Shape Description

Prof. Ioannis Pitas
Aristotle University of Thessaloniki
pitas@csd.auth.gr
www.aiia.csd.auth.gr
Version 2.7.1

Artificial Intelligence &
Information Analysis Lab

Shape Description

- **Introduction**
- Chain Codes
- Polygonal Approximations
- Fourier Descriptors
- Quadtrees
- Pyramids
- Shape Features
- Moment Descriptors
- Thinning Algorithms

Introduction

2D shapes can be described in two different ways:

a) *External representation*: Description using the object boundary and its features.

- Linked to edge detection, contour following.

b) *Internal representation*: Description by the object region (set of pixels on the image plane).

- Linked to image region segmentation.

Introduction

Desirable shape representation properties:

- ***Uniqueness***:
  - It is of crucial importance in object recognition.
- ***Invariance*** under geometrical transformations:
  - translation, rotation, scaling and reflection.
  - Very important for object recognition applications.
- ***Completeness***:
  - This refers to its ability to represent any shape.

Introduction

- ***Sensitivity***:
  - Ability of a representation scheme to reflect easily the differences between similar objects.
- ***Abstraction from detail***:
  - Ability of the representation to represent only the basic shape features.
  - Directly related to the noise robustness of the representation.

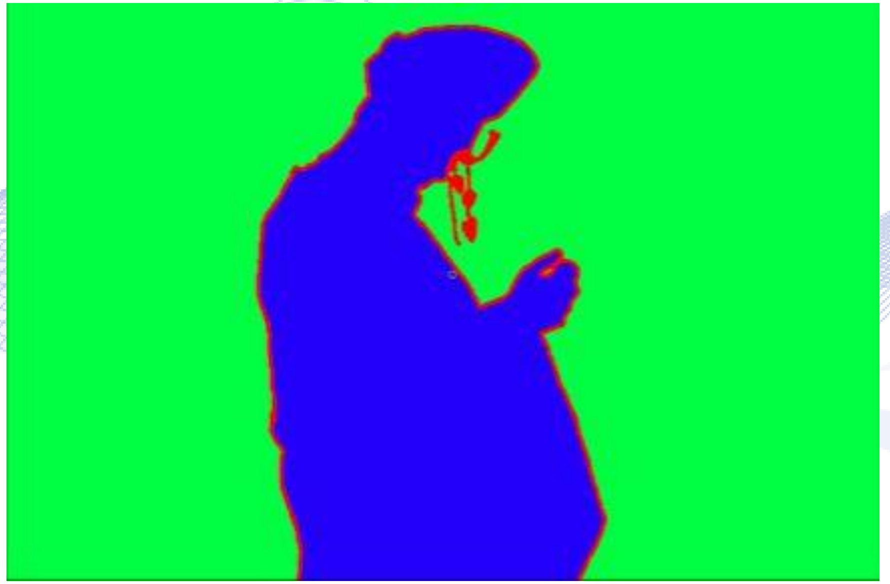
- Sensitivity and abstraction from detail may contradict each other.

Shape Description

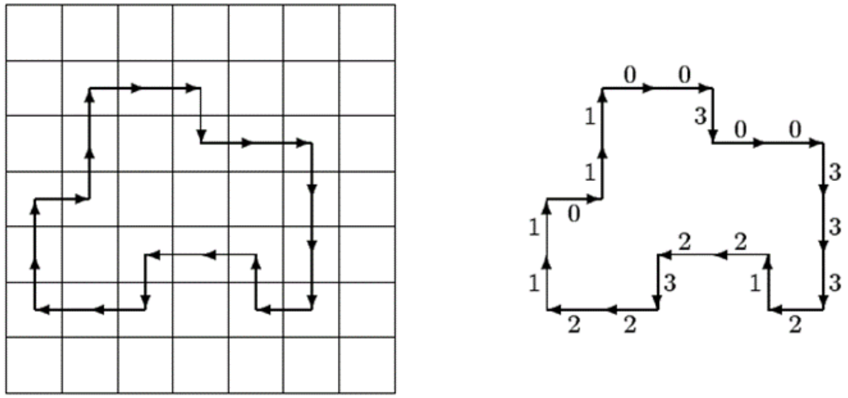
- Introduction
- **Chain Codes**
- Polygonal Approximations
- Fourier Descriptors
- Quadtrees
- Pyramids
- Shape Features
- Moment Descriptors
- Thinning Algorithms

Chain codes

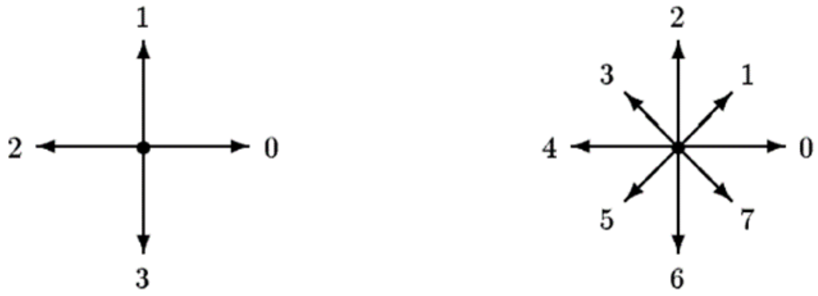
- Simplest object contour description: ordered list of contour pixels $[x_i, y_i]^T, i = 1, \ldots, N$.
- It is a verbose description that can be greatly compressed.
- This can be done, e.g., by ***chain codes***.



