Chapter 8.1

Representation and Description

Image Processing and Computer Vision

LE Thanh Sach
Faculty of Computer Science and Engineering
Ho Chi Minh University of Technology, VNU-HCM

Overview

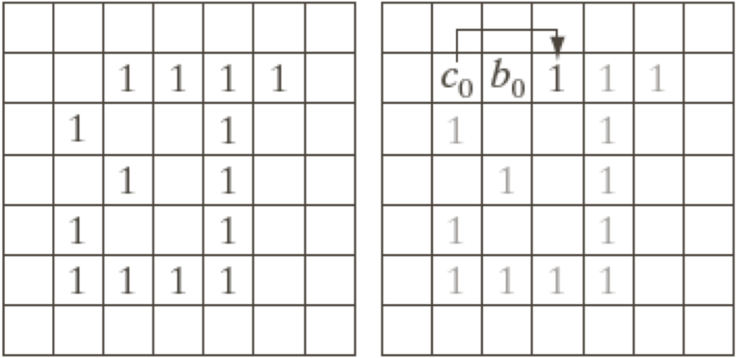
1. **Representation**
   - Boundary following
   - Chain codes
   - Signatures
2. **Boundary Descriptors**
   - Basic Descriptors
   - Fourier Descriptors
3. **Regional Descriptors**
   - Simple Descriptors
   - Topological Descriptors
   - Texture
   - Spectral Descriptors
4. **Moments**
5. **Principal Components for Description**
   - Principal Components

Representation: Boundary Following

Algorithm:

Step 1: Initialization

- b_0: The uppermost and leftmost point belonging to the region
- c_0: The west neighbor of b_0

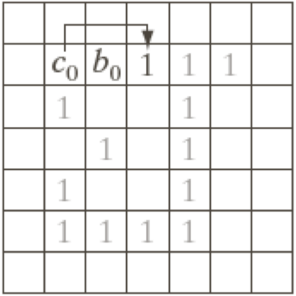


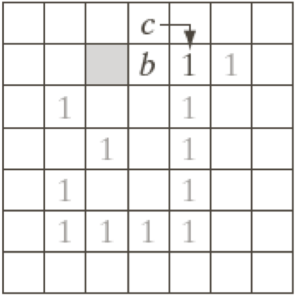
Representation: Boundary Following

Algorithm:

Step 2: Finding next point

- Let the 8-neighbors of b_0, starting at c_0 and proceeding in the clockwise direction be denoted as $n_1, n_2, .., n_8$
- Find the first n_k in this sequence such that $n_k = 1$
- Update $b_1 = n_k$ and $c_1 = n_{k-1}$



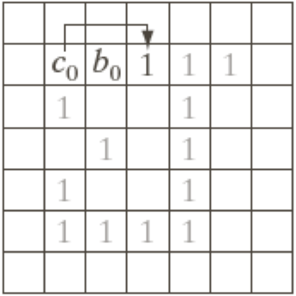


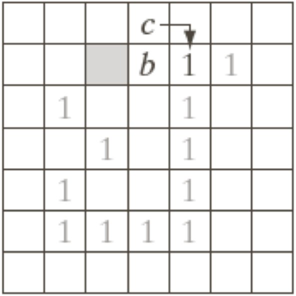
Representation: Boundary Following

Algorithm:

Step 3: Finding next point

- Let $b = b_1$ and $c = c_1$
- Let the 8-neighbors of b, starting at c and proceeding in the clockwise direction be denoted as $n_1, n_2, .., n_8$
- Find the first n_k in this sequence such that $n_k = 1$
- Update $b = n_k$ and $c = n_{k-1}$



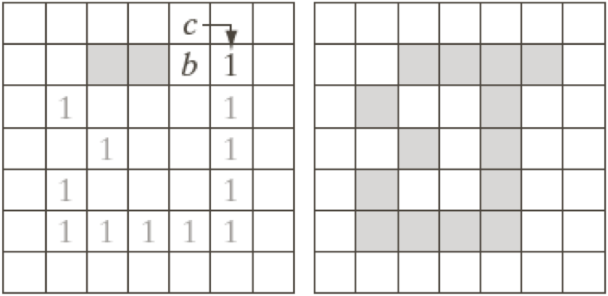


Representation: Boundary Following

Algorithm:

Step 4: Iteration

- Repeat **Step 3** until
  1. $b = b_0$ **AND**
  2. The next boundary point of b is b_1

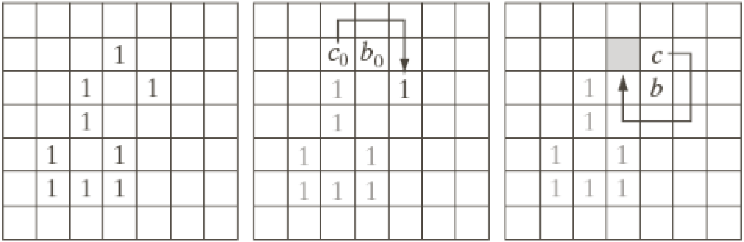


Representation: Boundary Following

Algorithm:

Questions

1. How does the algorithm work, if the stopping condition is only $b = b_0$, as shown in the following figure?
2. How can we detect the boundary of holes, i.e., zero-values surrounded with 1-values.

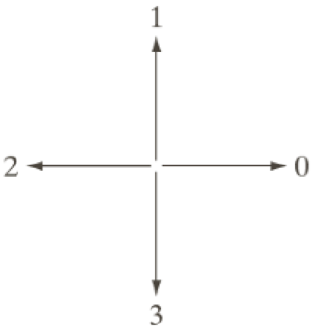


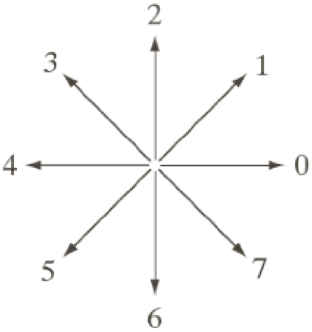
Representation: Chain codes

Definition

Feeman chain codes $\equiv$ **A sequence of straight-line segments** of specified length and direction.

