Image Feature Representation & Description

Amartya Kundu Durjoy

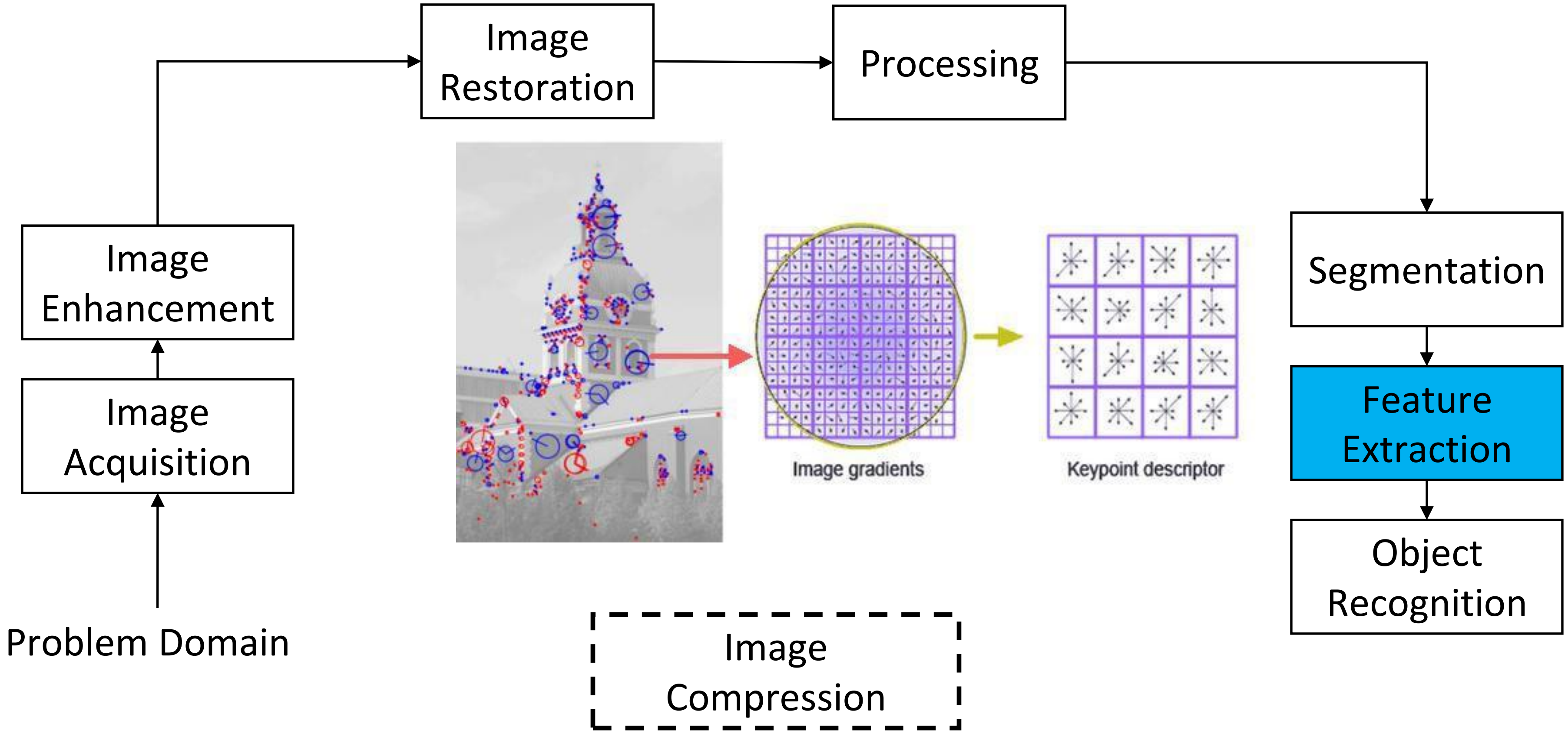
Dept. of CSE, UGV

Motivation

- One of the major concern of Computer Vision is image (object)recognition
  - Objects are represented as a collection of pixels in an image.
- Our Task: To describe the region based on the chosen representation
- Steps:
  - Image acquisition => digital image
  - Preprocessing => better image
  - Segmentation => basic features
  - Representation and description => advanced features
  - Object recognition

Representation & Description

Recall the Intro lecture



Introduction

- The common **goal** of feature extraction and representation techniques is to *convert the segmented objects into representations that better describe their main features and attributes.*
- The type and complexity of the resulting representation depend on many factors, such as
  - the type of image (e.g., binary, gray-scale, or color),
  - the level of granularity (entire image or individual regions) desired, and
  - the context of the application that uses the results
    - (e.g., a two-class pattern classifier that tells circular objects from noncircular ones
    - Or an image retrieval system that retrieves images judged to be similar to an example image).

Introduction

- “**Feature extraction** is the process by which certain features of interest within an image are detected and represented for further processing.”
- It is a critical step in most computer vision and image processing solutions because it marks the transition from pictorial to non-pictorial (alphanumerical, usually quantitative) data representation.
- The resulting representation can be subsequently used as an input to a number of pattern recognition and classification techniques, which will then *label*, *classify*, or *recognize* the semantic contents of the image or its objects.

Representation

- Representation means that we make the object information **more accessible** *for computer-interpretation* .
- Two types of representation
  - Using boundary (External characteristics)
  - Using pixels of region (Internal characteristics)

Prof. Dr. SMM Ahsan, CSE, KUET

Description

- Description means that ***we quantify*** our representation of the object
- **Boundary Descriptors**
  - Geometrical descriptors : Diameter, perimeter, eccentricity, curvature
  - Shape Numbers
  - Fourier Descriptors
  - Statistical Moments
- **Regional Descriptors**
  - Geometrical descriptors: Area, compactness, Euler number
  - Texture
  - Moments of 2D Functions

Desirable Properties of Descriptors

- Two objects must have the same descriptors if and only if *they have the same shape .*
- They should be *invariant to* Rotation, Scaling and Translation (RST)
- A descriptor should only contain information about what makes an object **unique**, or different from the other objects.
- The quantity of information used to describe this characterization should be ***less than*** the information necessary to have a complete description of the object itself.
- They should be robust
  - *Work well against Noise and Distortion*
- They should have low computational complexity

FEATURE VECTORS & VECTOR SPACES

- A ***feature vector*** is a $n \times 1$ array that <u>encodes the n features</u> (or measurements) of an image or object.
- The array **contents** may be
  - symbolic (e.g., a string containing the name of the predominant color in the image),
  - numerical (e.g., an integer expressing the area of an object, in pixels),
  - or both.
- Mathematically, a numerical feature vector **x** is given by

$$x = (x_1, x_2, \ldots, x_n)^T$$

  - where n is the total number of features and
  - T indicates the *transpose* operation.

FEATURE VECTORS & VECTOR SPACES

- The feature vector is a *compact representation of an image* (or object within the image), which can be associated with
  - the notion of a *feature space*,
  - an *n*-dimensional *hyperspace* that allows the visualization (for *n* <4) and
  - interpretation of the feature vectors' contents, their relative distances, and so on.

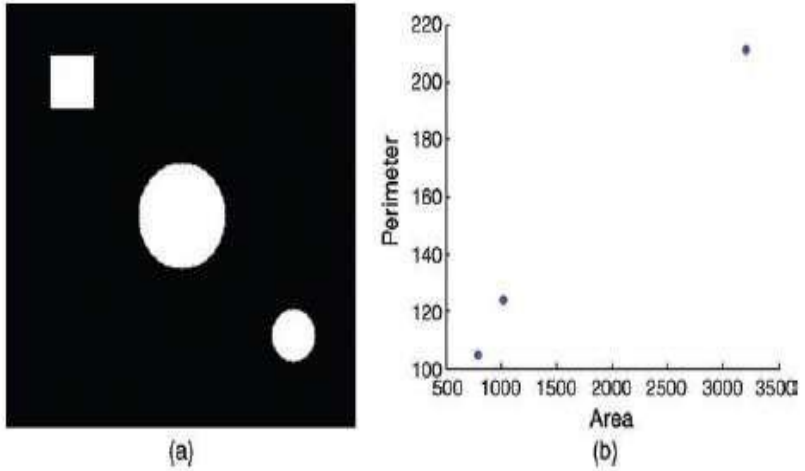


FIGURE 18.1 Test image (a) and resulting 2D feature vectors (b).

The resulting feature vectors will be as follows:

$Sq = (1024, 124)^T$

$LC = (3209, 211)^T$

$SC = (797, 105)^T$

Figure 18.1b shows the three feature vectors plotted in a 2D graph whose axes are the selected features, namely, *area* and *perimeter*.

Invariance & Robustness

- A common requirement for feature extraction and representation techniques is that the features used to represent an image be invariant to rotation, scaling, and translation, collectively known as RST.
- RST invariance ensures that a machine vision system will still be able to recognize objects even when they appear at different size, position within the image, and angle (relative to a horizontal reference).

Binary Object Features

- A binary object is a connected region within a binary image $f(x, y)$, which will be denoted as O_i, $i > 0$
- Mathematically, we can define a function $O_i(x, y)$ as follows:

  - $O_i(x, y) = \begin{cases} 1 & if(f(x, y) \in O_i) \\ 0 & otherwise \end{cases}$

- Area

Boundary Descriptors

- These techniques assume that the contour (or boundary) of an object can be represented in a convenient coordinate system (Cartesian—the most common, polar, or tangential) and rely exclusively on boundary pixels to describe the region or object.

- Object boundaries can be represented by different techniques, ranging from simple polygonal approximation methods to more elaborated techniques involving piecewise polynomial interpolations such as B-spline curves.

Boundary Descriptors

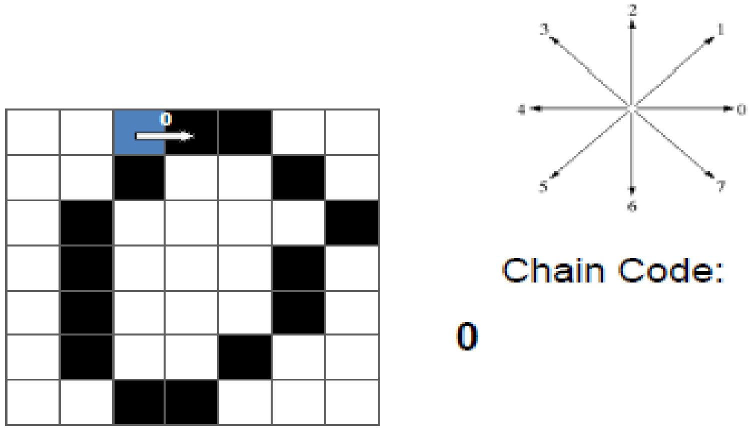
- The techniques described in this section assume that the pixels belonging to the boundary of the object (or region) can be traced, starting from any background pixel, using an algorithm known as bug tracing that works as follows:
  - As soon as the conceptual bug crosses into a boundary pixel, it makes a left turn and moves to the next pixel;
  - if that pixel is a boundary pixel, the bug makes another left turn, otherwise it turns right; the process is repeated until the bug is back to the starting point.
  - As the conceptual bug follows the contour, it builds a list of coordinates of the boundary pixels being visited.

Chain Code, Freeman Code, & Shape Number

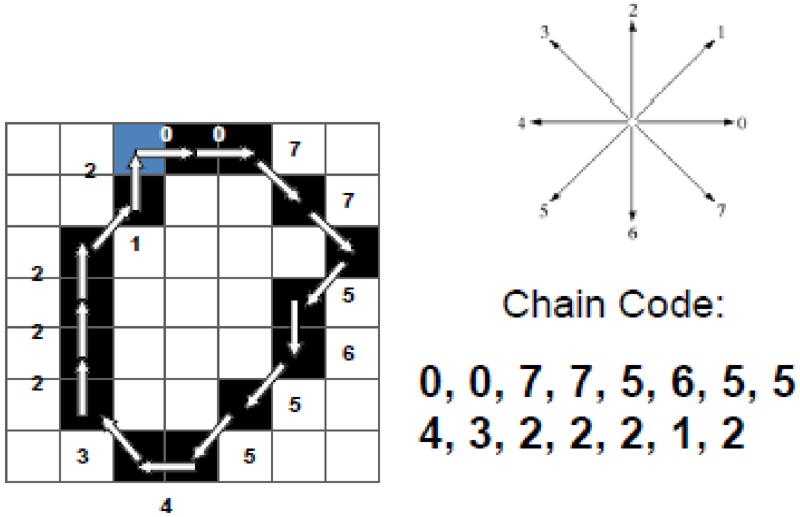
- Chain codes are alternative methods for *tracing* and *describing a **contour***.
- A chain code is a boundary representation technique by which "***A contour is represented as a sequence of straight line segments of specified length (usually 1) and direction***".
- The simplest chain code mechanism, also known as *crack code*, consists of assigning a number to the direction followed by a **bug tracking algorithm** as follows: right (0), down (1), left (2), and up (3).
- By allocating numbers based on directions, the boundary of an object is reduced to a sequence of numbers

Freeman Chain Code

- Steps for construction chain codes
  - Select some starting point of the boundary and represent it by its absolute coordinates in the image
  - Represent every consecutive point by a chain code showing transition needed to go from current point to next point on the boundary
  - Stop if the next point is the *initial point* or the *end of the boundary*



BOUNDARY



Freeman Chain Code

- Issues:
  - The resulting chain would be quite long
  - any small disturbances along the boundary due to noise or imperfect segmentation would cause changes in the code
- Solve:
  - resample the boundary by selecting a larger grid spacing

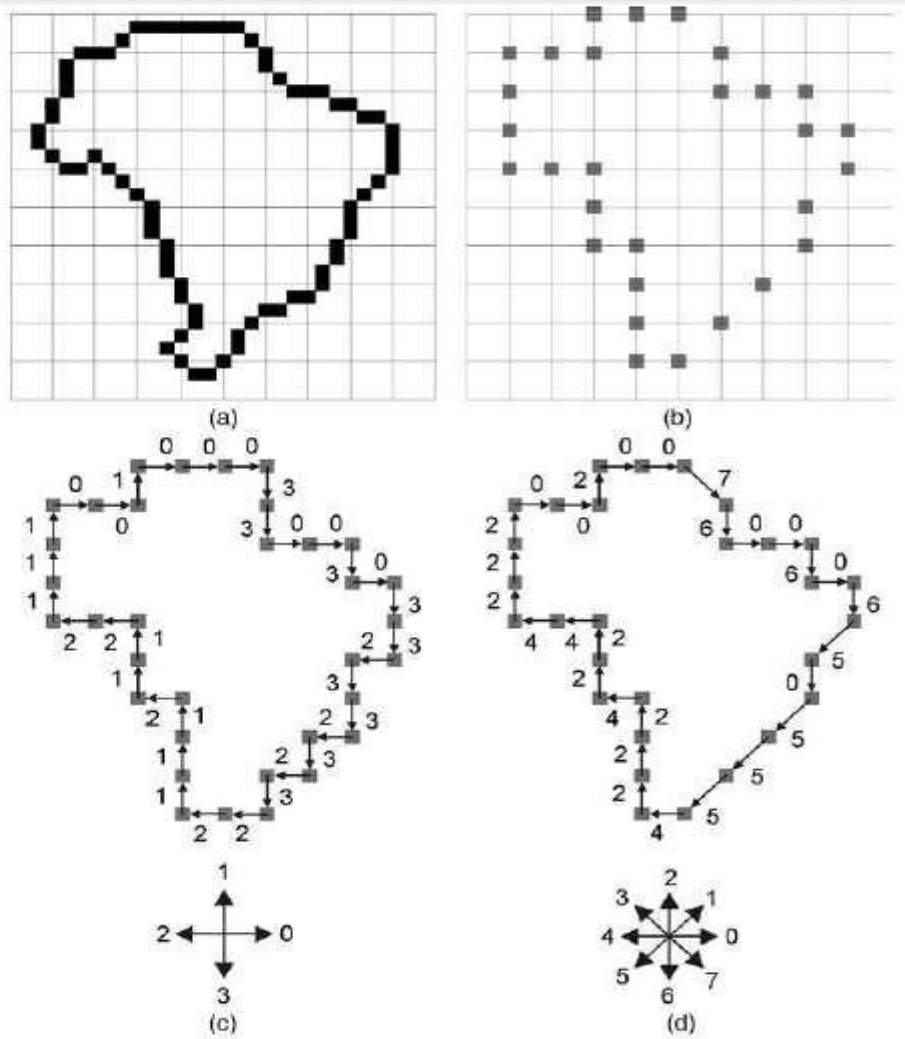


FIGURE 18.10 Chain code and Freeman code for a contour: (a) original contour; (b) subsampled version of the contour; (c) chain code representation; (d) Freeman code representation.

