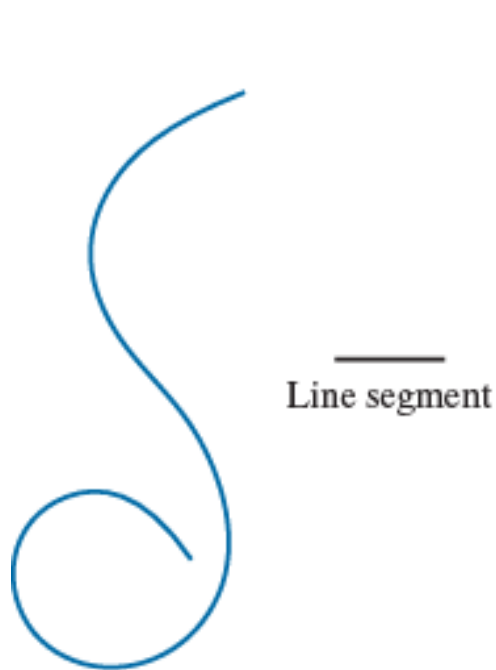
(f) Connected points from (e).



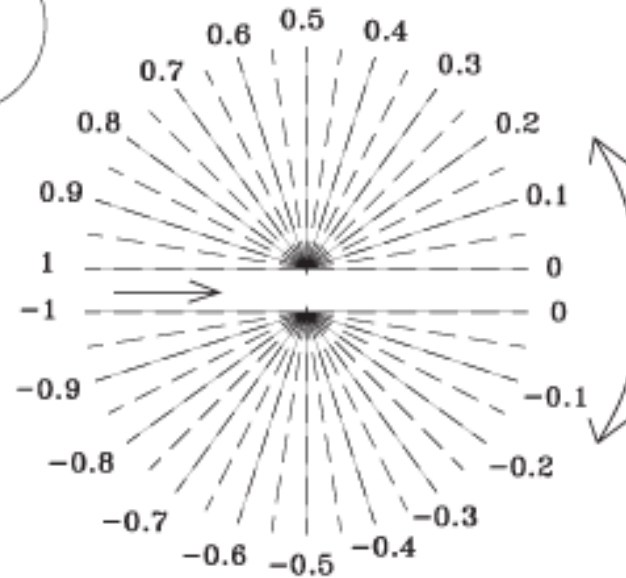
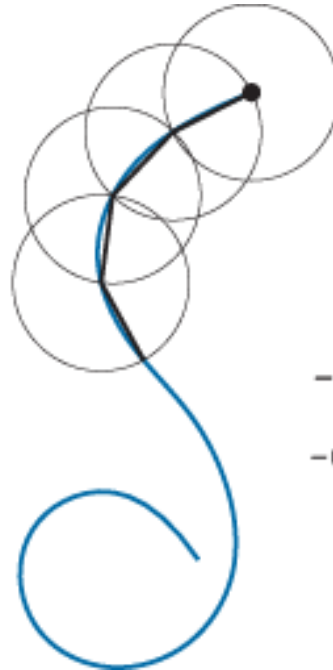
# Chain Code Smoothing

- Slope Chain Code (SCC)

Traversing the curve using circumferences to determine slope changes; the dot is the origin (starting point).

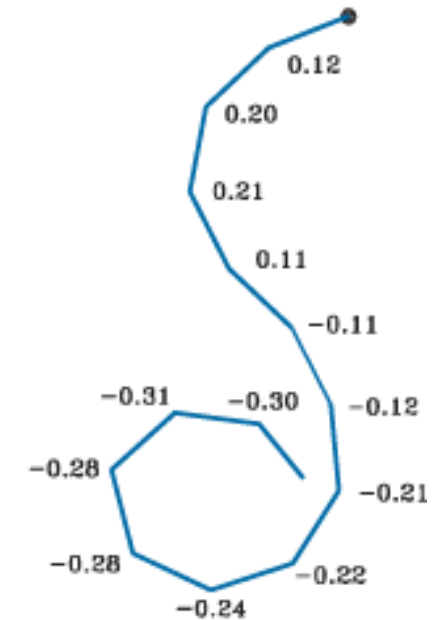


An open curve.



Range of slope changes in the open interval  $(-1, 1)$  (the arrow in the center of the chart indicates direction of travel). There can be ten subintervals between the slope numbers shown.

Resulting coded curve showing its corresponding numerical sequence of slope changes.

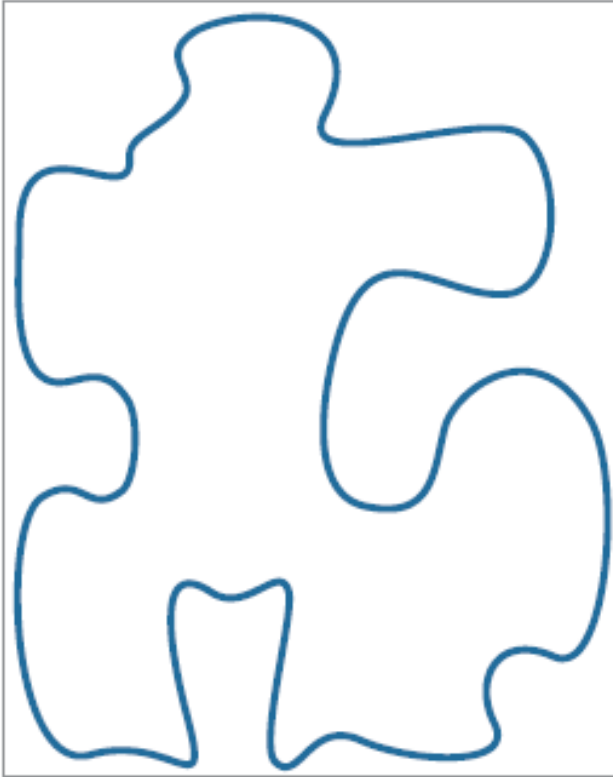


# Polygonal Approximations

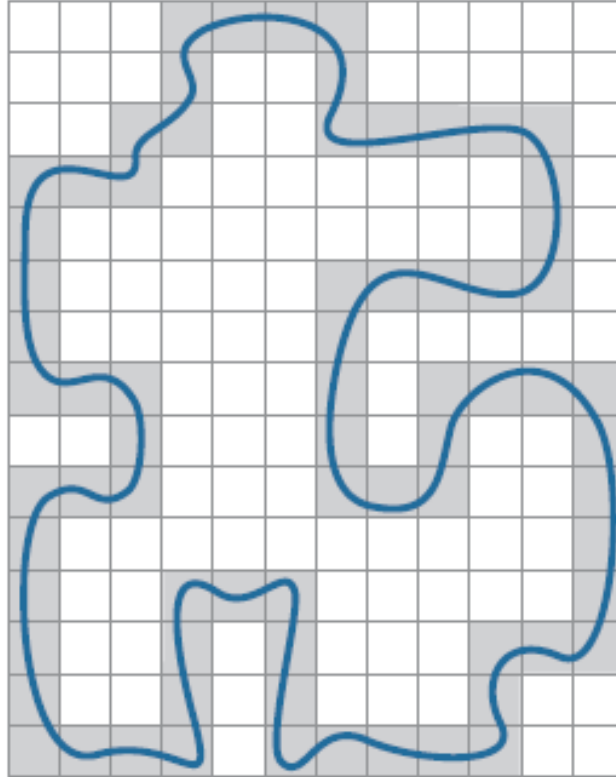
---

- Boundary can be approximated with arbitrary accuracy by a polygon.
- For a closed curve, the approximation is exact when the number of the segments in the polygon is equal to the number of points in the boundary.
- Goal: Try to capture the “essence” of the boundary shape with the fewest possible polygonal segments.
- Three methods:
  - Minimum perimeter polygons (MPP)
  - Merging techniques
  - Splitting techniques

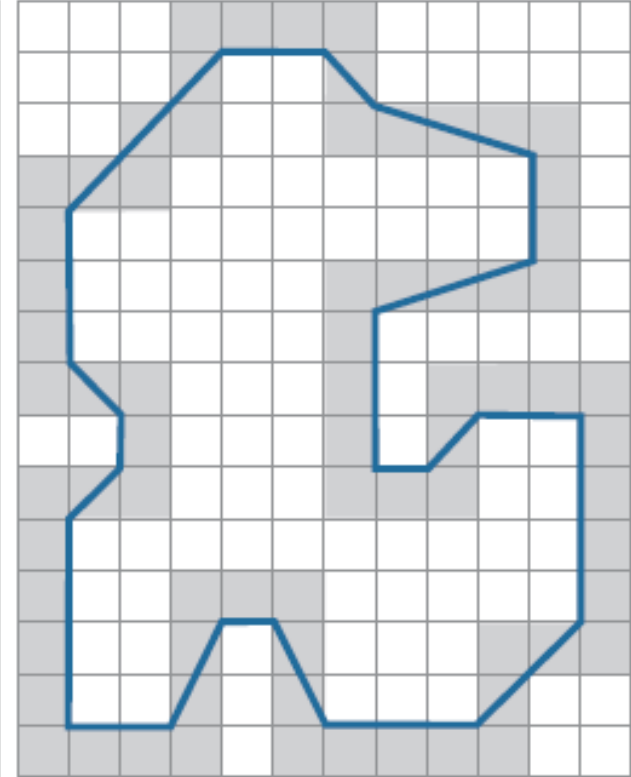
# Minimum Perimeter Polygons



An object boundary.

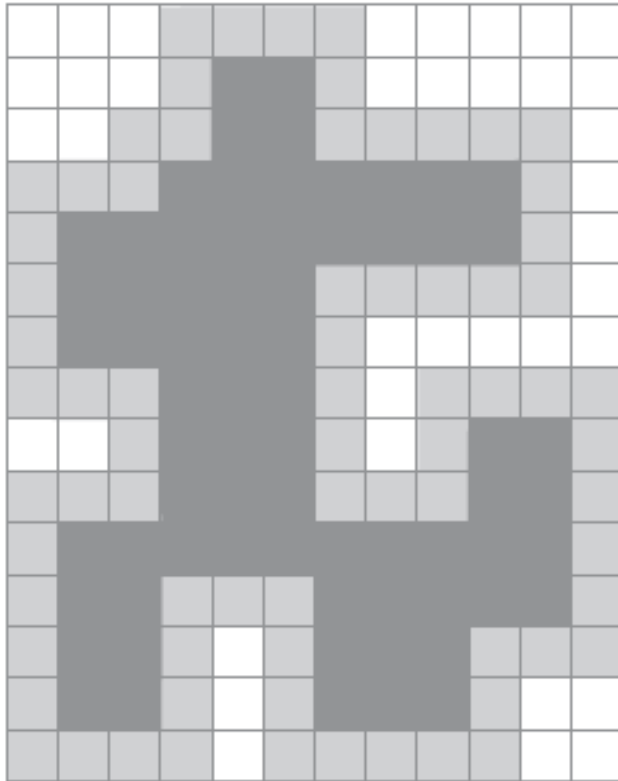


Boundary enclosed by cells (shaded).

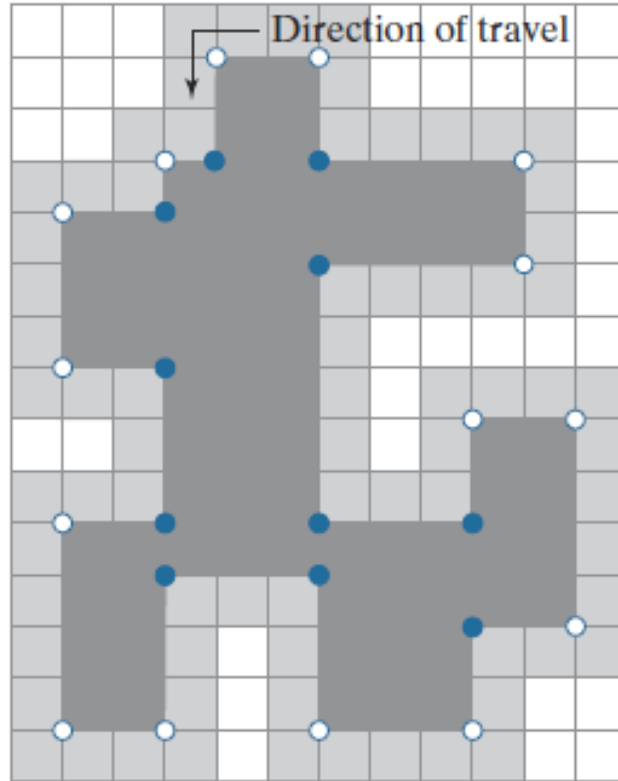


Minimum-perimeter polygon obtained by allowing the boundary to shrink. The vertices of the polygon are created by the corners of the inner and outer walls of the gray region.

