CHAPTER 7

SHAPE DESCRIPTION

Contents

- **INTRODUCTION**
- **CHAIN CODES**
- **POLYGONAL APPROXIMATIONS**
- **FOURIER DESCRIPTORS**
- **QUADTREES**
- **PYRAMIDS**
- **SHAPE FEATURES**

Contents

- **MOMENT DESCRIPTORS**
- **THINNING ALGORITHMS**
- **MATHEMATICAL MORPHOLOGY**
- **GREYSCALE MORPHOLOGY**
- **SKELETONS**
- **SHAPE DECOMPOSITION**

Introduction

Two-dimensional shapes can be described in two different ways:

Á) ***Use of the object boundary and its features (e.g. boundary length)***. This method is directly connected to edge and line detection. The resulting description schemes are called ***external represantations.***

Â) ***Description of the region occupied by the object on the image plane.*** This method is linked to the region segmentation techniques. The resulting representation schemes are called ***internal representations.***

Introduction

Shape representation schemes must have certain desirable properties:

Uniqueness. This is of crucial importance in object recognition, because each object must have a unique representation

Completeness. This refers to unambiguous representations

Invariance under geometrical transformations. Invariance under translation, rotation, scaling and reflection is very important for object recognition applications.

Introduction

Sensitivity. This is the ability of a representation scheme to reflect easily the differences between similar objects

Abstraction from detail. This refers to the ability of the representation to represent the basic features of a shape and to abstract from detail. This property is directly related to the noise robustness of the representation.

Chain codes

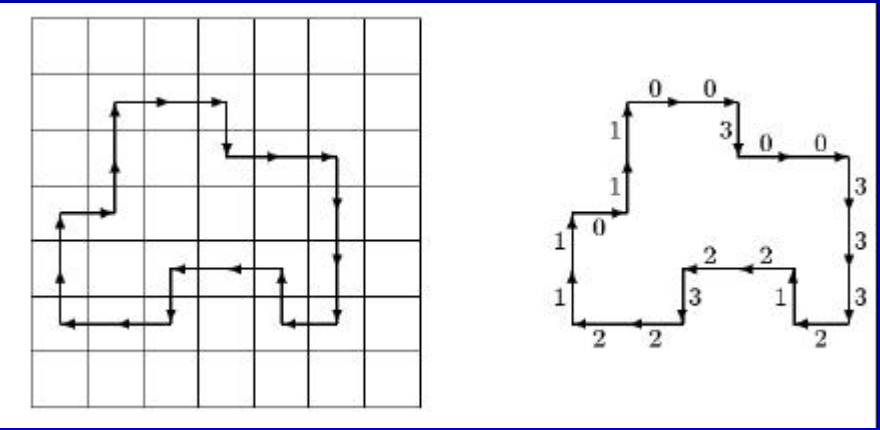
• The chain code depends on the start point of boundary following.

• An advantage of chain code is that it is translation invariant.

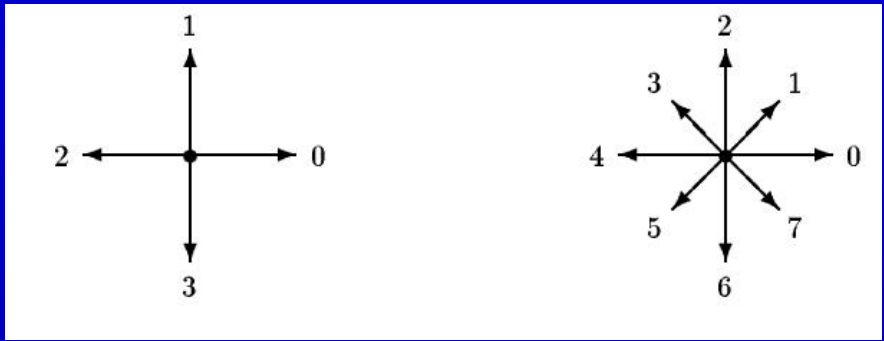
• Scale invariance can be obtained by changing the size of the sampling grid, producing seldom, however, exactly the same chain code.

• Rotation invariance is obtained by using the ***difference chain code.***

Chain codes



Chain code of the digital boundary



(a) (b)

Figure 1: Directions of boundary segments of a chain code for (a) a 4-connected chain; (b) an 8-connected chain.

Chain codes

The ***difference code chain*** is given by:

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

• Chain codes provide a good compression of boundary description.

• Chain codes can also be used to calculate certain boundary features.

Chain codes

The boundary perimeter T is given by:

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

where: $$n_i = \begin{cases} 1 & \text{if } x_i \bmod 2 = 0 \\ \sqrt{2} & \text{if } x_i \bmod 2 = 1 \end{cases}$$ *(in case of an 8-connected chain code)*

The object width w and height h are given by:

$$w = \sum_{i=1}^{N} w_i \quad , \quad h = \sum_{i=1}^{N} h_i$$ *(in case of an 4-connected chain code)*

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

Chain codes

Chain codes can be used in the calculation of object area

The boundaries of binary objects can be easilly followed by employing an algorithm similar to ***Papert's turtle:***

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

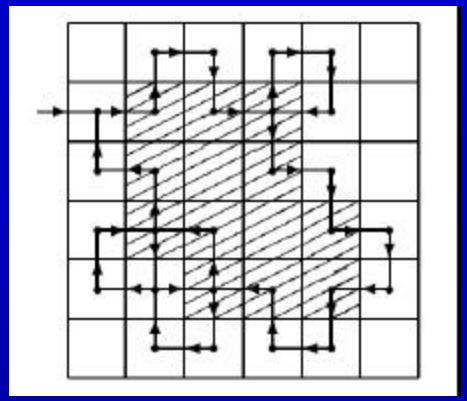


Figure 2: Turtle procedure in binary object boundary following

Polygonal approximations

- Digital boundaries carry information which may be superfluous for certain applications. Boundary approximations can be sufficient in such cases. ***Linear piecewise (polygonal) approximations*** are the most frequently used.
- ***The optimal linear piecewise approximation*** can be obtained by choosing the polygon vertices in such a way that the overall approximation error is minimized.
- Error measures:
  - Mean square $E_2 = \sum_{i=2}^{N-1} |x_i - d_i|^2$
  - Maximal $E_{max} = \max_{2 \le i \le N-1} |x_i - d_i|$

Polygonal approximations

- ***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.