Chain codes



Chain code of a digital image boundary.



(a) (b)

Chain codes for: a) a 4-connected chain; b) an 8-connected chain.

Chain codes

- The chain code depends on the start point of boundary following.
- Chain codes provide a good compression of boundary description.
- Chain codes can also be used to calculate certain boundary features.
- It is translation invariant.
- Scale invariance *may* be obtained by changing the sampling grid.

Chain codes

- Rotation invariance is obtained by using the ***difference chain code***:

$$d_i = \begin{cases} \text{diff}(x_i, x_{i-1}) = |x_i - x_{i-1}|, & \text{if } \ i \neq 1, \\ \text{diff}(x_i, x_N) = |x_i - x_{i-1}|, & \text{if } \ i = 1. \end{cases}$$

- Differences are calculated mod 2 for 4-connected chain, or mod 2 for 8-connected chains.

- Invariance is for multiples of 90 or 45 degrees, for 4-connected or 8-connected chains.

Chain codes

- Object boundary ***perimeter*** T is given by:

$$T = \sum_{i=1}^{N} n_i \,,$$

- In case of an 8-connected chain code:

$$n_i = \begin{cases} 1, & \text{if } x_i \bmod 2 = 0, \\ \sqrt{2}\,, & \text{if } x_i \bmod 2 = 1. \end{cases}$$

Chain codes

- ***Object width*** w and ***height*** h: are given by:

$$w = \sum_{i=1}^{N} w_i\,, \qquad h = \sum_{i=1}^{N} h_i\,,$$

where:

$$w_i = \begin{cases} 0, & \text{if } x_i = 1,2,3, \\ 1, & \text{if } x_i = 0, \end{cases} \qquad h_i = \begin{cases} 0, & \text{if } x_i = 0,2,3, \\ 1 & \text{if } x_i = 1, \end{cases}$$

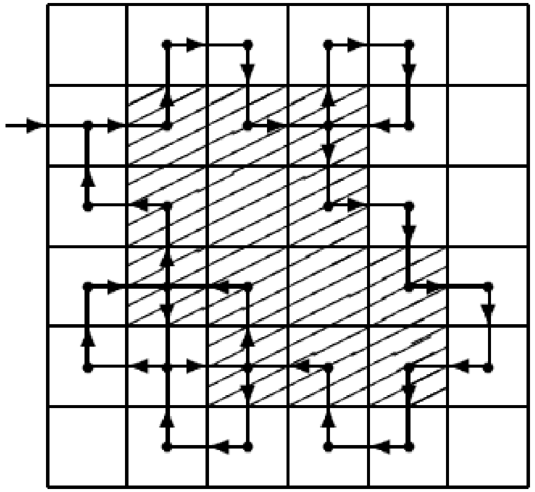
in case of an 8-connected chain code.

- Chain codes can also be used in the calculation of object area.

Chain codes

Papert's turtle follows binary object boundary:

- For pixel value $\begin{Bmatrix} 0 \\ 1 \end{Bmatrix}$ turn $\begin{Bmatrix} right \\ left \end{Bmatrix}$ and advance one pixel:



Papert's turtle in binary object boundary following.

Shape Description

- Introduction
- Chain Codes
- **Polygonal Approximations**
- Fourier Descriptors
- Quadtrees
- Pyramids
- Shape Features
- Moment Descriptors
- Thinning Algorithms

Polygonal approximations

Polygonal contour approximation:

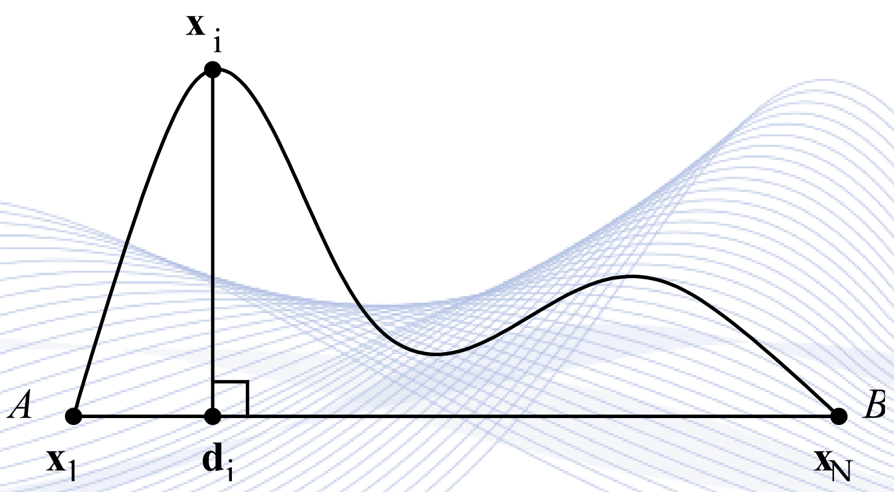
- Optimal linear piecewise contour approximation:
  - Choice of polygon vertices, so that the overall contour approximation error is minimized.
- Error measures:
  - ***Mean Square Error***:

$$E_2 = \sum_{i=2}^{N-1} |\mathbf{x}_i - \mathbf{d}_i|^2 .$$

Polygonal approximations

- Maximal approximation error:

$$E_{max} = \max_{2 \le i \le N-1} |\mathbf{x}_i - \mathbf{d}_i|.$$



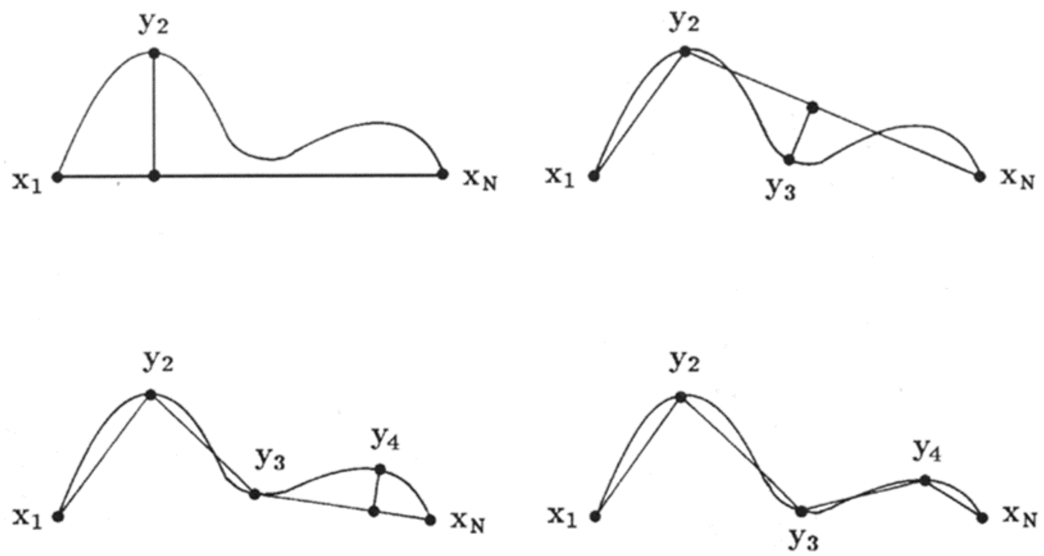
Curve approximation error.

Polygonal approximations

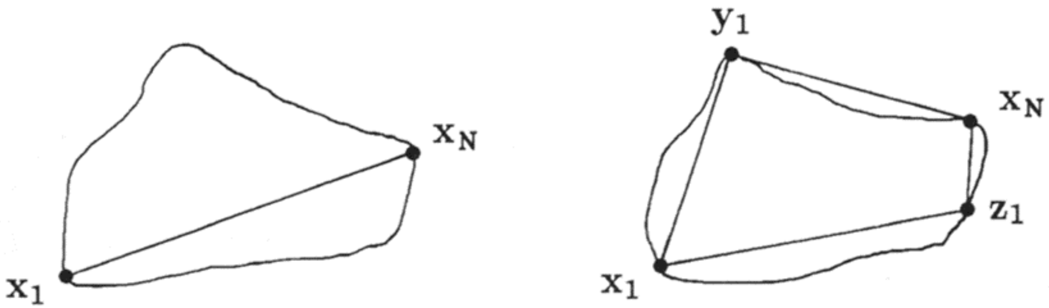
Polygonal splitting techniques:

- Divide a curve segment recursively into smaller segments, until each curve segment can be approximated by a linear segment within an acceptable error range.
- Curve inflection points can be easily detected and used in curve representation.

Polygonal approximations



Splitting method for polygonal approximations of a curve segment.



Splitting method for the linear piecewise approximation of a closed curve.

Polygonal approximations

Polygonal merging techniques:

- Polygon vertices do not coincide with curve inflection points.

- Combination of split and merge techniques.

Shape Description

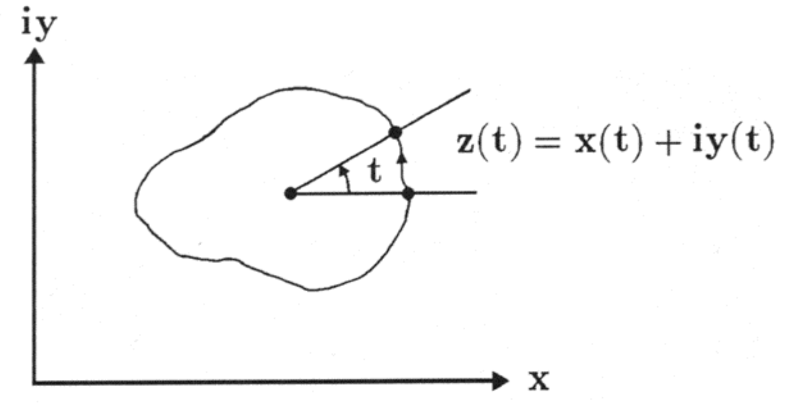
- Introduction
- Chain Codes
- Polygonal Approximations
- **Fourier Descriptors**
- Quadtrees
- Pyramids
- Shape Features
- Moment Descriptors
- Thinning Algorithms

Fourier descriptors

Fourier descriptors for contour representation:

$$Z(k)=\sum_{n=0}^{N-1} z(n)\exp\left(-i\frac{2\pi nk}{N}\right),$$

$$z(n)=\frac{1}{N}\sum_{k=0}^{N-1} Z(k)\exp\left(i\frac{2\pi nk}{N}\right).$$



Parametric curve representation.

