Digital Image Processing

Arithmetic/Logic/Morphological Operations

Arithmetic/Logic Operations

- Arithmetic/Logic operations are performed on the pixels of two or more images
- The operation is executed in a pointwise manner for all images taking part in
- Arithmetic:
  - Addition: p+q
  - Subtraction: p-q
  - Multiplication: p*q
  - Division: p/q
- Logic
  - And
  - Or
  - Xor
  - Not

Arithmetic Operations

- Addition: $C(x,y) = A(x,y) + B(x,y)$
- Applications:
  - Remove additive noise
  - Generative additive effect

Arithmetic Operations

- Addition:
  - $C(x,y) = \alpha A(x,y) + \beta B(x,y)$

Arithmetic Operations

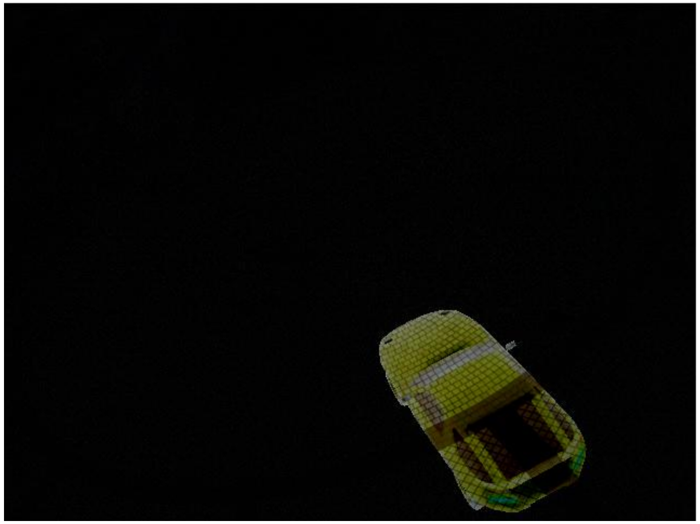
- Addition:Remove additive noise
  - $g_i(x,y) = f(x,y) + n_i(x,y)$, $i=1,2,\ldots,M$, $n \sim (0, \sigma)$, iid
  - Averaging: $g(x,y) = 1/M\,(g_0(x,y)+g_1(x,y)+\ldots+ g_M(x,y))$
  - $D(g)=1/M\ D(n)$

Arithmetic Operations

- Subtraction: $C(x,y) = A(x,y) - B(x,y)$
- Applications:
  - Foreground extraction
  - Motion detection
  - Enhancement