Chain Code

- **Problem**
  - A chain code sequence *depends on a starting point.*
- **Solution**
  - Treat a chain code as a circular sequence and redefine the starting point so that the resulting sequence of numbers forms an integer of minimum magnitude after circular shift

2 2 3 0 ⍰ 0 2 2 3

  - The first difference of a chain code is counting the number of direction change (in counter clockwise) between 2 adjacent elements of the code

Shape number

- The freeman chain code can be converted into a Rotation-Invariant Equivalent, known as the **first difference**.
  - It is obtained by encoding the number of direction changes, expressed in multiples of 90◦ (according to a predefined convention, for example, counter clockwise), between two consecutive elements of the Freeman code.
  - The first difference of Smallest magnitude is obtained by treating the resulting array as a **circular array** and rotating it cyclically until the resulting numerical pattern results in the **smallest possible number** is known as the **Shape number** of the contour.

Shape Number

- The shape number is Rotation invariant and Insensitive to the starting point used to compute the original sequence.
- Figure 18.11 shows an example of a contour, its chain code, first differences, and shape number.

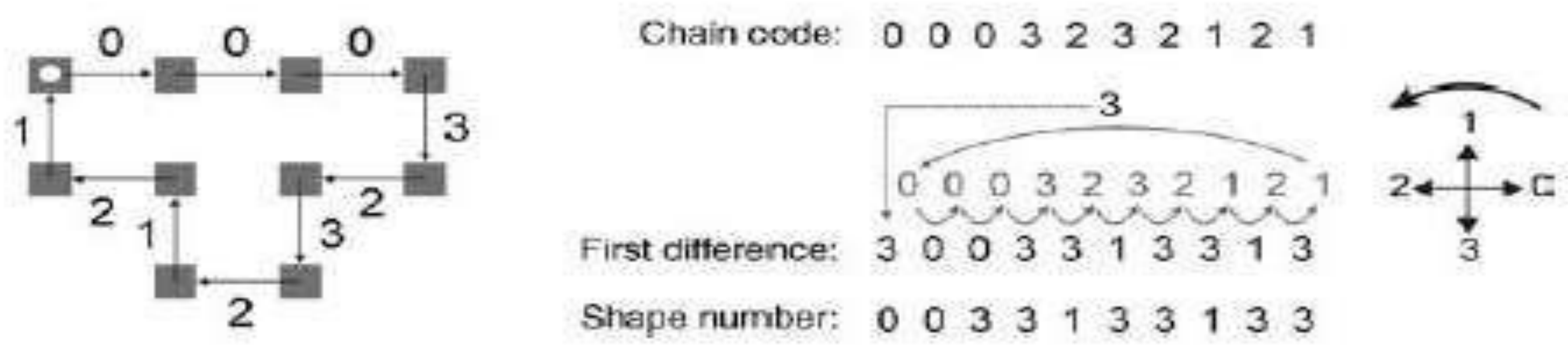
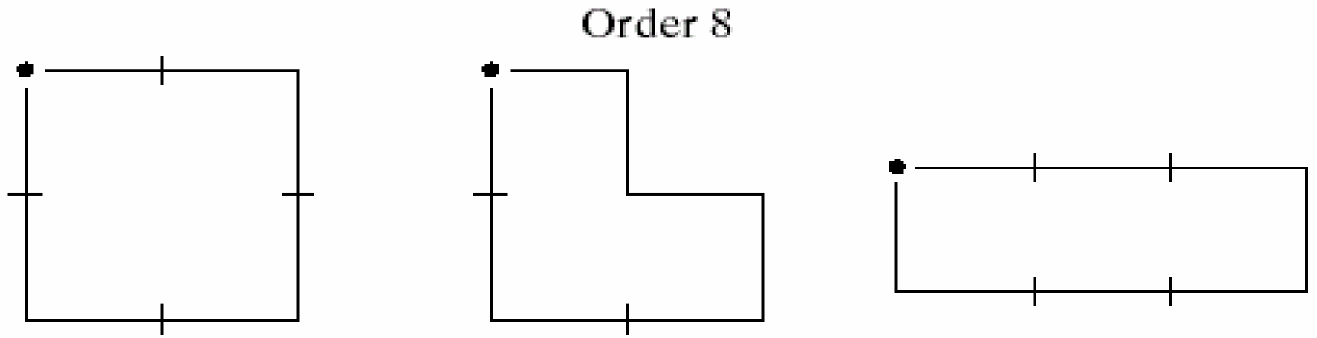


FIGURE 18.11 Chain code, first differences, and shape number.

Shape Number

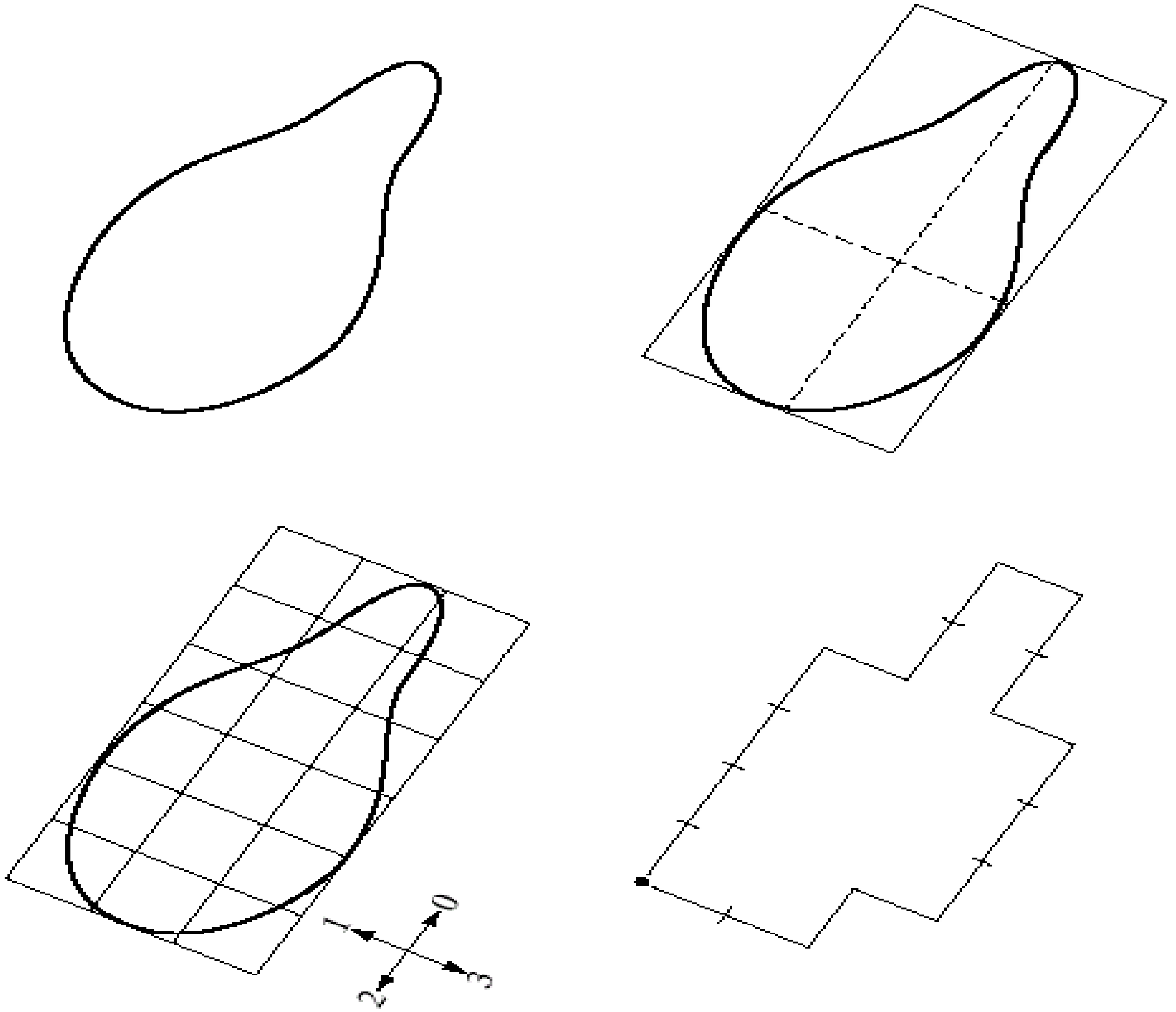
- The shape number of a boundary is defined as the first difference of smallest magnitude
- The order *n* of a shape number is defined as the number of digits in its representation



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

Algorithm for making a shape number

- Goal: To represent a given boundary by a shape number of order n
  - Step-1: Obtain the major axis of the shape and consider it as one of the coordinate axis
  - Step-2: Find the basic (smallest) rectangle that has sides parallel to major axis and just covers the shape
  - Step-3: From possible rectangles of order n, find one which best approximates rectangle of step-2
  - Step-4: Orient the rectangle, so that its major axis coincides with that of the shape
  - Step-5: Obtain the first difference chain code of minimum magnitude after circular shift



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

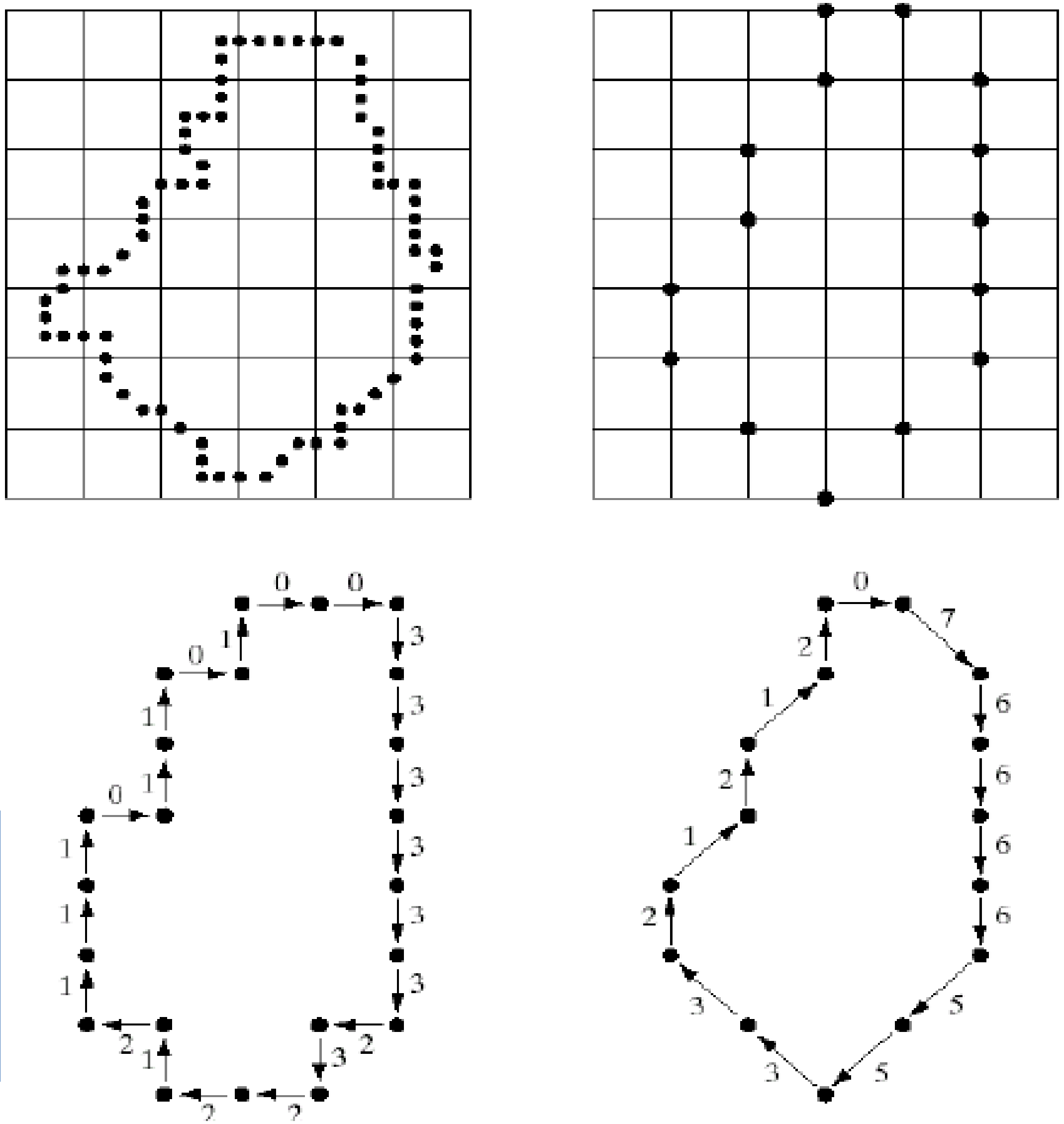
Difference: 3 0 0 0 3 1 0 3 3 0 1 3 0 0 3 1 3 0

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

Chain Code

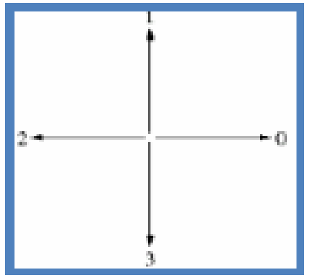
- **Advantages**
  - Preserves the information of interest
  - Provides good compression of boundary description
  - They are translation invariant
- **Problems**
  - Long chains of codes
  - No invariance to Rotation and Scale
  - Sensitive to Noise
- **Solution**
  - *Re-sample* the image to a lower resolution before calculating the code

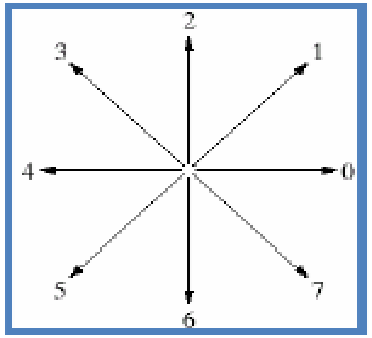
Chain Codes

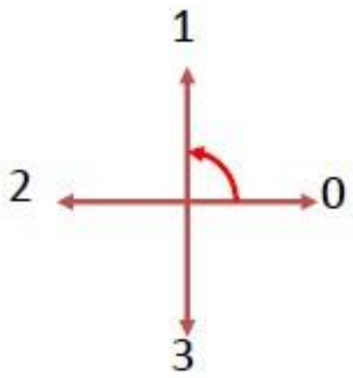


a b
c d

FIGURE 11.2 (a) Digital boundary with resampling grid superimposed. (b) Result of resampling. (c) 4-directional chain code. (d) 8-directional chain code.