Fourier descriptors

Fourier descriptor properties can be used in object recognition applications:

- Fourier coefficient $Z(0)$ represents the curve ***center of gravity***.
- Fourier coefficients $Z(k)$ represent slowly and rapidly varying shape trends for small and large indices k, respectively.

- ***Curve translation*** by $z_0 = x_0 + iy_0$:

$$z_t(n) = z(n) + z_0,$$

affects only the Fourier DC term $Z(0)$:

$$Z_t(0) = Z(0) + z_0.$$

Fourier descriptors

- ***Curve rotation*** by angle θ:

$$z_r(n) = z(n)e^{i\theta},$$

results in a phase shift of the Fourier coefficients:

$$Z_r(k) = Z(k)e^{i\theta}.$$

- ***Curve coordinate scaling*** by a factor a, results in Fourier coefficients scaling:

$$z_s(n) = \alpha z(n),$$
$$Z_s(k) = \alpha Z(k).$$

Fourier descriptors

Fourier descriptor properties:

- A ***change in the starting point*** of curve traversal:

$$z_t(n) = z(n - n_0),$$

produces modulation of the Fourier descriptors:

$$Z_t(k) = Z(k)e^{-i2\pi n_0 k/N}.$$

- Error measure for matching two curves $Z_1(n)$, $Z_2(n)$:

$$E = \sum_{k=0}^{N-1} (|Z_1(k) - Z_2(k)|)^2.$$

()

Shape Description

- Introduction
- Chain Codes
- Polygonal Approximations
- Fourier Descriptors
- Quadtrees
- Pyramids
- Shape Features
- Moment Descriptors
- Thinning Algorithms

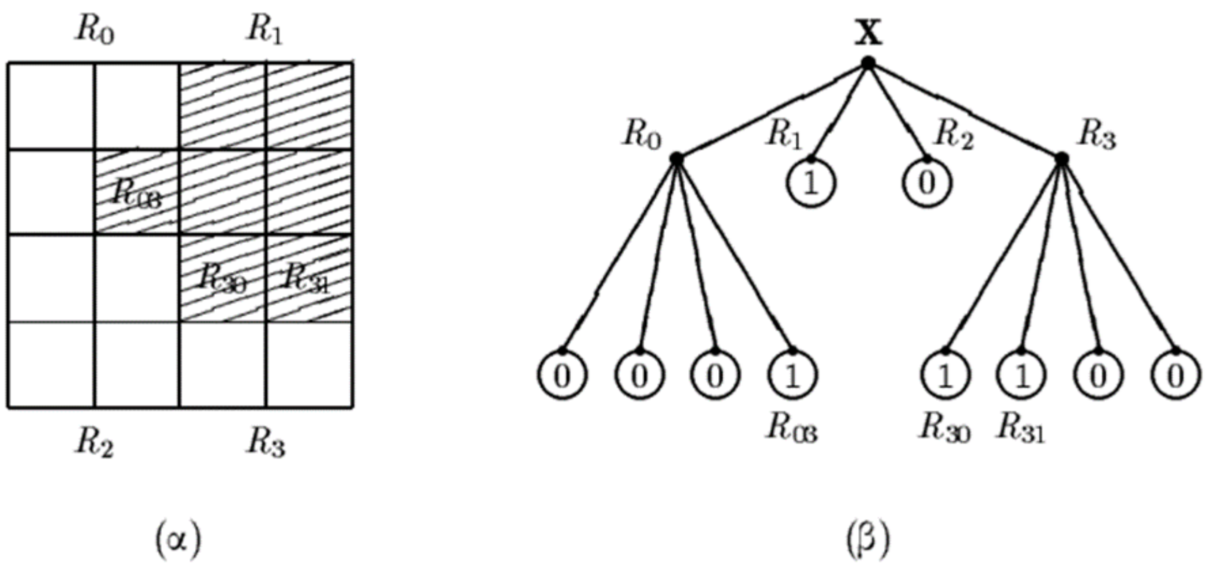
Quadtrees

Quadtree recursive computation:

- if a binary image region of size $2^n \times 2^n$ is inhomogeneous, it is split into four square subregions $\mathcal{R}_0, \mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3$ having size $2^{n-1} \times 2^{n-1}$ each.
- This continues until all subregions are homogeneous.
- The resulting shape representation is a quadtree.
- Maximal number of quadtree nodes:

$$N = \sum_{k=0}^{n} 4^k \approx \frac{4}{3} 4^n.$$

Quadtrees



a) Binary image; b) Quadtree representation.

Shape Description

- Introduction
- Chain Codes
- Polygonal Approximations
- Fourier Descriptors
- Quadtrees
- **Pyramids**
- Shape Features
- Moment Descriptors
- Thinning Algorithms

Pyramids

Image pyramids:

- Multiresolution image representations
- They employ several image copies at different resolutions.
- Both greyscale and binary images representations.
- An image pyramid is an image array series $f_k(i,j), k = 0, \dots, n$, each having size $2^k \times 2^k$.