Arithmetic Operations

- Subtraction:
  - Foreground extraction

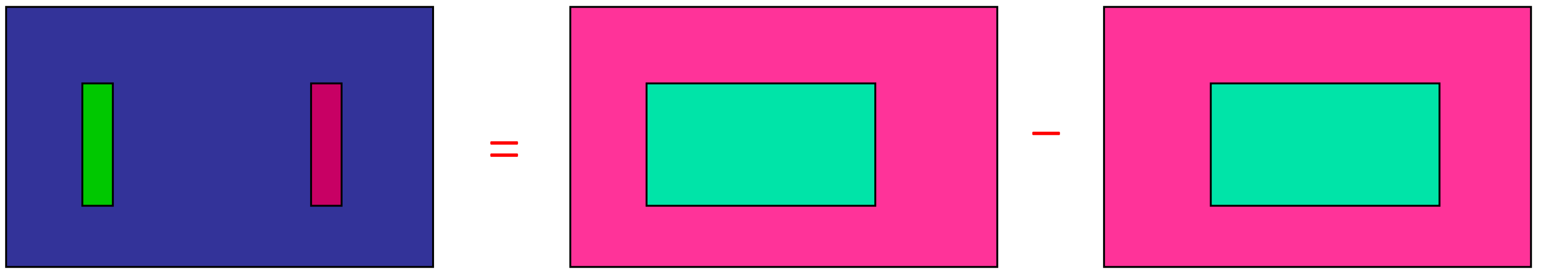


Arithmetic Operations

- Subtraction:
  - Motion detection

T1(x,y) : Image at time 1,
T2(x,y) : Image at time 2,

$$g(x,y) = T2\,(x,y) - T1(x,y)$$



Arithmetic Operations

- Subtraction:
  - Edge Enhancement

Image

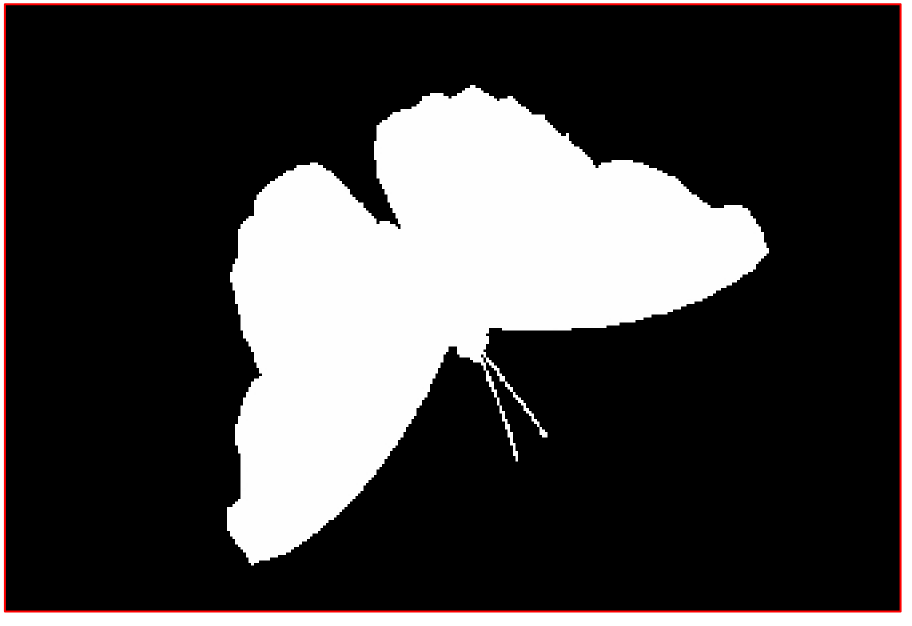
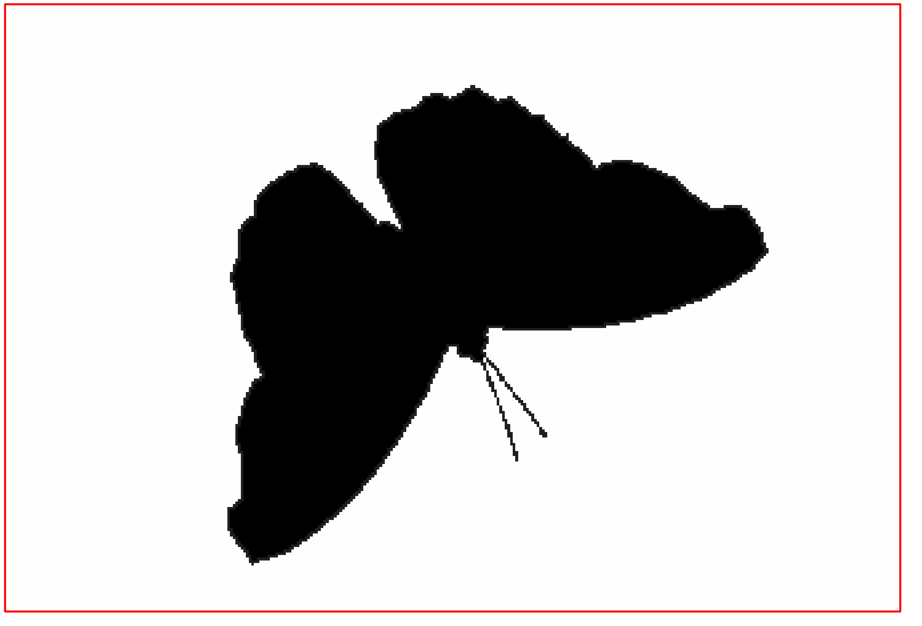
-