Chain code : The first difference

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

Example:

- a chain code: 10103322
- The first difference = 3133030
- Treating a chain code as a circular sequence, we get the first difference = 33133030

Polygonal Approximation

- "Approximates the boundary by a **set of connected line segments** "
- Polygonal approximation provides a simple representation of the ***Planar Object Boundary.***

- Mathematical Definition
  - Let the set of points of boundary be $\boldsymbol{X} = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_n}\}$
  - Divide this set into segment $\Lambda = \{\boldsymbol{\lambda_1}, \boldsymbol{\lambda_2}, \ldots, \boldsymbol{\lambda_n}\}$
  - Approximate each segment by straight line by minimization of objective function

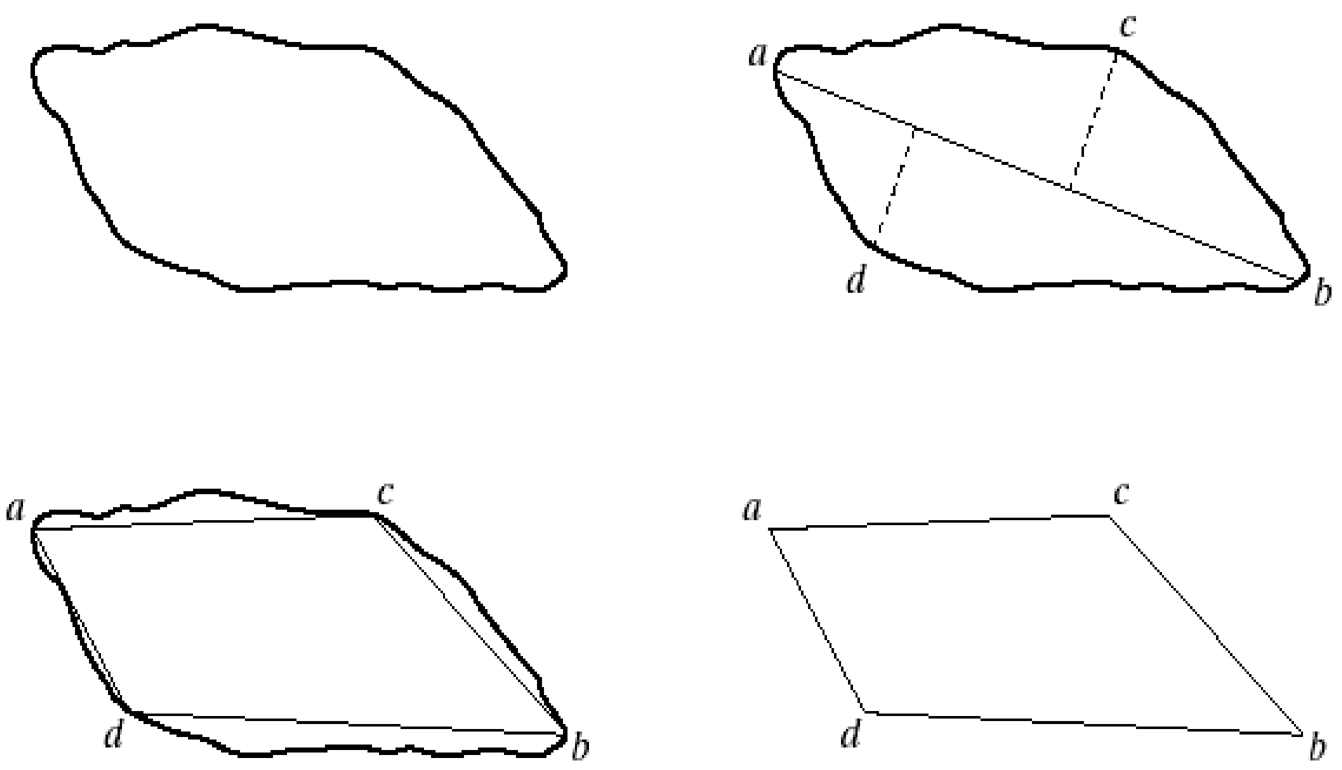
$$J = \sum_i d(x_i, l_j), x_i \in \lambda_j$$

Polygonal Approximation

- Approximation **leads to Loss of Information**
  - The number of straight line segments used determines the accuracy of the approximation
  - For a closed boundary, approximation becomes exact when no. of segments of the polygons is equal to the no. of points in the boundary
- However, the **goal** of approximation is
  - To capture the essence of the object shape with minimal loss
  - Thus, it saves the no. of bytes required for boundary representation

The Split Method (Top-down)

- **Iteratively decompose a boundary** into a set of small segments and represent the segment by a straight line
- **Algorithm:**
  - Step-1: Take the line segment connecting the end points of the boundary (if the boundary is closed, consider the line segment connecting the two farthest points).
  - Step-2: Find the boundary point with maximum distance from the line segment
  - Step-3: If the distance is above threshold, split the segment into two segments at that point (i.e., new vertex).
  - Step-4: Repeat the same procedure for each of the two sub segments until the distance is below threshold



a b
c d

FIGURE 11.4
(a) Original boundary.
(b) Boundary divided into segments based on extreme points. (c) Joining of vertices.
(d) Resulting polygon.

The Merge Method (Bottom-up)

- Operate in a direction opposite to that of splitting method
- **Algorithm**:
  - Step-1: Use the first two boundary points to define a line segment
  - Step-2: Add a new point if it does not deviate too far from the current line segment
  - Step-3: Update the parameters of the line segment using least-squares
  - Step-4: Start a newline segment when boundary points deviate too far from the line segment

The Split & Merge Algorithm

- Problems of the split and merge methods
  - Depending on threshold , vertices of polygon not necessarily correspond to points of inflections (such as corners) in the boundary
- Combine split and merge method
  - After recursive subdivision (split), allow adjacent segments to be replaced by a single segment (merge)

Signatures

- "**Signature is a 1D representation of a boundary**"
- It is obtained by *representing the boundary in a polar coordinate system* then
  - Computing the **distance *r*** between each Pixel along the boundary and the Centroid of the region, and
  - The **angle ϑ** subtended between a straight line connecting the boundary pixel to the centroid and a horizontal reference (Figure 18.12, top).
- The resulting plot of all computed values for $0 \leq \vartheta \leq 2\pi$ (Figure 18.12, bottom) provides a concise representation of the boundary that is translation invariant can be made rotation invariant (if the same starting point is always selected), but is *not scaling invariant*.
- Figure 18.13 illustrates the effects of noise on the signature of a contour.

Signatures

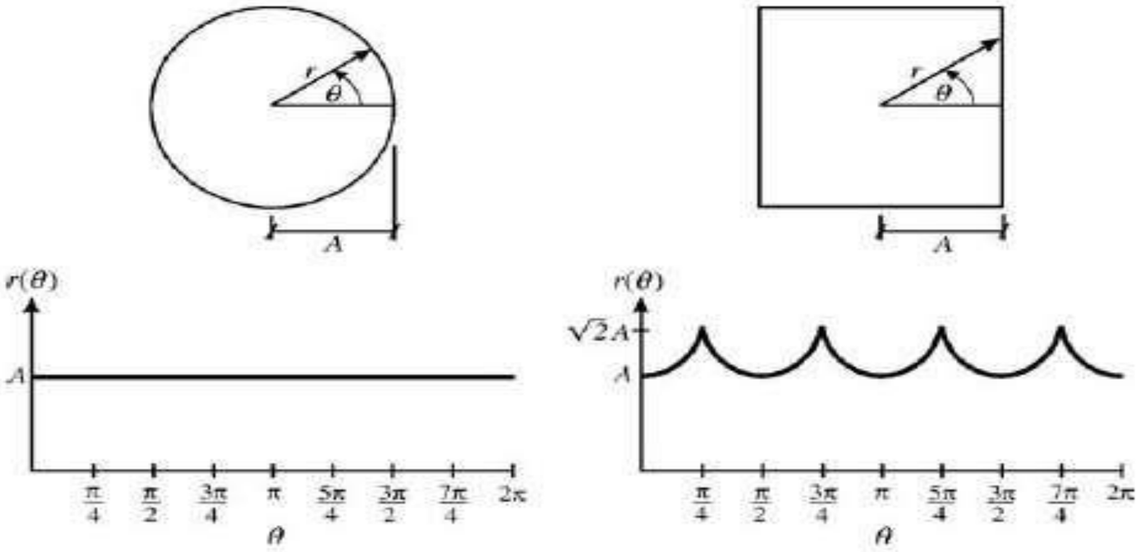


FIGURE 18.12 Distance × angle signatures for two different objects. Redrawn from [GW08].

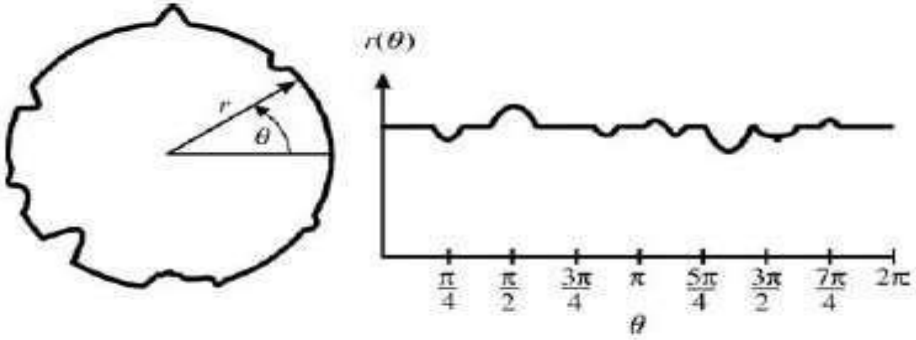


FIGURE 18.13 Effect of noise on signatures for two different objects. Redrawn from [GW08].

Signatures

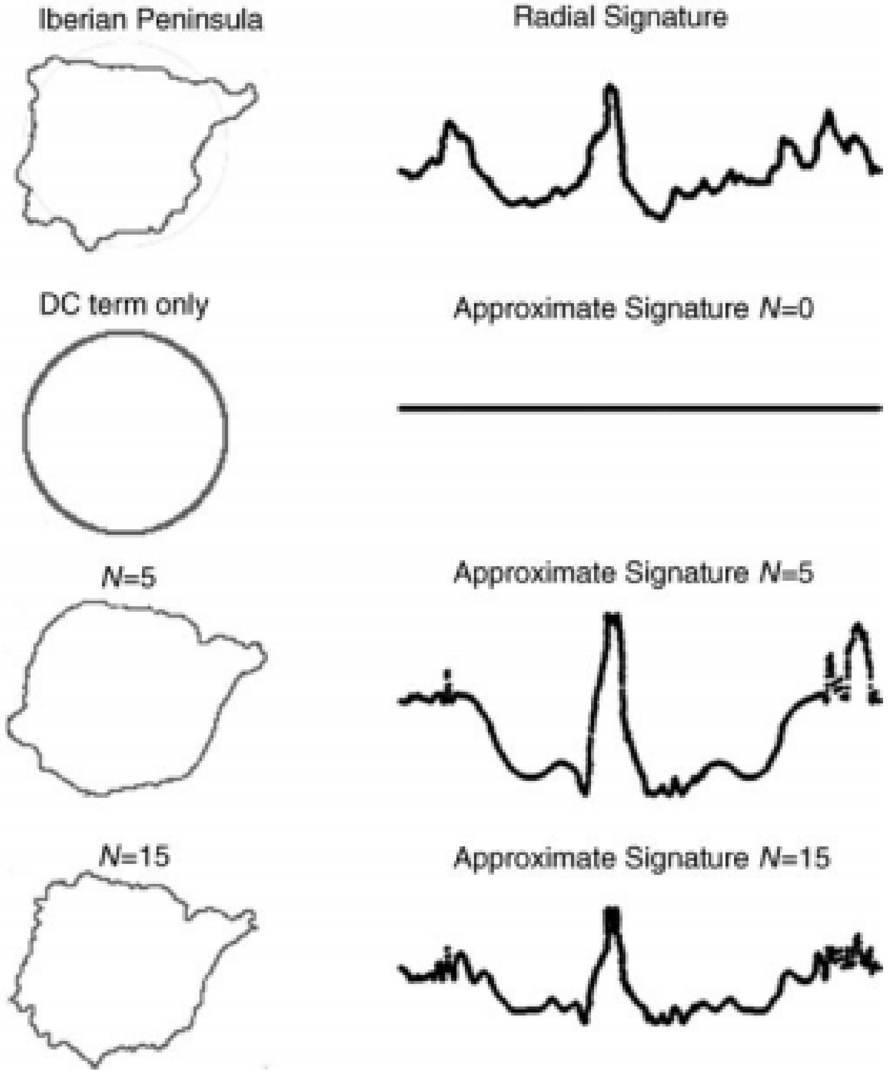


Figure 9.3 An example of the use of the radial Fourier expansion technique to approximate the shape of the Iberian Peninsula using $N = 0$, 5 and 15 terms

Signature

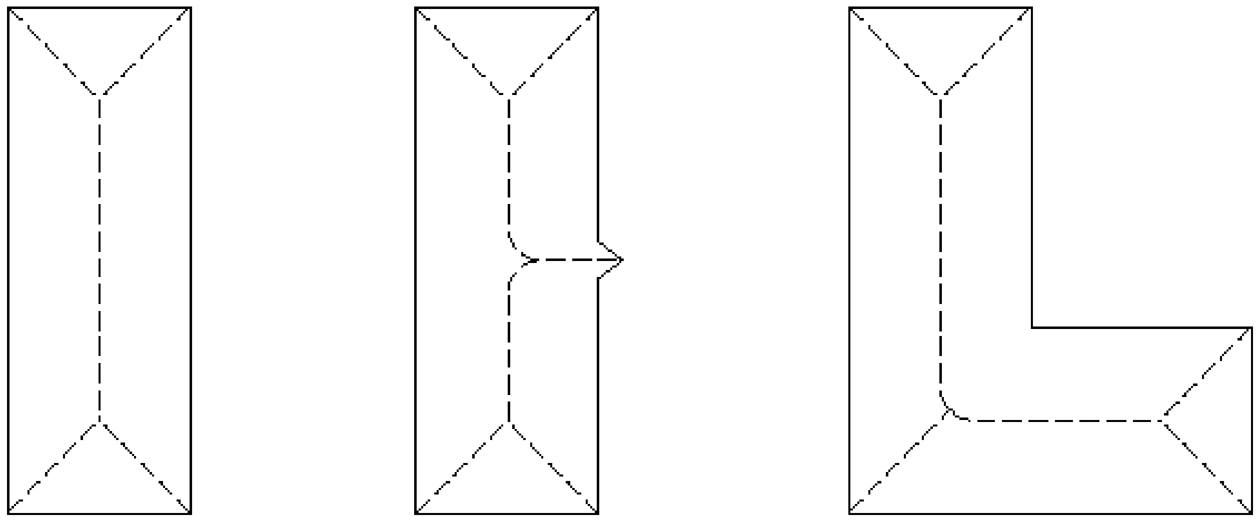
- Signatures are invariant to location, but will depend on rotation and scaling.
  - **Rotation invariance** can be improved by selecting a unique starting point (e.g. based on major axis)
  - **Scale invariance** can be achieved by normalizing amplitude of signature (divide by variance)

Skeletons

- Skeletons produce a **one pixel wide graph** that has the same basic shape of the region, like a stick figure of a human
- Hence, they provide a compact and often highly intuitive representation
- It can be used to analyse the *geometric structure of a region*
- Also popular tool in object recognition

Medial Axis Transform (MAT)

- **Provides skeleton of an object.**
- The MAT of a region R with border B is defined as follows:
  - For each point p of R, we find its closest neighbour in B.
  - If p has more then one such points, it is said to belong to the medial axis (skeleton) of R .



a b c

FIGURE 11.7 Medial axes (dashed) of three simple regions.