Pyramids

- Mapping function from one pyramid level to the one above:

$$f_k(i,j) = g(f_{k+1}(2i,2j), f_{k+1}(2i,2j+1), f_{k+1}(2i+1,2j), f_{k+1}(2i+1,2j+1)).$$

- Linear mapping by local averaging:

$$f_k(i,j) = \frac{1}{4}\sum_{l=0}^{1}\sum_{m=0}^{1} f_{k+1}(2i+l, 2j+m).$$

Pyramids

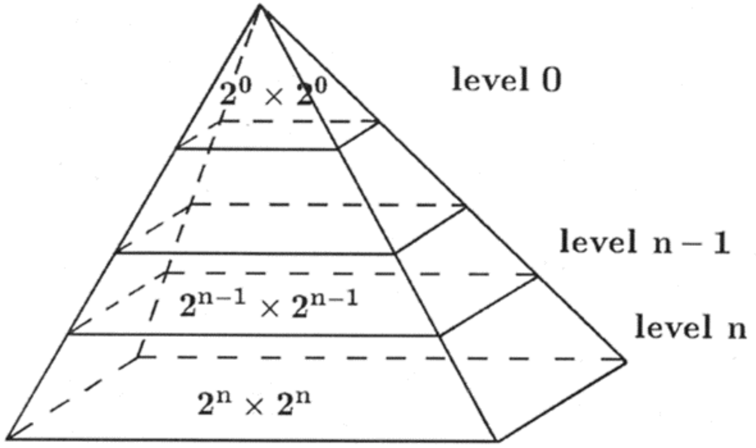
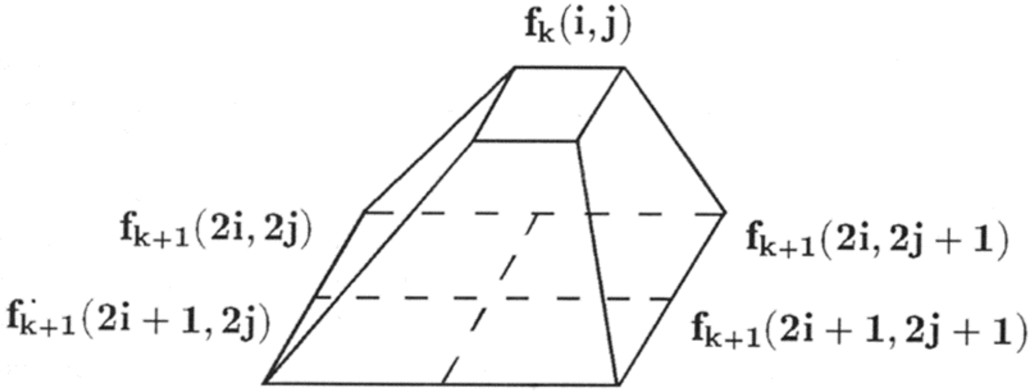


Image pyramid.



Mapping from one pyramid level to the next one.

Pyramids

- Pyramids offer abstraction from image details.
- They can be used in:
  - Image compression;
  - Multiresolution scaling-invariant image analysis.
- Pyramid storage on $n+1$ arrays of size $2^k \times 2^k, k = 0, \ldots, n$.
- Total space for pyramid storage for a $2^n \times 2^n$ image:

$$M = {}^4/_3 \times (2^n \times 2^n).$$

Pyramids

a) Original binary image.

b) Binary image Pyramid.

c) Output of the pyramid edge detector.

d) Edge Pyramid.

Shape Description

- Introduction
- Chain Codes
- Polygonal Approximations
- Fourier Descriptors
- Quadtrees
- Pyramids
- **Shape Features**
- Moment Descriptors
- Thinning Algorithms

Shape features

Geometrical shapes possess certain features (e.g., perimeter) that carry sufficient information for some object recognition applications. Such features can be used as object descriptors resulting in a significant data compression, because they can represent the geometrical shape by a relatively small feature vector.

Shape features can be grouped in two large classes:

- ***Boundary features***.
- ***Region features***.

Shape features

- ***Object perimeter***:

$$T = \int \sqrt{x^2(t) + y^2(t)}dt,$$

$$T = \sum_{i=1}^{N-1} d_i = \sum_{i=1}^{N-1} |\mathbf{x}_i - \mathbf{x}_{i+1}|_2.$$

- $\mathbf{x}_1, \dots, \mathbf{x}_n$: boundary coordinate list.
- ***Curvature magnitude***:

$$|k(t)|^2 = \left(\frac{d^2x}{dt^2}\right)^2 + \left(\frac{d^2y}{dt^2}\right)^2.$$

Shape features

- ***Curvature magnitude***:

$$|k(n)| = \frac{1}{\Delta^2}\sqrt{[x(n-1) - 2x(n) + x(n+1)]^2 + [y(n-1) - 2y(n) + y(n+1)]^2}.$$

- Curvature definition as local curve orientation change:

$$k(s) = \frac{d\phi(s)}{ds}$$

$$ds = \sqrt{dx^2 + dy^2}.$$

- $\phi(s)$: orientation of the local curve tangent at position s.