Blurred Image

=

Logic Operations

- Not: $g(x,y) = 255 - f(x,y)$

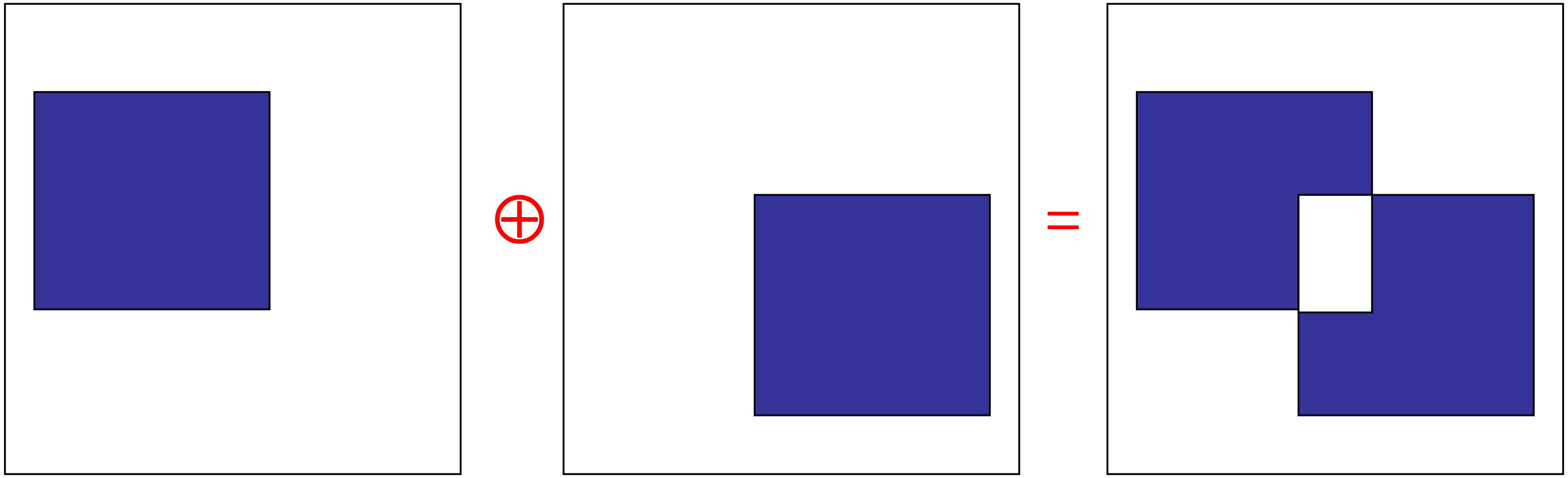



Logic Operations

- Not: $g(x,y) = 255 - f(x,y)$


Logic Operations

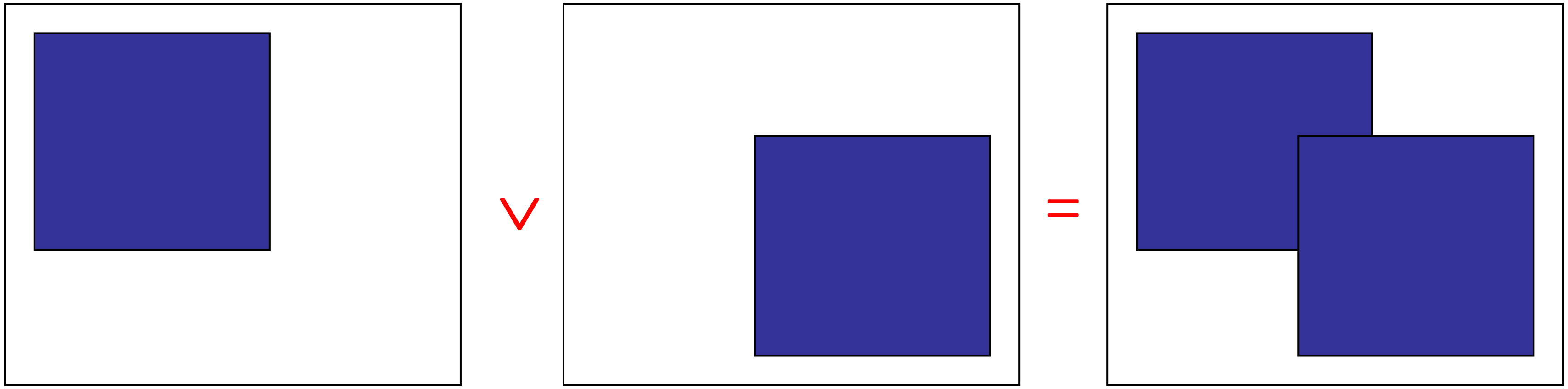
- Xor: $g(x,y) = f(x,y) \oplus h(x,y)$



Foreground: 1; Background: 0

Logic Operations

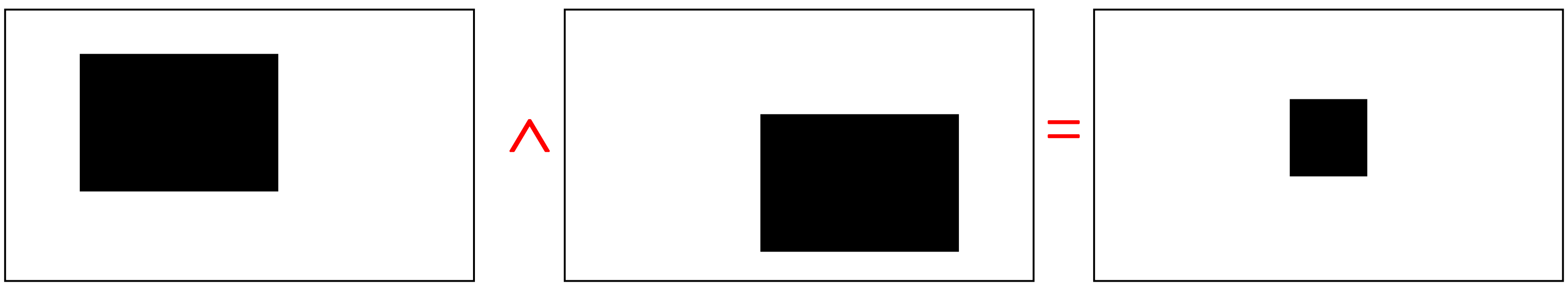
- Or: $g(x,y) = f(x,y) \vee h(x,y)$
- Application: Set union



