

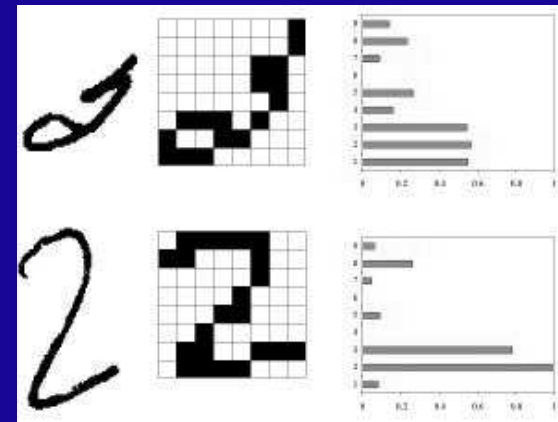
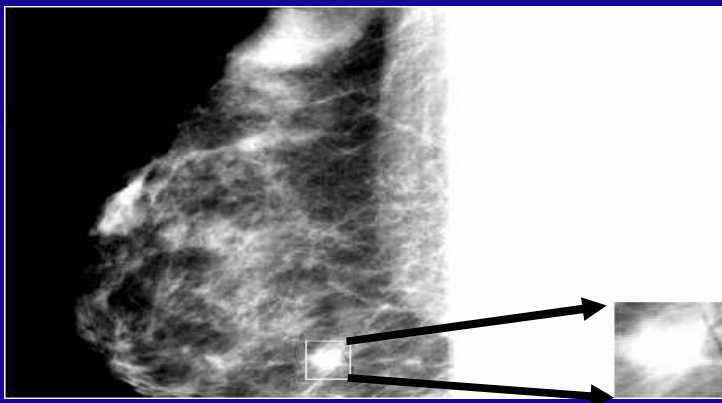
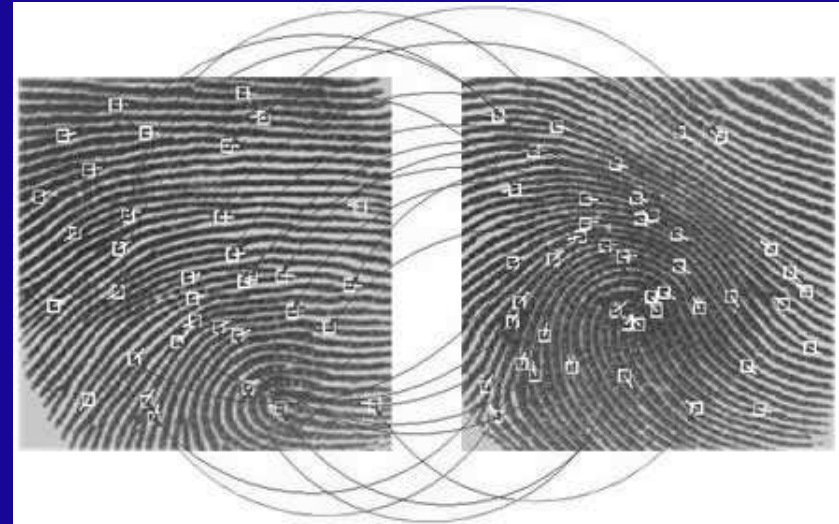
Multimedia Information Systems

Samson Cheung

EE 639, Fall 2004

Lecture 9: Content-based Image Retrieval
Shape

Why Shape?