Medial Axis Transform (MAT)

Step-1: Iteratively compute $f^k(x,y)$ as follows until $f^k(x,y) = f^{k-1}(x,y)$:

$$f^k(x,y) = f^0(x,y) + \min\left(f^{k-1}(p,q)\right)$$

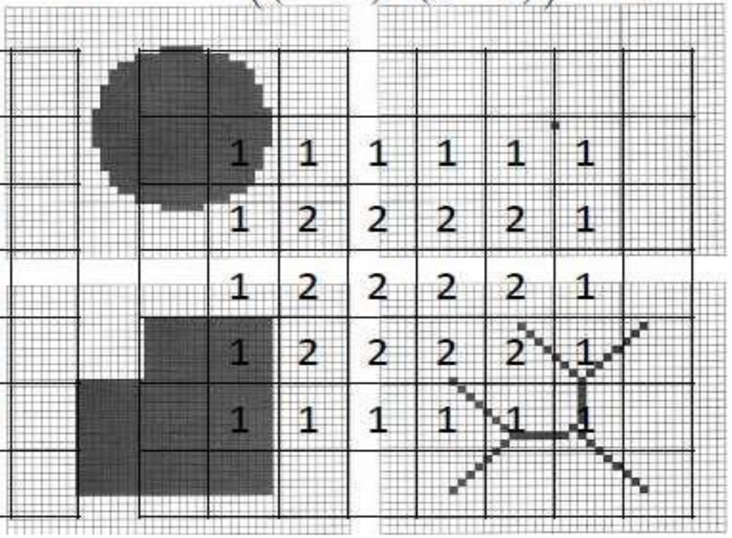
$$\forall (p,q) \text{ such that } d\left((x,y),(p,q)\right) \le 1$$

Step-2: Medial axis is given by all points such that:

$$f^k(x,y) \ge f^k(p,q)$$

$$\forall (p,q) \text{ such that } d\left((x,y),(p,q)\right) \le 1$$

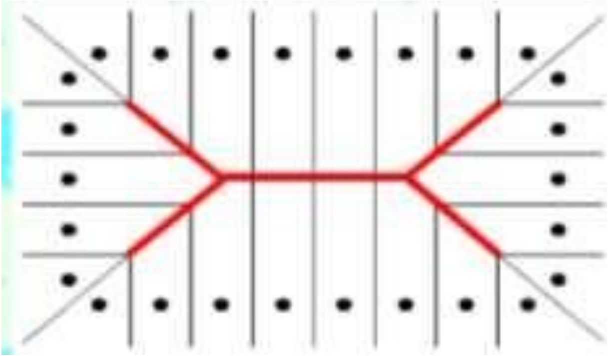
	1	1	1	1	1	1
	1	1	1	1	1	1
	1	1	1	1	1	1
	1	1	1	1	1	1
	1	1	1	1	1	1



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

Medial Axis Transform (MAT)

- Medial Axis augmented by radius function & Transformation is invertible
- The medial axis of a circle is its center.
- the medial axis of an ellipse is its center (the midpoint of the line that connects the two foci of the ellipse), too.
- Equilateral triangle : the segments connecting the middle of the bases and the center of the figure.
- Arbitrary triangle : the segments connecting the middle of the sides with the center of gravity (where all medians cross).



Medial Axis Transform (MAT)

- Application
  - Shape matching
  - Animation
  - Dimension reduction
  - Solid modelling
  - Smoothing or sharpening of shape
  - Motion planning
  - Mesh generation

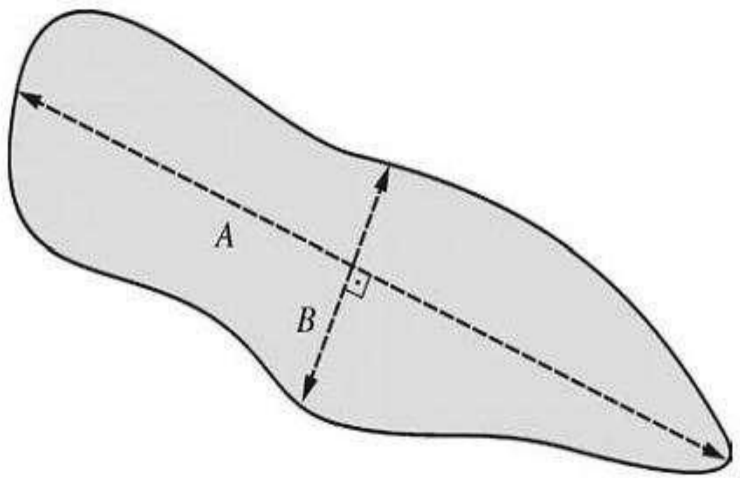
Other Boundary Descriptors

- There aresseveral simple geometric measures that can be useful for describing a boundary
- Length
  - the number of pixels along a boundary gives a rough approximation of its length
  - For a chain-coded curve with unit spacing
    - Length = the number of vertical and horizontal components + √2 * the number of diagonal components
- Diameter (Major Axis)

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

Other Boundary Descriptors

- Minor Axis
  - the line perpendicular to the major axis
- Eccentricity
  - Ratio of major axis to minor axis



Fourier Descriptors

- The idea behind Fourier descriptors is to *traverse the pixels belonging to a boundary*, starting from an arbitrary point, and record their coordinates.
- Each value in the resulting list of coordinate pairs $(x_0, y_0), (x_1, y_1), \ldots, (x_{K-1} y_{K-1})$ is then interpreted as a complex number $x_k + jy_k$, for $k = 0, 1, \ldots, K - 1$.
- "The discrete Fourier transform (DFT) of this list of complex numbers is the ***Fourier descriptor*** of the boundary".
- *The inverse DFT restores the original boundary.*
- Figure 18.14 shows a K-point digital boundary in the x-y plane and the first two coordinate pairs, (x_0, y_0) & (x_1, y_1)

Fourier Descriptors

- Following is a way of using the Fourier transform to analyse the shape of a boundary.

1. The *x-y* coordinates of the boundary are treated as the real and imaginary parts of a **complex number**
2. Then the list of coordinates is Fourier transformed using the DFT
3. The Fourier coefficients are called the **Fourier descriptors**.
4. The basic shape of the region is determined by the first several coefficients, which represent lower frequencies
5. Higher frequency terms provide information on the fine detail of the boundary

Fourier Descriptors

- **chief advantages**
  - ability to represent the **essence of the corresponding boundary using very few coefficients**.
  - This **property** is directly related to the ability of *the low-order coefficients of the DFT*
    - *That preserve the main aspects* of the boundary, while *the high-order coefficients encode the fine details*.

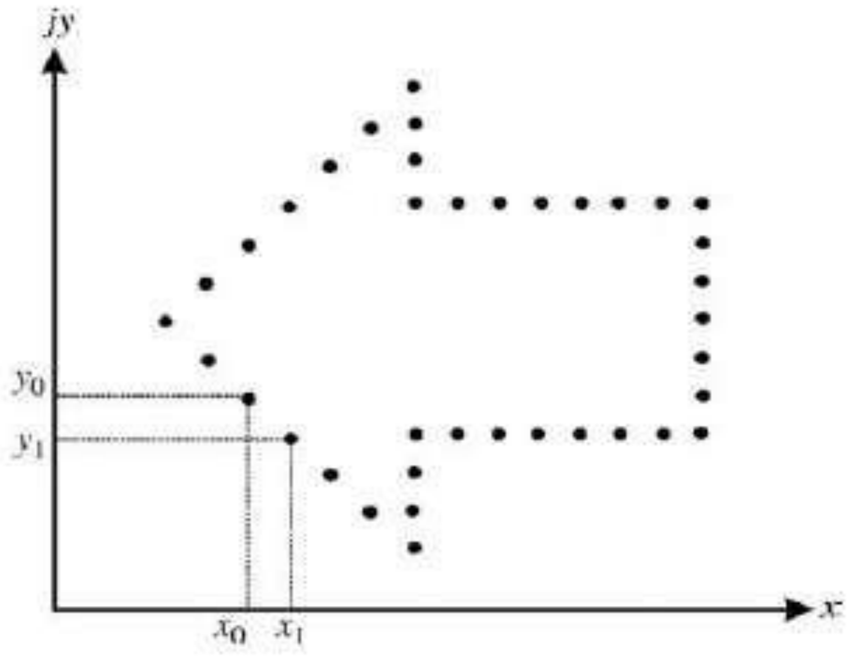
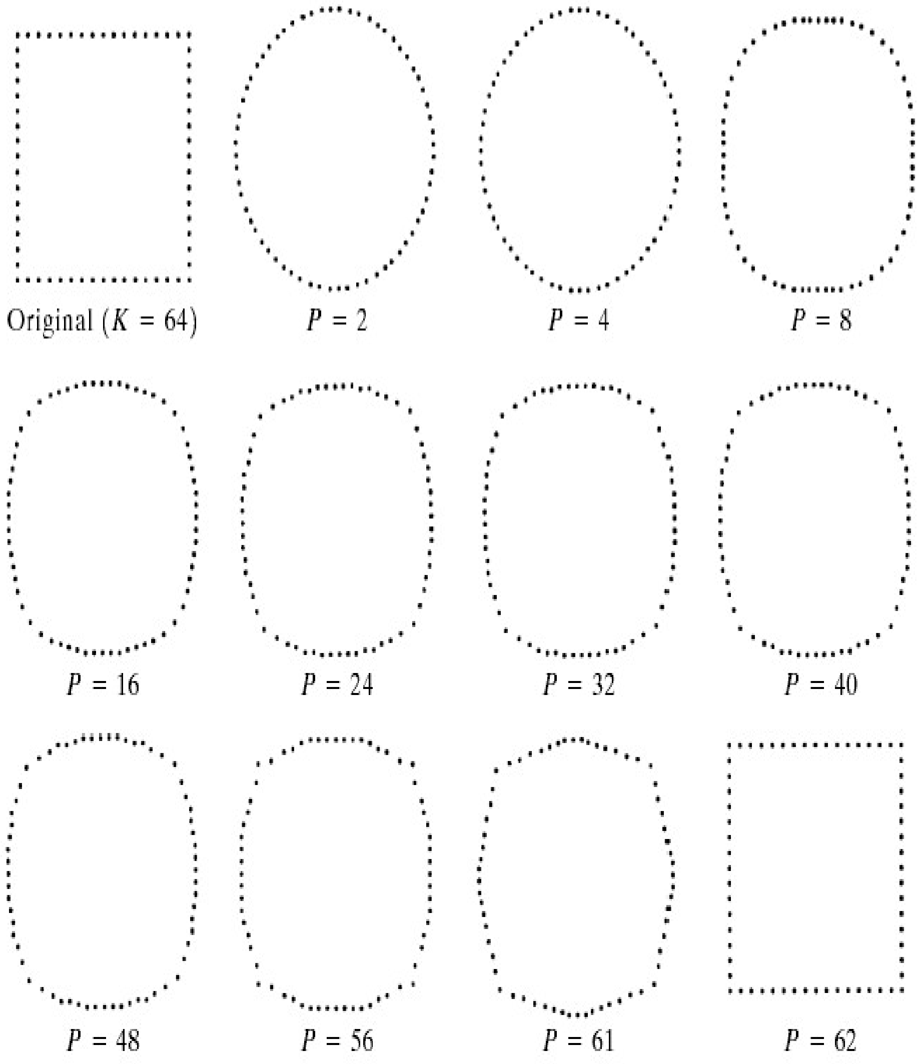
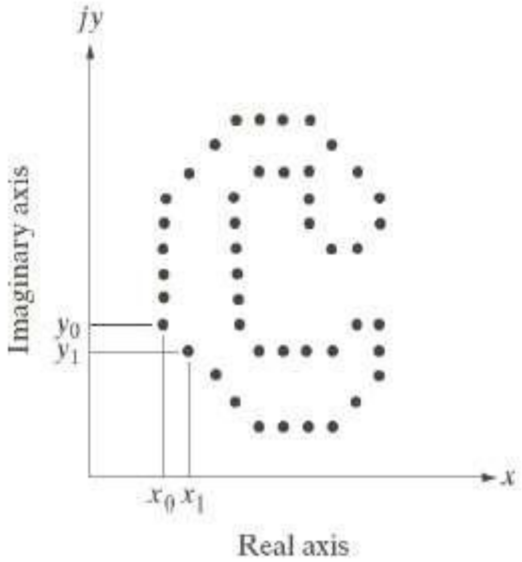


FIGURE 18.14 Fourier descriptor of a boundary.

Fourier Descriptors

FIGURE 11.14 Examples of reconstruction from Fourier descriptors. P is the number of Fourier coefficients used in the reconstruction of the boundary.