Foreground: 1; Background: 0

Logic Operations

- And: $g(x,y) = f(x,y) \wedge h(x,y)$
- Application: Set intersection

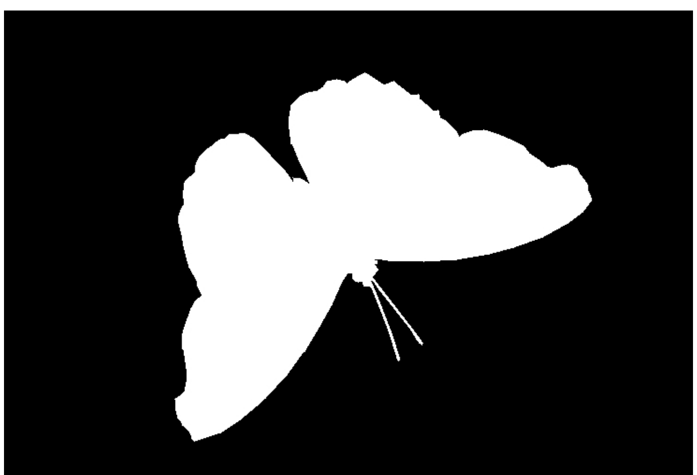


Logic Operations

- And: $g(x,y) = f(x,y) \wedge h(x,y)$

$=$

Is it possible to use OR to achieve this ?



$\wedge$

Binary Image Processing

- Introduction
- Set theory review
- Morphological filtering
  - Erosion and dilation
  - Opening and closing
  - Hit-or-miss, boundary extraction, …
- Skeleton via distance transform

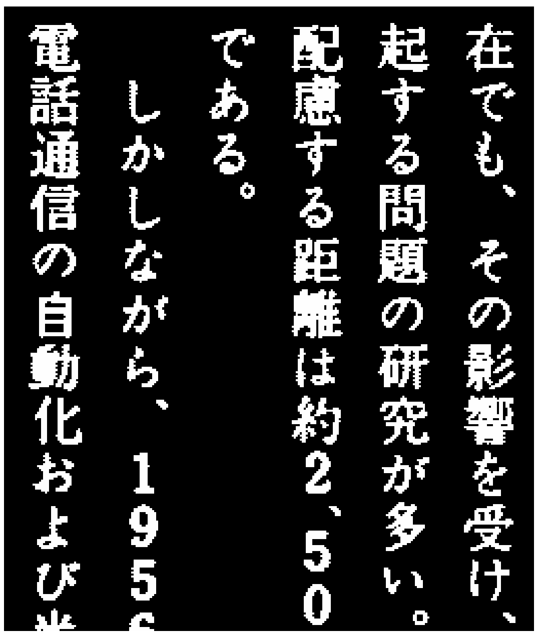
Binary Images

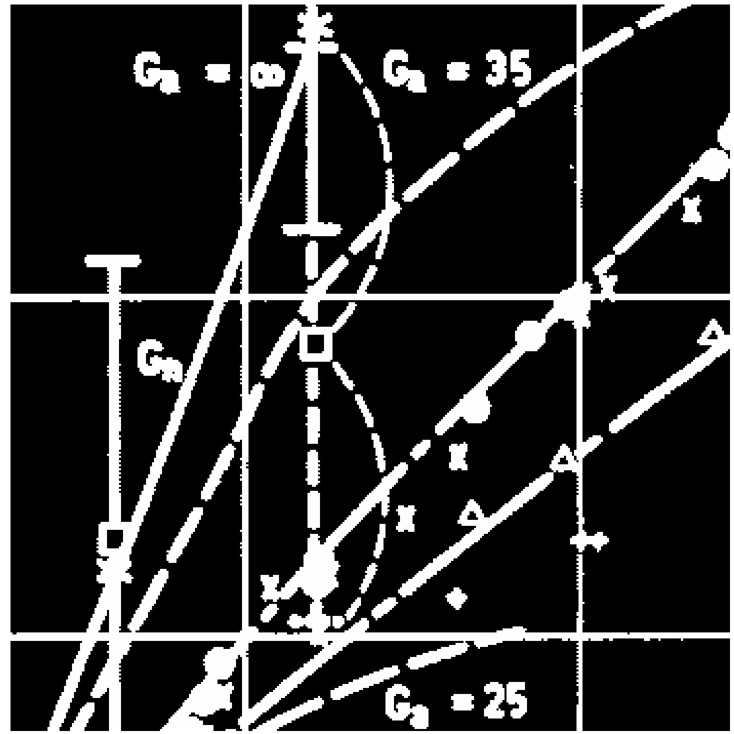
- Images only consist of two colors (tones): white or black

Numerical example (image of a square block)

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	255	255	255	255	0	0
0	0	255	255	255	255	0	0
0	0	255	255	255	255	0	0
0	0	255	255	255	255	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Binary Image Examples





Why are binary images special?