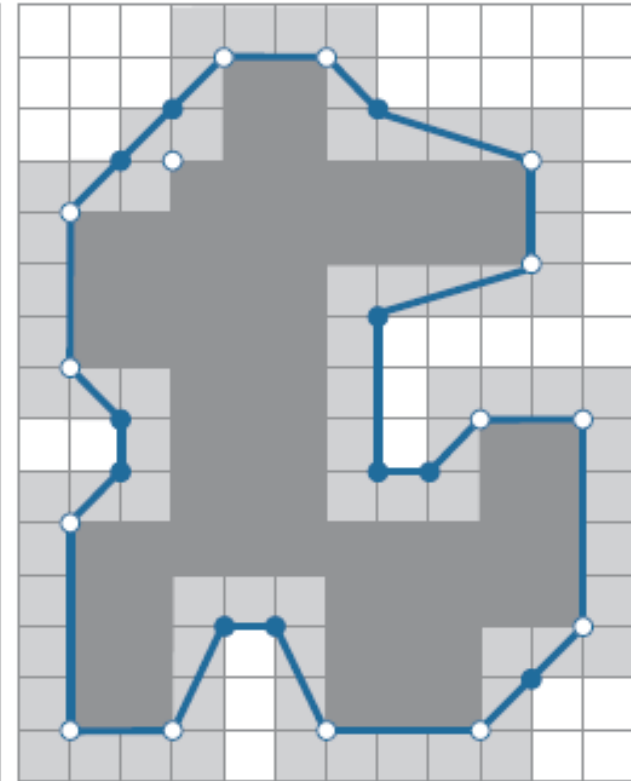
# Minimum Perimeter Polygons



Region (dark gray) resulting from enclosing the original boundary by cells

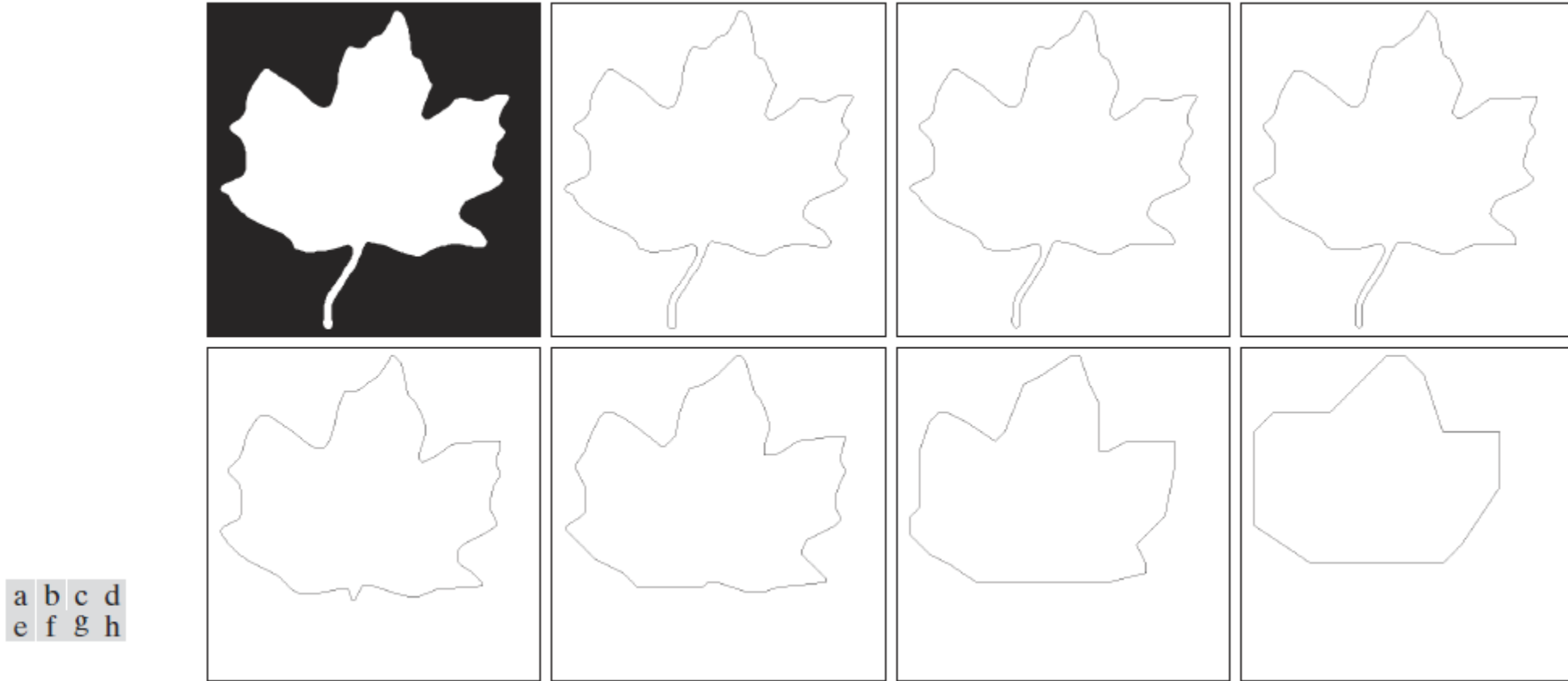


Convex (white dots) and concave (blue dots) vertices obtained by following the boundary of the dark gray region in the counterclockwise direction.



Concave vertices (blue dots) displaced to their diagonal mirror locations in the outer wall of the bounding region; the convex vertices are not changed. The MPP (solid boundary) is superimposed for reference.

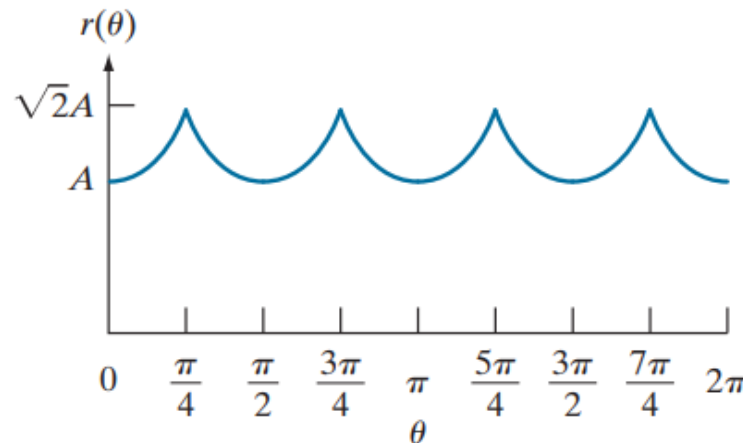
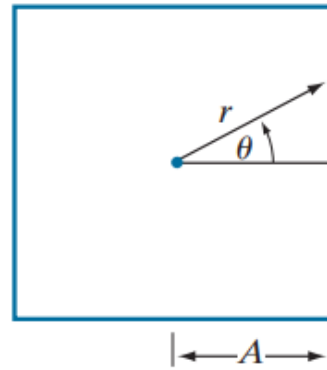
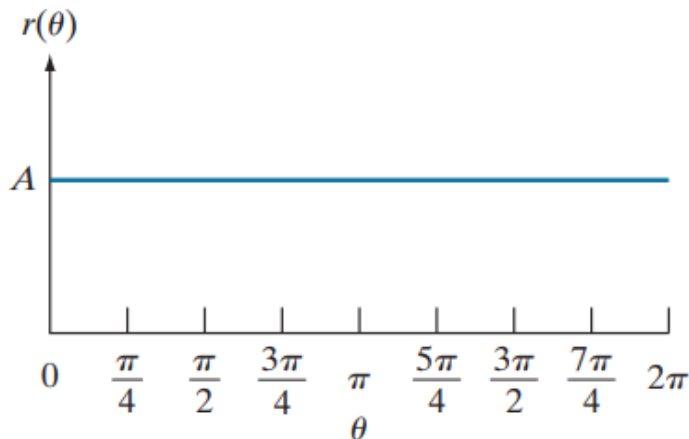
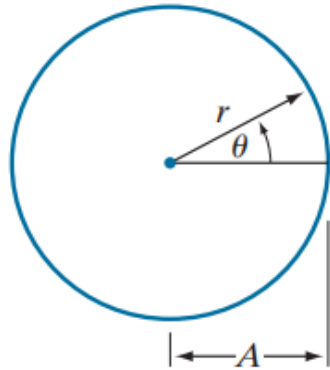
# Minimum Perimeter Polygons



(a)  $566 \times 566$  binary image. (b) 8-connected boundary. (c) through (h), MMPs obtained using square cells of sizes 2, 4, 6, 8, 16, and 32, respectively (the vertices were joined by straight-line segments for display). The number of boundary points in (b) is 1900. The numbers of vertices in (c) through (h) are 206, 127, 92, 66, 32, and 13, respectively. Images (b) through (h) are shown as negatives to make the boundaries easier to see.

# Signatures

- Map a 2D function to a 1D function.
- One of the simplest way is to plot the distance from the centroid (mean) to the boundary as a function of angle.



Distance-versus-angle signatures.

*Left:*  $r(\theta)$  is constant.

*Right:*, the signature consists of repetitions of the pattern  $r(\theta) = A \sec \theta$  for  $0 \leq \theta \leq \pi/4$ , and  $r(\theta) = A \csc \theta$  for  $\pi/4 < \theta \leq \pi/2$ .

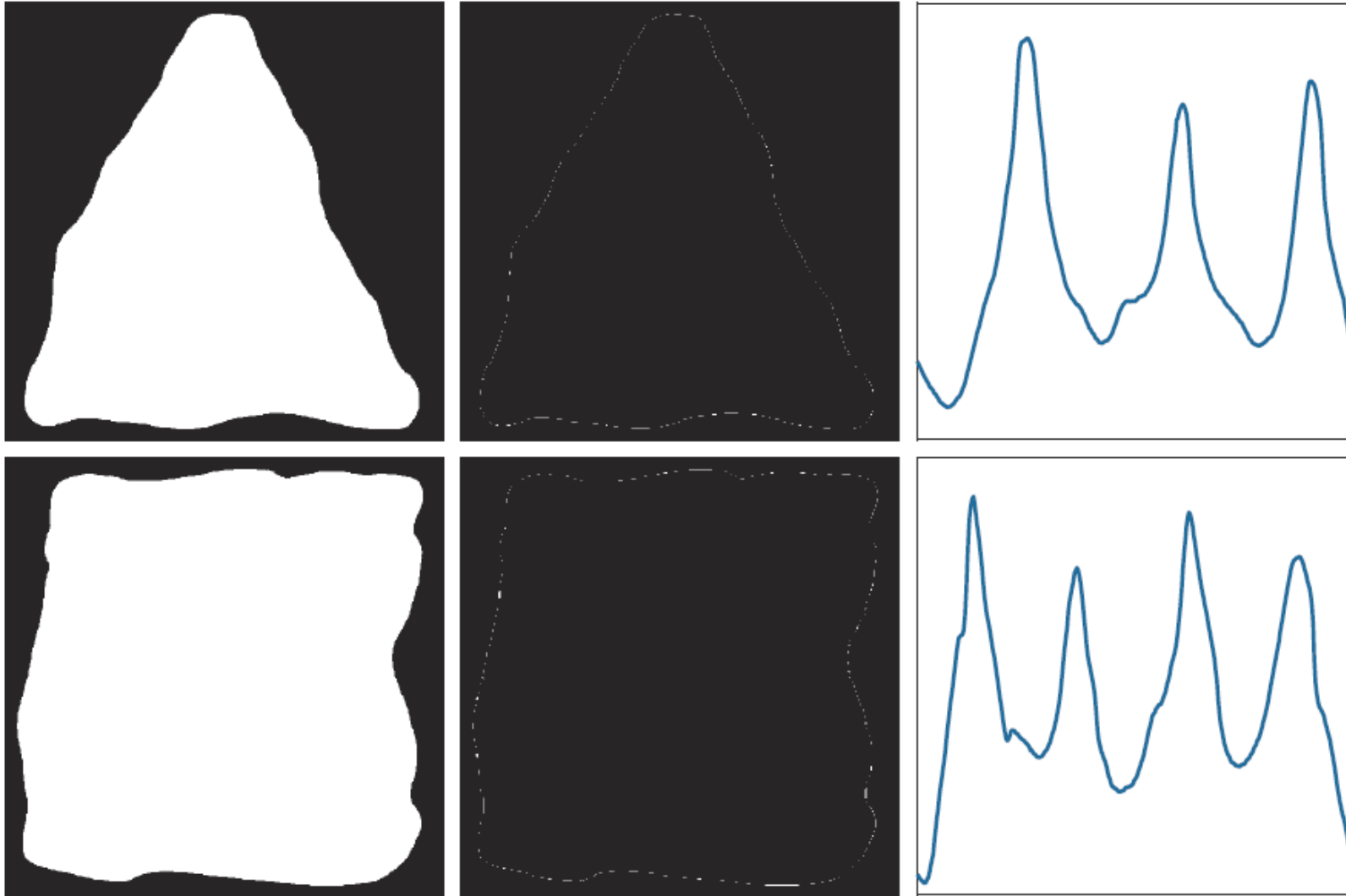
# Signatures

---

- Signatures are invariant to translation.
- Normalization with respect to rotation can be achieved by finding a way to select the same starting point.
- One way to do so is to select the starting point as the point farthest from the centroid, if the point happens to be unique and independent of rotational for each shape of interest.