Points on the boundary can be treated as a complex number

$$s(k)=x(k)+jy(k)$$

$$a(u)=\frac{1}{N}\sum_{k=0}^{N-1}s(k)\exp[-j2\pi uk/N]$$

Fourier Desc

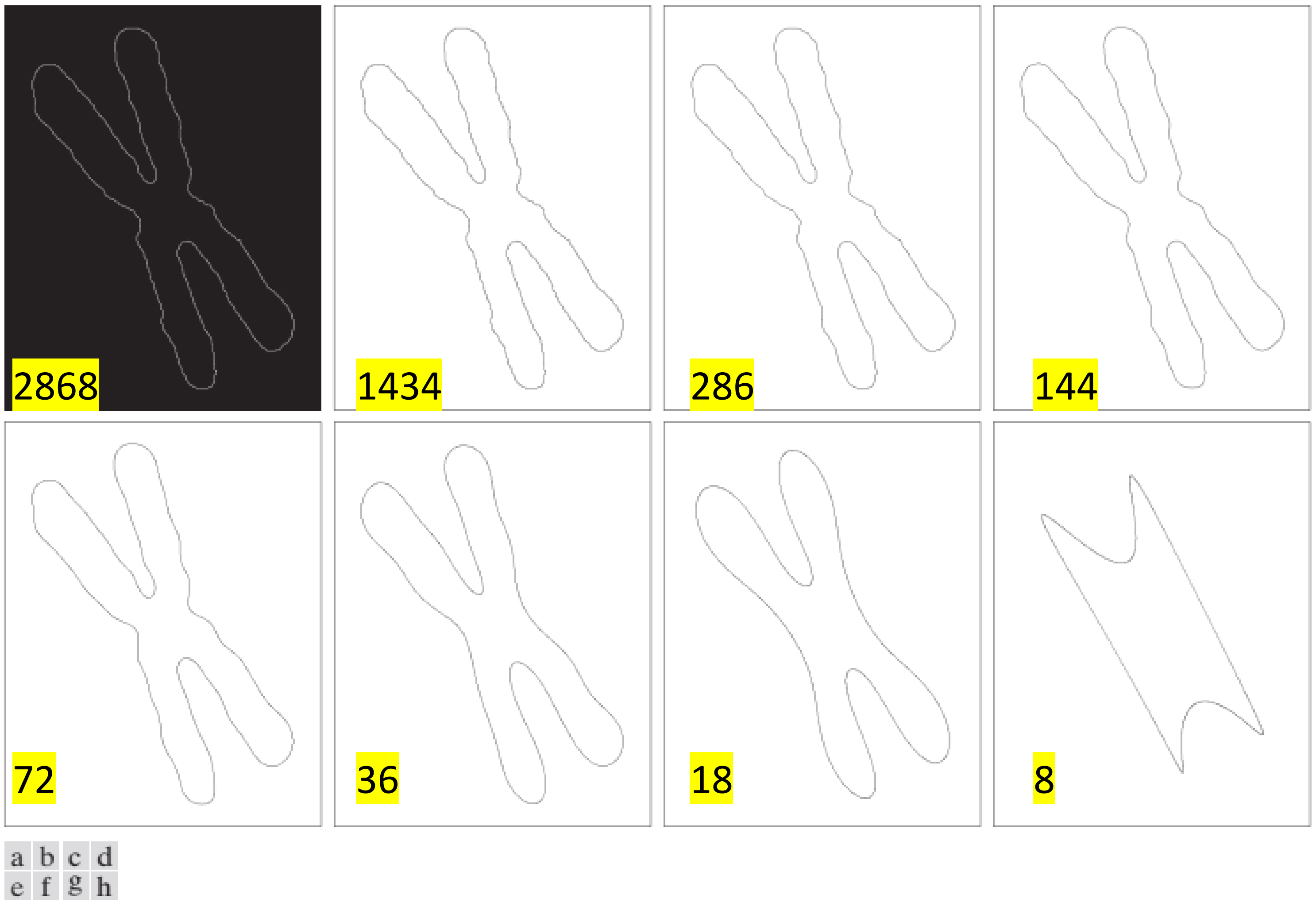


FIGURE 11.19 (a) Boundary of a human chromosome (2868 points). (b)–(h) Boundaries reconstructed using 1434, 286, 144, 72, 36, 18, and 8 Fourier descriptors, respectively. These numbers are approximately 50%, 10%, 5%, 2.5%, 1.25%, 0.63%, and 0.28% of 2868, respectively. Images (b)–(h) are shown as negatives to make the boundaries easier to see.

Properties of Fourier Descriptors

Translation

- Adding some constant to values of all coordinates
- So, we only change the zero-frequency component. (Mean position only nothing about the shape)
- So, except for the zero-frequency component, Fourier Descriptors are translation invariant.

Rotation

- Rotation in the complex plane by angle θ is multiplication by exp(jθ)
- So, rotation about the origin of the coordinate system only multiplies the Fourier descriptors by exp(jθ)

Properties of Fourier Descriptors

Scaling

- It means multiplying $x(k)$ and $y(k)$ by some constant.
- Hence, Fourier descriptors are scaled by the same constant (Again, we ignore the value of the zero-frequency component)

Starting Point

- Changing starting point is *equivalent to translation* of the one-dimensional signal $s(k)$ along the k dimension
- Hence, translation in the spatial domain (in this case, k) is a phase-shift in the transform.
- So, the magnitude part of $a(u)$ is invariant to the start point, and the phase part shifts accordingly

Transformation	Boundary	Fourier Descriptor
Identity	$s(k)$	$a(u)$
Rotation	$s_r(k) = s(k)e^{j\theta}$	$a_r(u) = a(u)e^{j\theta}$
Translation	$s_t(k) = s(k) + \Delta_{xy}$	$a_t(u) = a(u) + \Delta_{xy}\delta(u)$
Scaling	$s_s(k) = \alpha s(k)$	$a_s(u) = \alpha a(u)$
Starting point	$s_p(k) = s(k - k_0)$	$a_p(u) = a(u)e^{-j2\pi k_0 u/K}$

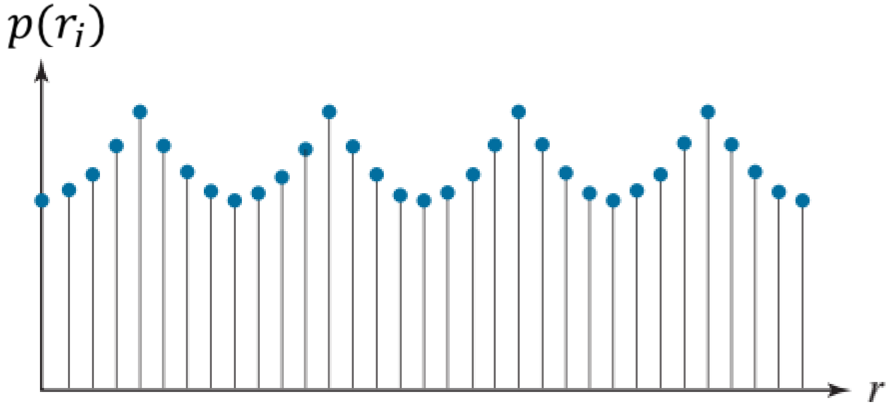
STATISTICAL FEATURES

STATISTICAL FEATURES

- Histogram-based features are also referred to as amplitude features
- Histograms provide a concise and useful representation of the intensity levels in a gray-scale image.
- The simplest histogram-based descriptor is the ***mean*** gray value of an image, representing its average intensity *m* and given by

$$m = \sum_{j=0}^{L-1} r_j \, p(r_j)$$

where r_j is the jth gray level (out of a total of L possible values), whose probability of occurrence is $p(r_j)$.



STATISTICAL FEATURES - Mean

- The mean gray value can also be computed directly from the pixel values from the original image $f(x, y)$ of size $M \times N$ as follows:

$$m = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y)$$

- The *mean is a very compact descriptor* (one floating-point value per image or object) that provides a measure of the overall brightness of the corresponding image or object.
- *It is also RST invariant*.
- On the negative side, it has very limited Expressiveness and Discriminative power.

STATISTICAL FEATURES – Std. Dev.

- The ***standard deviation***(as descriptor) σ of an image is given by

$$\sigma = \sqrt{\sum_{j=0}^{L-1} (r_j - m)^2 \, p(r_j)}$$

- where *m* is mean
- The square of the standard deviation is the ***variance***, which is also known as the *normalized* ***second-order*** *moment* of the image.
- The standard deviation provides a concise representation of the overall contrast.
- Similar to the mean, it is compact and RST invariant, but has limited expressiveness and discriminative power.

STATISTICAL FEATURES – Skew

- The ***skew*** of a histogram is a measure of its asymmetry about the mean level. It is defined as

$$skew = \frac{1}{\sigma^3}\sum_{j=0}^{L-1}(r_j - m)^3 p(r_j)$$

  - where σ is the standard deviation.
  - The **sign** of the skew indicates whether the *histogram's tail spreads* to the right (positive skew) or to the left (negative skew).
  - The skew is also known as the *normalized **third-order** moment* of the image.

STATISTICAL FEATURES – Skew

- If the image's mean value (m), standard deviation (σ), and mode (defined as the histogram's highest peak) are known, the skew can be calculated as follows:

$$skew = \frac{m - \text{mode}}{\sigma}$$

STATISTICAL FEATURES – Energy

- The ***energy/uniformity*** descriptor provides another measure of *how the pixel values are* ***distributed*** *along the gray-level range*:
  - images with a single constant value have maximum energy (i.e., energy = 1);
  - images with few gray levels will have higher energy than the ones with many gray levels. The energy descriptor can be calculated as

$$energy = \sum_{j=0}^{L-1} [p(r_j)]^2$$

STATISTICAL FEATURES – Entropy

- Histograms also provide information about the complexity of the image, in the form of ***entropy* descriptor**.
- The higher the entropy, the more complex the image
- Entropy and energy tend to vary inversely with one another. The mathematical formulation for entropy is

$$entropy = -\sum_{j=0}^{L-1} p(r_j)\log_2[p(r_j)]$$

- Histogram-based features and their variants are usually employed as texture descriptors, as we shall see in next slide.

Statistical Moments

nth moment

$$m_n = \int_{\infty}^{\infty} x^n p(x) \mathrm{d}x$$

central moments

$$\mathrm{M}_n = \int_{\infty}^{\infty} (x - \mu)^n p(x) \, \mathrm{d}x$$

*pq*th moment of a 2-D density function pðx; yÞ

$$m_{pq} = \int_{\infty}^{\infty} \int_{\infty}^{\infty} x^p y^q p(x, y) \mathrm{d}x\mathrm{d}y$$

(p- q)th central moment of 2-D shape I(x,y)

$$M_{pq} = \int_{\infty}^{\infty} \int_{\infty}^{\infty} (x - \mu_x)^p (y - \mu_y)^q I(x, y) \, \mathrm{d}x \, \mathrm{d}y$$

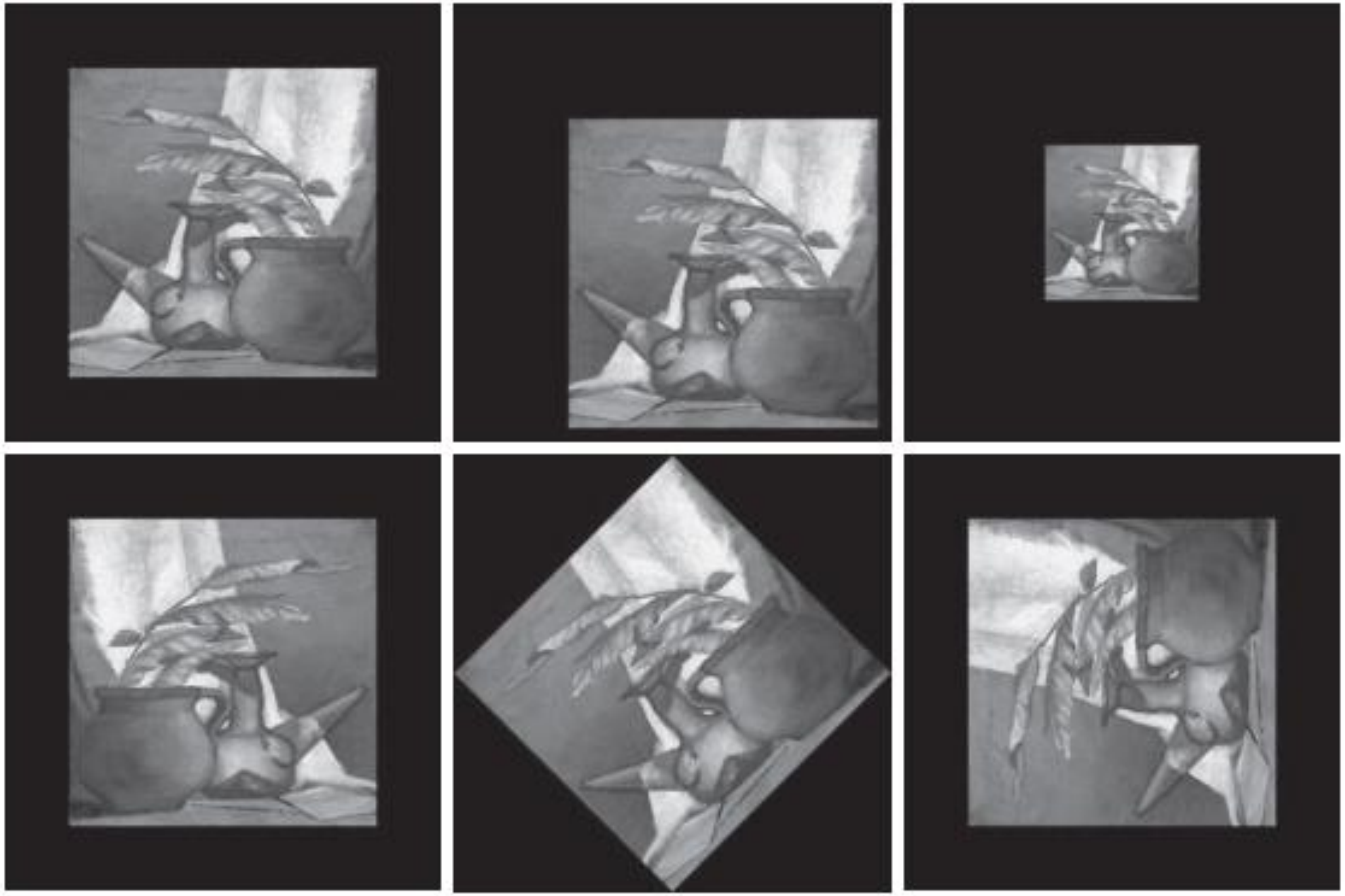
The $(p - q)$th normalized central moment is defined as

$$\eta_{pq} = \frac{M_{pq}}{M_{00}^{\beta}} \qquad \text{where } \beta = \frac{p+q}{2} + 1 \text{ and } p + q \geq 2$$

Hu Moments

$$
\begin{aligned}
\Lambda_1 &= \eta_{20} + \eta_{02} \\
\Lambda_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
\Lambda_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
\Lambda_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{30})^2 \\
\Lambda_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} - \eta_{03})^2] + (3\eta_{21} - \eta_{03})(\eta_{03} + \eta_{21}) \\
&\quad \times [3(\eta_{30} + \eta_{12})^2 - (\eta_{03} + \eta_{21})^2] \\
\Lambda_6 &= (\eta_{20} - \eta_{02})[(\eta_{12} + \eta_{30})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{21} + \eta_{03})(\eta_{12} + \eta_{30}) \\
\Lambda_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{03} + \eta_{21})^2] + (3\eta_{21} - \eta_{30})(\eta_{21} + \eta_{03}) \\
&\quad \times [3(\eta_{30} + \eta_{12})^2 - (\eta_{03} + \eta_{21})^2]
\end{aligned}
$$

Hu Moments



Hu Moments

TABLE 11.5
Moment invariants for the images in Fig. 11.37.