Each segment, as shown below, is denoted a number according 4- or 8-connectivity:





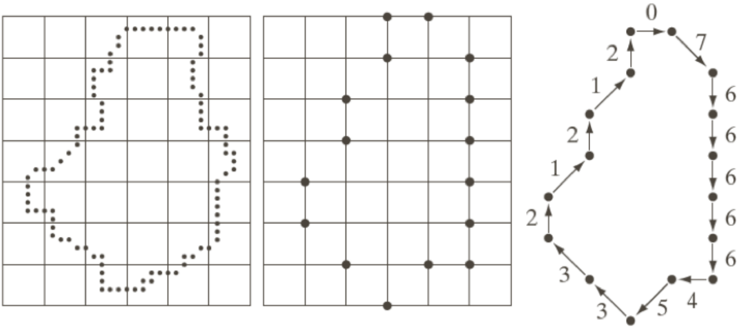
Representation: Chain codes

Algorithm for extracting chain codes

1. Re-sample the boundary by using a larger grid spacing
2. For each boundary point (during boundary traversing)
    1. Obtain the node in the larger grid that is nearest to the current boundary point.
    2. If this node is different to the node in previous step,
        - Compute a code using 4- or 8-connectivity from the node in the previous step to the current node
        - Output this code.

Representation: Chain codes

A example



Representation: Chain codes

Problems facing with chain codes

The chain code of a boundary depends on the starting point.

Solutions:

1. Sequence of codes forming the minimum integer
   - Consider the chain code as a circular sequence.
   - Re-define the starting point so that the sequence forms an integer with minimum magnitude.
2. Sequence of differences
   - Difference $\equiv$ number of direction changes (e.g., in counterclockwise) between two adjacent codes.

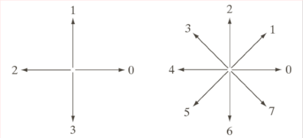
Representation: Chain codes

Problems facing with chain codes

The chain code of a boundary depends on the starting point.

Solutions:

2. Sequence of differences
   - Difference ≡ number of direction changes (e.g., in counterclockwise) between two adjacent codes. For 4-connectivity:



- From 1 to 0 ≡ 3
- From 1 to 3 ≡ 2
- From 2 to 0 ≡ 2

Representation: Chain codes

Problems facing with chain codes

The chain code of a boundary depends on the starting point.

Solutions:

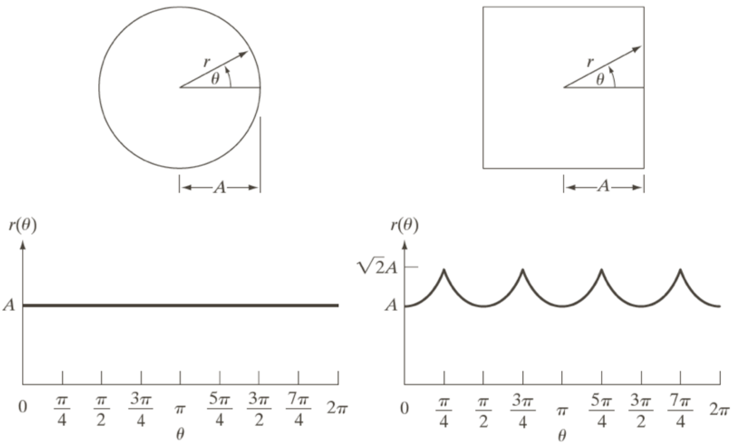
2. Sequence of differences
3. Represent the chain code by a sequence of differences instead of the chain code itself.
4. The first difference $\equiv$ the difference between the last and the first components in the original. For 4-connectivity:
    - $10103322 \rightarrow 33133030$

Representation: Chain codes

Questions

1. How can you extract the outer and the inner boundary of the ring in the figure? Note: there are noise in the input image.
2. How to extract the chain code that is invariant to rotation or orientation?

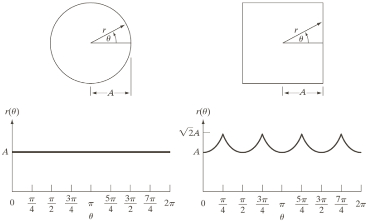
Representation: Signature



Ideas

Convert 2D boundary to 1D time series data.

Representation: Signature



A Method: Distance from the centroid

1. Determine the centroid of the boundary
2. For each angle θ (discretized angle)
    - Compute the distance from the centroid to the boundary point on the line oriented angle θ
    - Output this distance

Representation: Signature

Problem facing with signature

1. Signatures depend on the starting point, not invariant to orientation (rotation)

Solutions:

1. Starting point $\equiv$ The farthest point from the centroid (assume that it is unique)
2. Starting point $\equiv$ The farthest point on eigen-axis

Representation: Signature

Problem facing with signature

2. The magnitude of signatures depends on image scaling

Solutions:

1. Normalize the magnitude to a certain range, for examples, $[0, 1]$

Representation: Signature

Another Method: Histogram of angles

Input: **A reference line** L_{ref}

1. Extract boundary points
2. For each point p on the boundary
   - Compute tangent line $L_{tangent}$ to the boundary at p
   - Compute the angle θ between $L_{tangent}$ and L_{ref}
   - Output θ

- $L_{ref} \equiv Ox \Rightarrow$ The method
- $\equiv$ Histogram of tangent angle values
- $\equiv$ Slop-density function
- $\equiv$ Histogram of orientation for boundary points (HOG)

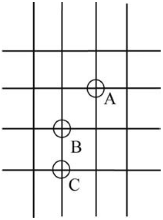
Basic Boundary Descriptors

Perimeter

Perimeter is the length of boundary.

Approximation methods:

1. $\equiv$ The number of pixels on the boundary.
2. $\equiv N_e + N_o\sqrt{2}$
   - N_e : Number of even codes in chain code representation
   - N_o : Number of odd codes in chain code representation



- distance(B,C) $= 1$; but, distance(A,B) $= \sqrt{2}$

Basic Boundary Descriptors

Diameter

Diameter is the largest distance between any two points on a boundary.