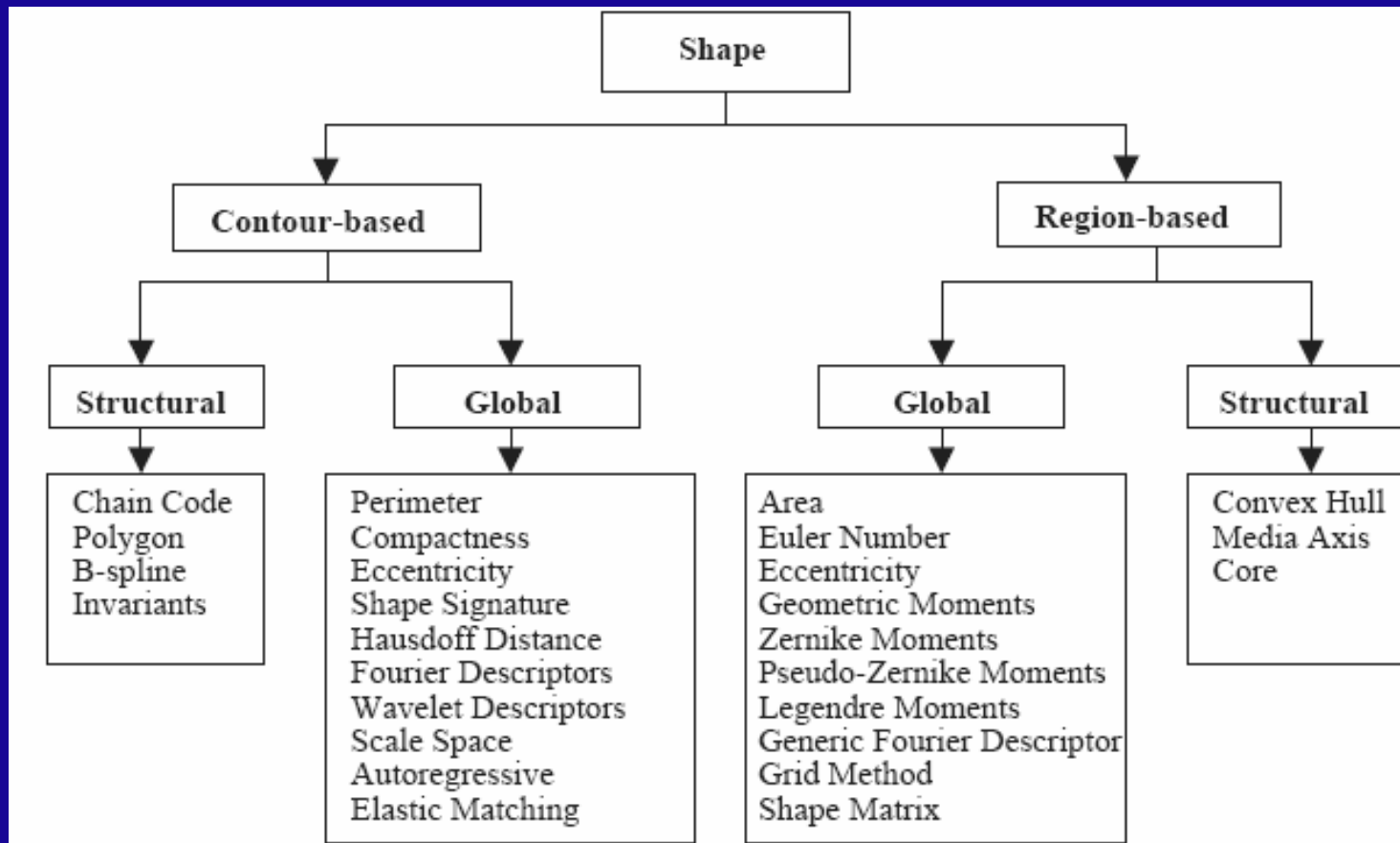
What is shape?

- **A numerical description of the spatial configurations in the image**
- **Assumptions:**
 - **2-D**
 - **Segmented image where each object has been labelled**
 - **Represented as**
 - Binary image
 - Contour

Considerations about shape descriptors

- **Is the descriptor complete?**
 - **Can the object be reconstructed from it?**
- **How are occlusions handled?**
 - **Can the descriptor be generated from partial objects?**
- **Invariant?**
 - **What transforms are allowed without significantly changing the descriptor?**
- **Robust against noise and digitization?**

Descriptor Taxonomy [Zhang,Lu04]



Types of shape descriptors

- **Contour based descriptors**
 - Correspondence approach
 - Curvature Scale Space
 - Chain Code
 - Fourier descriptor
- **Region based descriptors**
 - Simple characteristics
 - Topological descriptors
 - Convex hulls
 - Skeletons, graphs
 - Moments