# Signatures

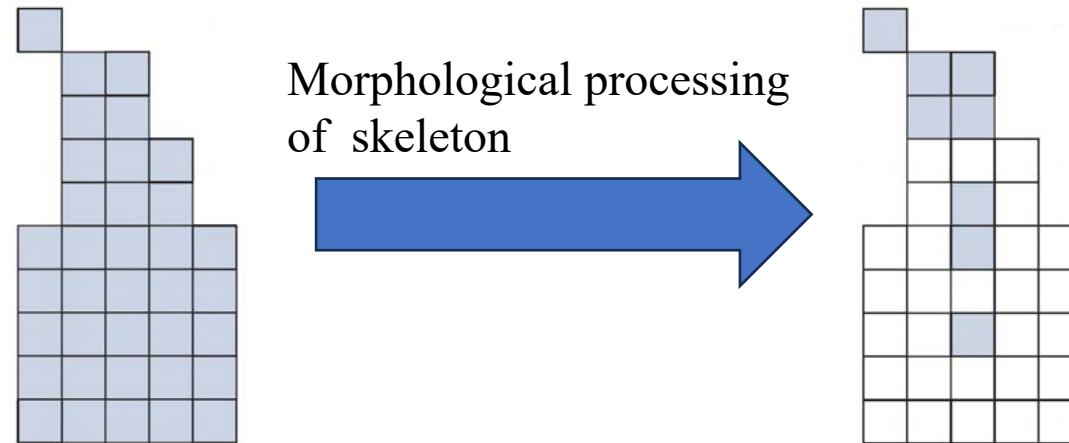


a	b	c
d	e	f

(a) and (d) Two binary regions, (b) and (e) their external boundaries, and (c) and (f) their corresponding  $r(\theta)$  signatures. The horizontal axes in (c) and (f) correspond to angles from 0° to 360°, in increments of 1°.

# Skeletons

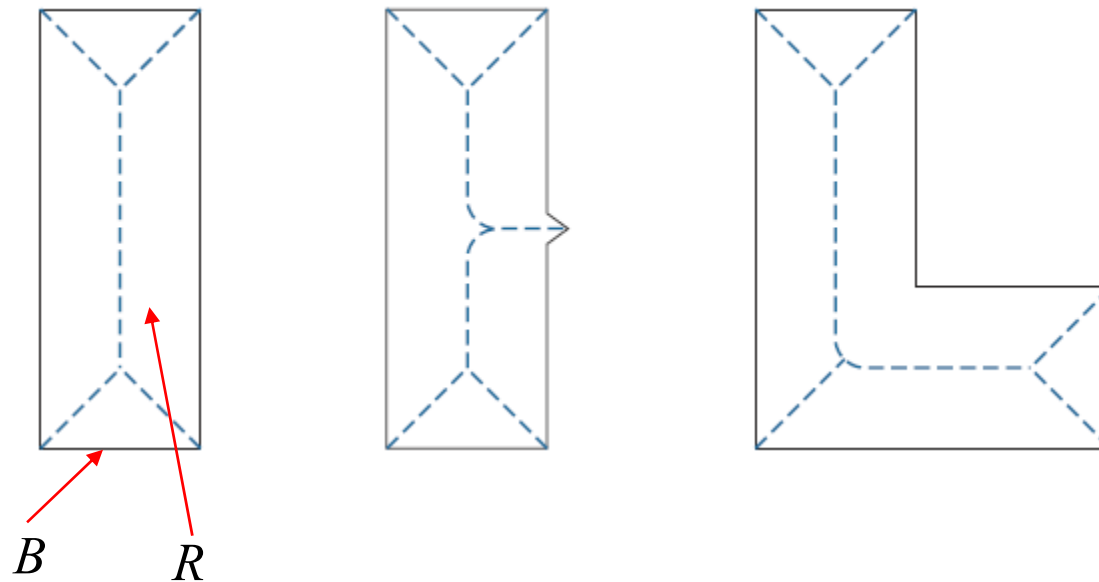
- We already discussed the basics of skeleton using morphology. However, as noted in that section, the procedure made no provision for keeping the skeleton connected.



- The algorithm developed here corrects that problem.

# Skeletons

- **Medial Axis Transformation (MAT):** For each point  $p$  in  $R$ , we find its closest neighbor in border  $B$ . If  $p$  has more than one such neighbor, it is said to belong to the medial axis (skeleton) of  $R$ .



Medial axes  
(dashed) of three  
simple regions.

# Skeletons

---

- In general, the MAT comes considerably closer than thinning to producing skeletons that “make sense”.
- However, computing the MAT of a region requires calculating the distance from every interior point to every point on the border of the region—an impractical endeavor in most applications.
- Instead, the approach is to obtain the skeleton equivalently from the *distance transform*, for which numerous efficient algorithms exist.

# Skeletons

- The **distance transform** of a region of foreground pixels in a background of zeros is the distance from every pixel to the nearest nonzero valued pixel.
- For the purpose of finding skeletons equivalent to the MAT, we are interested in the distance from the pixels of a region of foreground (white) pixels to their nearest background (zero) pixels, which constitute the region boundary. Thus, we compute the distance transform of the complement of the image.

0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0

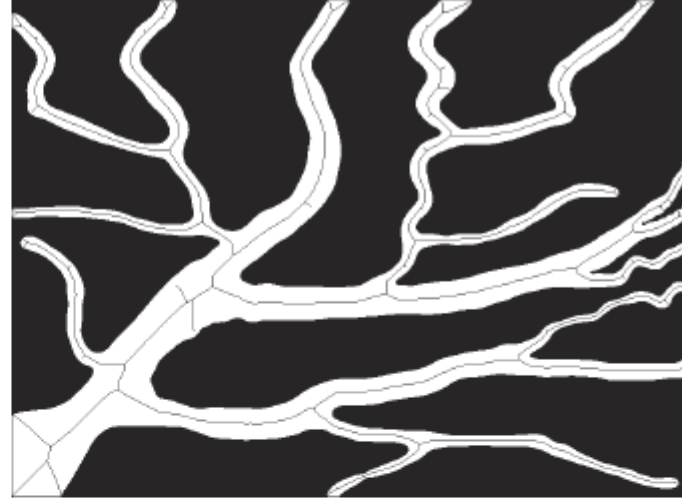
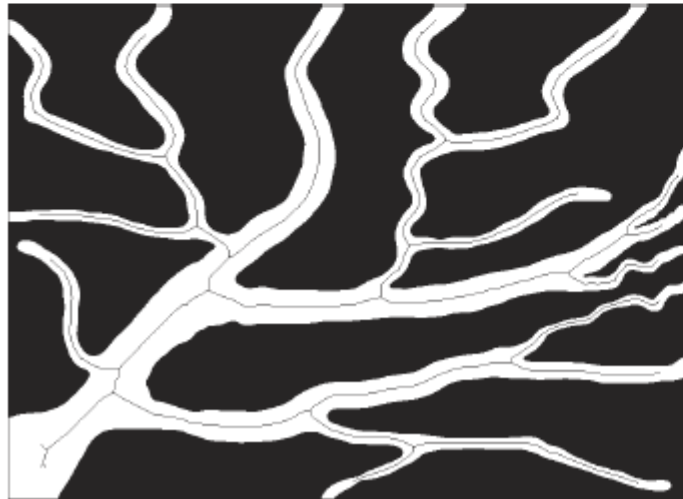
0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0
0	1	2	2	2	2	2	1	0
0	1	2	3	3	3	2	1	0
0	1	2	2	2	2	2	1	0
0	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0

# Skeletons

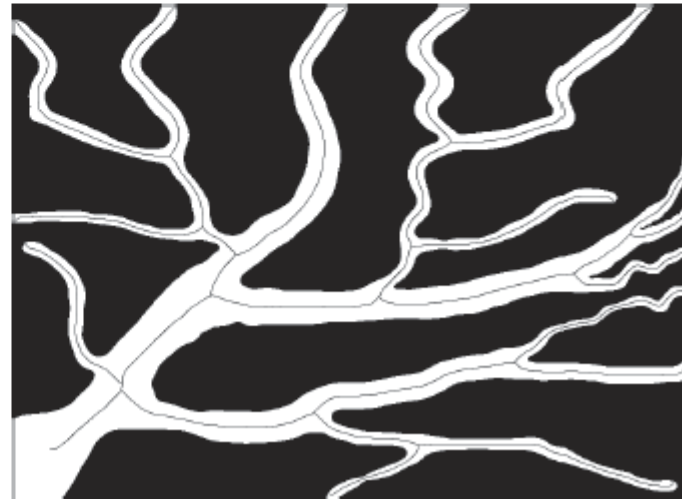
Thresholded image  
of blood vessels.



Result of 40 passes  
of spur removal.



Skeleton obtained by  
thinning, shown  
superimposed  
on the image (note the  
spurs).



Skeleton obtained  
using the distance  
transform.

# Boundary Descriptors

---

- Length
- Diameter
- Major axis and minor axis
- Eccentricity
- Curvature

# Direction

---

- *Length* of a boundary can be approximated by the number of pixels along the boundary.
  - If the boundary is represented by a polygonal curve, the length is equal to the sum of the lengths of the polygonal segments.

- *Diameter* of a boundary  $B$  is defined as

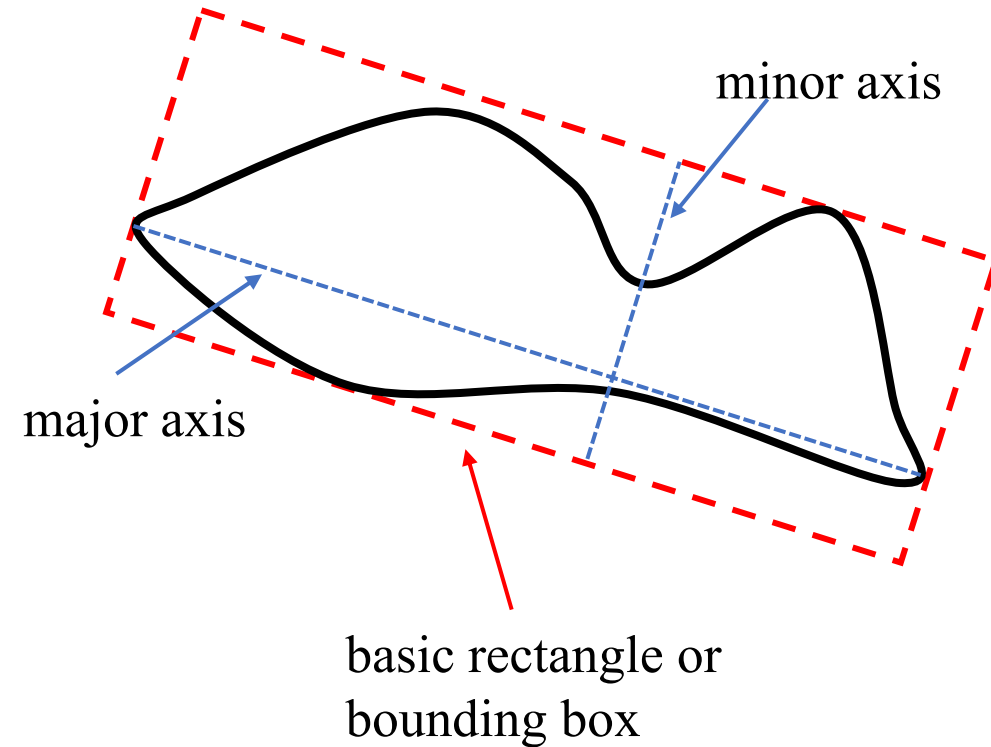
$$diameter(B) = \max_{i,j} [D(p_i, p_j)]$$

where  $D$  is a distance measure,  $p_i$  and  $p_j$  are points on the boundary  $B$ .