Polygonal approximations

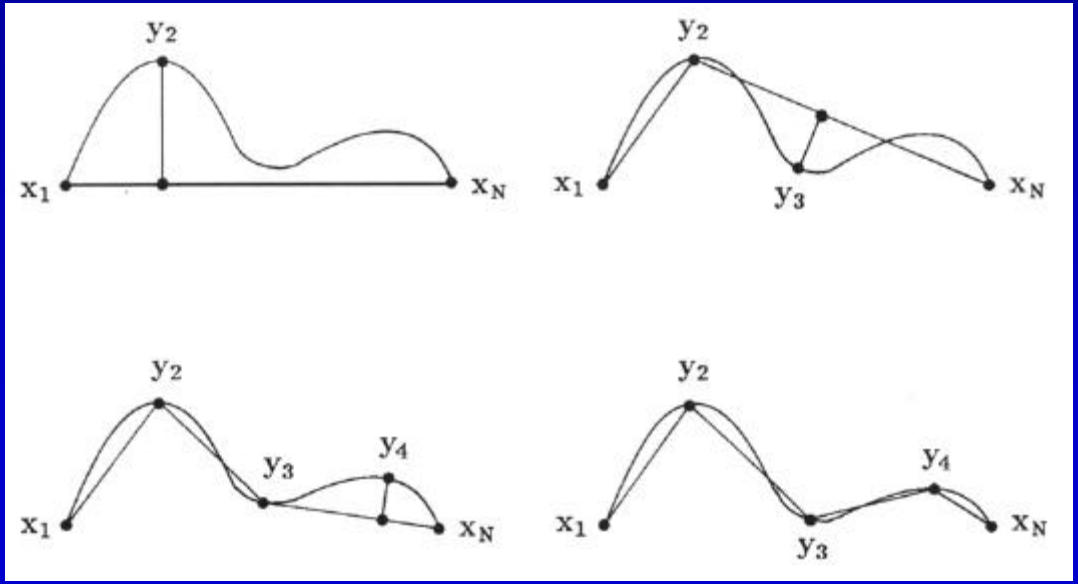
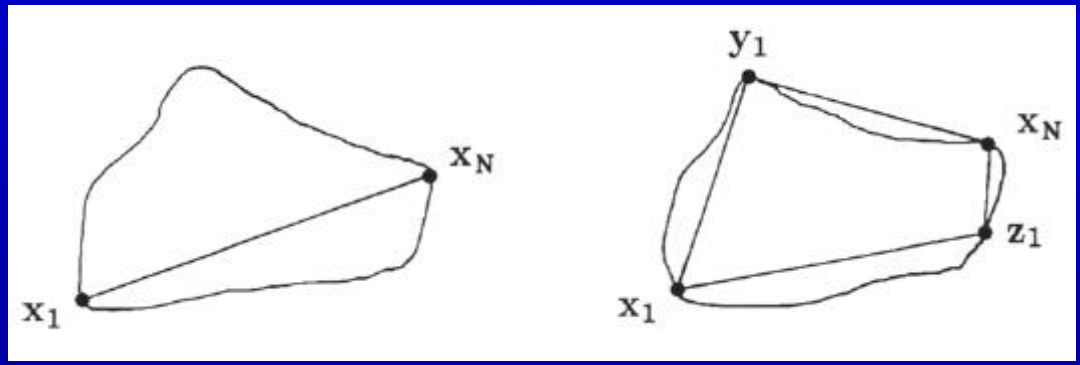


Figure 3:
Splitting method
for polygonal
approximations

Figure 4:
Splitting method
for the linear picewise
approximation
of a closed curve



Polygonal approximations

- A basic advantage of the splitting approach is that it can detect the inflection points on a curve and can use them in curve representation.

- ***Merge techniques in the polygonal approximation*** operate in the opposite way.

- The primary disadvantage of the merge algorithm is that polygon vertices do not coincide with curve inflection points.

- This problem can be solved by combining split and merge techniques.

Fourier descriptors

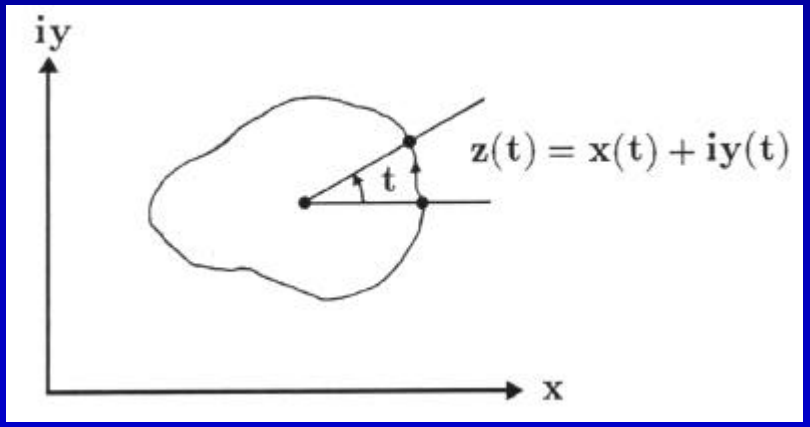


Figure 5: Parametric curve representation

Signal representation using ***Fourier descriptors***

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

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

Fourier representation properties

A) The coefficient *Æ(0)* represents the ***centre of gravity*** of the curve.

B) Fourier coefficients *Z(k)* represent slowly and rapidly varying shape trends for small and large indices *k* respectively.

C) A translation in curve coordinates by z_0 :

$$z_t(n)=z(n)+z_0 \ , \ z_0=x_0+iy_0$$

affects only the term *Z(0)* of the representation:

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

Fourier representation properties

D) A rotation of the curve coordinates by angle $\grave{e}$:

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

results in a phase shift of the transform coefficients by an equal amount:

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

E) A scaling operation by a factor $\acute{a}$, results in a scaling of Fourier coefficients by an equal amount:

$$z_s(n)=\acute{a}z(n)$$
$$Z_s(k)=\acute{a}Z(k)$$

Fourier descriptors

Fourier representation properties

F) A change in the starting point of curve traversal:

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

produces ***modulation*** of the Fourier descriptors:

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

Fourier descriptors have interesting invariance properties that can be used in object recognition applications.

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

(Error measure for matching two curves $z_1(n)$, $z_2(n)$)

Quadtrees

- Quadtrees are based on the following recursive approach: if a binary image region of size $2^n \times 2^n$ consists of both 0s and 1s, it is declared inhomogeneous and is split into four square subregions R_0, R_1, R_2, R_3, having size $2^{n-1} \times 2^{n-1}$ each.
- This procedure continues until all subregions are homogeneous.
- The resulting representation is a ***quadtree***.