Shape features

- Curvature approximation using chain codes:

$$k(n) \cong \frac{x_n - x_{n-1}}{L(x_n) - L(x_{n-1})},$$

$$L(x_i) = \begin{cases} {}^{1}\!/_{2}, & \text{for } x_i \text{ even}, \\ {}^{\sqrt{2}}\!/_{2}, & \text{for } x_i \text{ odd}. \end{cases}$$

Shape features

- ***Bending energy***:

$$E = \frac{1}{T}\int_0^T |k(t)|^2 dt\,,$$

$$E = \frac{1}{T}\sum_{i=0}^{n-1} |k(i)|^2, \qquad 1 < n < N.$$

- Fourier descriptors can be used for bending energy calculation:

$$E = \sum |Z(k)|^2 \left(\frac{2\pi k}{T}\right)^4,$$

Shape features

- Circle has the minimal bending energy:

$$E = \left(\frac{2\pi}{T}\right)^2.$$

- Bending energy normalization:

$$E_N = 1 - \frac{E_{circle}}{E_{object}} = 1 - \frac{4\pi^2}{T\sum_{i=1}^{n}|k(i)|^2}.$$

Artificial Intelligence & Information Analysis Lab

Shape features

- ***Area*** of object $\mathcal{R}$:

$$A = \iint_{\mathcal{R}} dx\, dy\,.$$

- Area can be approximated by counting pixel numbers:
  - dx, dy describe pixel size.
- ***Differential Geometry*** defines area through object contour $\partial\mathcal{R}$:

$$A = \int_{\partial\mathcal{R}} \left(y(t)\frac{dx}{dt} - x(t)\frac{dy}{dt} \right) dt\,.$$

Shape features

- Object ***compactness or circularity*** γ and its normalized version γ_N :

$$\gamma = \frac{T^2}{4\pi A}, \qquad \gamma_N = 1 - \frac{4\pi A}{T^2}.$$

- Object ***width and height***:

$$w = \max_t x(t) - \min_t x(t),$$
$$h = \max_t y(t) - \min_t y(t).$$

- Object ***diameter***:

$$D = \max_{\mathbf{x}_k, \mathbf{x}_l \in \mathcal{R}} d(\mathbf{x}_k, \mathbf{x}_l),$$

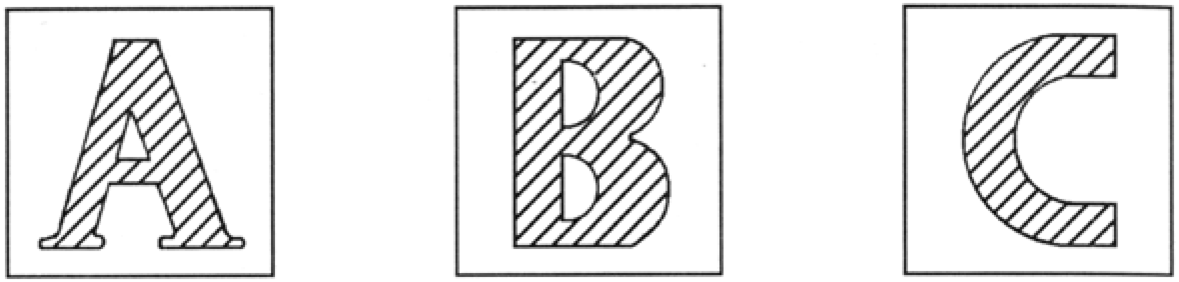
- $\mathbf{x}_k, \mathbf{x}_l$ are two pixels of $\mathcal{R}$ lying further apart.

Shape features

Topological descriptors can give useful global information about an object. Two important topological features are the ***holes*** H and the ***connected components*** C of an object.

Euler number:

$$E = C - H.$$



Letters A, B, C, have Euler numbers 0, -1, 1, respectively.

Shape Description

- Introduction
- Chain Codes
- Polygonal Approximations
- Fourier Descriptors
- Quadtrees
- Pyramids
- Shape Features
- **Moment Descriptors**
- Thinning Algorithms

Moment descriptors

- Image ***moments*** are given by:

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q f(x,y)dxdy. \qquad p,q = 0,1,2,\ldots$$

- ***Center of gravity*** of an object:

$$\bar{x} = \frac{m_{10}}{m_{00}}, \qquad \bar{y} = \frac{m_{01}}{m_{00}}.$$

- ***Central moments***:

$$\mu_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (x-\bar{x})^p (y-\bar{y})^q f(x,y)dxdy. \qquad p,q = 0,1,2,\ldots$$

Moment descriptors

- Moments of discrete images:

$$m_{pq} = \sum_i \sum_j i^p j^q f(i,j),$$

$$\mu_{pq} = \sum_i \sum_j (i - \bar{x})^p (j - \bar{y})^q f(i,j).$$

- Moments of binary images:

$$m_{pq} = \sum_i \sum_j i^p j^q,$$

$$\mu_{pq} = \sum_i \sum_j (i - \bar{x})^p (j - \bar{y})^q.$$

Moment descriptors

- Object ***barycenter*** (center of gravity):

$$\bar{x} = \frac{1}{N} \sum_{(i,j)\in\mathcal{R}} i\,, \qquad \bar{y} = \frac{1}{N} \sum_{(i,j)\in\mathcal{R}} j,$$