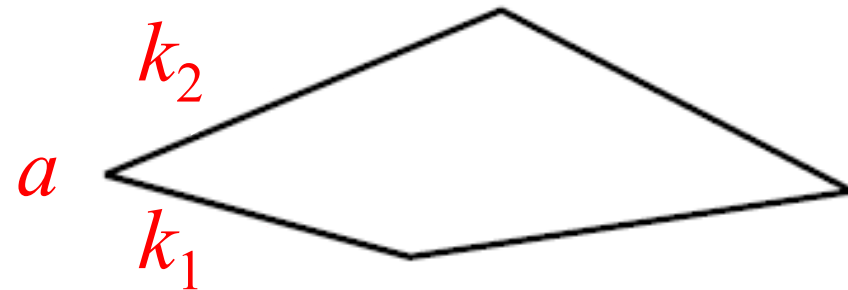
# Major Axis and Minor Axis

- *Major axis*: the line connecting the two extreme points that comprise the diameter.
- *Minor axis*: the line perpendicular to the major axis, and of such length that a box (*basic rectangle* or *bounding box*) passing through the outer four points of intersection of the boundary with the two axes completely encloses the boundary.
- *Eccentricity*: ratio of the major to the minor axis



# Curvature

- *Curvature*: the rate of change of slope.
  - Difficult to do as digital boundaries tend to be locally “ragged”.
  - Using the difference between the slopes of adjacent boundary segments (which are represented as straight lines).
  - Example:  $k_1$  and  $k_2$  are the slopes of two neighbor lines, and then the curvature at point  $a$  is  $k_1 - k_2$ .

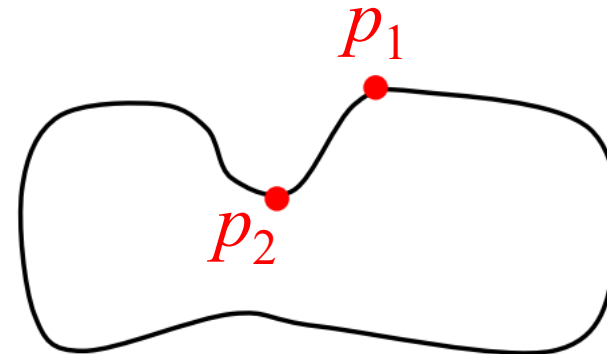


# Curvature

- As the boundary is traversed in the clockwise direction, a vertex point  $p$  is said to be the part of a **convex** if the change in slope at  $p$  is nonnegative. Otherwise,  $p$  is said to belong to a **concave**.
- Line (if the change is less than 10 degrees)
- Corner (if the changes exceeds 90 degrees)

Curvature of  $p_1 > 0$ : **Convex**

Curvature of  $p_2 < 0$ : **Concave**

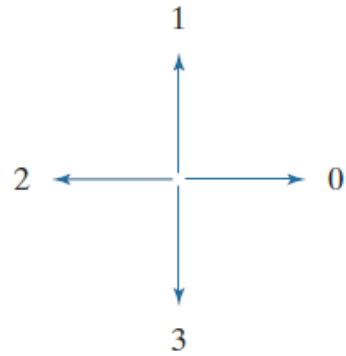


# Shape Numbers

---

- The shape number of a Freeman chain-coded boundary, based on the 4-directional code, is defined as **the first difference of smallest magnitude**.
- The *order*,  $n$ , of a shape number is defined as the number of digits in its representation.
- Moreover,  $n$  is even for a closed boundary, and its value limits the number of possible different shapes.

# Shape Numbers



Order 4



Chain code: 0 3 2 1

Difference: 3 3 3 3

Shape no.: 3 3 3 3

Order 6

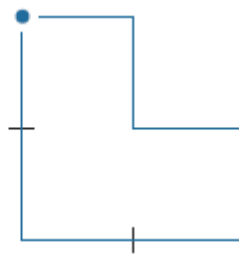
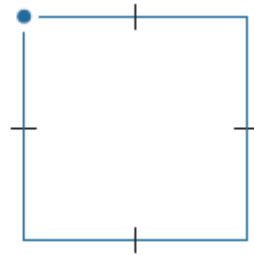


Chain code: 0 0 3 2 2 1

Difference: 3 0 3 3 0 3

Shape no.: 0 3 3 0 3 3

Order 8



Chain code: 0 0 3 3 2 2 1 1    0 3 0 3 2 2 1 1    0 0 0 3 2 2 2 1

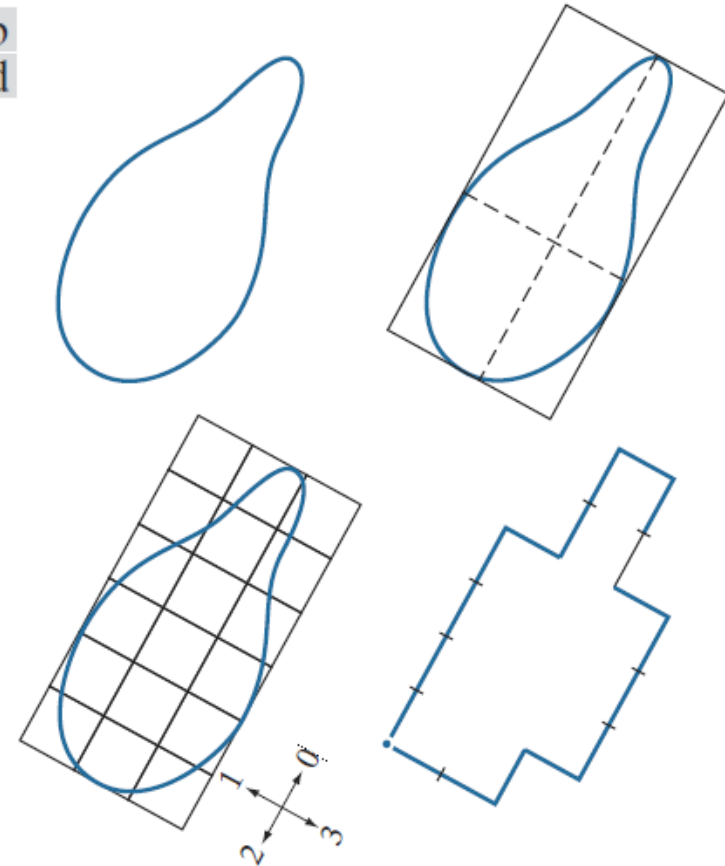
Difference: 3 0 3 0 3 0 3 0    3 3 1 3 3 0 3 0    3 0 0 3 3 0 0 3

Shape no.: 0 3 0 3 0 3 0 3    0 3 0 3 3 1 3 3    0 0 3 3 0 0 3 3

# Shape Numbers

- Suppose that  $n = 18$  is specified for the boundary in the left figure (a).
- To obtain a shape number of this order we first find the basic rectangle (b).
- Next we find the closest rectangle of order 18. It is a  $3 \times 6$  rectangle, requiring the subdivision of the basic rectangle shown in (c). The chain-code directions are aligned with the resulting grid.
- The final step is to obtain the chain code and use its first difference to compute the shape number, as shown in (d).

a	b
c	d



Chain code: 0 0 0 0 3 0 0 3 2 2 3 2 2 2 1 2 1 1

Difference: 3 0 0 0 3 1 0 3 3 0 1 3 0 0 3 1 3 0

Shape no.: 0 0 0 3 1 0 3 3 0 1 3 0 0 3 1 3 0 3

# Fourier Descriptors

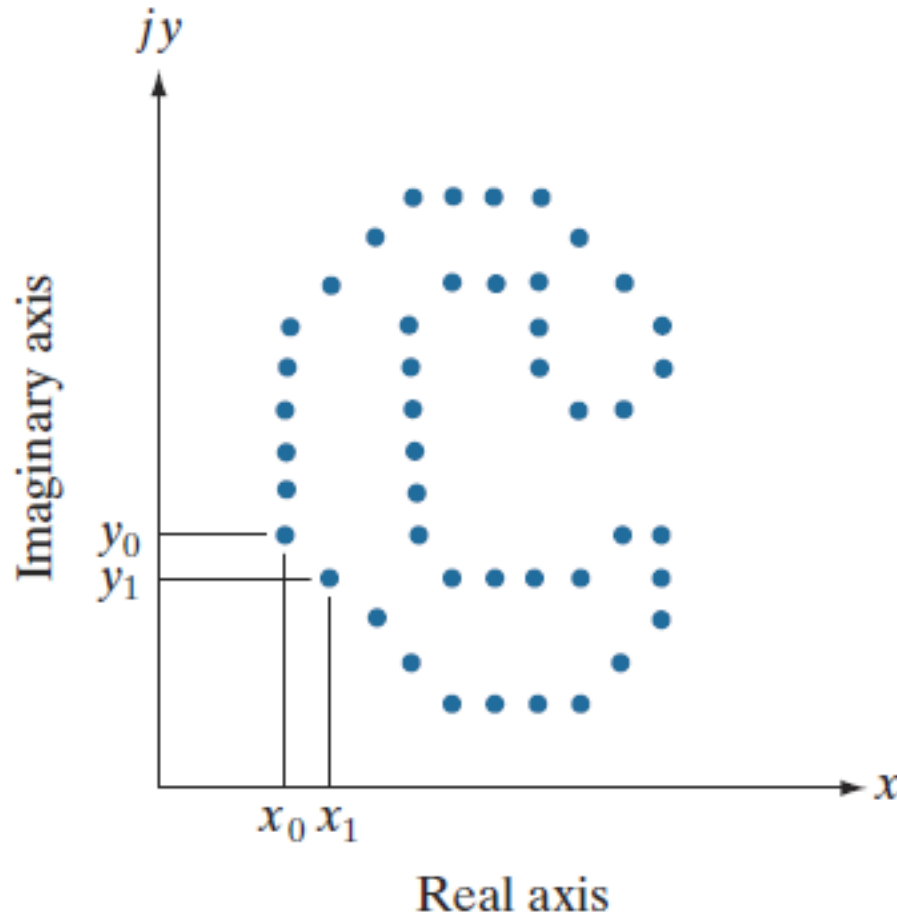
$$\text{boundary} = (x_0, y_0), \dots, (x_{K-1}, y_{K-1})$$

$$s(k) = [x(k), y(k)], \text{ for } k = 0, 1, \dots, K-1$$



$$s(k) = x(k) + jy(k), \text{ for } k = 0, 1, \dots, K-1$$

Map a 2D function to a 1D function



# Fourier Descriptors

---

- DFT of  $s(k)$  is

$$a(u) = \frac{1}{K} \sum_{k=0}^{K-1} s(k) e^{-j2\pi uk/K}, \text{ for } u = 0, 1, \dots, K-1$$

We call  $a(u)$  the  
Fourier descriptor

- IDFT of  $a(u)$  restores  $s(k)$

$$s(k) = \sum_{u=0}^{K-1} a(u) e^{j2\pi uk/K}, \text{ for } k = 0, 1, \dots, K-1$$

- $s(k)$  can be approximated by only the first  $P$  coefficients

$$\hat{s}(k) = \sum_{u=0}^P a(u) e^{j2\pi uk/K}, \text{ for } k = 0, 1, \dots, K-1$$