Correspondence-based shape matching

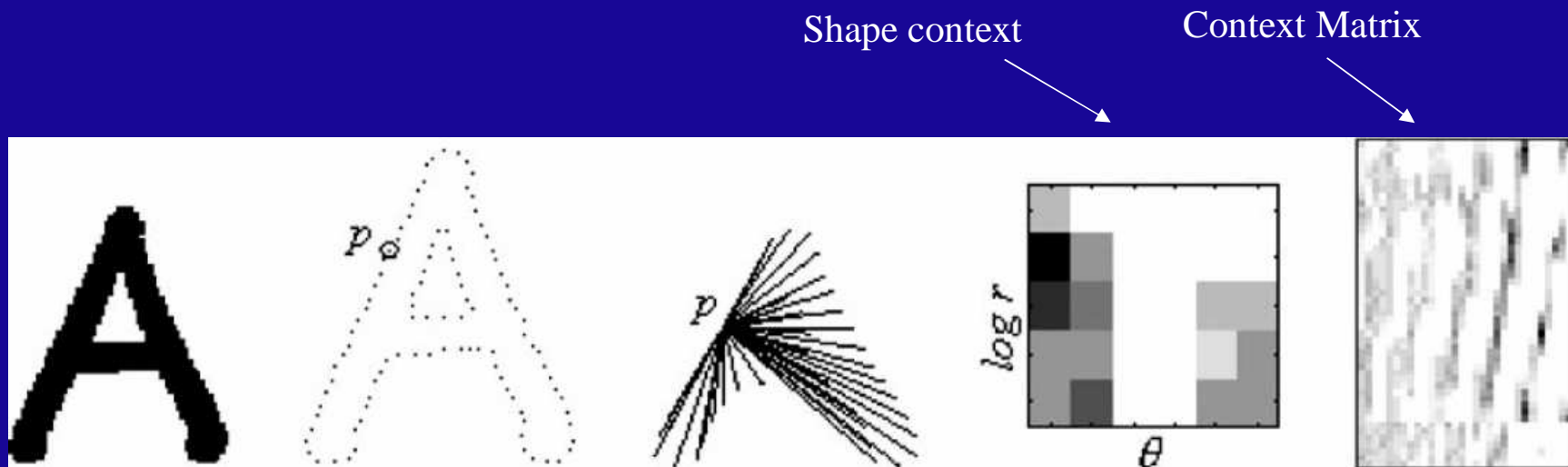
- **Hausdorff distance**

Two shapes : $A = \{a_1, \dots, a_p\}, B = \{b_1, \dots, b_q\}$

$$h(A, B) = \max_{a \in A} \min_{b \in B} \|a - b\|$$

- **Support partial matching**
- **Could be made rotational and scale invariant by preprocessing**
- **Sensitive to noise and sampling**
- **$O(N^2)$ complexity**

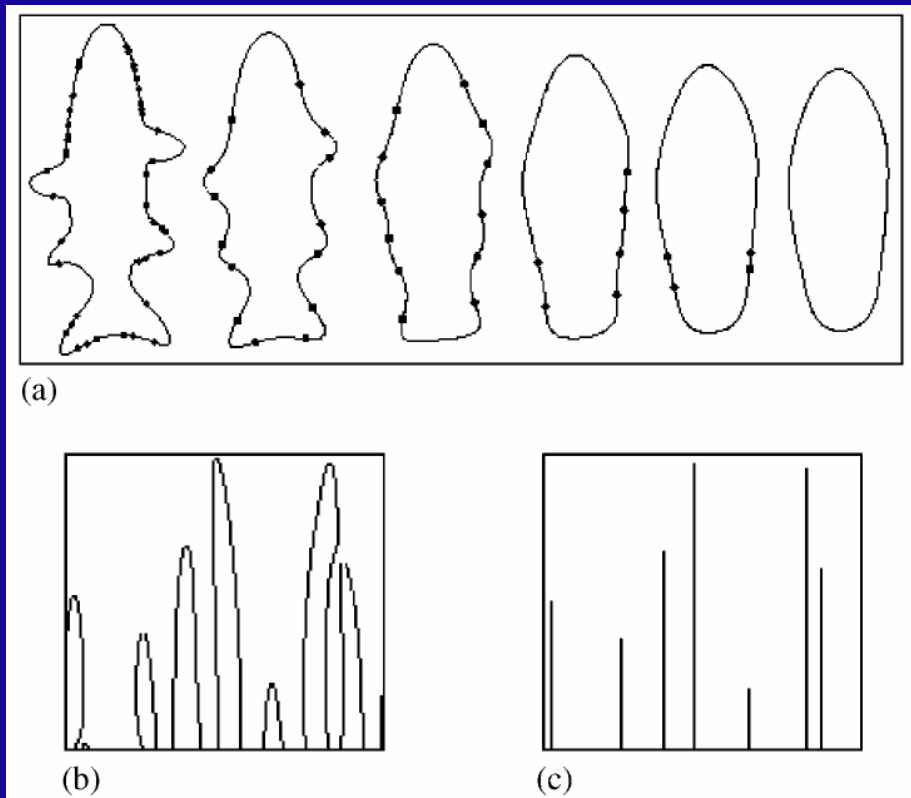
Shape context [Belongie et al01]



- Use a “context” rather than a single point
- Two context matrices are compared using Hausdorff distance

Curvature Scale Space

[Mokhtarian & Mackworth86]



For each scale

- Compute curvature

$$k_i = (\dot{x}_i \ddot{y}_i - \ddot{x}_i \dot{y}_i) / (\dot{x}_i^2 + \dot{y}_i^2)^{3/2}$$