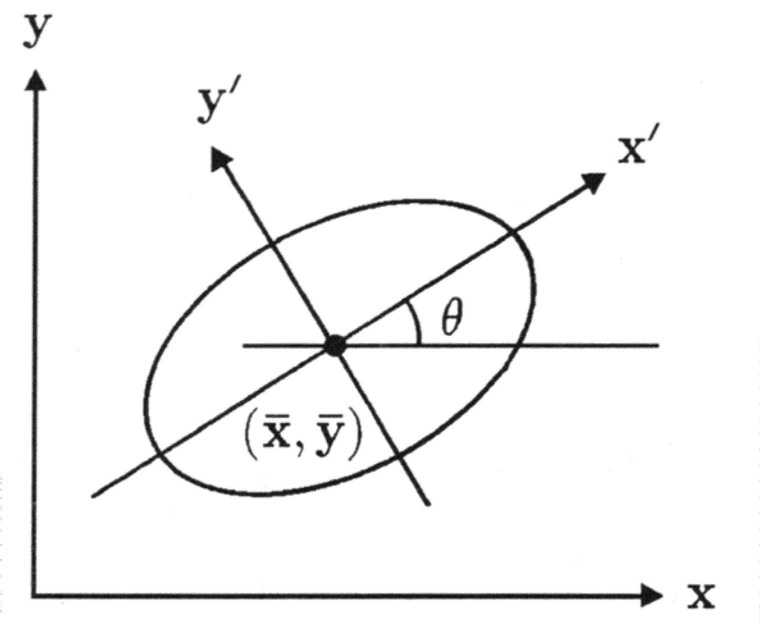
- N is the number of object pixels.

- Object ***orientation θ*** can be derived by minimizing the function:

$$\min_{\theta} S(\theta) = \sum_{(i,j)\in R} \sum [(i - \bar{x}) \cos\theta - (i - \bar{y}) \sin\theta]^2\,,$$

$$\theta = \frac{1}{2} arc \tan\left(\frac{2\mu_{11}}{\mu_{20} - \mu_{02}}\right).$$

Moment descriptors



Definition of object orientation.

Moment descriptors

- Object ***eccentricity***:

$$\varepsilon = \left[\frac{\mu_{02}\cos^2\theta + \mu_{20}\,sin^2\,\theta - \mu_{11}\sin 2\theta}{\mu_{02}\,sin^2\,\theta + \mu_{20}\,cos^2\,\theta - \mu_{11}\cos 2\theta}\right],$$

$$\varepsilon = \left[\frac{(\mu_{02} - \mu_{20})^2 + 4\mu_{11}}{A}\right].$$

- Object ***spread*** (or ***size***):

$$S = (\mu_{02} + \mu_{20}).$$

Shape Description

- Introduction
- Chain Codes
- Polygonal Approximations
- Fourier Descriptors
- Quadtrees
- Pyramids
- Shape Features
- Moment Descriptors
- **Thinning Algorithms**

Artificial Intelligence &
Information Analysis Lab

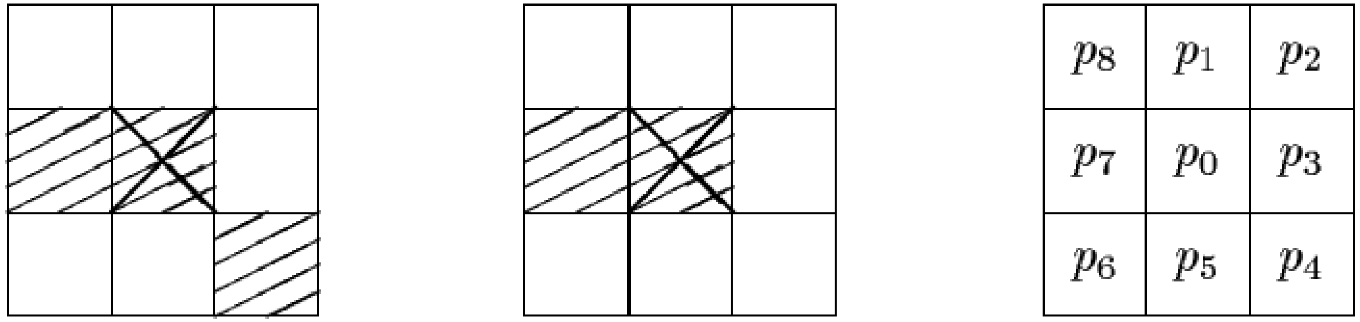
Thinning algorithms

Thinning can be defined heuristically as a set of successive erosions of the outermost layers of a shape, until a connected unit-width set of lines (skeleton) is obtained.

Thinning algorithms satisfy the following two constraints:

1. They maintain ***connectivity*** at each iteration. They do not remove border pixels that may cause discontinuities.
2. They ***do not shorten*** the end of thinned shape limbs.