Moment Invariant	Original Image	Translated	Half Size	Mirrored	Rotated 45°	Rotated 90°
ϕ_1	2.8662	2.8662	2.8664	2.8662	2.8661	2.8662
ϕ_2	7.1265	7.1265	7.1257	7.1265	7.1266	7.1265
ϕ_3	10.4109	10.4109	10.4047	10.4109	10.4115	10.4109
ϕ_4	10.3742	10.3742	10.3719	10.3742	10.3742	10.3742
ϕ_5	21.3674	21.3674	21.3924	21.3674	21.3663	21.3674
ϕ_6	13.9417	13.9417	13.9383	13.9417	13.9417	13.9417
ϕ_7	−20.7809	−20.7809	−20.7724	20.7809	−20.7813	−20.7809

REGION DESCRIPTORS

REGION DESCRIPTORS

$$\text{compactness} = \frac{p^2}{A}$$

perimeter p, of a region is the length of its boundary

$$\text{circularity} = \frac{4\pi A}{p^2}$$

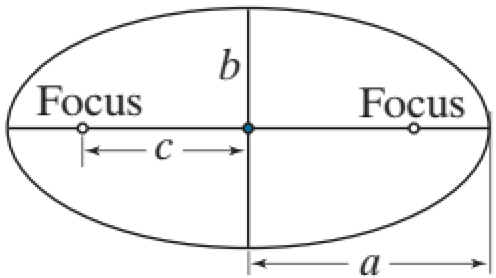
Area A, of a region is defined as the number of pixels in the region

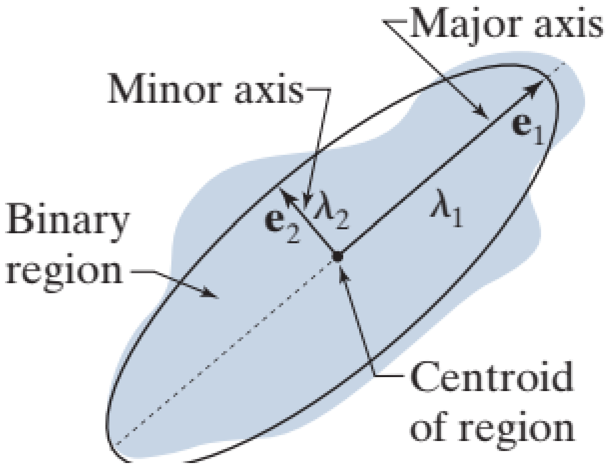
effective diameter

$$d_e = 2\sqrt{\frac{A}{\pi}}$$

$$c^2 = a^2 - b^2$$

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



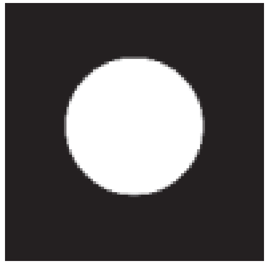
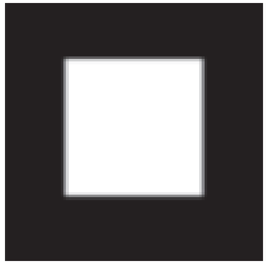


REGION DESCRIPTORS

Table 9.1 Some common, single-parameter descriptors for approximate shape in 2-D

Measure	Definition	Circle	Square	Rectangle as $b/a \rightarrow \infty$
Form factor	$\frac{4\pi \times \text{Area}}{\text{Perimeter}^2}$	1	$\pi/4$	$\rightarrow 0$
Roundness	$\frac{4 \times \text{Area}}{\pi \times \text{MaxDiameter}^2}$	1	$2/\pi$	$\rightarrow 0$
Aspect ratio	$\frac{\text{MaxDiameter}}{\text{MinDiameter}}$	1	1	$\rightarrow \infty$
Solidity	$\frac{\text{Area}}{\text{ConvexArea}}$	1	1	$\rightarrow 0$
Extent	$\frac{\text{TotalArea}}{\text{Area Bounding Rectangle}}$	$\pi/4$	1	indeterminate
Compactness	$\frac{\sqrt{(4 \times \text{Area})/\pi}}{\text{MaxDiameter}}$	1	$\sqrt{2/\pi}$	$\rightarrow 0$
Convexity	$\frac{\text{Convex Perimeter}}{\text{Perimeter}}$	1	1	1

REGION DESCRIPTORS - Example

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

REGION DESCRIPTORS - Example

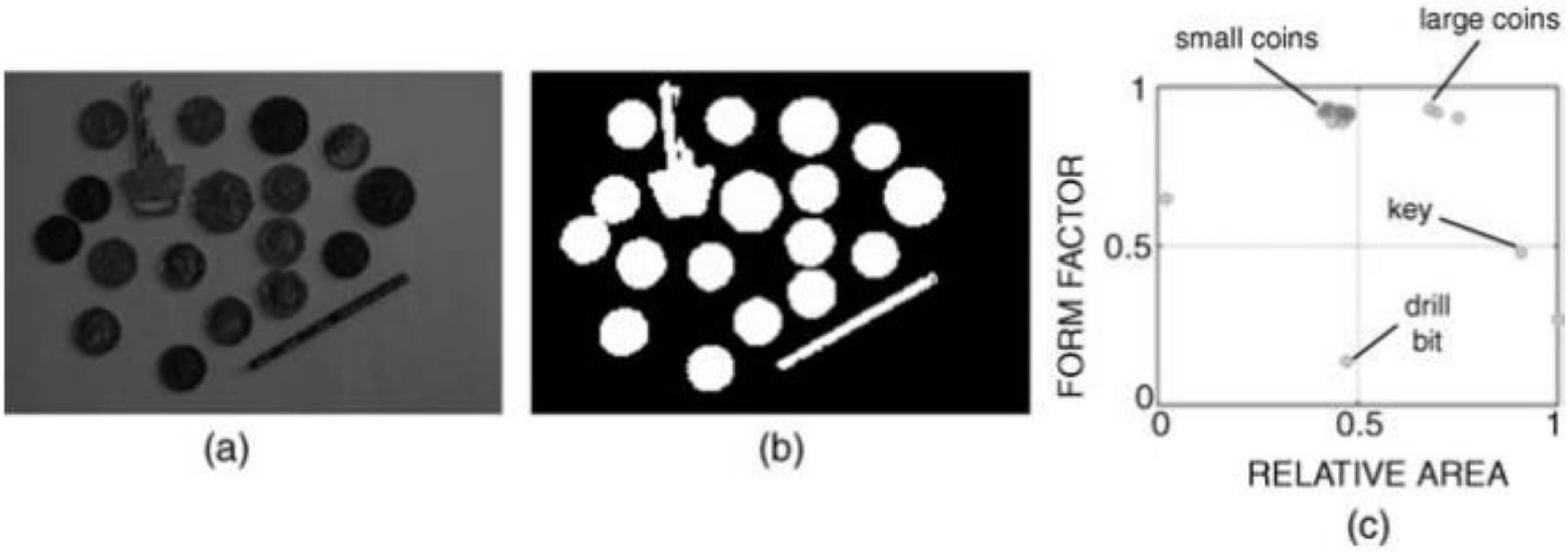


Figure 9.2 From left to right: (a) original image; (b) binary image after thresholding and morphological processing; (c) the normalized area and the form factor of each object in (b) are plotted in a 2-D feature space

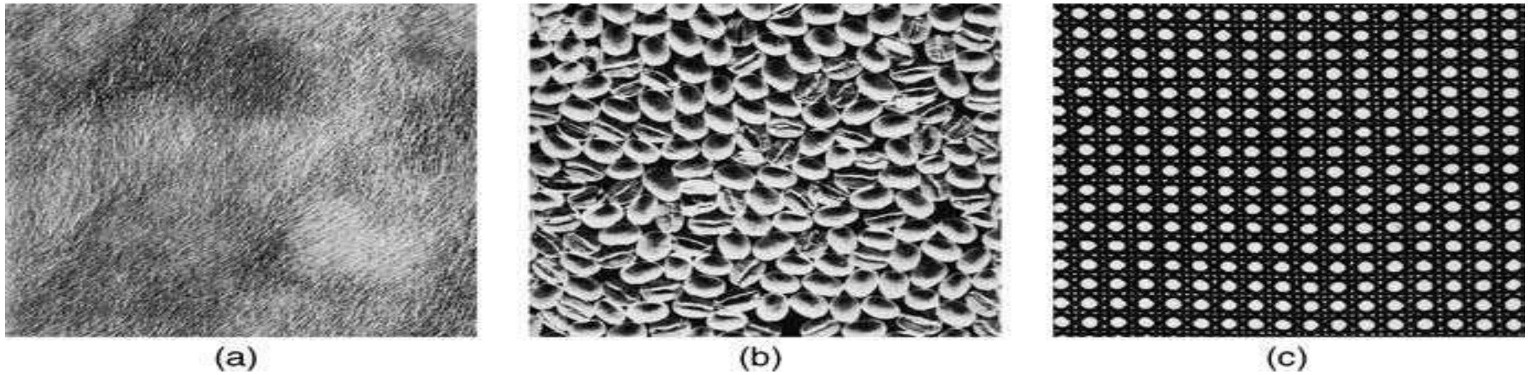
Texture Features/Descriptors

Texture Features

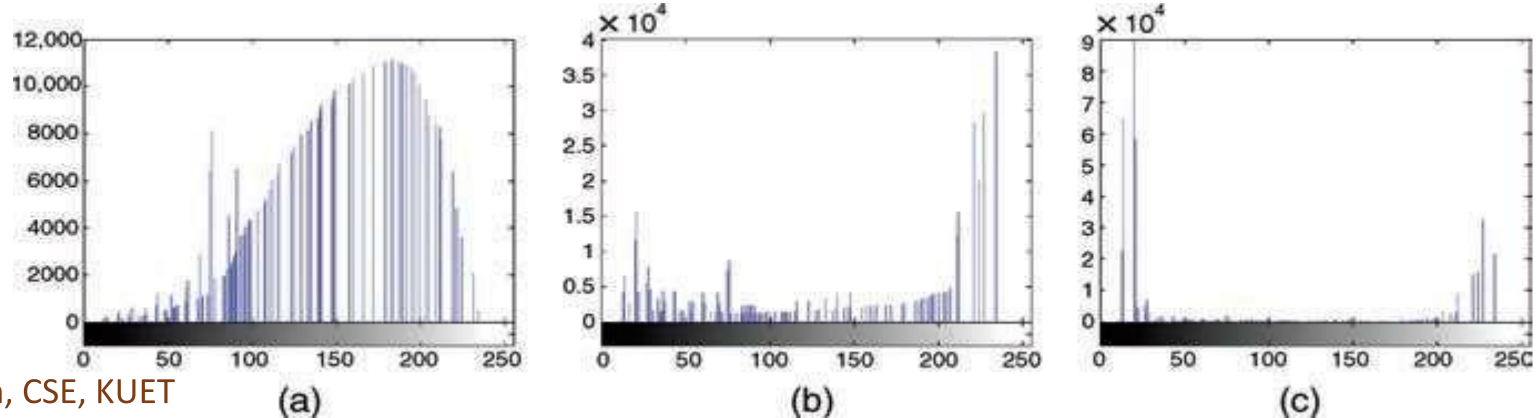
- Texture can be a powerful descriptor of an image (or one of its regions).
- Image processing techniques usually associate the notion of texture with image (or region) properties such as *Smoothness* (or its opposite, *roughness*), *Coarseness*, and *Regularity*.
- There are three main approaches to describe texture properties in image processing: Structural, Spectral, and Statistical.
- Most application focus on the statistical approaches, due to their popularity, usefulness and ease of computing.

Texture Features

- FIGURE 18.16 Example of images with smooth (a), coarse (b), and regular (c) texture. Images from the Brodatz textures data set.



- FIGURE 18.17 Histograms of images in Figure 18.16.



Texture Features

- One of the simplest set of statistical features for texture description consists of the following histogram-based descriptors of the image (or region): **mean**, **variance** (or its square root, the standard deviation), **skew**, **energy** (used as a measure of *uniformity*), and **entropy**, all of which were introduced in Section 18.5.
- The variance is sometimes used as a normalized descriptor of roughness (*R*), defined as

$$R = 1 - \frac{1}{1+\sigma^2}$$

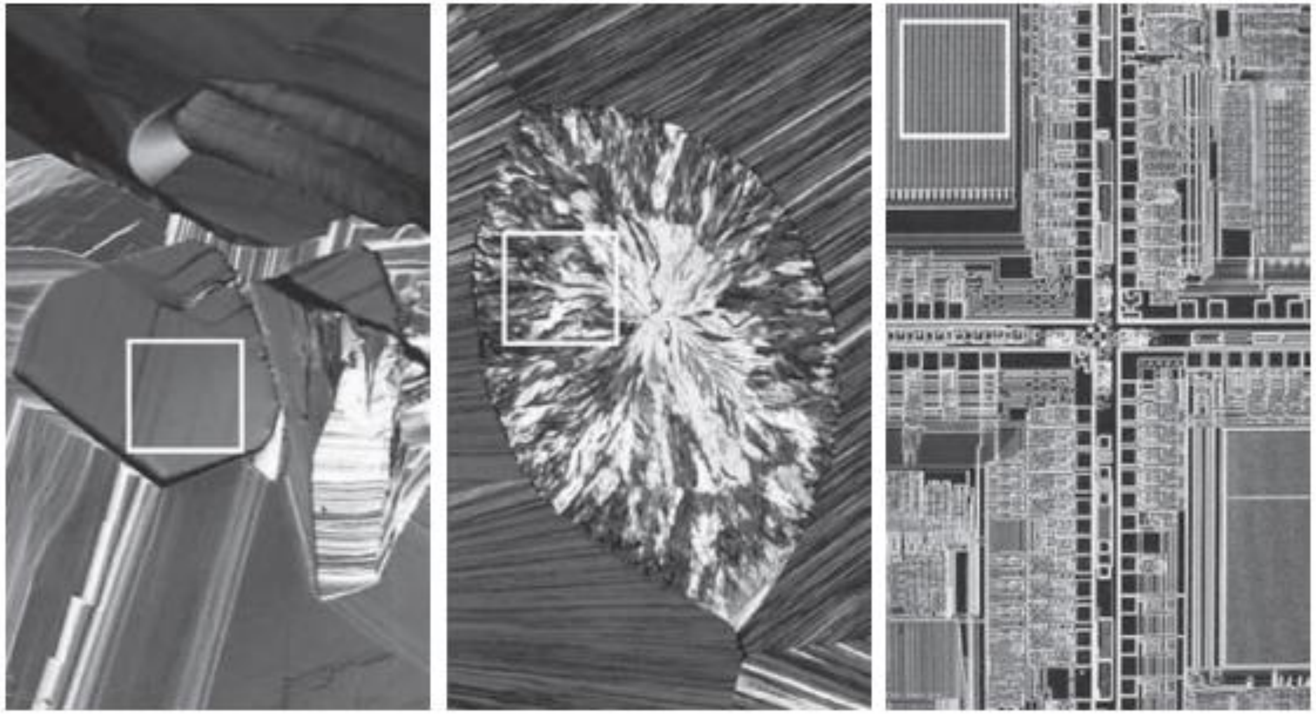
- Where, σ^2 is the normalized (to a [0, 1] interval) variance.
- *R* = 0 for areas of constant intensity, that is, smooth texture.