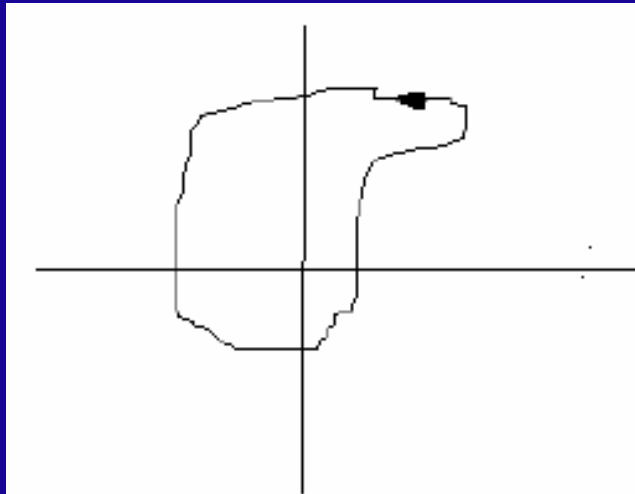
- Identify zero-crossings (inflection points)

CSS Image (b) : scale
versus zero-crossing
location

Chain code based shape descriptors

- **Derivative chain code [Bribiesca & Guzman 78]**
 - 1 = convex corner
 - 2 = Two successive links in the same direction
 - 3 = concave corner
 - Rotation and scale invariant
- **Unique representation**
 - All cyclic rotations and pick the one that represents the smallest ternary number
- **Similarity between two chain codes are measured by edit distances (substitution, deletion, and insertion)**

Fourier shape descriptor



$$z(t) = \sum_n T_n e^{int}$$

- Consider shape as a curve in the complex plane
- A moving point generates a periodic complex valued function
- Can be expressed as a Fourier series

Fourier Descriptor [Zahn72]

- Pick N points (x_i, y_i) around the boundary, compute:

$$r_i = \left[(x_i - x_c)^2 + (y_i - y_c)^2 \right]^{1/2}$$

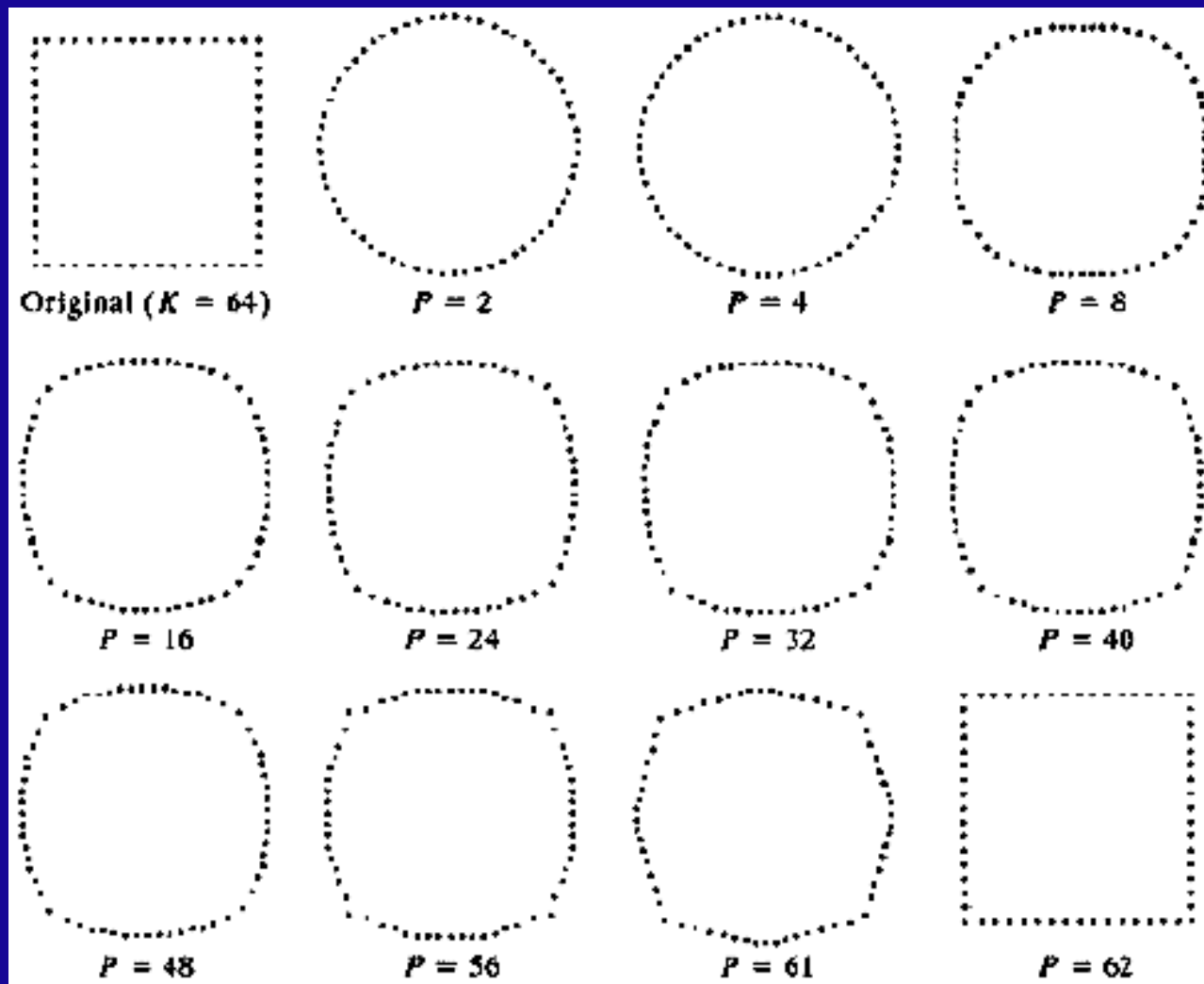
- Compute Fourier transform of r_i

$$u_n = \frac{1}{N} \sum_{i=0}^{N-1} r_i \exp\left(\frac{-j2\pi ni}{N}\right)$$

- Derive scale, translation, and rotational invariant descriptor:

$$FD = \left(\frac{|u_1|}{|u_0|}, \frac{|u_2|}{|u_0|}, \dots, \frac{|u_{N/2}|}{|u_0|} \right)$$

Fourier descriptor

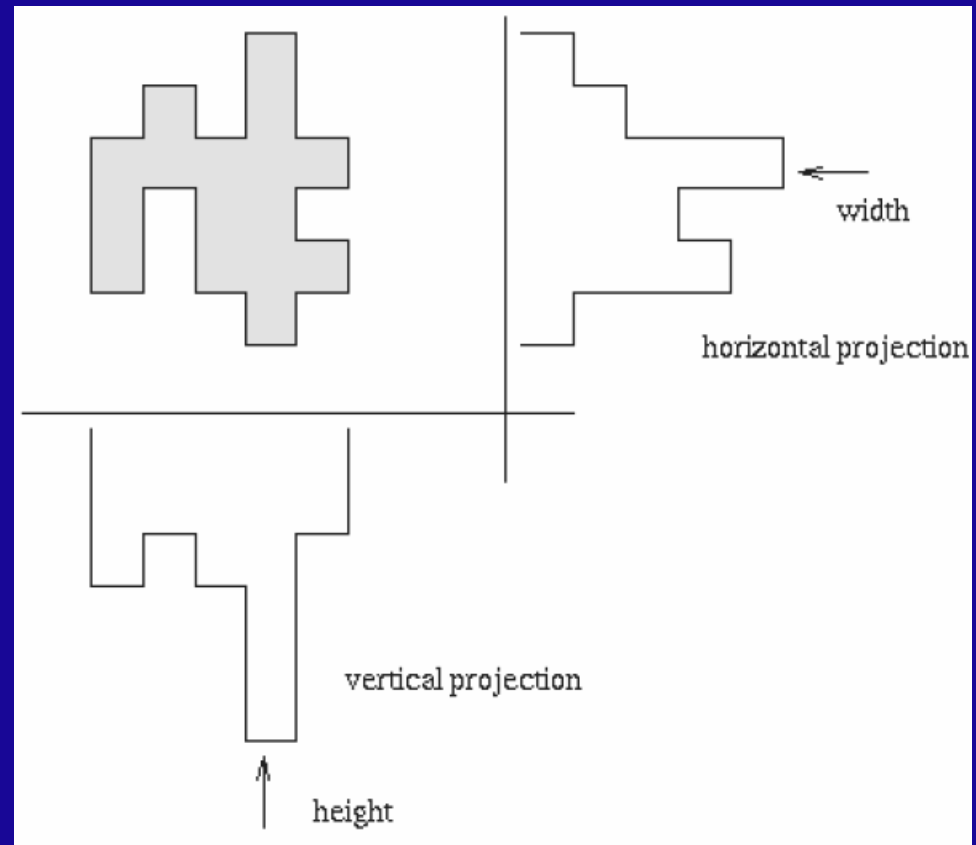


Types of shape descriptors

- **Contour based descriptors**
 - Correspondence approach
 - Curvature Scale Space
 - Chain Code
 - Fourier descriptor
- **Region based descriptors**
 - Simple characteristics
 - Topological descriptors
 - Convex hulls
 - Skeletons, graphs
 - Moments