Mathematical Definition:

- $Diam(B)$: Diameter of boundary B
- $D(p_i, p_j)$: The distance between two points p_i and p_j on the boundary B

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

Basic Boundary Descriptors

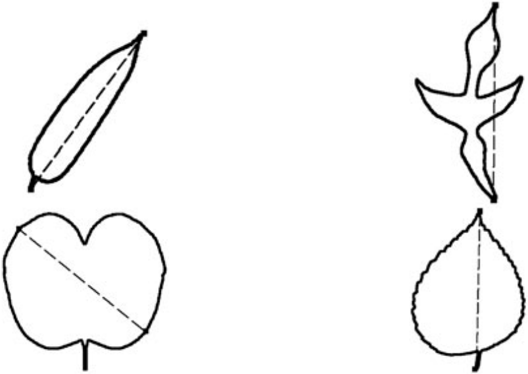


Figure: Diameter demonstration (Source: Book on Shape Classification and Analysis)

Basic Boundary Descriptors

Major and Minor Axes

- **Major Axis** : The direction along which the shape points are more dispersed. The distance between two farthest points on the major axis is the **diameter**.

- **Minor Axis** : The direction that is perpendicular to the major axis

Basic Boundary Descriptors

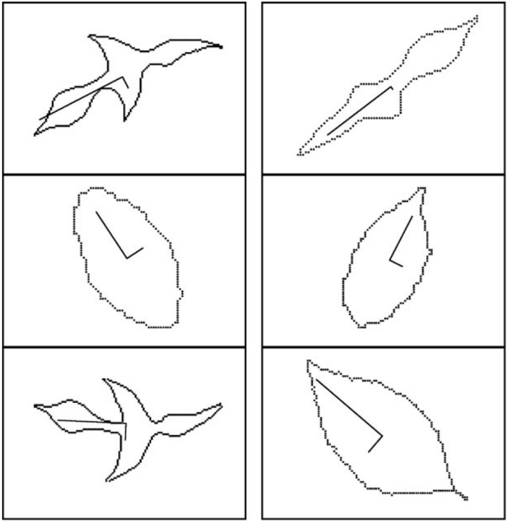


Figure: Demonstration of Major and Minor axes (Source: Book on Shape Classification and Analysis)

Basic Boundary Descriptors

Algorithm for obtaining major and minor axes

1. Extract boundary points, using boundary following algorithm. Assume that there are n boundary points
2. Store n boundary points into matrix X of size $n \times 2$
   - Each boundary point $p(x_p, y_p)$ is stored into a row in X
3. Compute covariance matrix K of X.
4. Calculate the eigenvectors and eigenvalues of K
   - Major axis is the eigenvector corresponding to the largest eigenvalue.
   - Minor axis is the remaining eigenvector.

Basic Boundary Descriptors

Basic rectangle and Eccentricity

- **Basic rectangle** : The rectangle is a smallest rectangle that is aligned with the major and the minor axes and completely encloses the boundary
- **Eccentricity** :

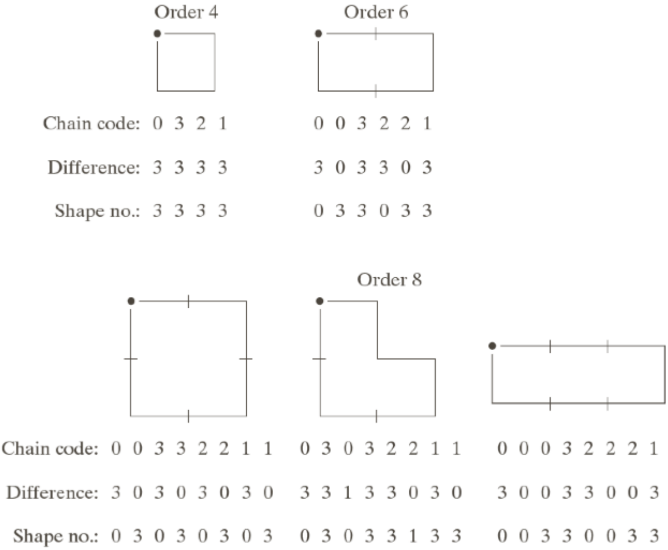
$$E = \frac{\text{length(major axis)}}{\text{length(minor axis)}}$$

Basic Boundary Descriptors

Shape number

- **Shape number** : The first difference (see chain-code) of smallest magnitude
- **Order of shape** :The order of shape $\equiv$ The number of digits in the shape number

Basic Boundary Descriptors



Dot is the starting point

Basic Boundary Descriptors

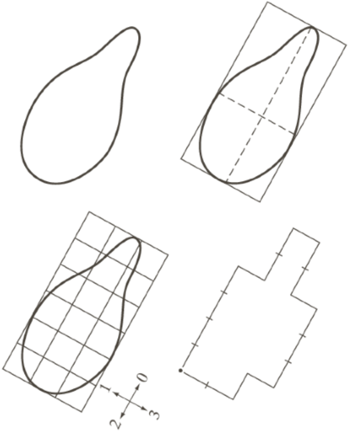
Questions

Given a order, How do we obtain encode a shape in this order?

Solutions

1. Determine the basic rectangle of the shape.
2. Calculate the eccentricity of the basic rectangle, called E_1
3. From the input order, find a rectangle, called R, that has the order and best approximates E_1.
   - Input order $n = 12$
   - $\Rightarrow n = 1 \times 12$ $(E = 12)$; $n = 2 \times 6$ $(E = 3)$; $n = 3 \times 4$ $(E = \frac{4}{3})$, etc.
4. Use R to create the grid size.
5. Use this grid to generate first difference and shape number.