- Since pixels are either white or black, the locations of white (black) pixels carry ALL information of binary images

<u>Example</u>

$$\begin{matrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{matrix} \quad \Longleftrightarrow \quad L=\{(3,3),(3,4),(4,3),(4,4)\}$$

f(m,n)

location of white pixels

matrix representation

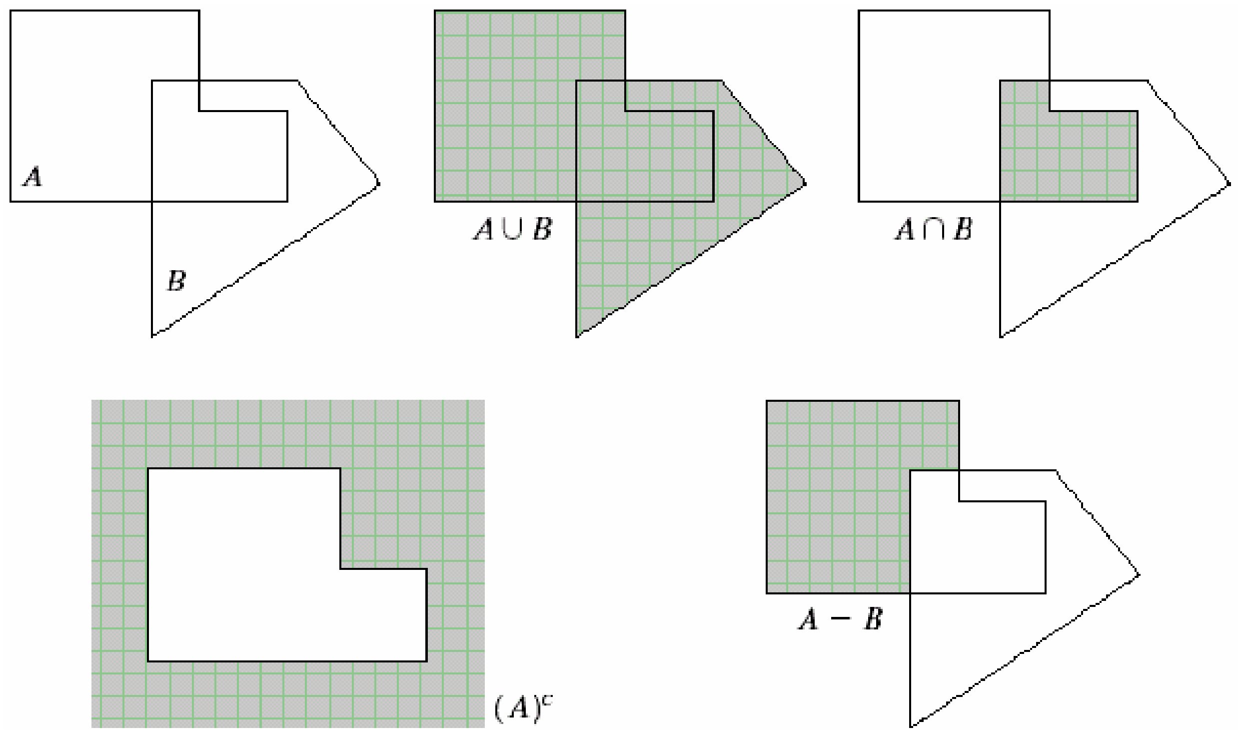
set representation

It is often more convenient to consider the set representation than the matrix representation for binary images

Binary Image Processing

- Introduction
- Set theory review
- Morphological filtering
  - Erosion and dilation
  - Opening and closing
  - Hit-or-miss, boundary extraction, …
- Skeleton via distance transform

Set Theory Review



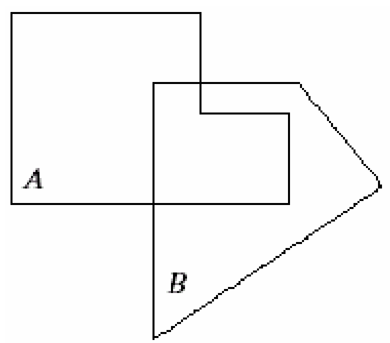
$$A^c = \{w \mid w \notin A\} \qquad A - B = \{w \mid w \in A, w \notin B\}$$

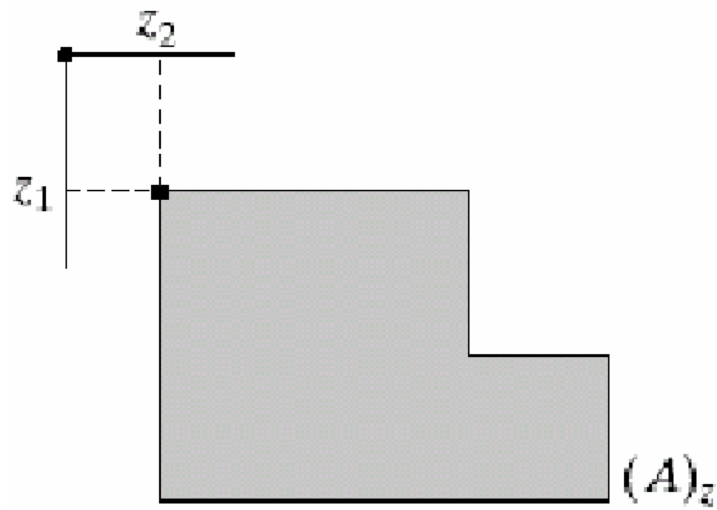
Think of sets A and B as the collections of spatial coordinates