Texture Features

- **Highest uniformity has lowest entropy**

Texture	Mean	Standard deviation	Roughness R	Skew	Uniformity	Entropy
Smooth	147.1459	47.9172	0.0341	−0.4999	0.0190	5.9223
Coarse	138.8249	81.1479	0.0920	−1.9095	0.0306	5.8405
Regular	79.9275	89.7844	0.1103	10.0278	0.1100	4.1181

Texture Features



a b c

FIGURE 11.29 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. (Courtesy of Dr. Michael W. Davidson, Florida State University.)

Statistical texture measures for the subimages in Fig. 11.29.

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

Texture Features

- Histogram-based texture descriptors are limited by the fact that the histogram does not carry any information about the spatial relationships among pixels.
- One way to circumvent this limitation consists in using an alternative representation for the pixel values that encodes their relative position with respect to one another.
- One such representation is the ***gray-level co-occurrence matrix G, defined as a matrix whose*** element *g(i, j) represents the number of times that pixel pairs with intensities zi and zj occur in image f (x, y) in the position specified by an operator* ***d.***
- ***The vector d is*** known as *displacement vector:*

Gray level Co-occurrence Matrix

0	1	5	5	2	0
3	6	3	0	7	6
7	7	5	7	0	1
3	2	6	3	1	7
6	3	6	3	5	1
4	7	5	3	5	4

(a)

i \ j	0	1	2	3	4	5	6	7
0	0	2	0	0	0	0	0	1
1	0	0	0	0	0	1	0	1
2	1	0	0	0	0	0	1	0
3	1	1	1	0	0	2	2	0
4	0	0	0	0	0	0	0	1
5	0	1	1	1	1	2	0	0
6	0	0	0	4	0	0	0	0
7	1	0	0	0	0	2	1	1

(b)

Displacement, d = [1, 0]

Gray level Co-occurrence Matrix

- The gray-level co-occurrence matrix can be normalized as follows:

$$N_g(i,j) = \frac{g(i,j)}{\sum_i \sum_j g(i,j)}$$

where Ng(i, j) is the normalized gray-level co-occurrence matrix.

- Since all values of Ng(i, j) lie between 0 and 1, they can be thought of as the probability that a pair of points satisfying **d will have values (zi, zj).**
- Co-occurrence matrices can be used to represent the texture properties of an image.
- Instead of using the entire matrix, more compact descriptors are preferred.
- These are the most popular texture-based features that can be computed from a normalized gray-level co-occurrence matrix Ng(i, j):

GLCM - descriptors

$$\text{Maximum probability} = \max_{i,j} N_g(i, j)$$

$$\text{Energy} = \sum_i \sum_j N_g^2(i, j)$$

$$\text{Entropy} = -\sum_i \sum_j N_g(i, j) \log_2 N_g(i, j)$$

$$\text{Contrast} = \sum_i \sum_j (i - j)^2 N_g(i, j)$$

$$\text{Homogeneity} = \sum_i \sum_j \frac{N_g(i, j)}{1 + |i - j|}$$

$$\text{Correlation} = \frac{\sum_i \sum_j (i-\mu_i)(j-\mu_j) N_g(i, j)}{\sigma_i \sigma_j}$$

where μ_i, μ_j are the means and σ_i, σ_j are the standard deviations of the row and column sums $N_g(i)$ and $N_g(j)$, defined as

$$N_g(i) = \sum_j N_g(i, j) \qquad (18.35)$$

$$N_g(j) = \sum_i N_g(i, j) \qquad (18.36)$$

GLCM - descriptors

Descriptor	Explanation
Maximum probability	Measures the strongest response of **G**. The range of values is [0, 1].
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.
Contrast	A measure of intensity contrast between a pixel and its neighbor over the entire image. The range of values is 0 (when **G** is constant) to $(K-1)^2$.
Uniformity (also called Energy)	A measure of uniformity in the range [0, 1]. Uniformity is 1 for a constant image.
Homogeneity	Measures the spatial closeness to the diagonal of the distribution of elements in **G**. The range of values is [0, 1], with the maximum being achieved when **G** is a diagonal matrix.
Entropy	Measures the randomness of the elements of **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 $2\log_2 K$.

GLCM - Example

L = 256 and the position operator, d= “one pixel immediately to the right.

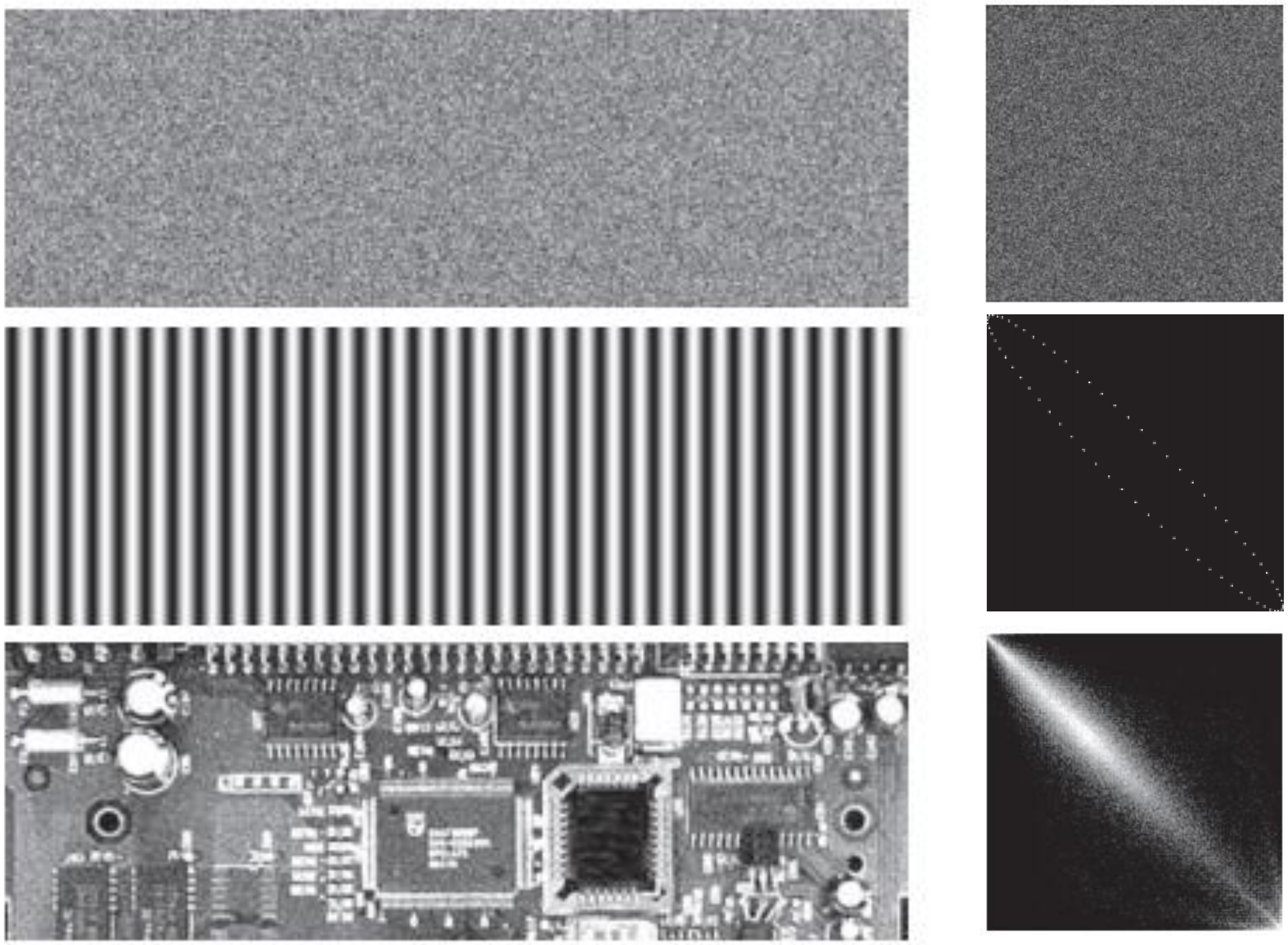
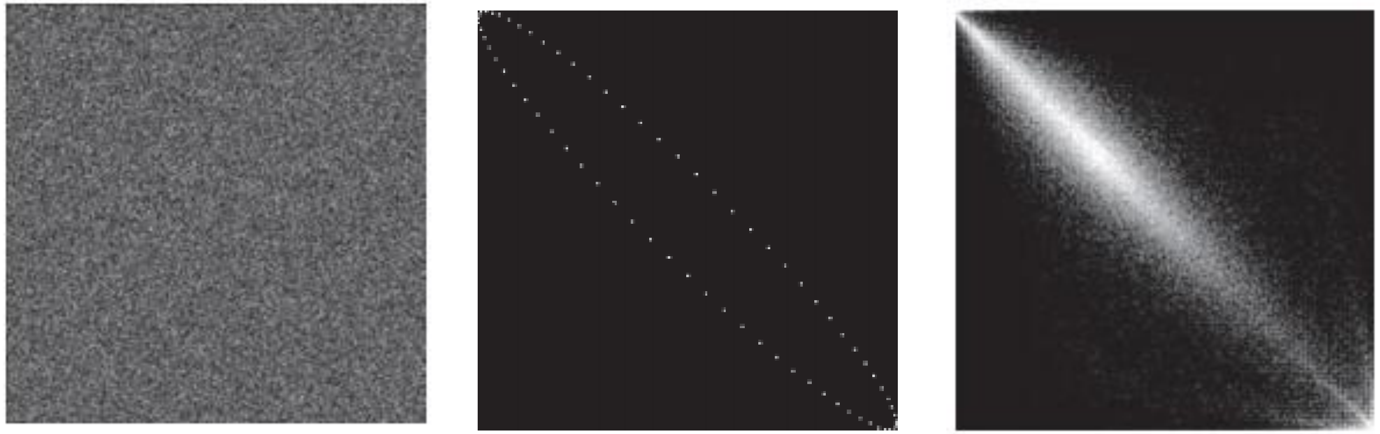


Image GLCM

GLCM - Example

L = 256 and the position operator, d= “one pixel immediately to the right.



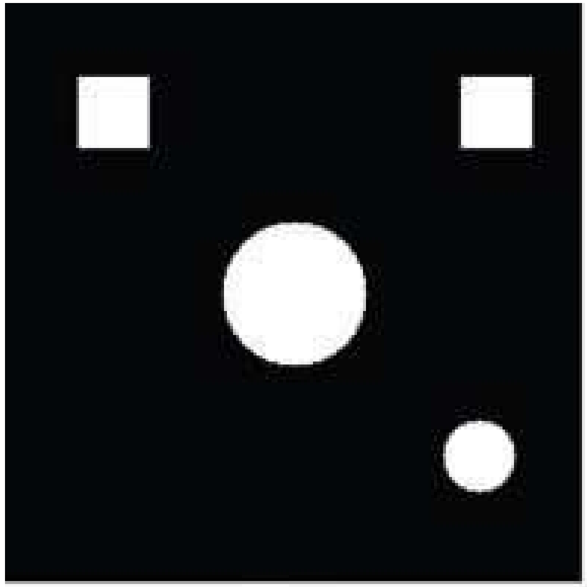
GLCM

Descriptors evaluated using the co-occurrence matrices displayed as images in Fig. 11.32.

Normalized Co-occurrence Matrix	Maximum Probability	Correlation	Contrast	Uniformity	Homogeneity	Entropy
$\mathbf{G}_1/n_1$	0.00006	−0.0005	10838	0.00002	0.0366	15.75
$\mathbf{G}_2/n_2$	0.01500	0.9650	00570	0.01230	0.0824	06.43
$\mathbf{G}_3/n_3$	0.06860	0.8798	01356	0.00480	0.2048	13.58

Example

- For the given binary image compute the descriptor values and fill them in the given table.



(a)

Table for feature

Object	Area	Centroid (row, col)	Orientation (degrees)	Euler number	Eccentricity	Aspect ratio	Perimeter	Thiness ratio
Top left square								
Big circle								
Small circle								
Top right square								

Question 2 *Do the results obtained for the extracted features correspond to your* expectations? Explain.

Question 3 *Which of the extracted features have the best discriminative power to* help tell squares from circles? Explain.

Question 4 *Which of the extracted features have the worst discriminative power* to help tell squares from circles? Explain.

Question 5 *Which of the extracted features are ST invariant, that is, robust to* changes in size and translation? Explain.

Question 6 *If you had to use only one feature to distinguish squares from circles,* in a ST-invariant way, which feature would you use? Why?

Other Features/Descriptors

- Harris Corner detector
- SIFT – Scale Invariant Feature Transform
- SURF – Speeded Up Robust Feature
- ORB – Oriented FAST and Rotated BRIEF

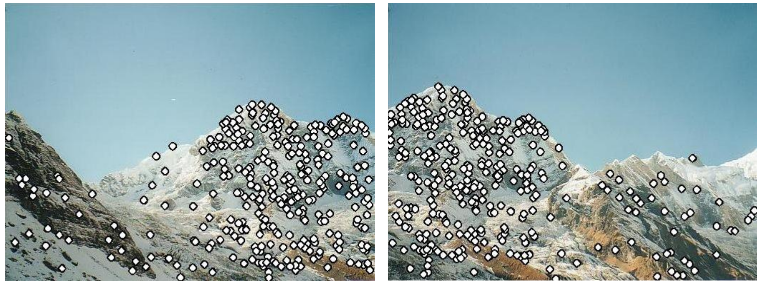
Feature Matching

Many slides from James Hays, Derek Hoiem, and Grauman&Leibe 2008 AAAI Tutorial

Local features: main components

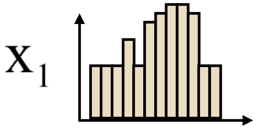
1) Detection:

Find a set of distinctive key points.

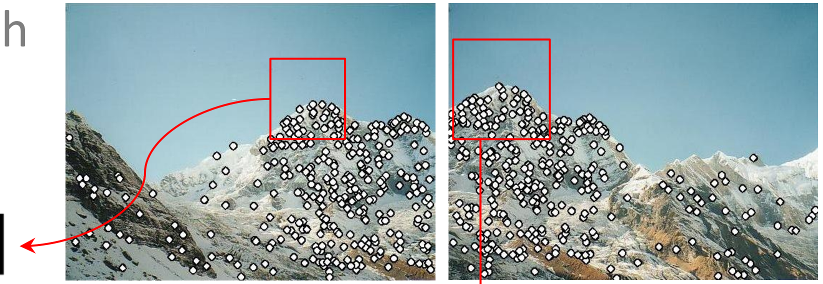


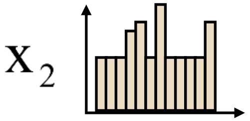
1) Description:

Extract feature descriptor around each interest point as vector.