Basic Boundary Descriptors



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

Figure: Demonstration for generating shape number from an order

Fourier Descriptors

Input

- A shape

Method

1. Extract boundary points: $(x_0, y_0), (x_1, y_1)$, etc
2. Define a sequence of complex numbers:

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

   - $x(k) = x_k; y(k) = y_k$
   - $k = 0, 1, .., N-1$

Fourier Descriptors

Method

3. Perform Discrete Fourier Transform (DFT)

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

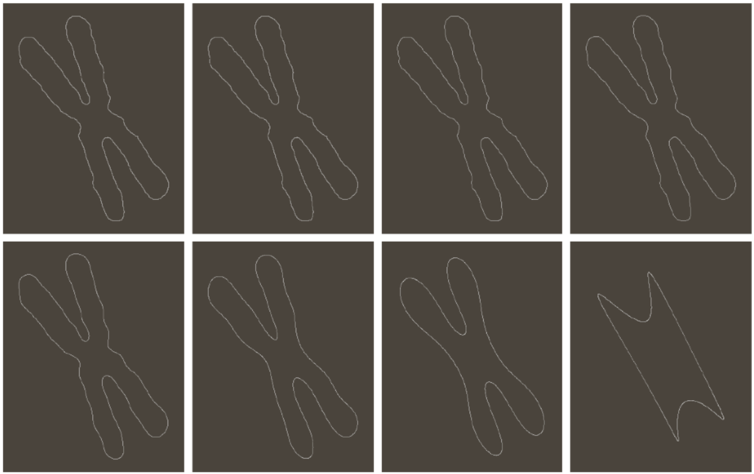
- $u = 0, 1, .., N-1$
- $a(u)$ are **Fourier Descriptors**

Inverse DFT

$s(k)$ can be approximated from the first P Fourier Descriptors as follows:

$$\hat{s}(k) = \frac{1}{P} \sum_{u=0}^{P-1} a(u) e^{j2\pi uk/N}$$

Fourier Descriptors: Example



a b c d
e f g h

FIGURE 11.20 (a) Boundary of 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.

Fourier Descriptors

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	$s_p(k) = s(k - k_0)$	$a_p(u) = a(u)e^{-j2\pi k_0 u/K}$

TABLE 11.1 Some basic properties of Fourier descriptors.

Regional Simple Descriptors

- **Perimeter**: defined in previous section
- **Area**: The number of pixels in the shape
- **Compactness**:

$$\textbf{Compactness} = \frac{\textbf{Perimeter}^2}{\textbf{Area}}$$

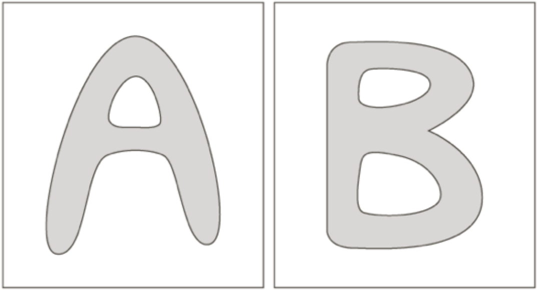
- **Circularity Ratio**: The ratio of the area of the input region to the area of the circle that has the same perimeter with the input region

$$R_c = \frac{4\pi A}{P^2}$$

  - A: Area of the region.
  - P: Perimeter of the region.
  - $R_c = 1 \Rightarrow$ Circle; $R_c = \pi/4 \Rightarrow$ Square;

Topological Descriptors

- **Input**: A region has H holes and C connected components
- **Euler number**: $E = C - H$



a b

FIGURE 11.25 Regions with Euler numbers equal to 0 and −1, respectively.

Topological Descriptors

- **Input**: A Region represented by straight-line segments
  1. F : Number of faces
  2. V : Number of vertices
  3. Q : Number of edges
- **Euler number**:

$$
\begin{aligned}
E &= C - H \\
&= V - Q + F
\end{aligned}
$$

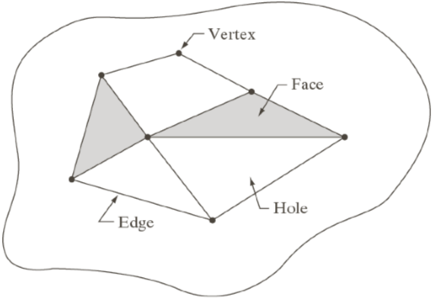


FIGURE 11.26 A region containing a polygonal network.

Texture: Statistical Approaches

Histogram-based descriptors:

Input:

- An image or a region

A Method

1. Compute histogram
   $p(z) = [p(z_0), p(z_1), p(z_i).., p(z_{L-1})]$
   - $i = 0, 1, ..., L-1$ ($L = 256$ for gray image)
2. Compute the mean of z

$$m = \sum_{i=0}^{L-1} z_i p(z_i)$$

Texture: Statistical Approaches

Histogram-based descriptors:

A Method

3. Compute the nth moments about the mean m

$$\mu_n = \sum_{i=0}^{L-1} (z_i - m)^n p(z_i)$$

Texture: Statistical Approaches

Histogram-based descriptors:

Moments' meaning and properties

1. $n = 0$: $\mu_0 = 1$
2. $n = 1$: $\mu_1 = 0$
3. $n = 2$: $\mu_2 = \sigma^2$: measure the variance of z (intensities)
4. **Relative Smoothness** R is defined as

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

- $R = 0$: for region of constant intensity
- $R \to 1$ (approaching 1) for large variance σ^2
- σ^2 should be normalized to 1 by $\sigma^2 \leftarrow \sigma^2/(L-1)^2$

Texture: Statistical Approaches

Histogram-based descriptors:

Moments' meaning and properties

5. $n = 3$: μ_3 measures the skewness of the region's histogram.
6. $n = 4$: μ_4 measures the flatness of the region's histogram.
7. $n \geq 5$: still provide further **discriminative** information of texture content.

Texture: Statistical Approaches

Histogram-based descriptors:

More descriptors using histogram

Uniformity U:

$$U = \sum_{i=0}^{L-1} p(z_i)^2$$

- U is maximum for images in which all intensities are equal.

Texture: Statistical Approaches

Histogram-based descriptors:

More descriptors using histogram

Average Entropy E:

$$E = -\sum_{i=0}^{L-1} p(z_i) \log_2 p(z_i)$$

- E measures the variation of intensities in images. $E = 0$ for constant images.