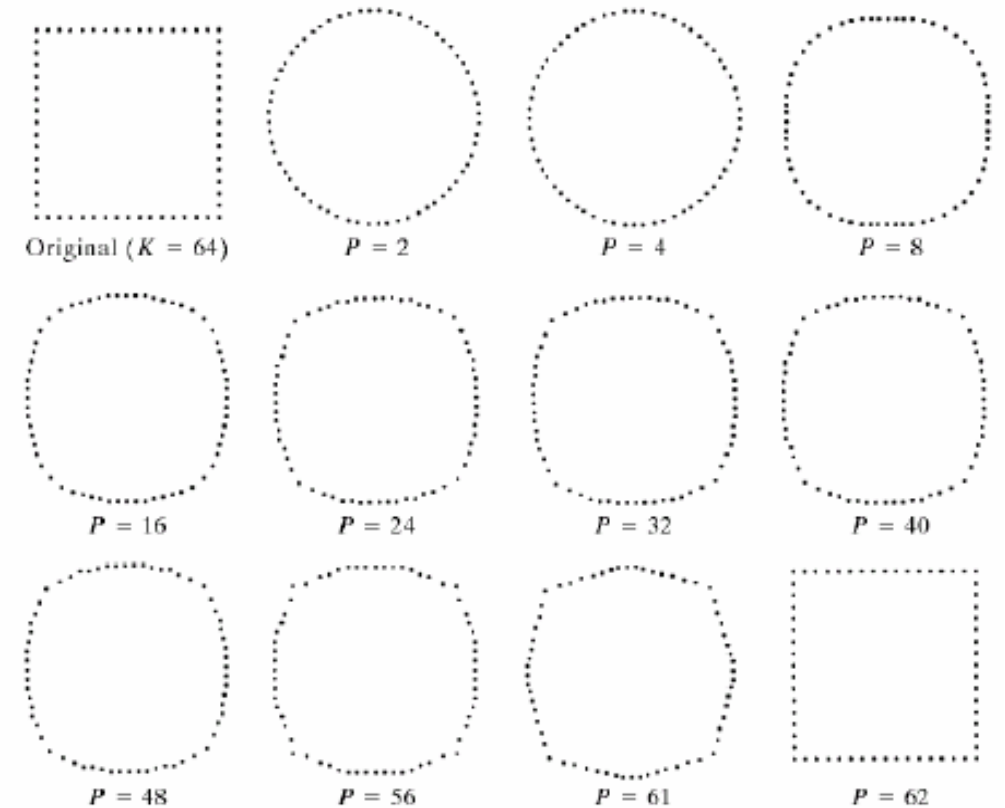
# Fourier Descriptors

---

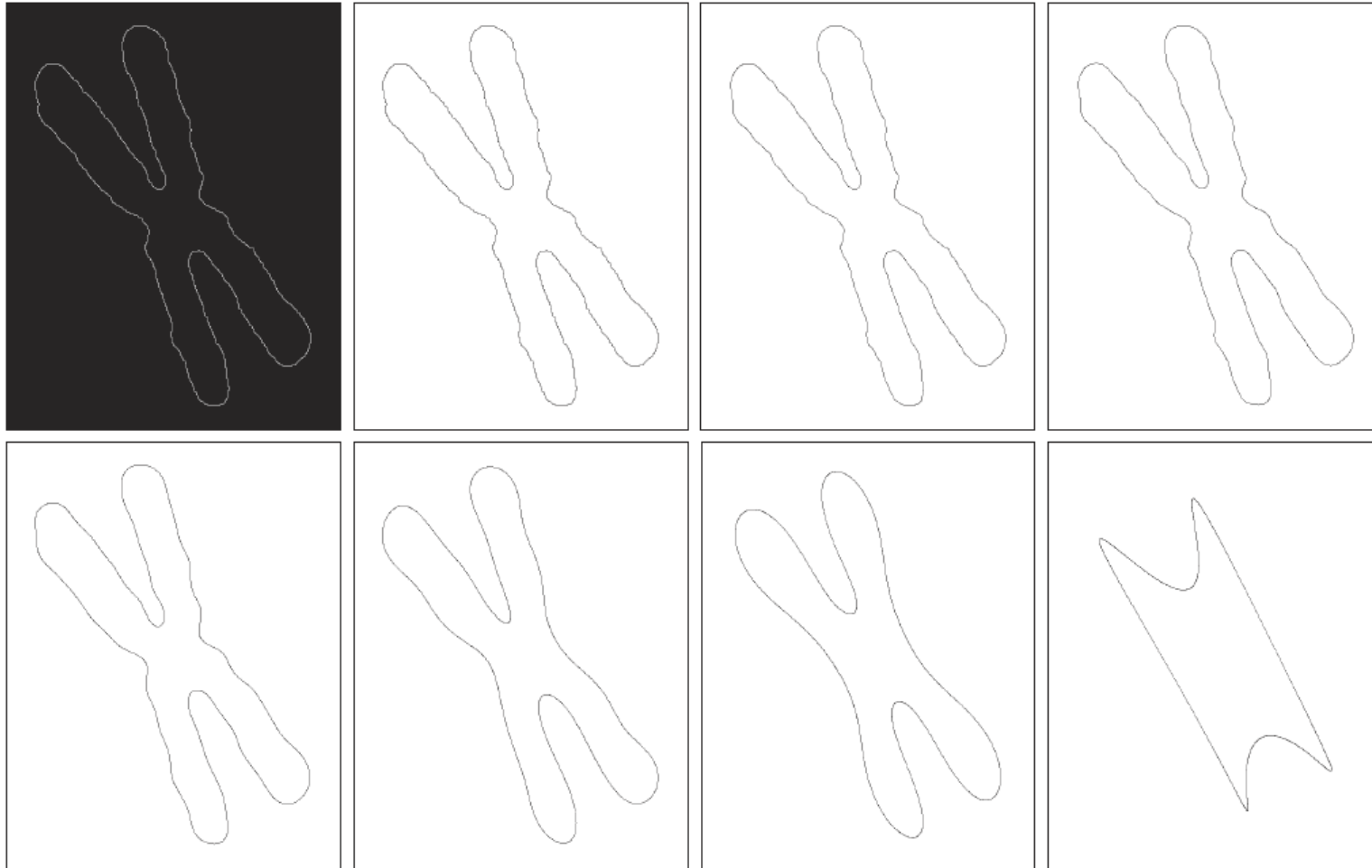
- The same number of points exists in the approximate boundary, but not as many terms are used in the reconstruction of each points.
- We know that high frequency components account for fine detail, and low frequency components determine global shape.
- Thus the smaller  $P$  becomes, the more detail that is lost on the boundary.

# Fourier Descriptors

- Low-order coefficients can reflect the general shape, high-order coefficients can precisely define the shape features.
- A few Fourier descriptors carry the shape information and can reflect the rough nature of the boundary.



# Fourier Descriptors



a	b	c	d
e	f	g	h

(a) Boundary of a human chromosome (2868 points).

(b)–(h) Boundaries reconstructed using 1434, 286, 144, 72, 36, 18, and 8 Fourier descriptors, respectively. These numbers are approximately 50%, 10%, 5%, 2.5%, 1.25%, 0.63%, and 0.28% of 2868, respectively

# Regional Descriptors

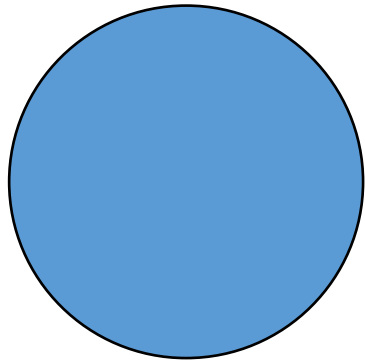
---

- Some simple regional descriptors
- Topological descriptors
- Texture
- Moments of two-dimensional function

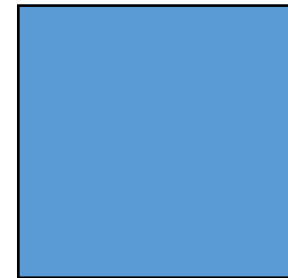
# Simple Regional Descriptors

---

- Area ( $A$ ): the number of pixels in the region
- Perimeter ( $p$ ): length of its boundary
- Compactness:  $\text{compactness} = p^2 / A$



$$\text{Compactness} = (2\pi r)^2 / (\pi r^2) = 4\pi$$

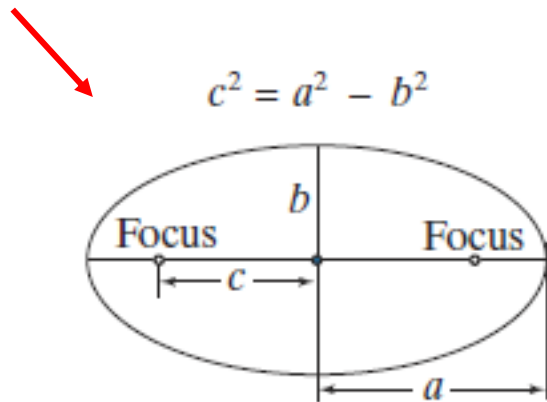


$$\text{Compactness} = (4a)^2 / a^2 = 16$$

# Simple Regional Descriptors

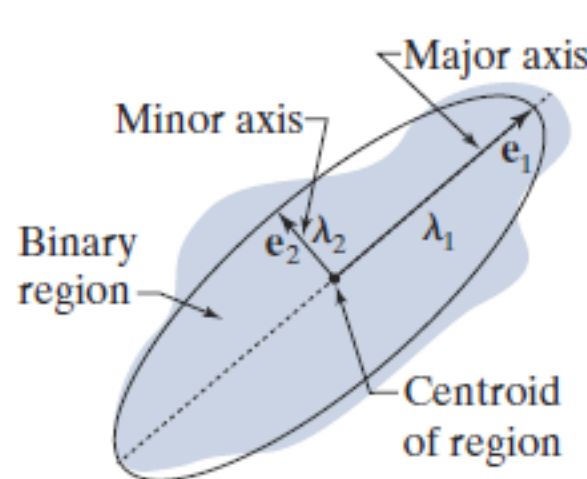
- Circularity (also called roundness):  $\text{circularity} = 4\pi A / p^2$
- Effective diameter:  $d_e = 2\sqrt{A / \pi}$
- Eccentricity:

$$\text{eccentricity} = \frac{c}{a} = \frac{\sqrt{a^2 - b^2}}{a}, a \geq b$$



An ellipse in standard form.





$$\text{eccentricity} = \frac{\sqrt{\lambda_2^2 - \lambda_1^2}}{\lambda_2}, \lambda_2 \geq \lambda_1$$

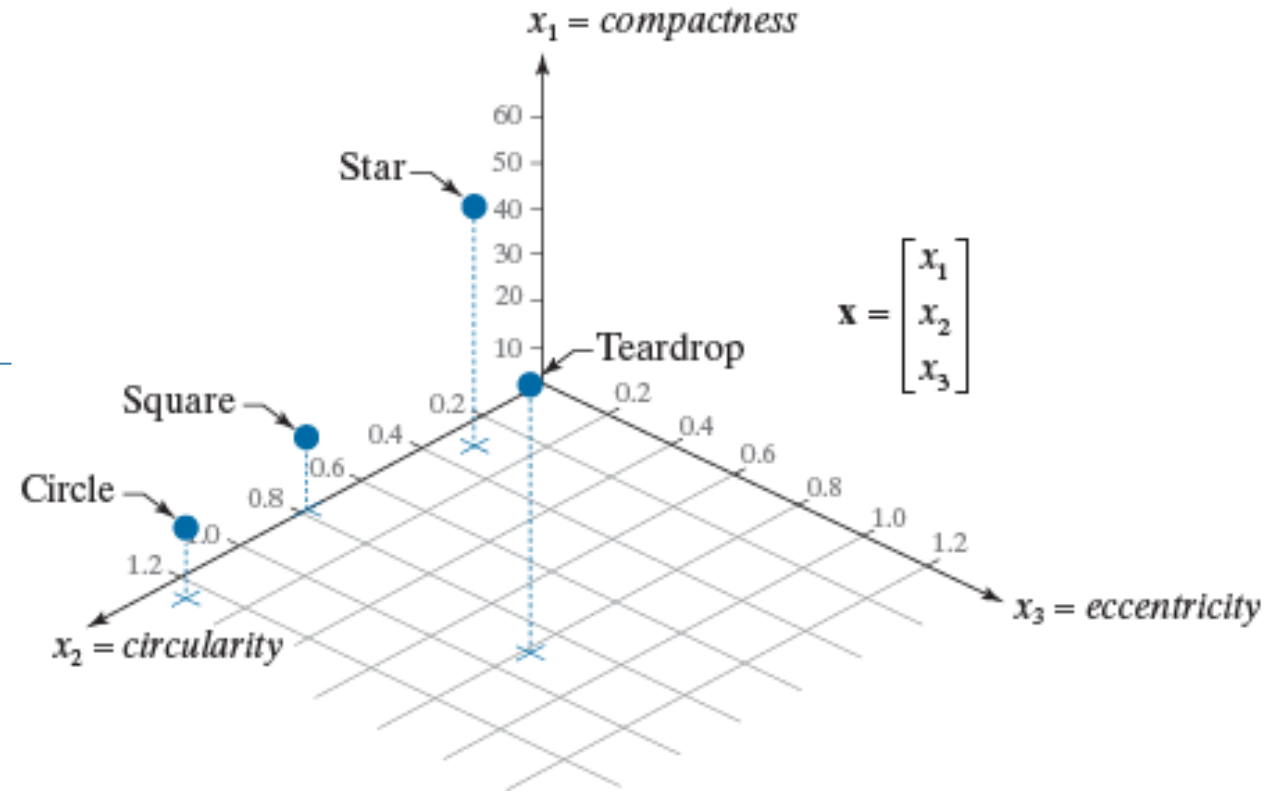


$e_1$   $\lambda_1$  and  $e_2$   $\lambda_2$  are the eigenvectors and corresponding eigenvalues of the covariance matrix of the coordinates of the region

An ellipse approximating a region in arbitrary orientation.