Thinning algorithms



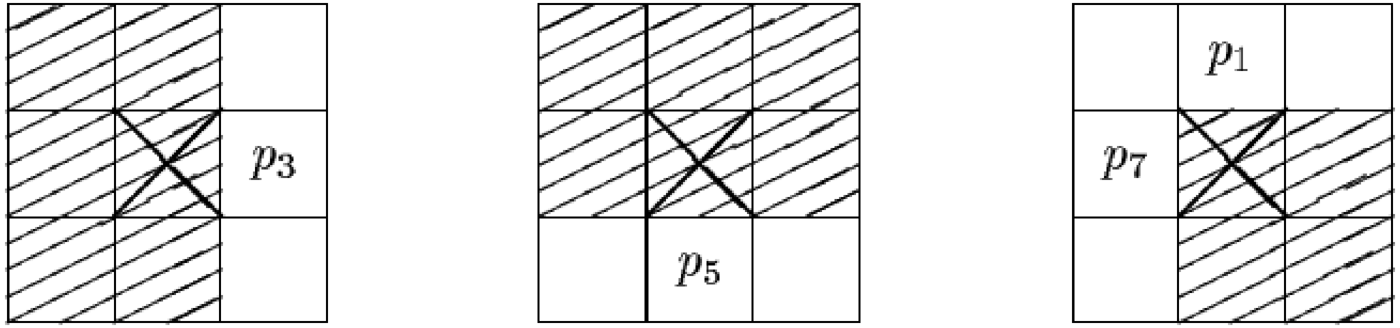
(a) (b) (c)

a) Border pixel, whose removal may cause discontinuities;

b) Border pixel, whose removal will shorten an object limb;

c) Local pixel notation used in connectivity check.

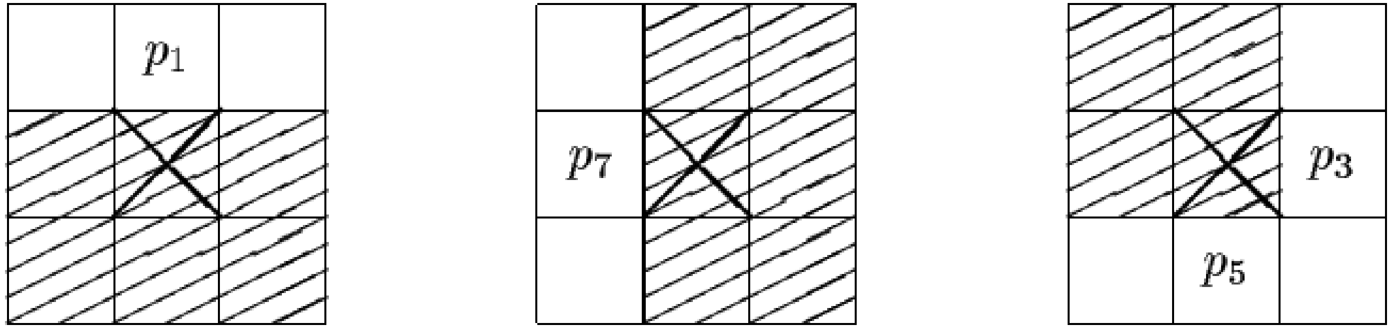
Thinning algorithms



(a) (b) (c)

Central window pixels belonging to: a) an East boundary; b) a South boundary; c) a North-West corner point.

Thinning algorithms



(a) (b) (c)

Central window pixels belonging to: a) a North boundary; b) a West boundary; c) a South-East corner.

Thinning algorithms

One-pass thinning algorithm:

- Check in a local 3×3 image neighborhood:
  - if the object pixel number $N(p_0)$ satisfies: $N(p_0) < 2$ or $N(p_0) > 7$, do nothing.
  - If $2 < N(p_0) < 8$, we check if the removal of the central pixel would break object connectivity:
    - The pixel sequence is formed $p_1 p_2 p_3 \dots p_8 p_1$.
    - If the number of $0 \rightarrow 1$ transitions therein is 1, then the central pixel that has value 1 is removed.

Thinning algorithms

Two-pass thinning algorithm:

- Step 1: a logical rule P_1 is applied in a 3×3 neighborhood and flags the border pixels ***that can be deleted*.**
- Step 2: a logical rule P_2 is applied in a 3×3 neighborhood and flags the border pixels ***that will be deleted*.**

$$P_1: (2 \leq N'(p_0) \leq 6) \&\& (T(p_0) = 1) \&\& (p_1 \cdot p_3 \cdot p_5 = 0) \&\& (p_3 \cdot p_5 \cdot p_7 = 0),$$
$$P_2: (2 \leq N'(p_0) \leq 6) \&\& (T(p_0) = 1) \&\& (p_1 \cdot p_3 \cdot p_7 = 0) \&\& (p_1 \cdot p_5 \cdot p_7 = 0),$$

- $N'(p_0)$: object pixel number in a local 3×3 image neighborhood, but the central pixel.
- $T(p_0)$ denotes the number of the $0 \rightarrow 1$ transitions.

Thinning algorithms

Sobel edge detector output.

Binary Image.

Output of the one-pass thinning Algorithm.

Output of the two-pass thinning Algorithm.

Bibliography

[PIT2019] I. Pitas, "Computer vision", Createspace/Amazon, in press.

[SZE2011] R.Szelinski, " Computer Vision " , Springer 2011

[PIT2017] I. Pitas, "Digital video processing and analysis " , China Machine Press, 2017 (in Chinese).

[PIT2013] I. Pitas, "Digital Video and Television " , Createspace/Amazon, 2013.

[PIT2000] I. Pitas, Digital Image Processing Algorithms and Applications, J. Wiley, 2000.

[NIK2000] N. Nikolaidis and I. Pitas, 3D Image Processing Algorithms, J. Wiley, 2000.

Artificial Intelligence & Information Analysis Lab

Q & A

Thank you very much for your attention!

More material in
http://icarus.csd.auth.gr/cvml-web-lecture-series/

Contact: Prof. I. Pitas
pitas@csd.auth.gr