Texture: Statistical Approaches

Histogram-based descriptors:

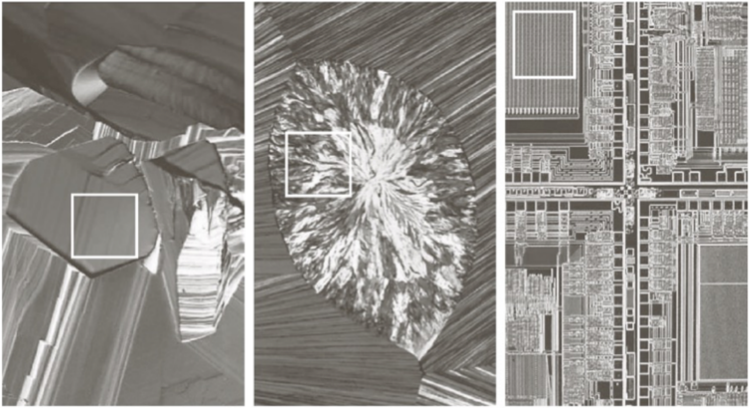


Figure: Three kinds of texture in sub-images

Texture: Statistical Approaches

Histogram-based descriptors:

Texture	Mean	Standard deviation	R (normalized)	Third 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

Figure: Measurements for the previous three kinds of textures

.

Find the **discrimination** on the above measurements

Texture: Statistical Approaches

Relative positions of pixels:

Input:

- An input image or a region
- Q an operator that defines the position of two pixels relative to each other.
- L: number of intensity levels.
- Intensities are transformed to range of $[1, L]$ instead of $[0, L-1]$

Texture: Statistical Approaches

Relative positions of pixels:

Concept

Co-occurrence matrix: $G = \{g_{ij}\}_{L \times L}$

- G has size of $L \times L$
- $i, j \in [1, L]$

Meaning of g_{ij}: g_{ij} is the number of pairs of two pixels p and q that satisfy the following predicates

- p and q satisfy operator Q, for examples, they are two consecutive pixels in the input image
- Gray level of pixel p is i
- Gray level of pixel q is j

Texture: Statistical Approaches

Relative positions of pixels:

Examples

- Q : one pixel immediately to the right
- $L = 8$
- Input image f of small size, 6×6

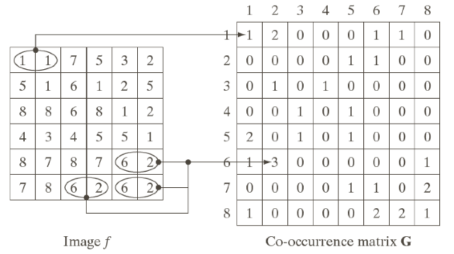


Image f

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

Co-occurrence matrix **G**

	1	2	3	4	5	6	7	8
1	1	2	0	0	0	1	1	0
2	0	0	0	0	1	1	0	0
3	0	1	0	1	0	0	0	0
4	0	0	1	0	1	0	0	0
5	2	0	1	0	1	0	0	0
6	1	3	0	0	0	0	0	1
7	0	0	0	0	1	1	0	2
8	1	0	0	0	0	2	2	1

Figure: Demonstration of building co-occurrence matrix

Texture: Statistical Approaches

Relative positions of pixels:

Normalized Co-occurrence Matrix: G_{norm}

$$G_{norm} = \{p_{ij} = \frac{g_{ij}}{n}\}_{L \times L}$$

- n : the total number of pairs that satisfy Q

$$n = \sum_{i=1}^{L} \sum_{j=1}^{L} g_{ij}$$

- p_{ij} : probability of occurring a pair of two pixels p and q that satisfy the positional operator Q and that they have gray-levels i and j respectively.

Texture: Statistical Approaches

Relative positions of pixels:

Descriptors bases on G_{norm}

1. **Maximum Probability** :

$$\max_{(i,j)}\{p_{ij}\}$$

   - Measures the strongest response of the co-occurrence

Relative positions of pixels:

Descriptors bases on G_{norm}

❷ Correlation :

$$\sum_{i=1}^{K}\sum_{j=1}^{K}\frac{(i-m_r)(j-m_c)p_{ij}}{\sigma_r\sigma_c}$$

- Measures how correlated a pixel is to its neighbors over the entire image
- Range of values: $[-1, 1]$, i.e., perfect negative and perfect positive.
- $\sigma_r \neq 0$ and $\sigma_c \neq 0$

Texture: Statistical Approaches

Relative positions of pixels:

Descriptors bases on G_{norm}

2. **Correlation** :

$$m_r = \sum_{i=1}^{K} i \sum_{j=1}^{K} p_{ij}$$

$$m_c = \sum_{j=1}^{K} j \sum_{i=1}^{K} p_{ij}$$

$$\sigma_r = \sum_{i=1}^{K} (i - m_r)^2 \sum_{j=1}^{K} p_{ij}$$

$$\sigma_c = \sum_{j=1}^{K} (j - m_c)^2 \sum_{i=1}^{K} p_{ij}$$

Texture: Statistical Approaches

Relative positions of pixels:

Descriptors bases on G_{norm}

3. **Contrast** :

$$\sum_{i=1}^{K}\sum_{j=1}^{K}(i-j)^2 p_{ij}$$

- Measures the intensity contrast between a pixel to its neighbors over the entire image.
- Range of values: $[0, (K-1)^2]$
- Contrast $= 0 \equiv G$ is constant.