# Simple Regional Descriptors

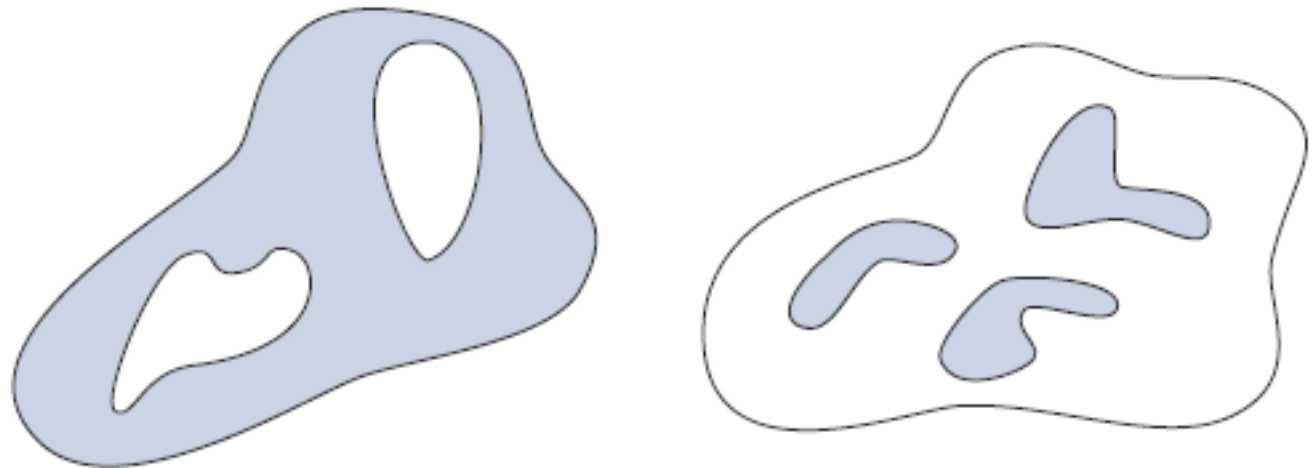
Descriptor				
<i>Compactness</i>	10.1701	42.2442	15.9836	13.2308
<i>Circularity</i>	1.2356	0.2975	0.7862	0.9478
<i>Eccentricity</i>	0.0411	0.0636	0	0.8117



# Topological Descriptors

- Topological properties are useful for global description of regions in the image plane.
- Topology is the study of properties of an image that are unaffected by any deformation, as long as there is no tearing or joining of the figure.
- Topological descriptors can be defined by the number of holes and connected components in the region.

- (left) A region with two holes.
- (right) A region with three connected components.





# Topological Descriptors

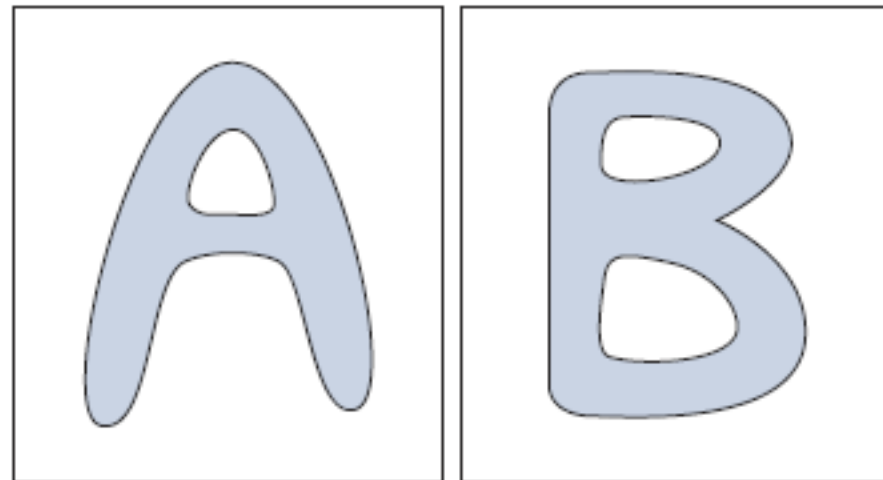
---

- Euler number  $E$ :  $E = C - H$ ,

where  $C$  is the number of connected components,

$H$  is the number of holes.

Regions with Euler numbers equal to 0 (“A”) and  $-1$  (“B”), respectively.



# Topological Descriptors

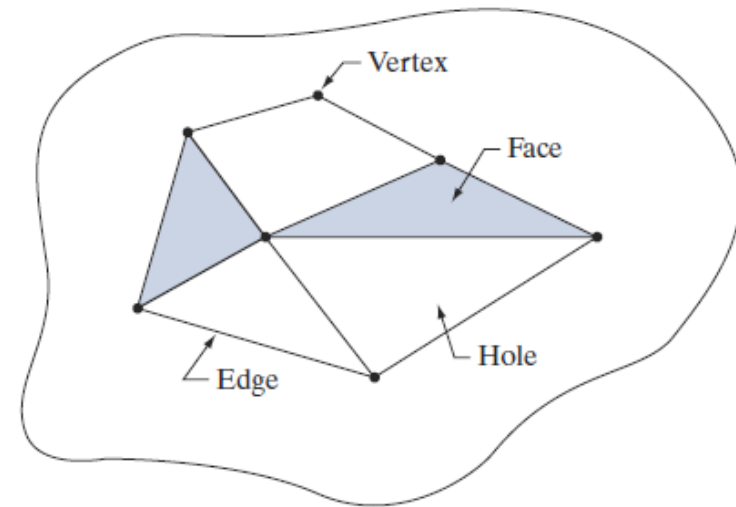
- Denoting the number of vertices by  $V$ , the number of edges by  $Q$ , and the number of faces by  $F$  gives the following relationship, called the **Euler formula**:

$$V - Q + F = C - H$$

which can be expressed as

$$V - Q + F = E$$

The network has 7 vertices, 11 edges, 2 faces, 1 connected region, and 3 holes, thus the Euler number is  $-2$ .

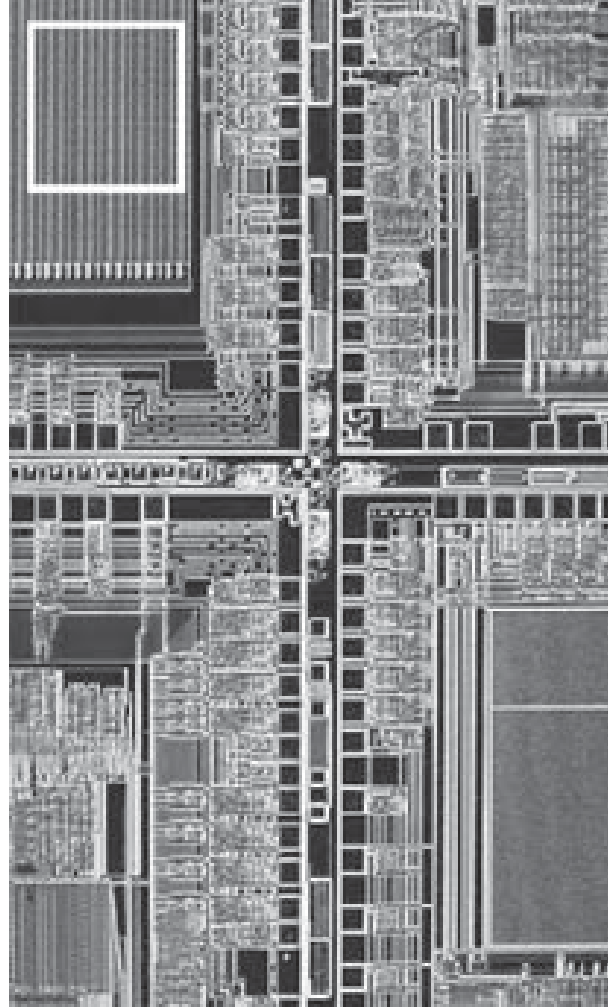
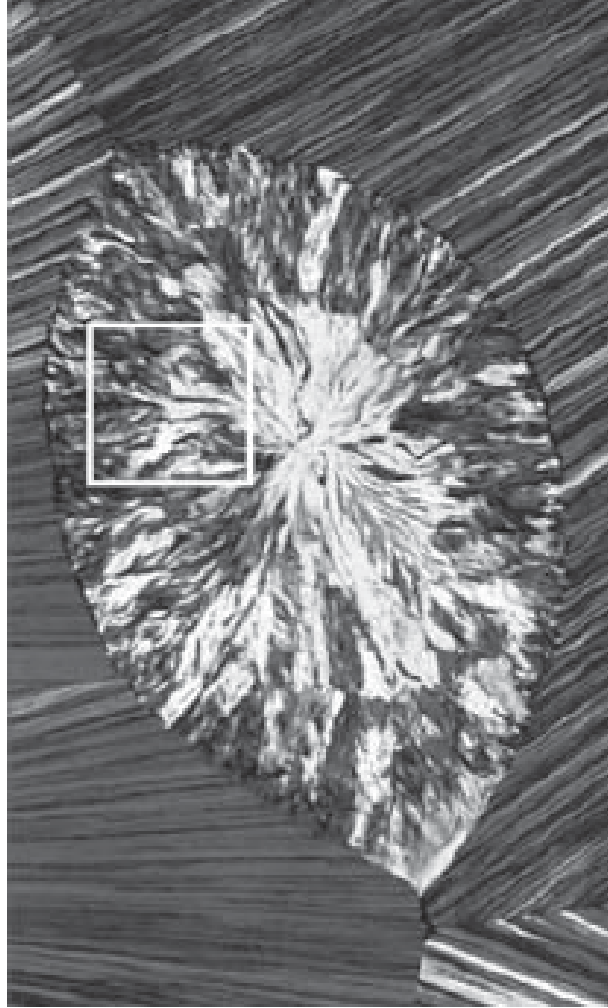


# Texture Descriptors

---

- An important approach to region description is to quantify its texture.
- Now no formal definition of texture exists.
- Texture is a set of primitive *patterns* in some regular or repeated relationship, or have similar properties such as constant color, gray level, average intensity, etc...
- Coarse textures are built from larger primitive patterns; Fine textures from smaller primitive patterns.
- Texture descriptor provides measures of properties such as **smoothness, coarseness, and regularity.**

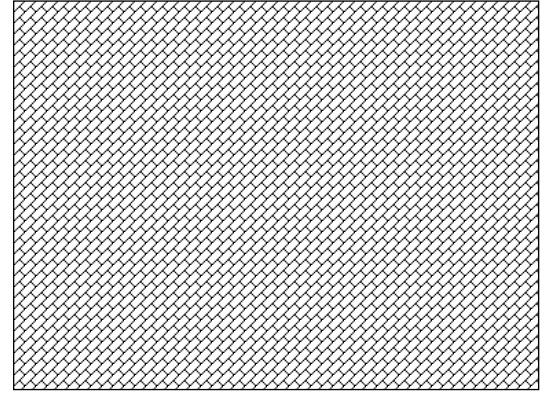
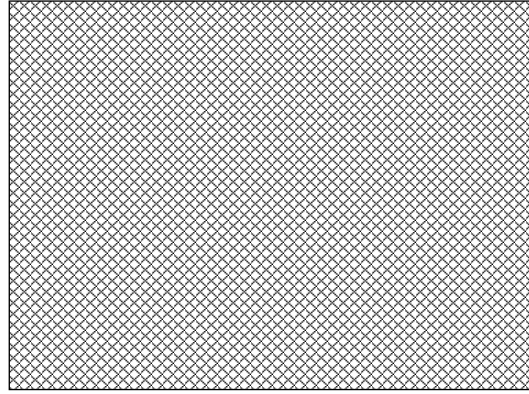
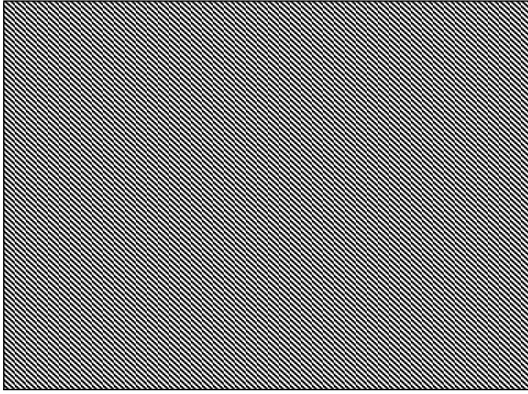
# Texture



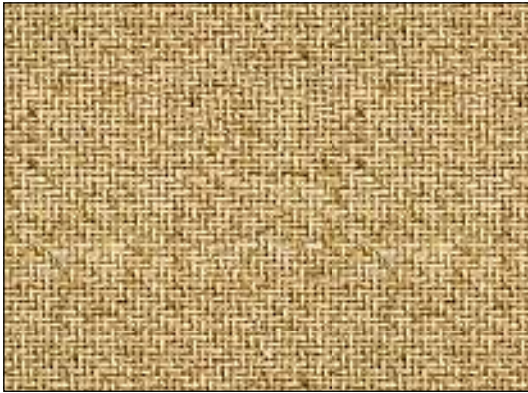
The white squares mark, from left to right, smooth, coarse, and regular textures. These are optical microscope images of a superconductor, human cholesterol, and a microprocessor.