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

Maximal number of nodes

Quadtrees

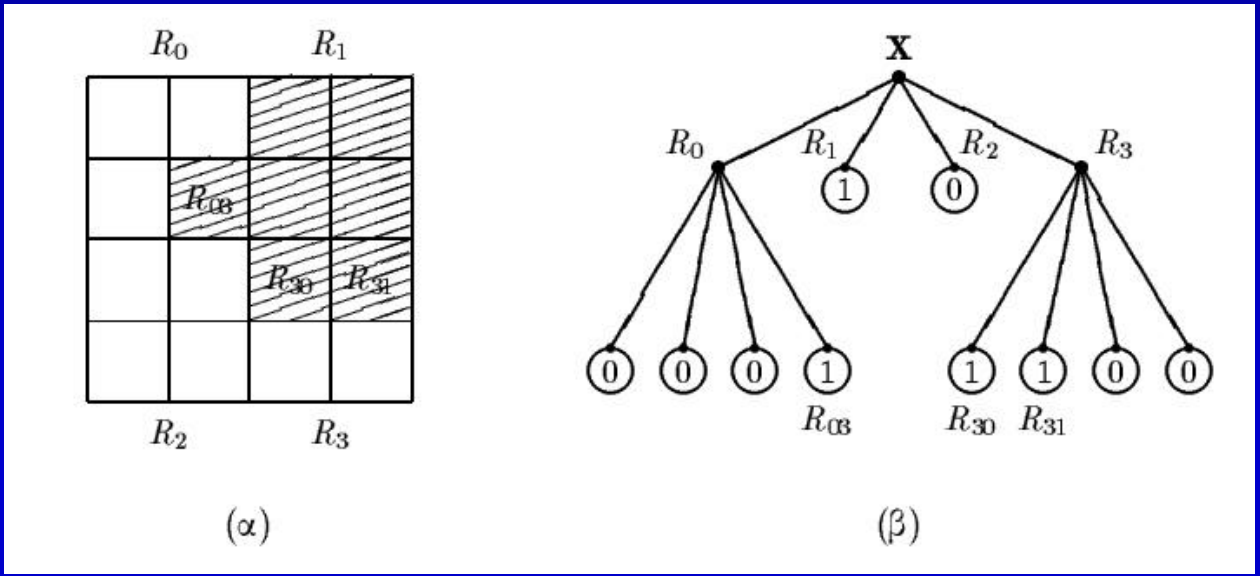


Figure 6: (a) Binary image (b) Quadtree representation

Pyramids

- ***Multiresolution*** representations employ several copies of the same image at different resolutions.
- Multiresolution techniques applied to greyscale or binary images lead to the so-called ***image pyramids.***
- An image pyramid is a series $f_k(i,j)$, $k=0,\ldots,n$ of image arrays, each having size $2^k \times 2^k$.

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

$g(\cdot)$ is a mapping function

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

Pyramids

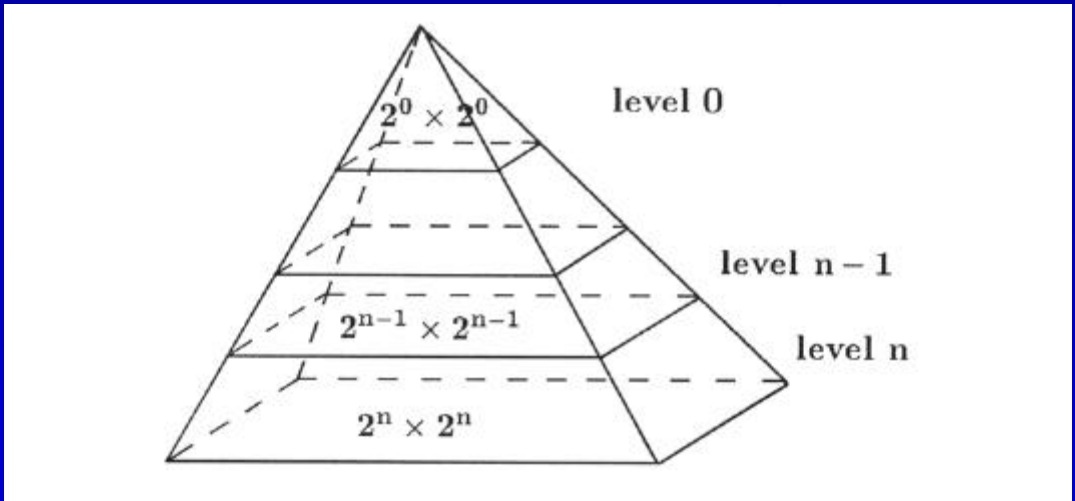
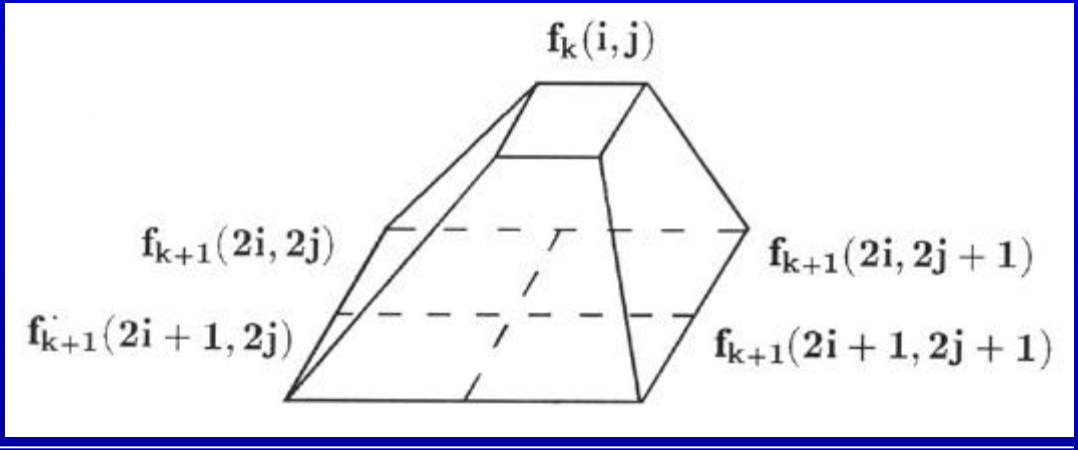


Figure 7:
(a) Image pyramid

(b) Mapping from one pyramid level to the next level.



Pyramids

- Pyramids techniques enjoy a certain popularity for image analysis and compression applications, because they offer abstraction from image details.

- Binary image pyramids can be used in multiresolution edge detection and region segmentation.

- The total space required for the storage of a pyramid (and of a quadtree) is $4/3 \times (2^n \times 2^n)$ where $2^n \times 2^n$ is the size of the original image. Of course, the pyramid can be simply stored on $n+1$ arrays of size $2^k \times 2^k$, $k=0,..,n$.

Pyramids

Figure 8:
(a) Original binary image

(b) Image pyramid

(c) Output of the pyramid edge detector

(d) Edge pyramid

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:***

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

$$T = \sum_{i=1}^{N-1} d_i = \sum_{i=1}^{N-1} | x_i - x_{i+1} |$$

$\boldsymbol{x}_1,\ldots,\boldsymbol{x}_N$: *boundary coordinate list*

Curvature magnitude:

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

Shape features

Curvature magnitude:

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

Another curvature definition:

$$k(s) = \frac{d\phi(s)}{ds} \quad \text{where} \quad ds = \sqrt{dx^2 + dy^2}$$

Approximaton of the local curvature:

$$k(n) \cong \frac{x_n - x_{n-1}}{L(x_n) - L(x_{n-1})} \quad \text{where} \quad L(x_i) = \begin{cases} \frac{1}{2} & \text{,for } x_i \text{ even} \\ \frac{\sqrt{2}}{2} & \text{, for } x_i \text{ odd} \end{cases}$$

Shape features

Bending energy:

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

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

$$E = \sum |Z(k)|^2 \left(\frac{2pk}{T}\right)^4$$ *Calculated from boundary Fourier descriptors*

$$E = \left(\frac{2p}{T}\right)^2$$ Circle bending energy

Normalization of bending energy

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

Shape features

Object area

$$A = \iint_R dxdy$$

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

using a differential geometry formula

Compactness or circularity

$$g = \frac{T^2}{4pA}$$

$$g_N = 1 - \frac{4pA}{T^2}$$

normalized version

Shape features

Object ***width and height***

$$w = \max_t x(t) - \min_t x(t)$$

$$h = \max_t y(t) - \min_t y(t)$$

Object ***diameter***

$$D = \max_{X_k, X_l \in R} d(X_k, X_l)$$

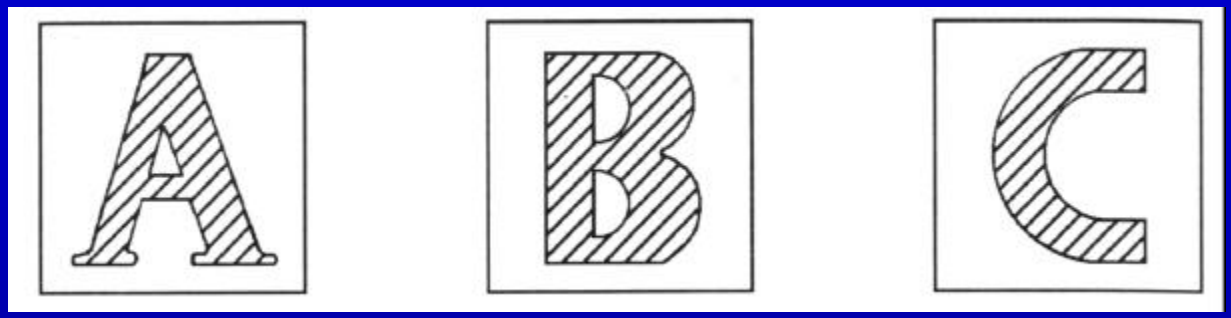
where $X_k X_l$ is the direction of the line segment

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.

Moment descriptors

The ***moments*** of an image ***f(x,y)*** are given by:

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

Centre of gravity of an object

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

Central moments

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

Moment descriptors

Moment relations for discrete images

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

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

Moment relations for binary images

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

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

Moment descriptors

Coordinates of the centre of gravity

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

where $Í$ is the area of an image in pixels

Object ***orientation è***: can be derived by minimizing the function:

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

$$q = \frac{1}{2} \arctan\left(\frac{2m_{11}}{m_{20} - m_{02}} \right)$$

Moment descriptors

Object ***eccentricity***

$$e = \left[\frac{m_{02}\cos^2 q + m_{20}\sin^2 q - m_{11}\sin 2q}{m_{02}\sin^2 q + m_{20}\cos^2 q - m_{11}\cos 2q} \right]$$

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

Object ***spread or size***

$$S = (m_{02} + m_{20})$$

Moment descriptors

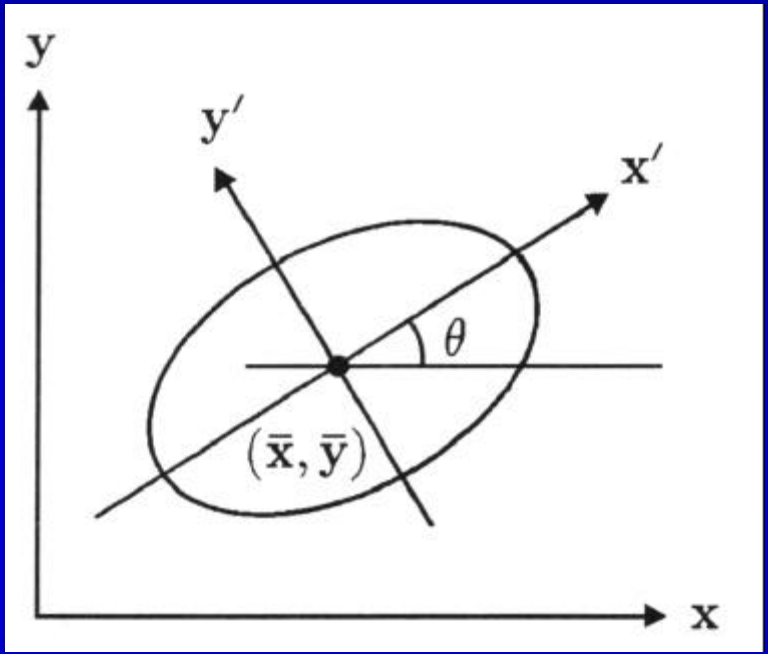


Figure 9: Definition of object orientation

Thinning algorithms

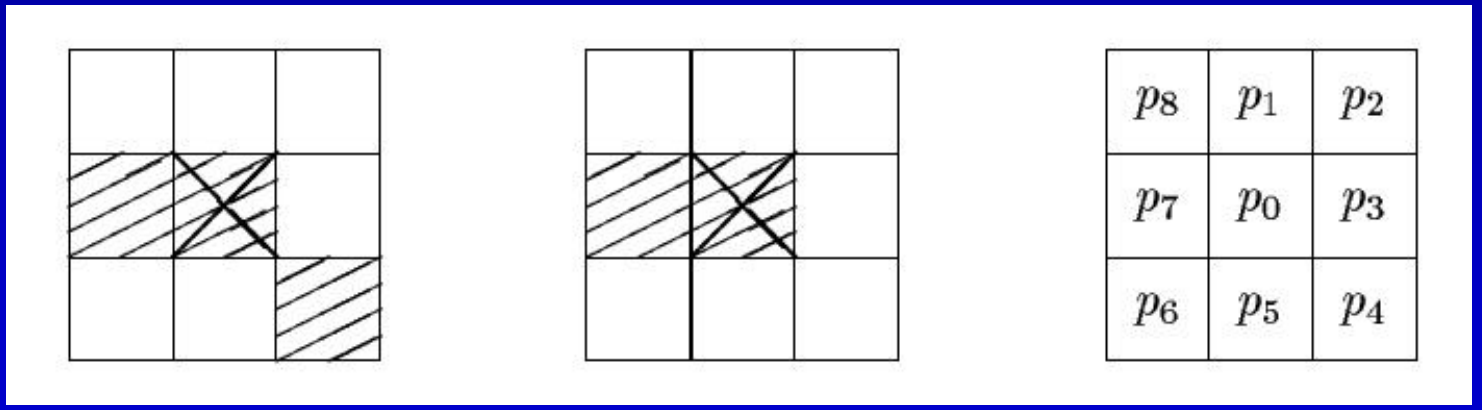
Thinning can be defined heuristically as a set of succesive erosions of the outermost layers of a shape, until a connected unit-width set of lines (skeleton) is obtained.

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.

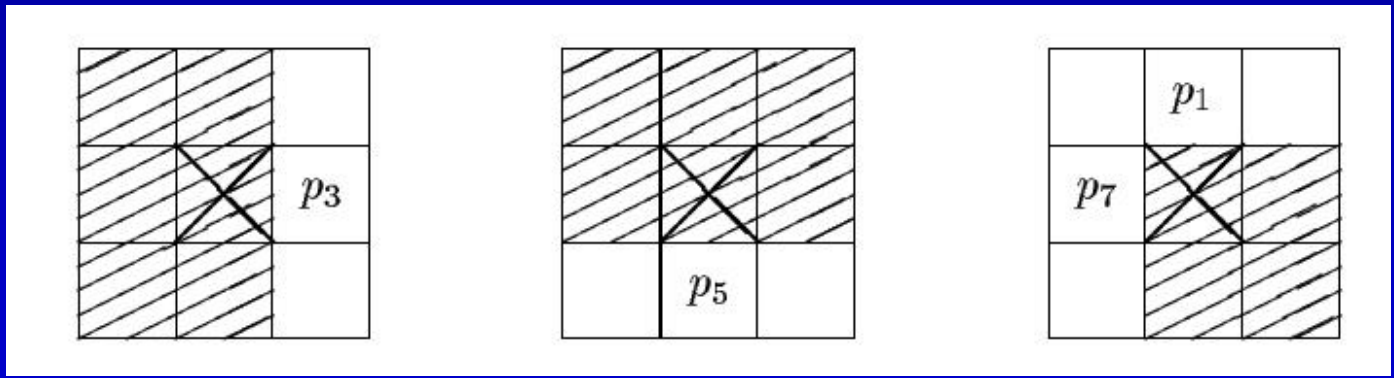
Thinning algorithms



(a) (b) (c)

Figure 10:
(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.