Texture: Statistical Approaches

Relative positions of pixels:

Descriptors bases on G_{norm}

4. **Uniformity** :

$$\sum_{i=1}^{K}\sum_{j=1}^{K} p_{ij}^2$$

- Measures the uniformity
- Range of values: $[0, 1]$
- Uniformity $= 1$ for constant images

Texture: Statistical Approaches

Relative positions of pixels:

Descriptors bases on G_{norm}

5. **Homogeneity** :

$$\sum_{i=1}^{K}\sum_{j=1}^{K}\frac{p_{ij}}{1+|i-j|}$$

- Measures the spatial closeness of the distribution of elements in G to the diagonal
- Range of values: $[0,1]$
- Homogeneity $= 1 \equiv G$ is diagonal matrix

Texture: Statistical Approaches

Relative positions of pixels:

Descriptors bases on G_{norm}

❻ **Entropy** :

$$\sum_{i=1}^{K}\sum_{j=1}^{K} p_{ij}\log_2 p_{ij}$$

- Measures the randomness of elements in G
- Entropy $= 0$: all elements in G are zeros
- Entropy $= 1$: all elements in G are equal
- Max entropy $= 2 \times \log_2 K$

Texture: Statistical Approaches - Demonstration

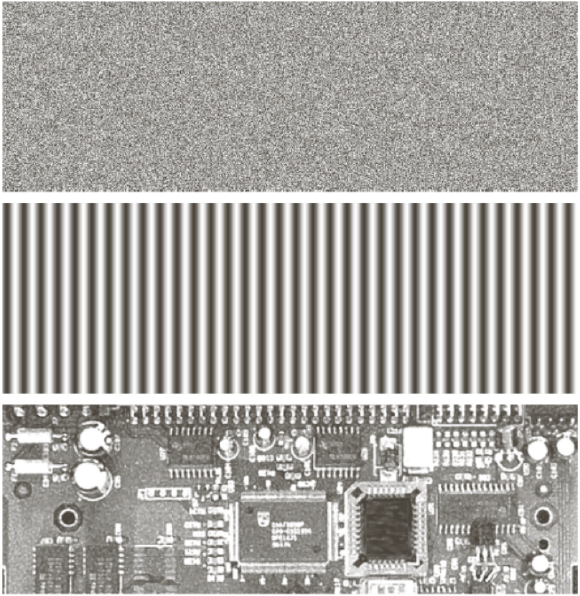
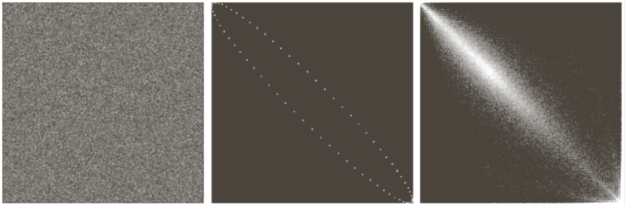


FIGURE 11.30 Images whose pixels have (a) random, (b) periodic, and (c) mixed texture patterns. Each image is of size 263 × 800 pixels.

Texture: Statistical Approaches - Demonstration



a b c

FIGURE 11.31 256 × 256 co-occurrence matrices, $\mathbf{G}_1$, $\mathbf{G}_2$, and $\mathbf{G}_3$, corresponding from left to right to the images in Fig. 11.30.

- Q: One pixel immediately to the right
- Note: these matrices are discriminative

Texture: Statistical Approaches - Demonstration

Normalized Co-occurrence Matrix	Descriptor					
	Max 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	570	0.01230	0.0824	6.43
$\mathbf{G}_3/n_3$	0.06860	0.8798	1356	0.00480	0.2048	13.58

- Measurements are discriminative

Texture: Statistical Approaches

Questions

How can you reduce the size of co-occurrence matrices?

Spectral Descriptors

Method

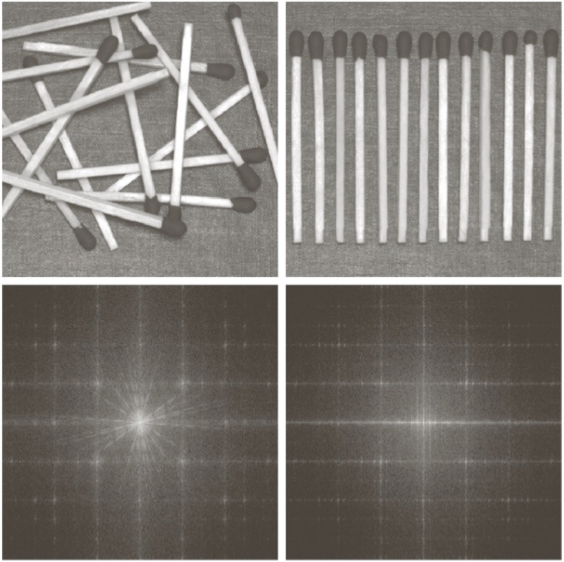
1. Compute FFT, called $F(u.v)$, for the input image $f(x, y)$ for sub-image containing the input region.
2. Convert $F(u.v)$ to polar system $T(r, \theta)$
3. Compute measurements $S(r)$ and $S(\theta)$

$$S(r) = \sum_{\theta=0}^{\pi} T(r, \theta)$$

$$S(\theta) = \sum_{r=1}^{R_0} T(r, \theta)$$

- R_0 : the radius of a circle centered at the origin.

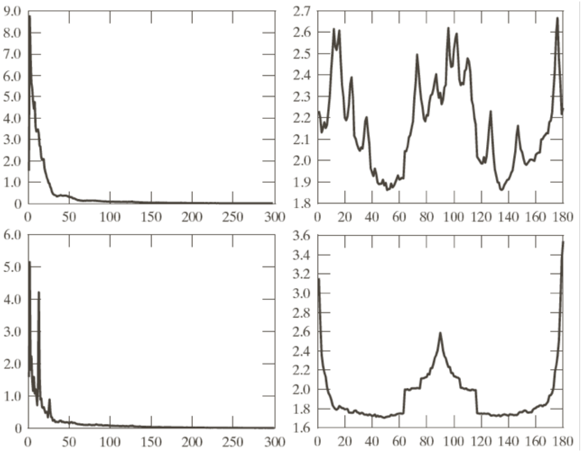
Spectral Descriptors - Demonstration



a b
c d

FIGURE 11.35 (a) and (b) Images of random and ordered objects. (c) and (d) Corresponding Fourier spectra. All images are of size 600 × 600 pixels.