Translation Operator

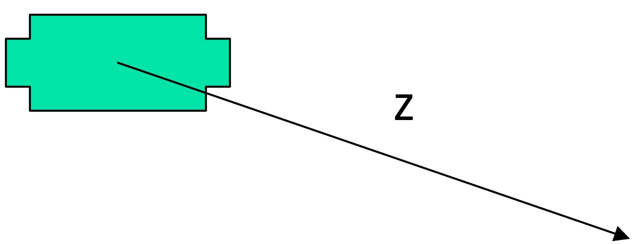
$$(A)_z = \{w \mid w = a + z, a \in A\}$$

Example





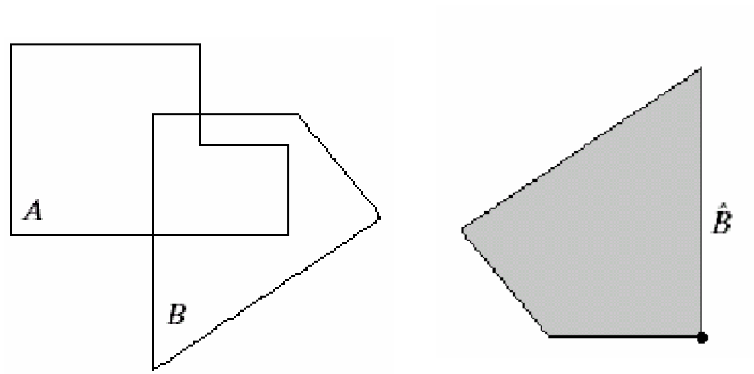
$$z = (z_1, z_2)$$

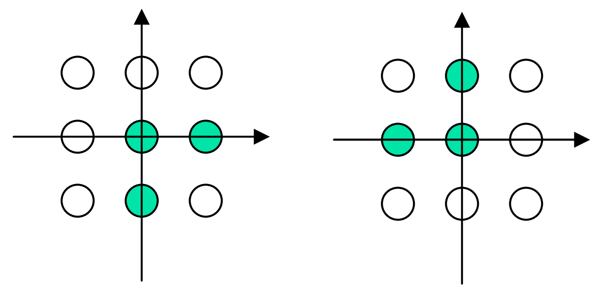


Reflection Operator

$$\hat{B} = \{ w \mid w = -b, b \in B \}$$

Example





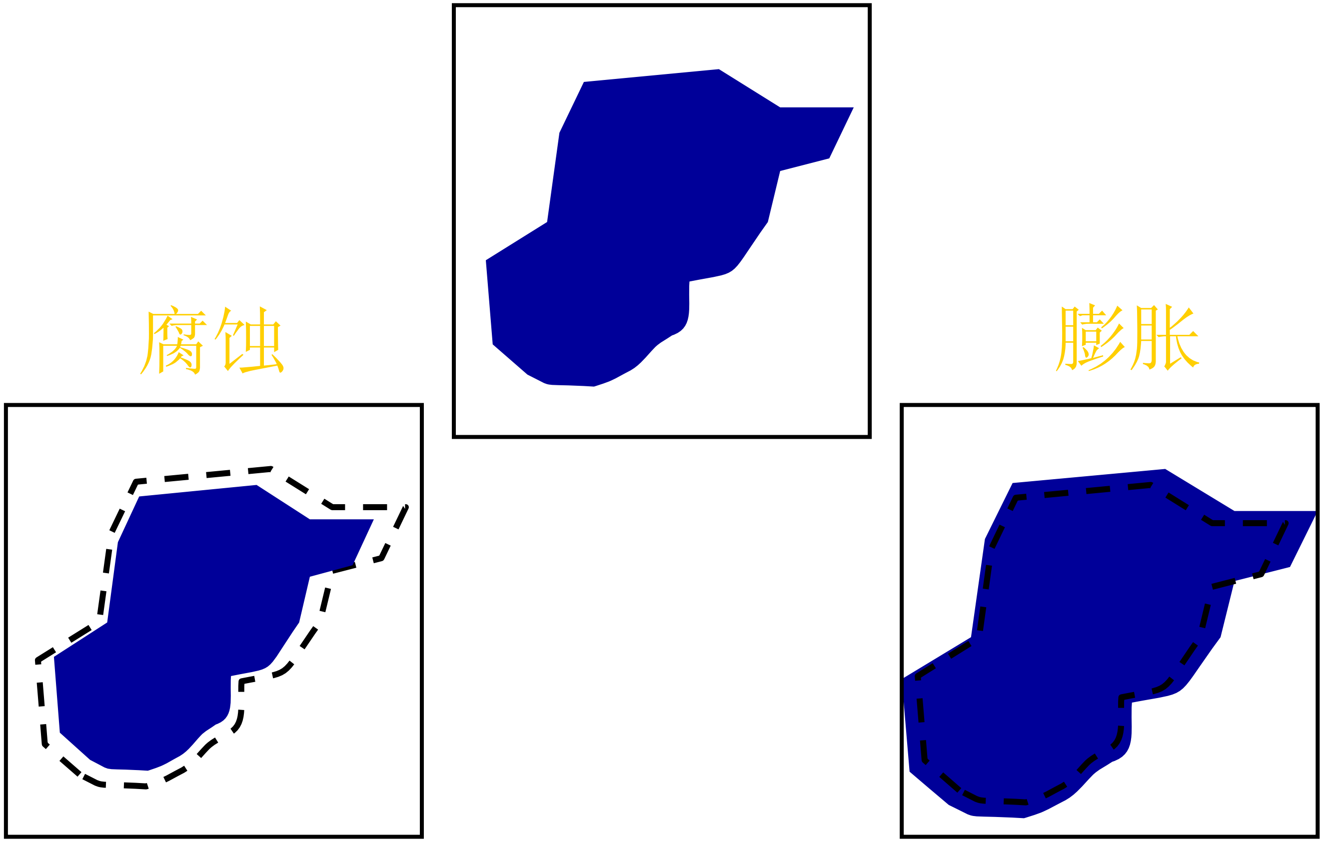
B $\quad\quad$ $\hat{B}$

Binary Image Processing

- Introduction
- Set theory review
- Morphological filtering
  - Erosion and dilation
  - Opening and closing
  - Hit-or-miss, boundary extraction, …
- Skeleton via distance transform

Morphological filtering