$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$



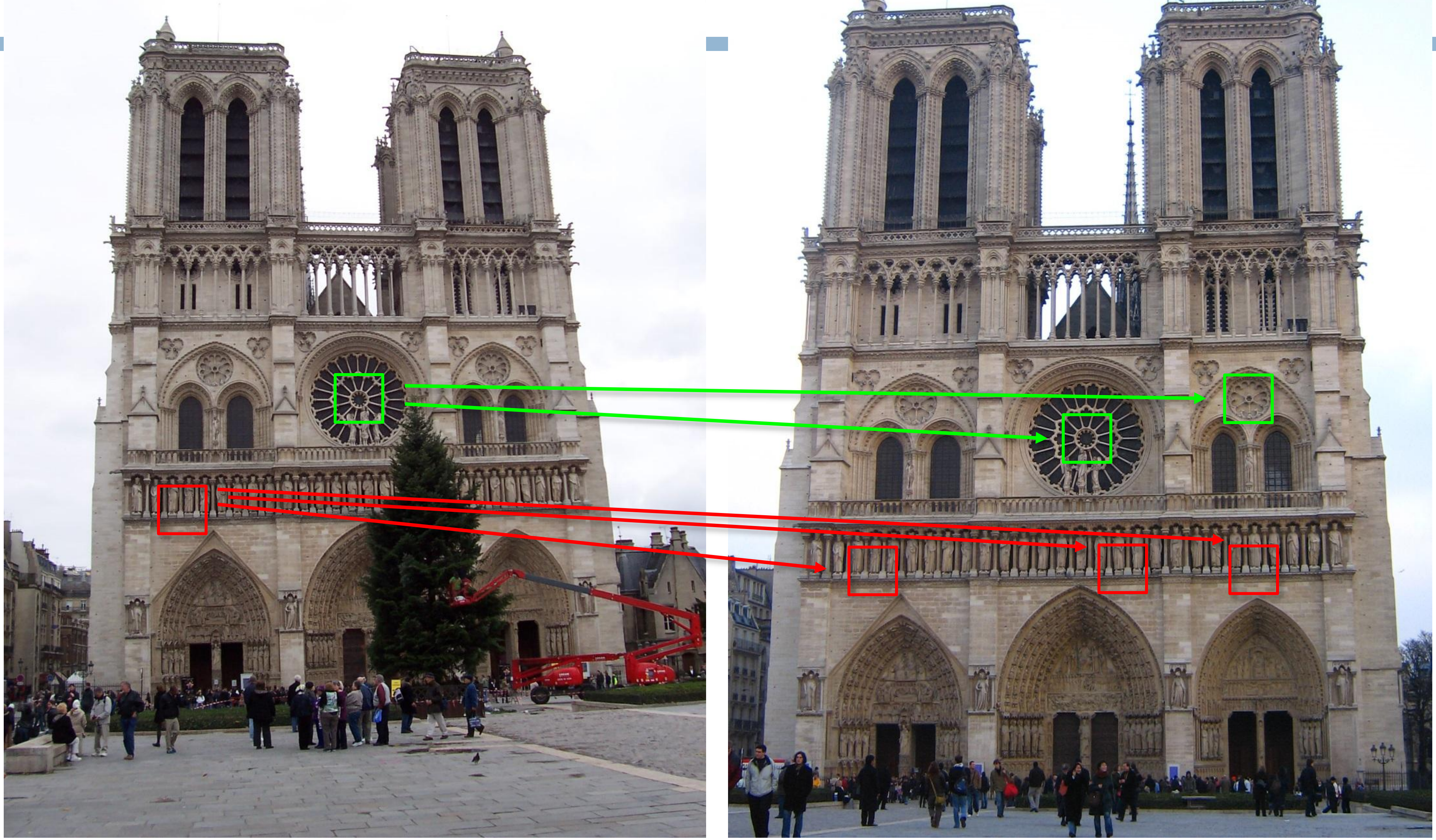


$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

1) Matching:

Compute distance between feature vectors to find correspondence.

How do we decide which features match?

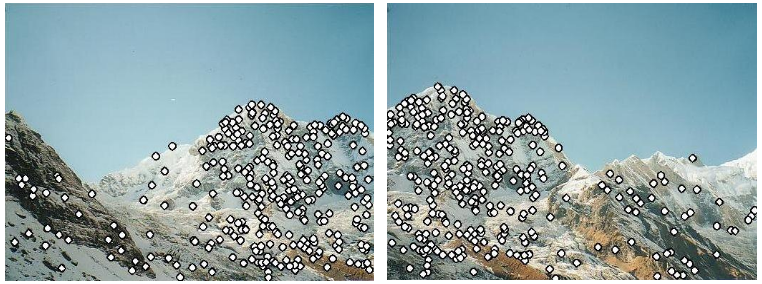


Distance: 0.34, 0.30, 0.40

Distance: 0.61, 1.22

Think-Pair-Share

- *Design a feature point matching scheme.*
- Two images, I_1 and I_2



- Two sets X_1 and X_2 of feature points
  - Each feature point $\mathbf{x}_1$ has a descriptor

$$\mathbf{x}_1 = [x_1^{(1)}, \dots, x_d^{(1)}]$$

- Distance, bijective/injective/surjective, noise, confidence, computational complexity, generality…

Euclidean distance vs. Cosine Similarity

- Euclidean distance:

$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) = \sqrt{(q_1-p_1)^2 + (q_2-p_2)^2 + \cdots + (q_n-p_n)^2}$$

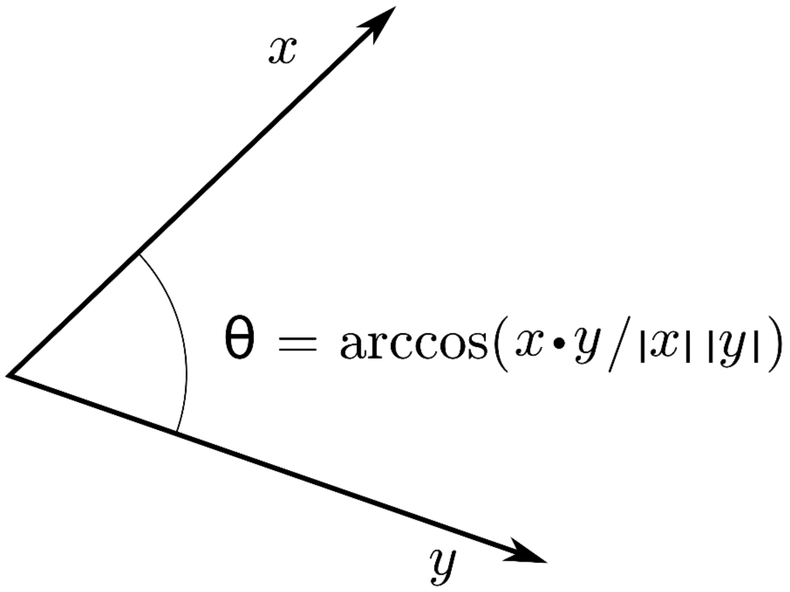
$$= \sqrt{\sum_{i=1}^{n}(q_i-p_i)^2}.$$

$$\|\mathbf{q}-\mathbf{p}\| = \sqrt{(\mathbf{q}-\mathbf{p})\cdot(\mathbf{q}-\mathbf{p})}.$$

- Cosine similarity:

$$\mathbf{a}\cdot\mathbf{b} = \|\mathbf{a}\|_2\|\mathbf{b}\|_2\cos\theta$$

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|_2\|\mathbf{B}\|_2}$$



Wikipedia

Other Distance Measures

Minkowski-form distance:

$$d_{L_r}(H, K) = \left(\sum_{\mathbf{i}} |h_{\mathbf{i}} - k_{\mathbf{i}}|^r \right)^{1/r}$$

Histogram intersection:

$$d_{\cap}(H, K) = 1 - \frac{\sum_{\mathbf{i}} \min(h_{\mathbf{i}}, k_{\mathbf{i}})}{\sum_{\mathbf{i}} k_{\mathbf{i}}} .$$

Kullback-Leibler divergence and Jeffrey divergence: The Kullback-Leibler (K-L) divergence [14] is defined as:

$$d_{KL}(H, K) = \sum_{\mathbf{i}} h_{\mathbf{i}} \log \frac{h_{\mathbf{i}}}{k_{\mathbf{i}}} .$$

χ^2 statistics:

$$d_{\chi^2}(H, K) = \sum_{\mathbf{i}} \frac{(h_{\mathbf{i}} - m_{\mathbf{i}})^2}{m_{\mathbf{i}}} , \qquad m_{\mathbf{i}} = \frac{h_{\mathbf{i}} + k_{\mathbf{i}}}{2}$$

Other Distance Measures

Quadratic-form distance: this distance was suggested for color based retrieval in [17]:

$$d_A(H,K) = \sqrt{(\mathbf{h}-\mathbf{k})^T \mathbf{A}(\mathbf{h}-\mathbf{k})} \ ,$$

Match distance:

$$d_M(H,K) = \sum_i |\hat{h}_i - \hat{k}_i| \ ,$$

Kolmogorov-Smirnov distance:

$$d_{KS}(H,K) = \max_i(|\hat{h}_i - \hat{k}_i|) \ .$$

$$\mathrm{EMD}(P,Q) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} d_{ij} f_{ij}}{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}} \ ,$$

Feature Matching

- Criteria 1:
  - Compute distance in feature space, e.g., Euclidean distance between (eg. 128-dim SIFT) descriptors
  - Match point to lowest distance (nearest neighbor)

- Problems:
  - Does everything have a match?

Feature Matching

- Criteria 2:
  - Compute distance in feature space, e.g., Euclidean distance between (eg. 128-dim SIFT) descriptors
  - Match point to lowest distance (nearest neighbor)
  - Ignore anything higher than threshold (no match!)

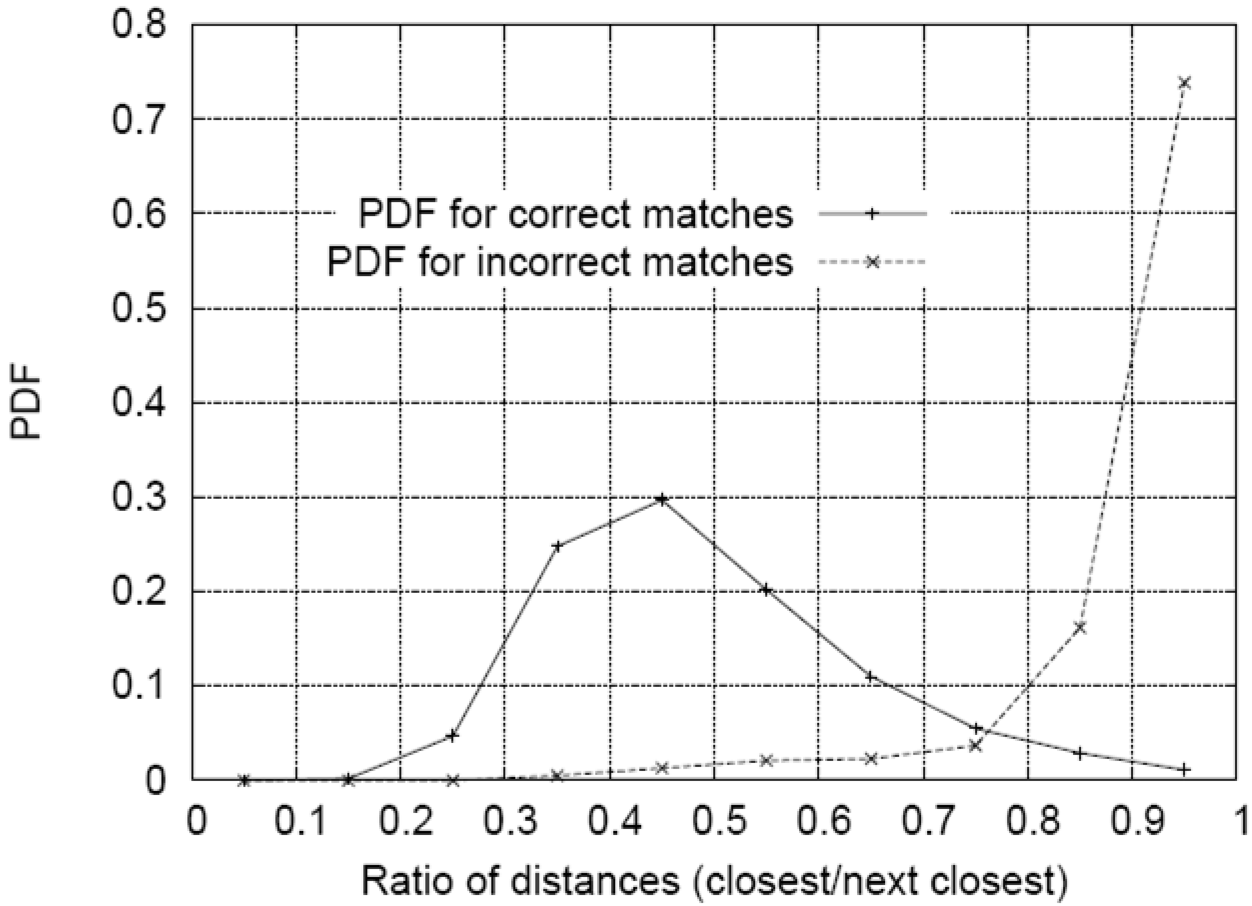
- Problems:
  - Threshold is hard to pick
  - Non-distinctive features could have lots of close matches, only one of which is correct

Nearest Neighbor Distance Ratio

- *Compare distance of closest (NN1) and second-closest (NN2) feature vector neighbor.*
  - If NN1 ≈ NN2, ratio $\frac{NN1}{NN2}$ will be ≈ 1 -> matches too close.
  - As NN1 << NN2, ratio $\frac{NN1}{NN2}$ tends to 0.
- Sorting by this ratio puts matches in order of confidence.
- Threshold ratio – but how to choose?

Nearest Neighbor Distance Ratio

- Lowe computed a probability distribution functions of ratios
- 40,000 keypoints with hand-labeled ground truth



Ratio threshold depends on your application's view on the trade-off between the number of false positives and true positives!

Lowe IJCV 2004

Efficient compute cost

- Naïve looping: Expensive

- Operate on matrices of descriptors
- E.g., for row vectors,

  `features_image1 * features_image2`^{T}

  produces matrix of dot product results
  for all pairs of features (cosine similarity).
  What can we do for Euclidean distance?

Ref.

- Fundamentals of Digital Image Processing
  - Chris Solomon, Toby Breckon
- https://www.slideshare.net/Jaddu44/image-feature-extraction