Spectral Descriptors - Demonstration



a b
c d

FIGURE 11.36 Plots of (a) $S(r)$ and (b) $S(\theta)$ for Fig. 11.35(a). (c) and (d) are plots of $S(r)$ and $S(\theta)$ for Fig. 11.35(b). All vertical axes are $\times 10^5$.

Moments

Inputs

1. Image $f(x, y)$ of size $M \times N$

Moments of order (p + q)

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

- $p = 0, 1, 2, \ldots$
- $q = 0, 1, 2, \ldots$

Moments

Central Moments of order (p + q)

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

- $p = 0, 1, 2, ...$
- $q = 0, 1, 2, ...$

$$\bar{x} = \frac{m_{10}}{m_{00}}$$

$$\bar{y} = \frac{m_{01}}{m_{00}}$$

Moments

Set of 7 moments

- See textbook on pp.863
- These moments are invariant to translation, rotation, scaling and mirroring

Moments - Demonstration

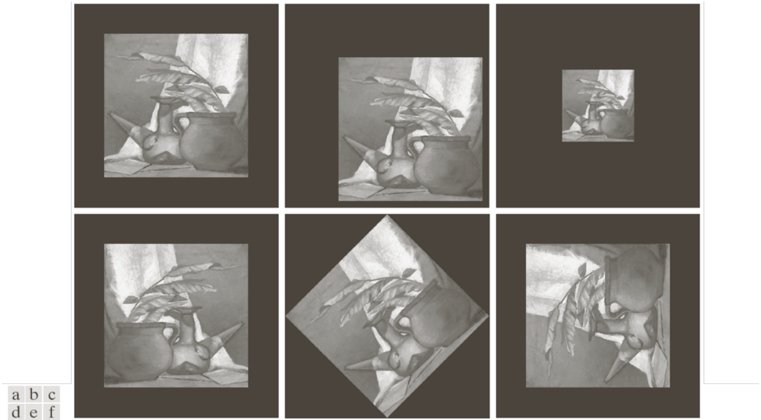


FIGURE 11.37 (a) Original image. (b)–(f) Images translated, scaled by one-half, mirrored, rotated by 45° and rotated by 90°, respectively.

Moments - Demonstration

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

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

- Note: The value of moments just **change slightly** with rotation, scalling, translation, and mirroring.

Principal Components

Inputs

1. A set X of K vectors, each has size of $n \times 1$

$$\mathbf{x}_k = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ x_n \end{bmatrix}$$

- $\mathbf{x}$: an observation (measurement) for n variables (features).
- The variables can be width, height, length, or any other extracted features.
- $\mathbf{x}$: can be thought as a point in n-dimension space.

Principal Components

Basic Ideas

- Components or variables in vectors of X are correlated or uncorrelated.
- Principal Component Analysis (PCA) will propose a **transforms**. When we apply this transforms for each vector in X, we obtain a new set of vectors that their variables (or components) are **uncorrelated**.

Principal Components

Steps

1. Compute the mean vector of X

$$\mathbf{m_x} = E\{\mathbf{x}\} \approx \frac{1}{K}\sum_{k=1}^{K}\mathbf{x}_k$$

Principal Components

Steps

2. Compute the variance-covariance matrix (called covariance matrix) of X

$$\mathbf{C_x} = E\{(\mathbf{x} - \mathbf{m_x})(\mathbf{x} - \mathbf{m_x})^T\}$$
$$\approx \frac{1}{K}\sum_{k=1}^{K} \mathbf{x}_k \mathbf{x}_k^T - \mathbf{m_x}\mathbf{m_x}^T$$

- $\mathbf{C_x}$: real and symmetric matrix

Principal Components

Meaning of the mean vector and the covariance matrix

- **Mean vector**:
  - A vector that contains the mean value for each component (variable)
  - The centroid of the shape, if vectors are points in space.
- **Covariance matrix**:
  - Elements on diagonal: The variance of components
  - Elements off-diagonal: The covariance between component x and component y. These covariances are either negative or positive.
    - Positive covariance: Large value on component x tends to occur with large value on component y. Small value on component x tends to occur with small value on component y
    - Negative covariance: Large value on component x tends to occur with small value on component y and vice versa.

Principal Components

Steps

3. Calculate eigenvectors and eigenvalues of $\mathbf{C_x}$

Let $\mathbf{e}_i$ and λ_i be the eigenvectors and corresponding eigenvalues of $\mathbf{C_x}$, for $i = 1, 2, .., n$

Principal Components

Steps

4. Form transforms matrix **A** as follows:
   - Matrix **A** has size of $n \times n$
   - Eigenvectors are put to rows of **A** sin such a way that their corresponding eigenvalues are in descending-order from the first row to the last row in matrix **A**

Transforms Matrix's Properties

- **A** is an orthogonal matrix, so
- $\mathbf{A} = \mathbf{A}^T$

Principal Components

A is used to transform input vectors in X to principal components

Steps

5. Perform the transformation

$$\mathbf{y} = \mathbf{A}(\mathbf{x} - \mathbf{m_x})$$

Principal Components

Properties of set of vector $\mathbf{y}$

1. The mean vector of $\mathbf{y}$ is vector $\mathbf{0}$

$$\begin{aligned}\mathbf{m_y} &= E\{\mathbf{y}\}\\ &= \mathbf{0}\end{aligned}$$

2. Variables of $\mathbf{y}$ are uncorrelated $\equiv$ Covariance matrix of vectors $\mathbf{y}$ has all elements zeros, accept ones on diagonal.

$$\begin{aligned}\mathbf{C_y} &= \mathbf{A C_x A}^T\\ &= \begin{bmatrix} \lambda_1 & 0 & 0 & .. & 0 \\ 0 & \lambda_2 & 0 & .. & 0 \\ 0 & 0 & \lambda_3 & .. & 0 \\ \\ 0 & .. & & 0 & \lambda_n \end{bmatrix}\end{aligned}$$