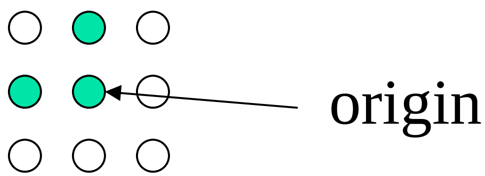
- Erosion and dilation(腐蚀与膨胀)



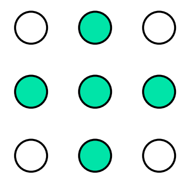
Structuring Element B

Definition: a set of local neighborhood with specified origin

<u>Examples</u>



B_1



B_2

Note: different structuring element leads to different filtering result

Dilation(膨胀)

Definition $Y = X \oplus B = \{z \mid (\hat{B})_z \cap X \neq \phi\}$

or

Example $\boxed{Y = X \oplus B = \bigcup_{b \in B} X_b = \bigcup_{x \in X} B_x = B \oplus X}$

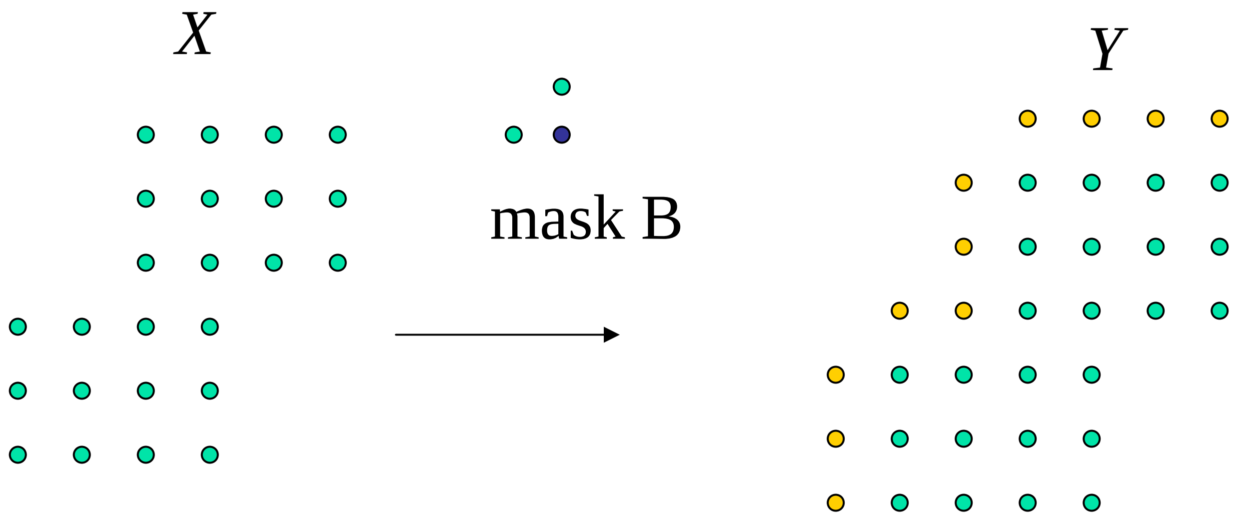
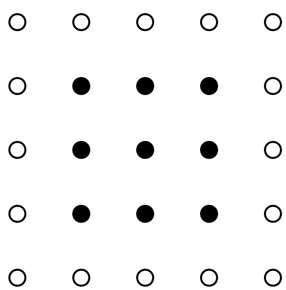
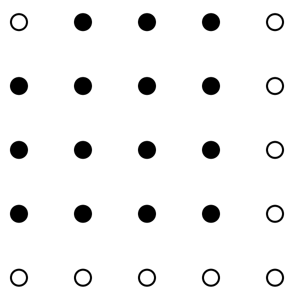