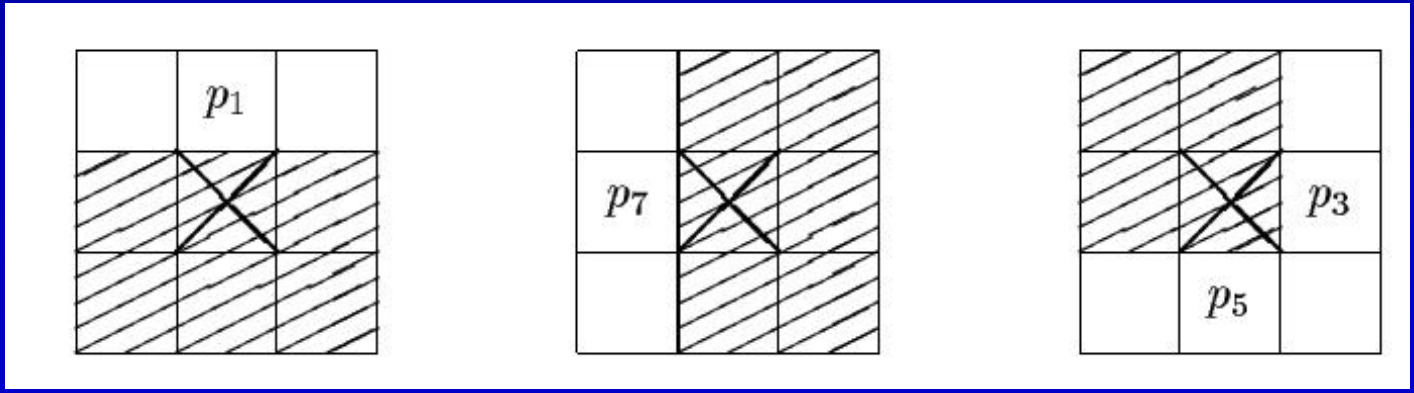
Thinning algorithms



(a) (b) (c)

Figure 11: Central window pixels belonging to: (a) an East boundary; (b) a South boundary; (c) a North-West corner point.

Thinning algorithms



(a) (b) (c)

Figure 12: Central window pixels belonging to: (a) a North boundary; (b) a West boundary; (c) a South-East corner.

Thinning algorithms

First thinning algorithm

- Check in a local neighborhood 3×3
- If the number of the pixels of the object (except the central) $\acute{I}(p_0)$ is: $2 < \acute{I}(p_0) < 8$)
  - we check if the removal of the central pixel would break object connectivity.

    Check
    - The pixel sequence is formed $p_1p_2p_3...p_8p_1$.
    - If the number of $0 \rightarrow 1$ transitions is 1, then the central pixel that has value 1 is removed.

Second thinning algorithm

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

$$P_1: (2 \le N(p_0) \le 6) \&\& (T(p_0) = 1) \&\& (p_1 . p_3 . p_5 = 0) \&\& (p_3 . p_5 . p_7 = 0)$$

$$P_2: (2 \le N(p_0) \le 6) \&\& (T(p_0) = 1) \&\& (p_1 . p_3 . p_7 = 0) \&\& (p_1 . p_5 . p_7 = 0)$$

where $\hat{O}(p_0)$ denotes the number of the $0 \rightarrow 1$ transitions.

Thinning algorithms

Figure 13: Sobel edge detector output

Binary image

Output of the one-pass thinning algorithm

Output of the two-pass thinning algorithm

Mathematical morphology

Mathematical morphology uses a set theoretic approach to image analysis.

The morphological transformations must possess the following properties:

1. Translation invariance

$$Ø(X_z)=[Ø(X)]_z$$

2. Scale invariance

$$Ø_ë(X)=ëØ(ë^{-1}X)$$

Mathematical morphology

3. Local knowledge. Transformation Ø(×) must require only information within a local neighbourhood for its operation

4. Semicontinuity. The morphological transformation Ø(×) must possess certain continuity properties.

Basic morphological transformations

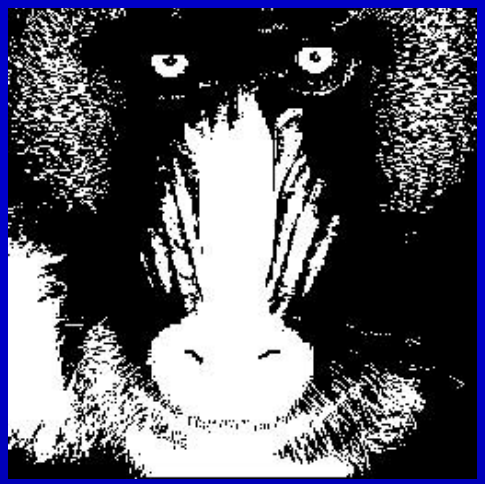
dilation $$X \oplus B^s = \bigcup_{b \in B} X_{-b} = \{z \in E : B_z \bigcap X \neq 0\}$$

erosion $$X \ominus B^s = \bigcap_{b \in B} X_{-b} = \{z \in E : B_z \subset X\}$$

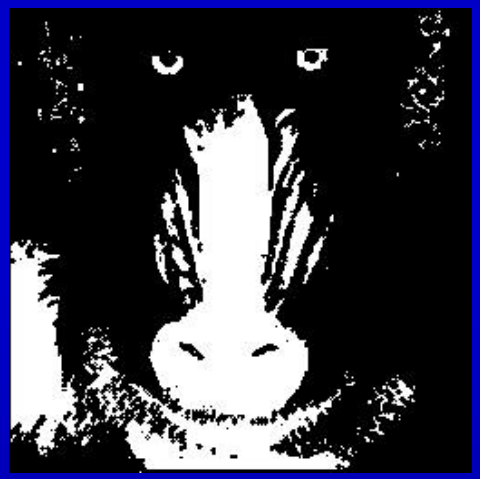
Mathematical morphology

Erosion and dilation are special cases of ***Minkowski set addition*** and ***Minkowski set subtraction***

$$X \oplus B = \bigcup_{b \in B} X_b \qquad X \ominus B^s = \bigcap_{b \in B} X_b$$



(a)



(b)

(c)

Figure 14: (a) thresholded image (b) eroded and (c) dilated image by the structuring elements SQUARE.