Region based shape descriptors

- Area = no. of pixels
- Height, width and projections in other directions



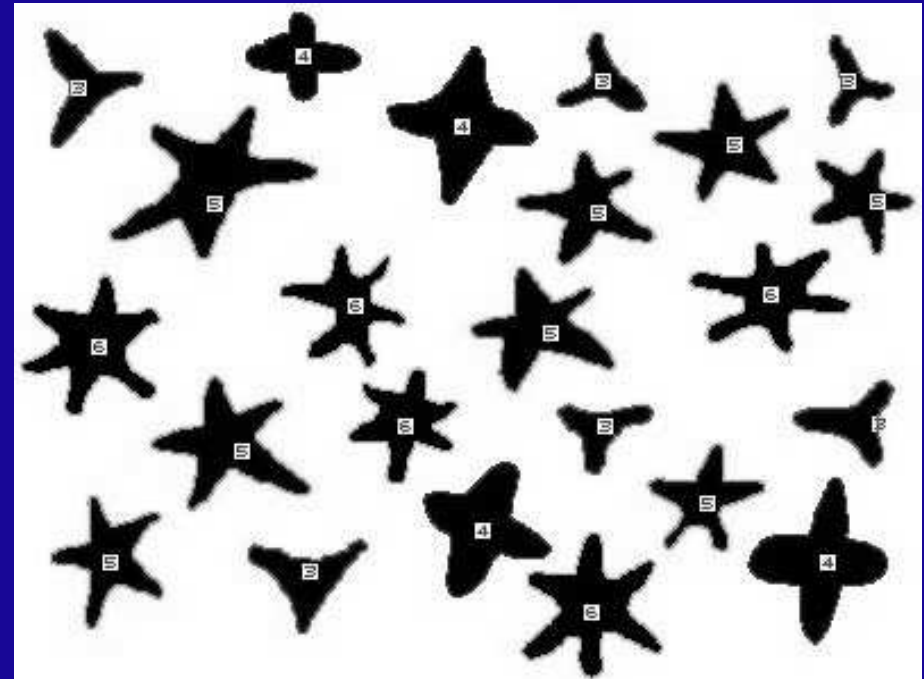
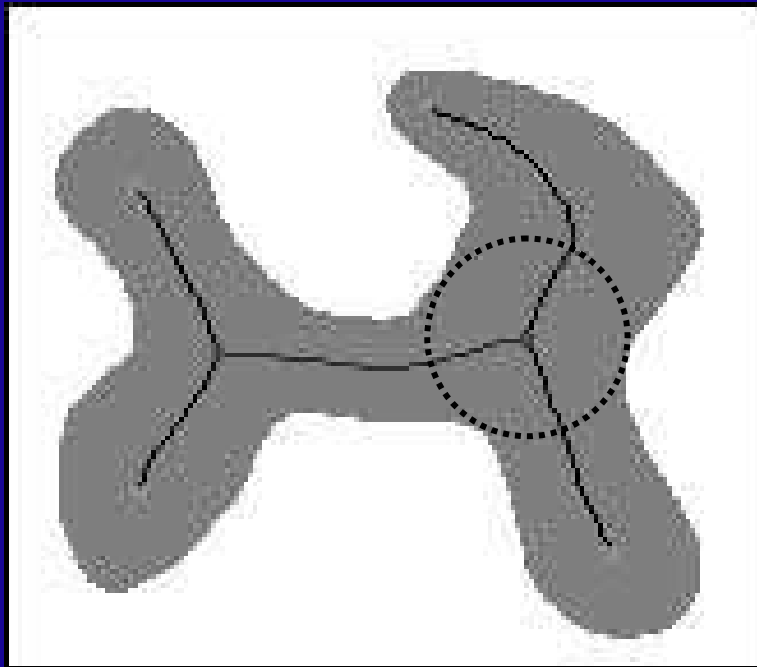
More region-based shape descriptors

- **Eccentricity = longest chord/max perpendicular width**
- **Elongatedness = $\text{area}/(2d)^2$, $d=\text{max radius}$**
- **Rectangularity = $\text{max}(\text{region/bounding rectangle})$ in difference rotations**
- **Compactness = $\text{perimeter}^2/\text{area}$**