# Texture

---



Artificial texture



Nature texture

# Texture

---

- There are three principal approaches used in texture analysis:
  - Statistical approaches
    - Moments of the gray level histogram
    - Gray level co-occurrence matrix
  - Structural approaches
- *Statistical*: convenient for natural textures.
- *Structural*: convenient for artificial textures.

# Texture

- Statistical approaches: Let  $z$  be a random variable denoting gray levels and let  $p(z_i)$ ,  $i = 1, 2, \dots, L - 1$ , be the corresponding histogram. The  $n^{th}$  moment of  $z$  about the mean is

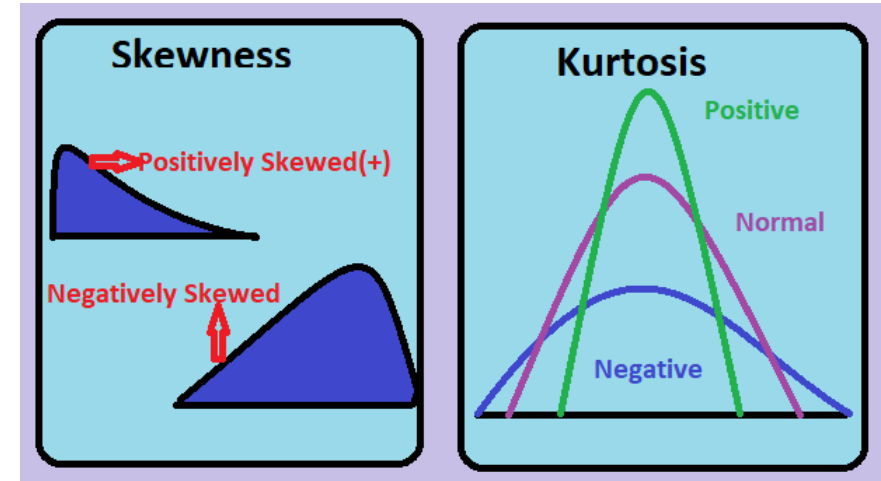
$$\mu_n(z) = \sum_{i=0}^{L-1} (z_i - m)^n p(z_i), \quad \text{where the mean is } m = \sum_{i=0}^{L-1} z_i p(z_i)$$

$$\mu_0 = 1, \mu_1 = 0, \text{ and } \mu_2 = \sigma^2$$

- The measure of gray-level **contrast**:  $R = 1 - \frac{1}{1 + \sigma^2}$ 
  - $R$  is 0 for the area of constant intensity (the variance is zero here), and approaches 1 for large value of  $\sigma^2$ .

# Texture

- The third moment is a measure of the **skewness** of the histogram. The fourth moment is a measure of **relative flatness (kurtosis)**.

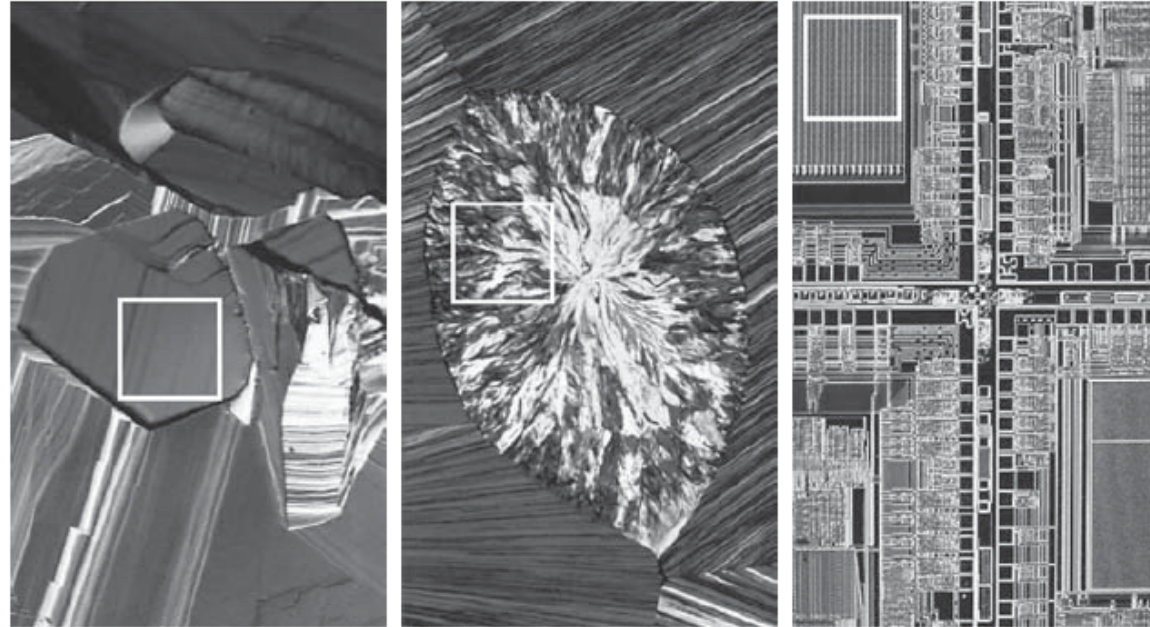


- Some additional texture measures based on histograms:

- uniformity 
$$U = \sum_{i=0}^{L-1} p^2(z_i)$$
- average entropy 
$$e = -\sum_{i=0}^{L-1} p(z_i) \log_2 p(z_i)$$



# Texture

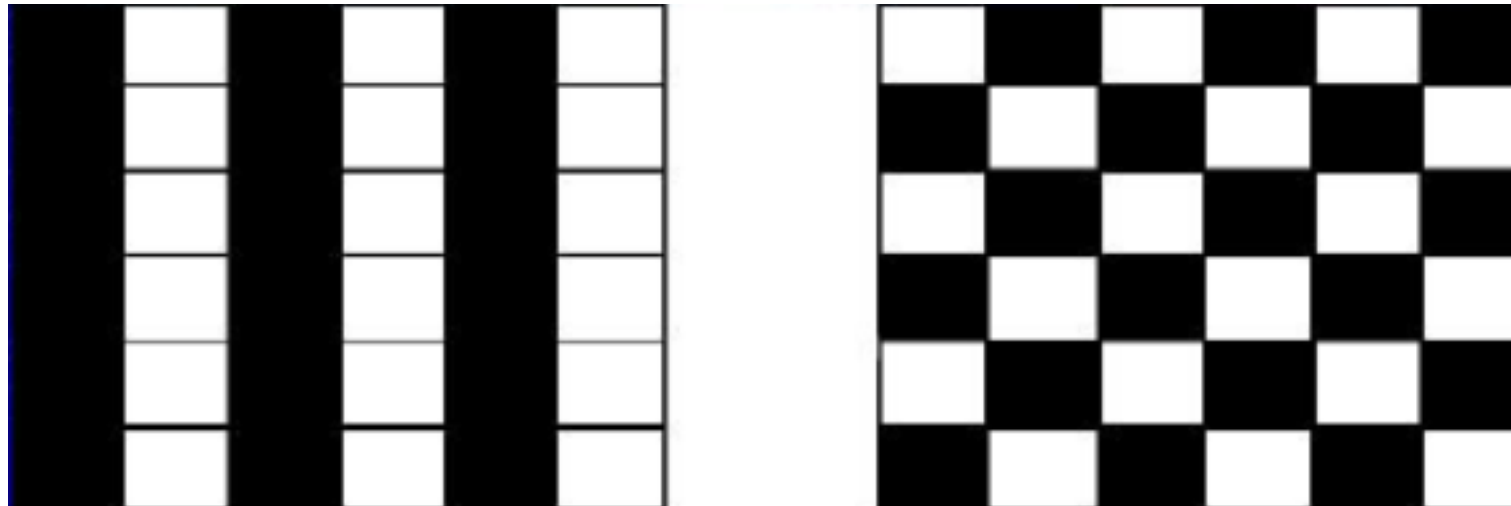


Statistical texture measures for the above subimages in the white squares

Texture	Mean	Standard deviation	$R$ (normalized)	3rd moment	Uniformity	Entropy
Smooth	82.64	11.79	0.002	-0.105	0.026	5.434
Coarse	143.56	74.63	0.079	-0.151	0.005	7.783
Regular	99.72	33.73	0.017	0.750	0.013	6.674

# Gray Level Co-occurrence Matrix

- Measures of texture computed using only histograms suffer from the limitation that they carry no information regarding the relative position of pixels with respect to each other.
- **Gray level co-occurrence matrix**: a way is to consider not only the distribution of intensities, but also the positions in texture analysis.



Images with same histograms

# Gray Level Co-occurrence Matrix

- A co-occurrence matrix, a.k.a. a co-occurrence distribution, is defined over an image to be the distribution of co-occurring values at a given offset.
- Let  $f(x,y)$  be a gray level image, then the co-occurrence matrix  $C_P$  is defined as follows:

$$C_P(i, j) = \# \{ (m, n) \in f(x, y) \mid f(m, n) = j, \text{ and } f(m + dm, n + dn) = i \}$$

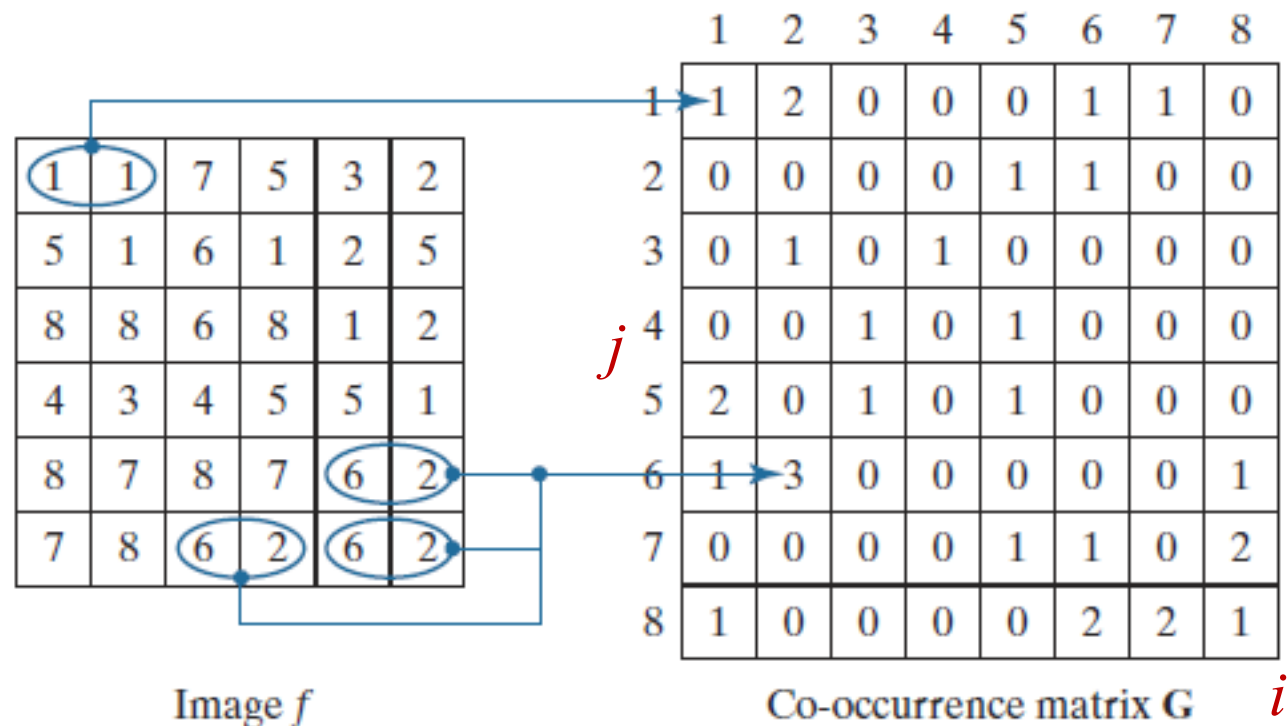
$P = (dm, dn) : \text{displacement}$

$\# : \text{presence times}$

- The value of  $C_P(i, j)$  indicates how many times the value  $i$  co-occurs with the value  $j$  in some designed spatial relationship.

# Gray Level Co-occurrence Matrix

- An example of how to construct a co-occurrence matrix  $\mathbf{G}$  using  $L = 8$  and a position operator  $Q$  defined as “one pixel immediately to the right” (i.e., the neighbor of a pixel is defined as the pixel immediately to its right).