Mathematical morphology

Erosion, dilation, Minkowski set addition and subtraction have the following interesting properties:

Commutativity:

$$A \oplus B = B \oplus A$$

Associativity:

$$A \oplus (B \oplus C) = (A \oplus B) \oplus C$$

Mathematical morphology

Translation invariance:

$$A_z \oplus B = (A \oplus B)_z$$

$$A_z \ominus B = (A \ominus B)_z$$

$$A \ominus B_z = (A \ominus B)_z$$

Increasing property:

$$A \subseteq B \Rightarrow A \oplus D \subseteq B \oplus D$$

$$A \subseteq B \Rightarrow A \ominus D \subseteq B \ominus D$$

Distributivity:

$$(A \bigcup B) \oplus C = (A \oplus C) \bigcup (B \oplus C)$$

$$A \oplus (B \bigcup C) = (A \oplus B) \bigcup (A \oplus C)$$

$$A \ominus (B \bigcup C) = (A \ominus B) \bigcap (A \ominus C)$$

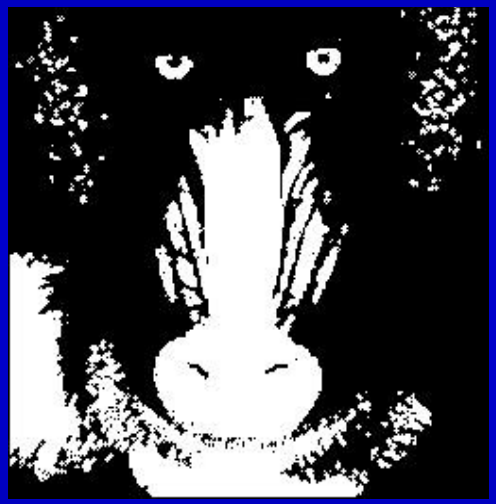
$$(A \bigcap B) \ominus C = (A \ominus C) \bigcap (B \ominus C)$$

$$A \ominus (B \oplus C) = (A \ominus B) \ominus C$$

Mathematical morphology

Opening $\times_{\hat{A}}$: $$X_B = (X \ominus B^s) \oplus B = \bigcup \{B_z : B_z \subset X\}$$

Closing X^B: $$X^B = (X \oplus B^s) \ominus B = \bigcap \{B_z^c : B_z \subset X^c\}$$



(a)

(b)

Figure 15: (a) opened image (b) closed image.

Opening and closing properties

Duality:

$$(X^{B})^{c} = (X^{c})_{B}$$

$$(X_{B})^{c} = (X^{c})^{B}$$

Extensivity and antiextensivity:

$$X_{B} \subset X$$

$$X^{B} \supset X$$

Mathematical morphology

Increasing property:

$$X_1 \subset X_2 \Rightarrow (X_1)_B \subset (X_2)_B$$

$$X_1 \subset X_2 \Rightarrow (X_1)^B \subset (X_2)^B$$

Idempotence:

$$(X_B)_B = X_B$$

$$(X^B)^B = X^B$$

Mathematical morphology

Definition of binary dilation

$$X \oplus B^s = \left\{ z \in E : B_z \bigcap X \neq 0 \right\}$$

Definition of binary erosion

$$X \ominus B^s = \left\{ z \in E : B_z \subset X \right\}$$

An alternative way for the calculation of binary erosion and dilation

$$X \oplus B^s = \bigcup_{b \in B} X_{-b}$$

$$X \ominus B^s = \bigcap_{b \in B} X_{-b}$$

Greyscale morpholgy

The tools for greyscale morphological operations are simple functions g(x) having domain G. They are called ***structuring functions***

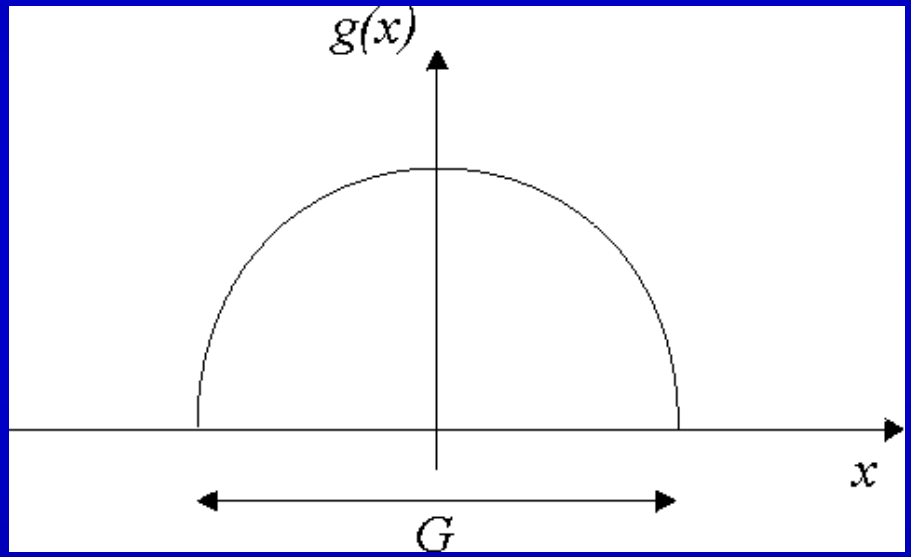


Figure 16: A example of a structuring function.

Greyscale morpholgy

Greyscale dilation and erosion of a function f(x) by g(x)

$$[f \oplus g^s](x) = \max_{z \in D, z-x \in D} \{f(z) + g(z-x)\}$$

$$[f \ominus g^s](x) = \min_{z \in D, z-x \in D} \{f(z) - g(z-x)\}$$

Greyscale opening and closing

$$f_g(x) = [(f \ominus g^s) \oplus g](x) = [f(x) \ominus g(-x)] \oplus g(x)$$

$$f^g(x) = [(f \oplus g^s) \ominus g](x) = [f(x) \oplus g(-x)] \ominus g(x)$$

Greyscale morpholgy

Implementation of greyscale dilation and erosion in pipeline

$$f \oplus g = (\ldots((f \oplus g_1) \oplus g_2) \oplus \ldots \oplus g_k)$$

$$f \ominus g = (\ldots((f \ominus g_1) \ominus g_2) \ominus \ldots \ominus g_k)$$

Dilation and erosion of a function by a set

$$[f \oplus G^s](x) = [f \oplus g^s](x) = \max\{f(i-v), \ldots, f(i), \ldots, f(i+v)\}$$

$$[f \ominus G^s](x) = [f \ominus g^s](x) = \min\{f(i-v), \ldots, f(i), \ldots, f(i+v)\}$$

Opening and closing of a function by a set

$$f_G(x) = [(f \ominus G^s) \oplus G](x)$$

$$f^G(x) = [(f \oplus G^s) \ominus G](x)$$

Greyscale morpholgy

Close-opening filter (CO)

$$y = [(f^{G})_{G}](x)$$

open-closing filter (OC)

$$y = [(f_{G})^{G}](x)$$

The algebraic difference $y=f(x)-f_{nB}(x)$ is a nonlinear high-pass filter, called ***top-hat transformation.***