Topological shape descriptors

- **Unchanged under rubber-sheet transformation**
- **Number of connected components**
- **Number of holes**
- **Euler-Poincare characteristics:**
 - **Euler number**
= #Faces - #Edges + #Vertices (any triangular meshes)

Skeleton



Convex hull

- **Minimal enclosing convex region**
- **Convexity:**
 - **Every line inside an object joining any two points on the boundary is within the object**
- **Convex deficiency is a useful shape descriptor**

Concavity tree

- Generate convex hulls and deficiencies recursively to create concavity tree
- Represented by area, bridge length, max curvature

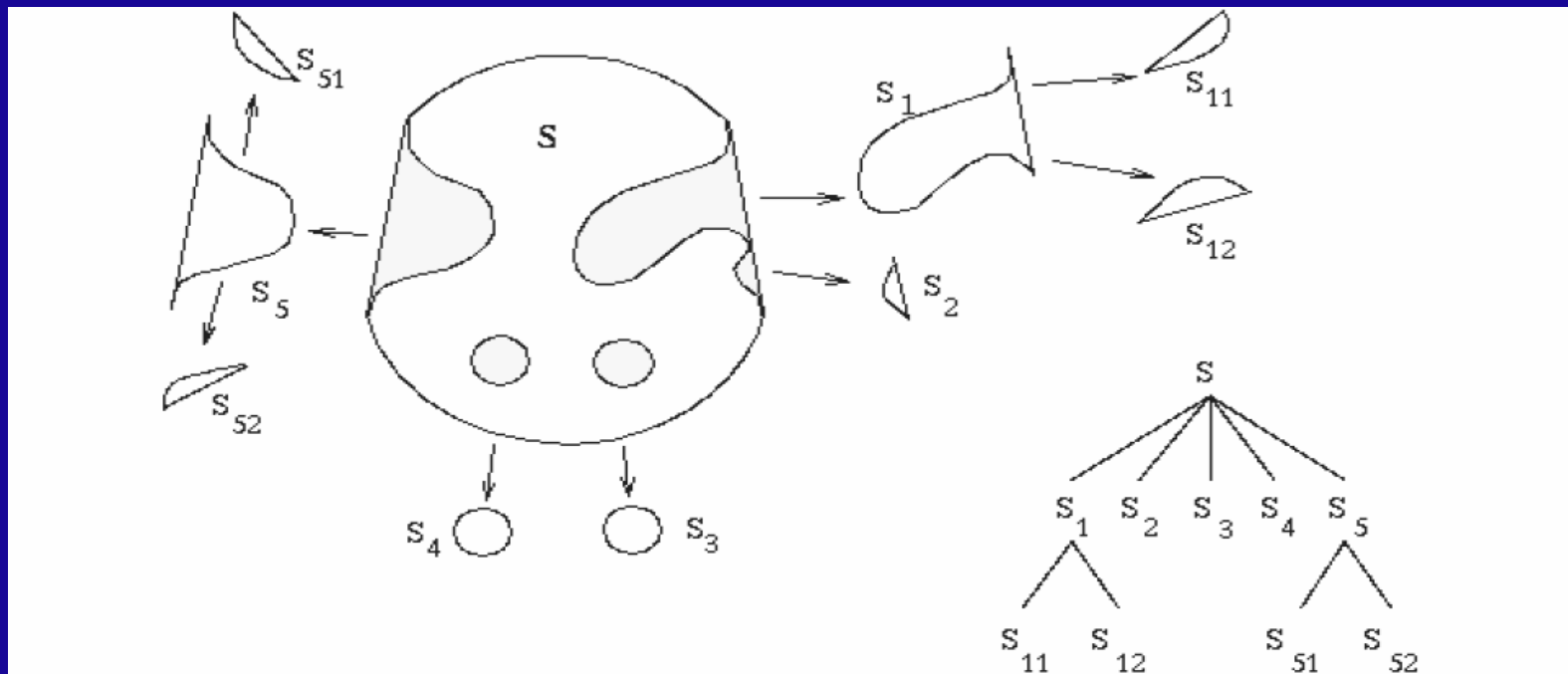
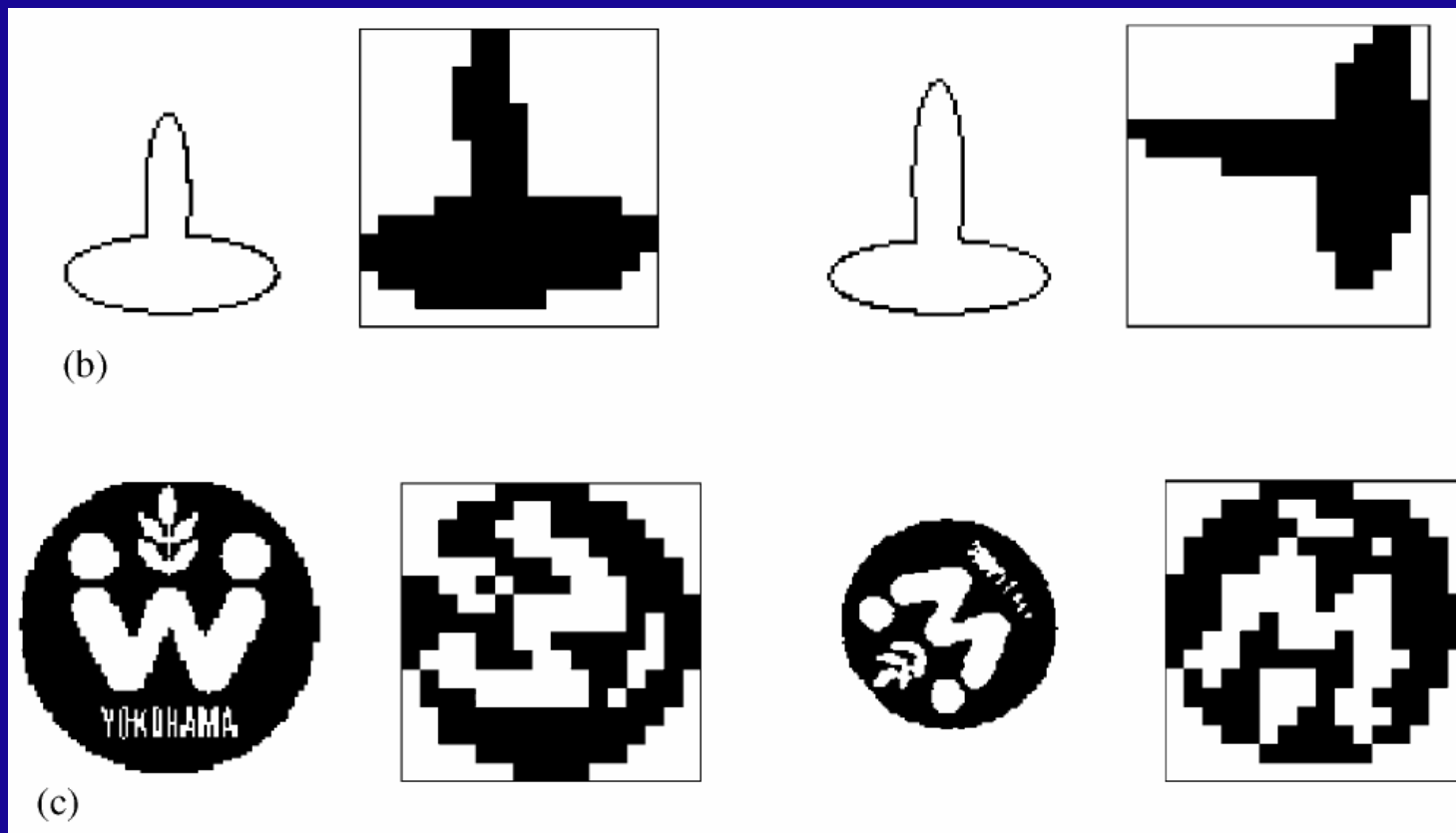


Figure 6.30 Concavity tree construction: (a) Convex hull and concave residues, (b) concavity tree.

Grid Based methods

- Orientation determined by principal axis
- Bit-patterns compared using hamming distance



Moments

- Treating the image as a probability mass function

$$m_{pq} = \frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M i^p j^q f(i, j)$$

- Central moments give translation invariance

$$\mu_{pq} = \frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M (i - x_c)^p (j - y_c)^q f(i, j)$$

where $x_c = m_{10} / m_{00}$, $y_c = m_{01} / m_{00}$

- Normalized moments give scale invariance

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_0^\gamma}$$

where $\gamma = (p + q)/2 + 1$ for $(p + q) \geq 2$

Hu's Invariant Moments

- They are rotational and reflection invariant as well.
- But lots of redundant information ...

$$I_1 = \eta_{20} + \eta_{02}$$

$$I_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$

$$I_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$

$$I_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$

$$I_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

$$I_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2 + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})]$$

$$I_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ + (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

Moment features

- View $i^p j^q$ as basis functions
- Use orthogonal basis functions instead
 - Legendre polynomials (not rotation invariant)
 - Zernike polynomials (very good!!)
 - MPEG-7 Angular Radial Transform (ART)