# Gray Level Co-occurrence Matrix

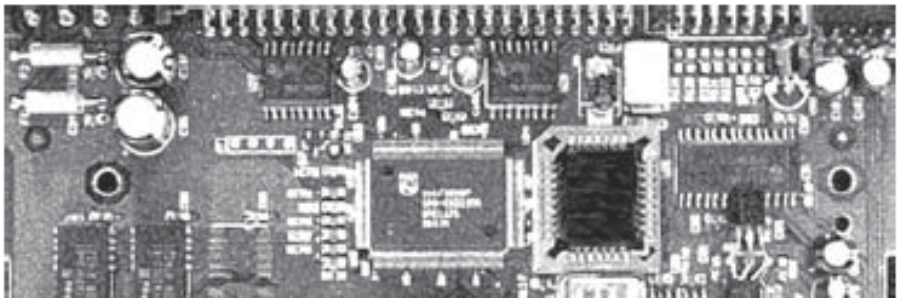
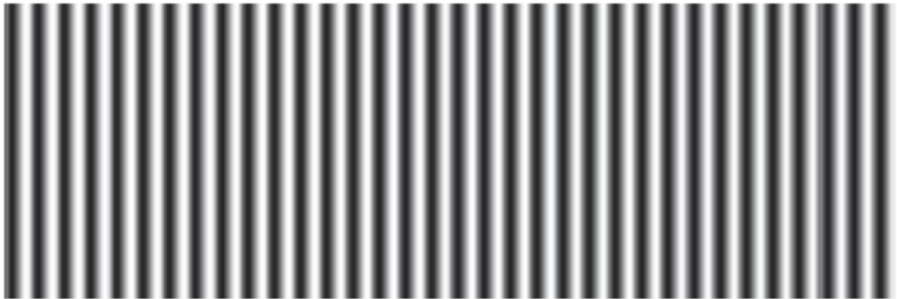
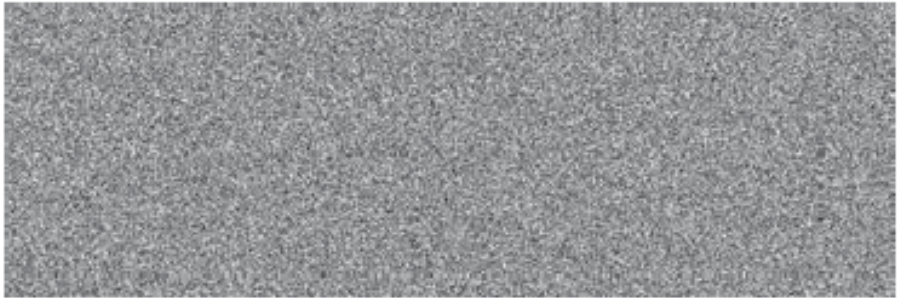
Descriptors used for characterizing co-occurrence matrices of size  $K \times K$ . The term  $p_{ij}$  is the  $ij$ -th term of  $\mathbf{G}$  divided by the sum of the elements of  $\mathbf{G}$ .

Descriptor	Explanation	Formula
Maximum probability	Measures the strongest response of $\mathbf{G}$ . The range of values is $[0, 1]$ .	$\max_{i,j}(p_{ij})$
Correlation	A measure of how correlated a pixel is to its neighbor over the entire image. The range of values is 1 to $-1$ corresponding to perfect positive and perfect negative correlations. This measure is not defined if either standard deviation is zero.	$\frac{\sum_{i=1}^K \sum_{j=1}^K (i - m_r)(j - m_c) p_{ij}}{\sigma_r \sigma_c}$ $\sigma_r \neq 0; \sigma_c \neq 0$
Contrast	A measure of intensity contrast between a pixel and its neighbor over the entire image. The range of values is 0 (when $\mathbf{G}$ is constant) to $(K - 1)^2$ .	$\sum_{i=1}^K \sum_{j=1}^K (i - j)^2 p_{ij}$
Uniformity (also called Energy)	A measure of uniformity in the range $[0, 1]$ . Uniformity is 1 for a constant image.	$\sum_{i=1}^K \sum_{j=1}^K p_{ij}^2$
Homogeneity	Measures the spatial closeness to the diagonal of the distribution of elements in $\mathbf{G}$ . The range of values is $[0, 1]$ , with the maximum being achieved when $\mathbf{G}$ is a diagonal matrix.	$\sum_{i=1}^K \sum_{j=1}^K \frac{p_{ij}}{1 +  i - j }$
Entropy	Measures the randomness of the elements of $\mathbf{G}$ . The entropy is 0 when all $p_{ij}$ 's are 0, and is maximum when the $p_{ij}$ 's are uniformly distributed. The maximum value is thus $2 \log_2 K$ .	$-\sum_{i=1}^K \sum_{j=1}^K p_{ij} \log_2 p_{ij}$

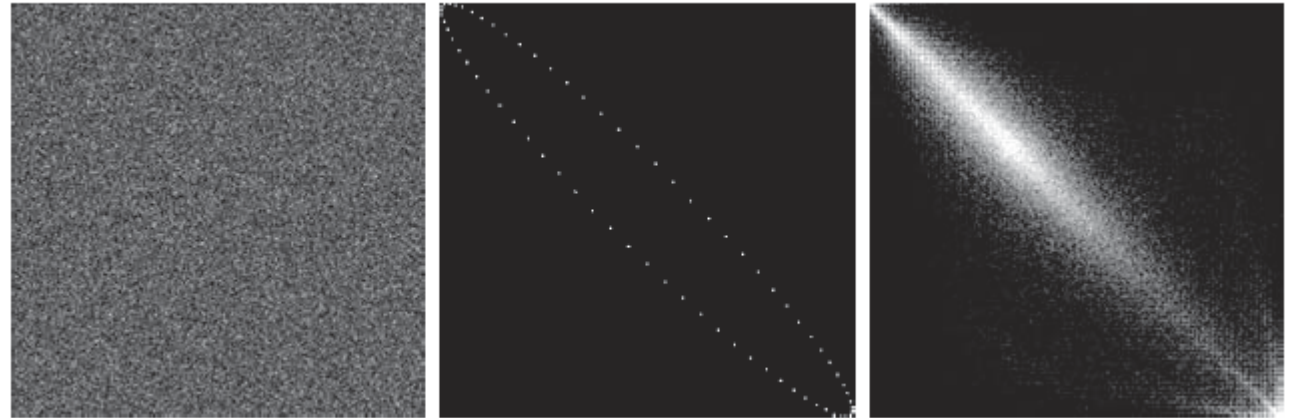


# Gray Level Co-occurrence Matrix

Images whose pixels have random, periodic, and mixed texture patterns.



Co-occurrence matrices  $G_1$ ,  $G_2$ , and  $G_3$ , corresponding from left to right to the images on the left.



Descriptors evaluated using the co-occurrence matrices displayed above.

Normalized Co-occurrence Matrix	Maximum Probability	Correlation	Contrast	Uniformity	Homogeneity	Entropy
$G_1/n_1$	0.00006	-0.0005	10838	0.00002	0.0366	15.75
$G_2/n_2$	0.01500	0.9650	00570	0.01230	0.0824	06.43
$G_3/n_3$	0.06860	0.8798	01356	0.00480	0.2048	13.58

# Moments of Two Dimensional Functions

---

- Moments can be used to represent the geometrical characters in image. It is also called **moment invariants**, because it is insensitive to **rotation**, **translation**, and **scale**.
- The 2-D *moment* of order  $(p + q)$  of an  $M \times N$  digital image,  $f(x, y)$ , is defined as

$$m_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} x^p y^q f(x, y)$$

where  $p = 0, 1, 2, \dots$  and  $q = 0, 1, 2, \dots$  are integers

# Moments of Two Dimensional Functions

---

- The corresponding *central moment* of order  $(p + q)$  is defined as

$$\mu_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (x - \bar{x})^p (y - \bar{y})^q f(x, y) dx dy$$

for  $p = 0, 1, 2, \dots$  and  $q = 0, 1, 2, \dots$  are integers, where

$$\bar{x} = \frac{m_{10}}{m_{00}} \quad \text{and} \quad \bar{y} = \frac{m_{01}}{m_{00}}$$

- The *normalized central moment* of order  $(p + q)$ , denoted  $\eta_{pq}$ , is defined

as 
$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}, \quad \text{where } \gamma = \frac{p + q}{2} + 1 \quad \text{for } p + q = 2, 3, \dots$$



# Moments of Two Dimensional Functions

- A set of seven, 2-D *moment invariants* can be derived from the second and third normalized central moments

$$\phi_1 = \eta_{20} + \eta_{02}$$

$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$

$$\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$

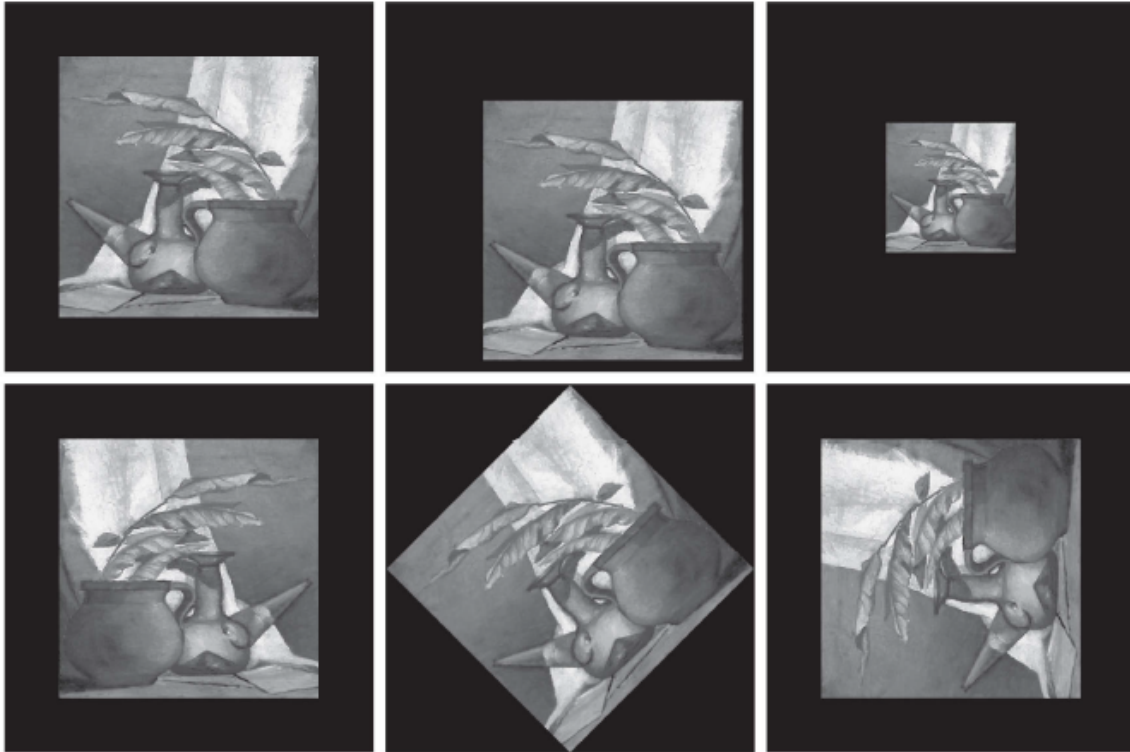
$$\begin{aligned}\phi_5 = & (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ & + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]\end{aligned}$$

$$\begin{aligned}\phi_6 = & (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\ & + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})\end{aligned}$$

$$\begin{aligned}\phi_7 = & (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ & + (3\eta_{12} - \eta_{30})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]\end{aligned}$$

# Moments of Two Dimensional Functions

(a) Original image. (b)-(f) Images translated, scaled by one-half, mirrored, rotated by 45 and rotated by 90 respectively.



Moment invariants for the images on the left

Moment Invariant	Original Image	Translated	Half Size	Mirrored	Rotated 45°	Rotated 90°
$\phi_1$	2.8662	2.8662	2.8664	2.8662	2.8661	2.8662
$\phi_2$	7.1265	7.1265	7.1257	7.1265	7.1266	7.1265
$\phi_3$	10.4109	10.4109	10.4047	10.4109	10.4115	10.4109
$\phi_4$	10.3742	10.3742	10.3719	10.3742	10.3742	10.3742
$\phi_5$	21.3674	21.3674	21.3924	21.3674	21.3663	21.3674
$\phi_6$	13.9417	13.9417	13.9383	13.9417	13.9417	13.9417
$\phi_7$	-20.7809	-20.7809	-20.7724	20.7809	-20.7813	-20.7809

# Other Descriptors

---

- Relational descriptors
- Principal components as feature descriptors
- Whole-image features
- Scale-invariant feature transform (SIFT)
- and many others...

# Summary

---

- In this lecture we have learnt:
  - Background of feature extraction
  - Boundary feature descriptor
  - Region feature descriptor
  - Other descriptors

# Optional Homework

---

Check the Textbook!

- **Chapter 11: Problems 11.4, 11.10(a), 11.13(a), 11.15(a)(b), 11.16(a), 11.19**
- Homework answers will be provided at the end of each week.