Greyscale morpholgy

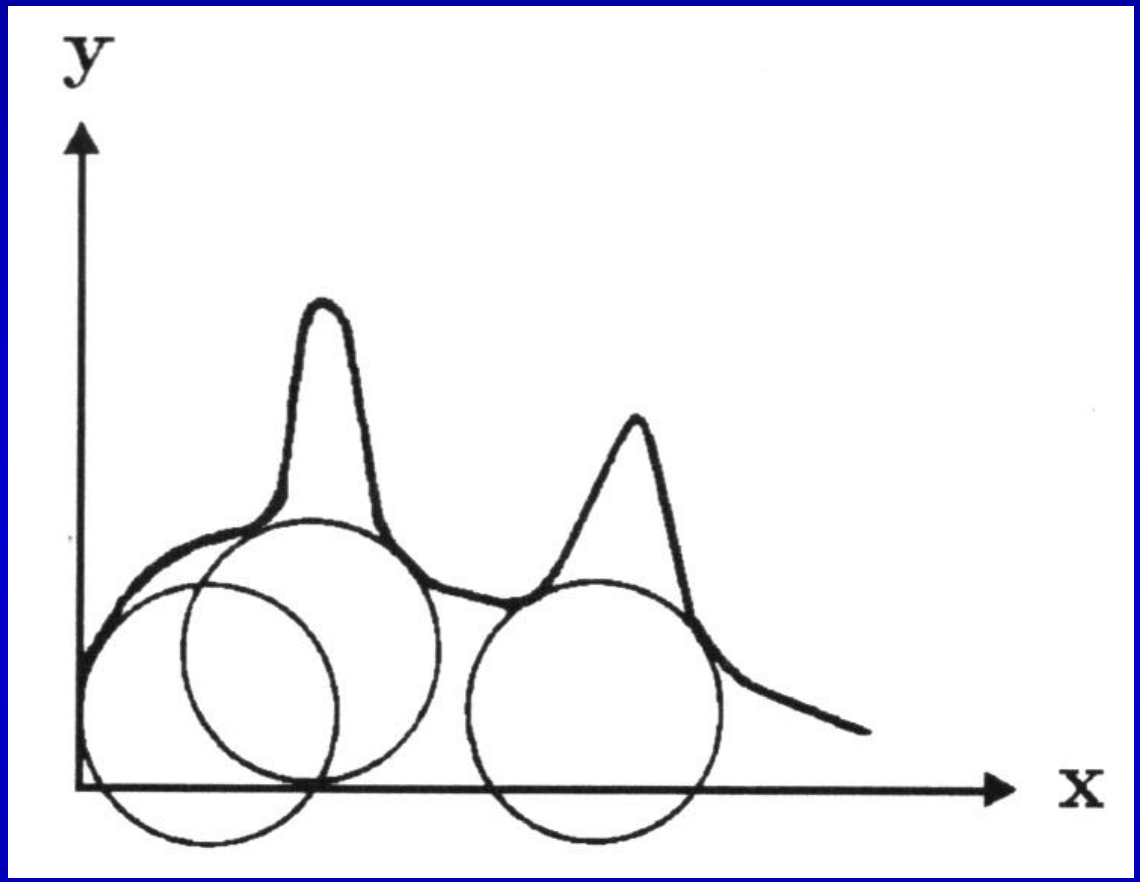
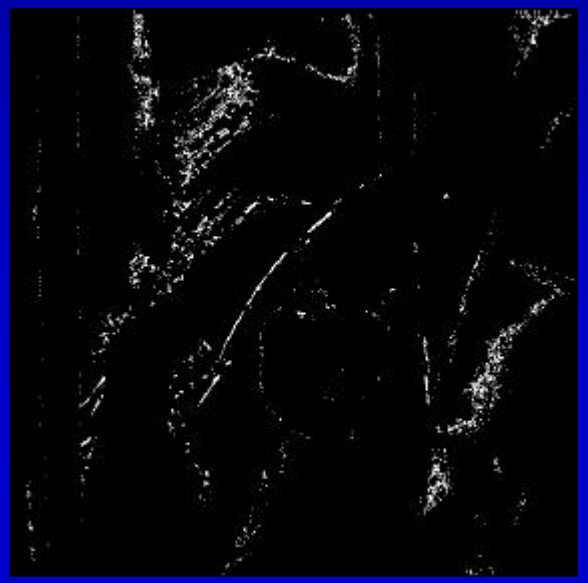


Figure 17: Opening as a rolling ball transformation

Greyscale morpholgy

(a)

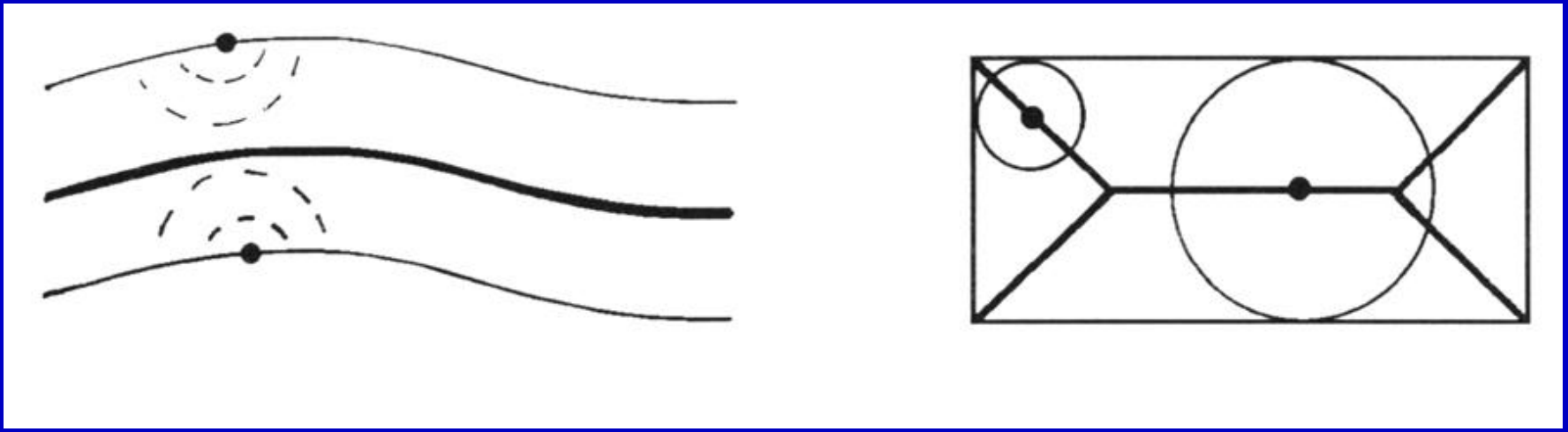


(b)

Figure 18: (a) Thresholded image, (b) Result of top-hat filtering

Skeletons

Object ***skeleton*** is an important topological descriptor of a two-dimensional binary object



(a) (b)

Figure 19: (a) Grassfire propagation model of medial axis;
(b) maximal disk definition of skeleton.