Principal Components

Reconstruction x from y - perfect reconstruction

$$\mathbf{x} = \mathbf{A}^T\mathbf{y} + \mathbf{m_x}$$

Principal Components

- $\mathbf{A}_k$: created from $\mathbf{A}$ by keeping the first k rows
  - $\Rightarrow$ $\mathbf{A}_k$ has size: $k \times n$
  - $\Rightarrow$ $\mathbf{A}_k^T$ has size: $n \times k$

Reconstruction x from y - Approximation

1. Keep k important components of $\mathbf{y}$:

$$\mathbf{y} = \mathbf{A}_k(\mathbf{x} - \mathbf{m_x})$$

2. Reconstruct $\mathbf{x}$ from $\mathbf{y}$:

$$\hat{\mathbf{x}} = \mathbf{A}_k^T\mathbf{y} + \mathbf{m_x}$$

- $\Rightarrow$ $\mathbf{y}$ has size: $k \times 1$
- $\Rightarrow$ $\hat{\mathbf{x}}$ has size: $n \times 1$
- $\hat{\mathbf{x}}$ is an approximation of $\mathbf{x}$ using the first k important components.

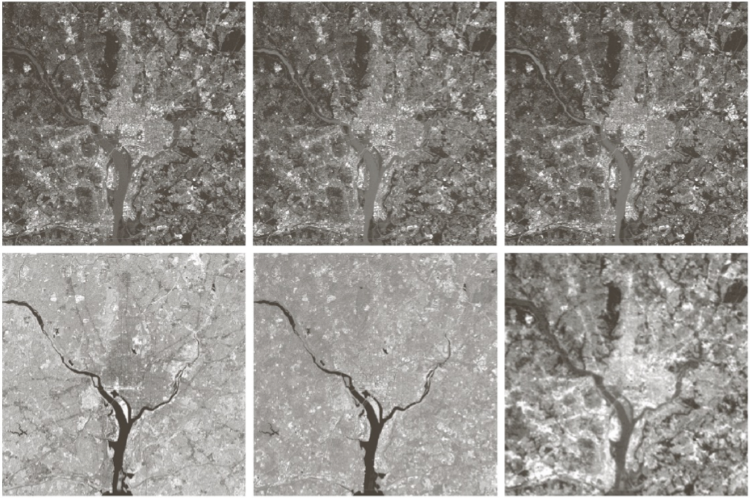
Principal Components: Applications

Dimension reduction

$$\mathbf{y} = \mathbf{A}_k(\mathbf{x} - \mathbf{m}_\mathbf{x})$$

1. The first k important component can be used as descriptor, so
2. Number of dimensions are reduced from n down to k
   - Size of input $\mathbf{x}$: n
   - Size of output $\mathbf{y}$: k

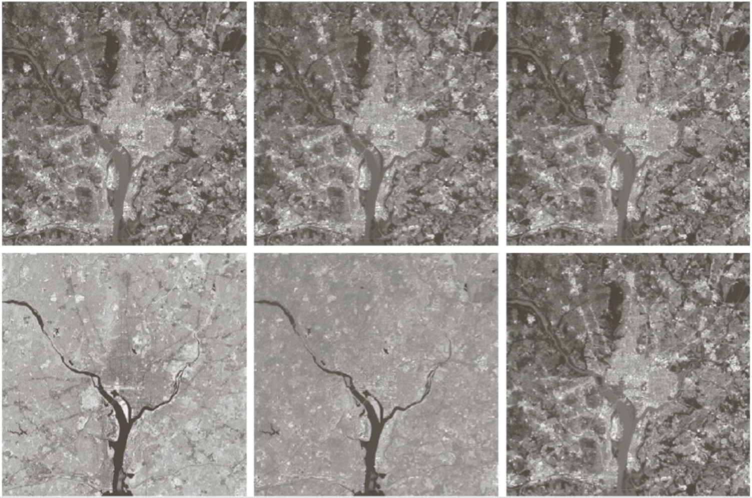
Principal Components: Applications



a b c
d e f

FIGURE 11.38 Multispectral images in the (a) visible blue, (b) visible green, (c) visible red, (d) near infrared, (e) middle infrared, and (f) thermal infrared bands. (Images courtesy of NASA.)

Principal Components: Applications



a b c
d e f

FIGURE 11.41 Multispectral images reconstructed using only the two principal component images corresponding to the two principal component images with the largest eigenvalues (variance). Compare these images with the originals in Fig. 11.38.

Principal Components: Applications

$$\mathbf{y} = \mathbf{A}(\mathbf{x} - \mathbf{m}_{\mathbf{x}})$$

Properties of y

Transformed features **y** has the following advantages compared to **x**:

1. Invariant to translation and rotation
   - **y** has been shifted to the centroid by $\mathbf{m}_{\mathbf{x}}$
   - **y** has been aligned with principal directions (eigenvectors)
2. Invariant to scaling can be achieved by dividing **y** for eigenvalues.

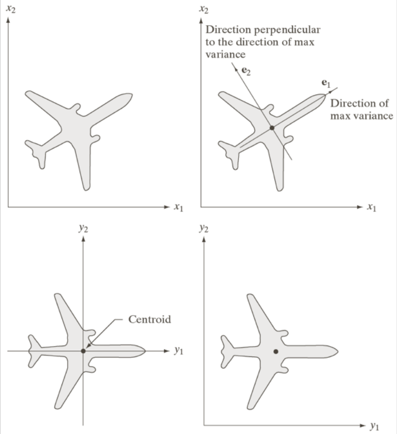
Principal Components: Applications

$$\mathbf{m}_x = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$$\mathbf{C}_x = \begin{bmatrix} 3.333 & 2.00 \\ 2.00 & 3.333 \end{bmatrix}$$

$$\mathbf{e}_1 = \begin{bmatrix} 0.707 \\ 0.707 \end{bmatrix}$$

$$\mathbf{e}_2 = \begin{bmatrix} -0.707 \\ 0.707 \end{bmatrix}$$



a b
c d

FIGURE 11.43 (a) An object. (b) Object showing eigenvectors of its covariance matrix. (c) Transformed object, obtained using Eq. (11.4-6). (d) Object translated so that all its coordinate values are greater than 0.