Skeletons

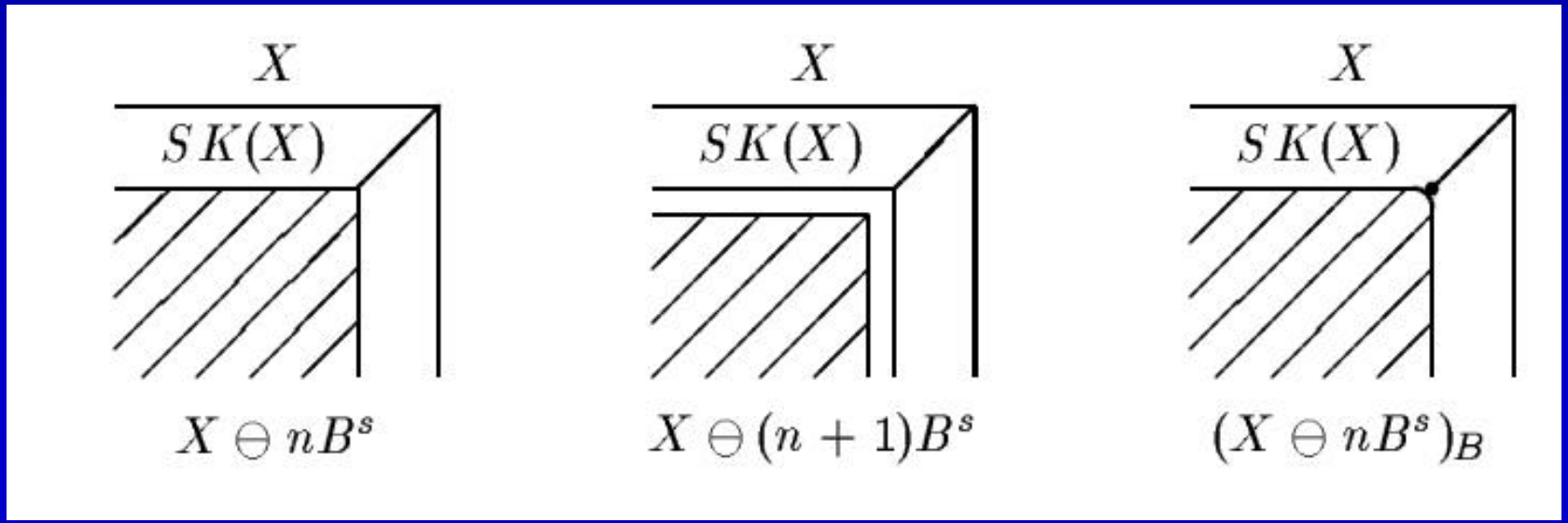
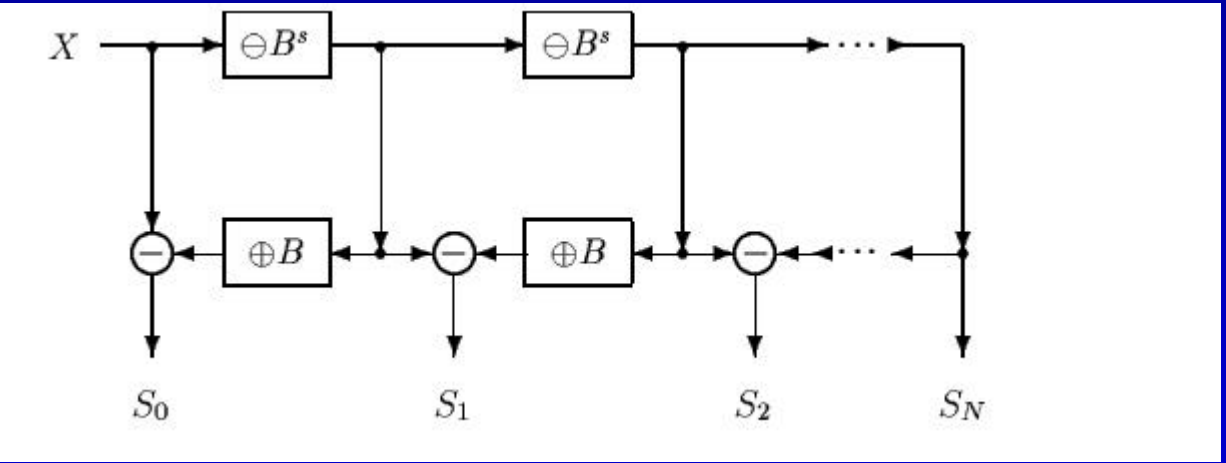
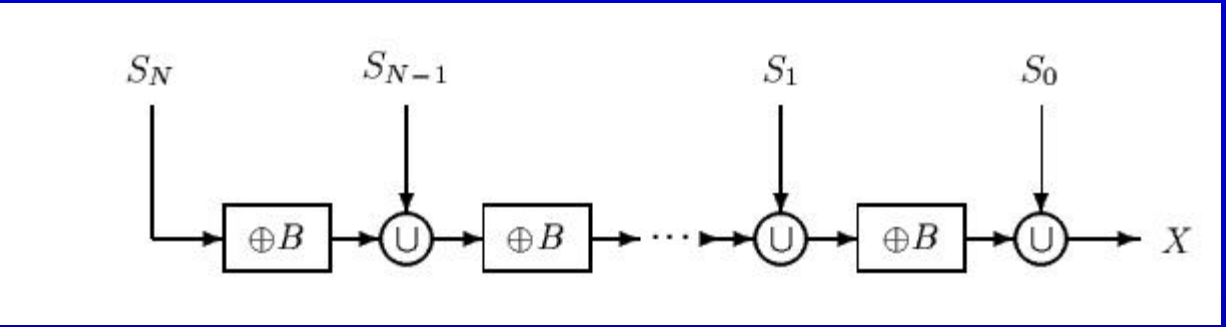


Figure 20: Illustration of morphological skeletonization

Skeletons



(a)



(b)

Figure 21: (a) Fast skeletonization algorithm;
(b) fast object reconstruction from skeleton subsets.

Shape decomposition

• A complex object X can be decomposed into a union of 'simple' subsets $X_1,\ldots, X_n$, thus providing an intuitive object description scheme called ***shape decomposition.***

• Shape decomposition must use simple geometrical primitives in order to conform with our intuitive notion of simple shapes.

• The complexity of the decomposition must be small compared with the original description of X.

• A small noise sensitivity is desirable.

Shape decomposition

Morphological shape decomposition

Recursive relation:

$$X_i = (X - X'_{i-1})_{n_i B}$$

$$X'_i = \bigcup_{j=1}^{i} X_j$$

$$X'_0 = \emptyset$$

Stopping condition: $(X - X'_K) \ominus B^s = \emptyset$

Figure 22: (a) Original binary image;
(b) first 16 components of its morphological shape decomposition.

Shape decomposition

Blum ribbons

Simple objects X_i of the form:

$$X_i = L_i \oplus n_i B$$

Disadvantages of morphological shape decomposition

It is susceptible to boundary noise.

The representation produced is not close to human shape perception if the object consists of unions, intersections and differences of various geometrival primitives. This can be alleviated by combining morphological techniques with ***constructive solid geometry, (CSG).***

Shape decomposition

Main advantage of CSG over skeleton representation or morphological shape decomposition

• CSG uses a multitude of geometrical primitives (e.g. squares) instead of one.

• This fact not only enhances the descriptive power of CSG but also conforms to our intuitive notion of simple geometrical shapes.