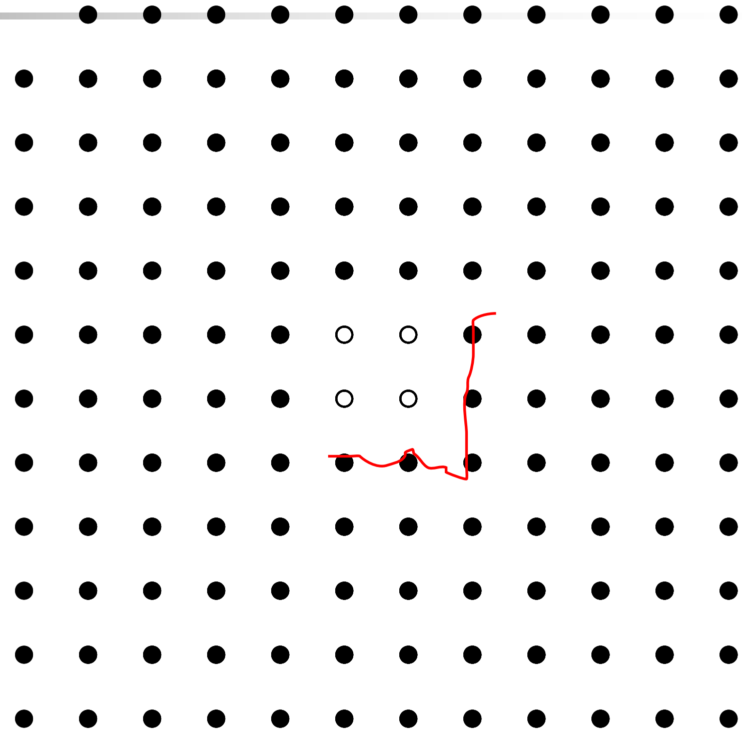
Illustration by Animation



X

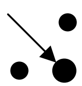


$X \oplus B$



$X^c \oplus B$

origin



B

B={a,b,c}

a=(0,0),b=(0,-1),c=(-1,0)

Erosion(腐蚀)

Definition

$$\boxed{Y = X \ominus B = \{x : B_x \subset X\}}$$

Example

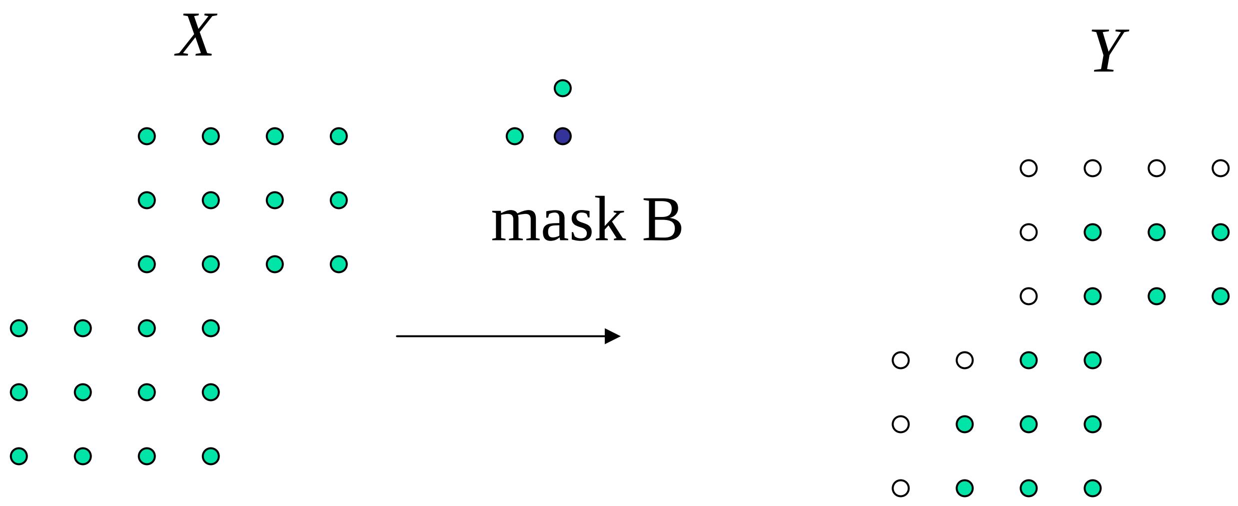
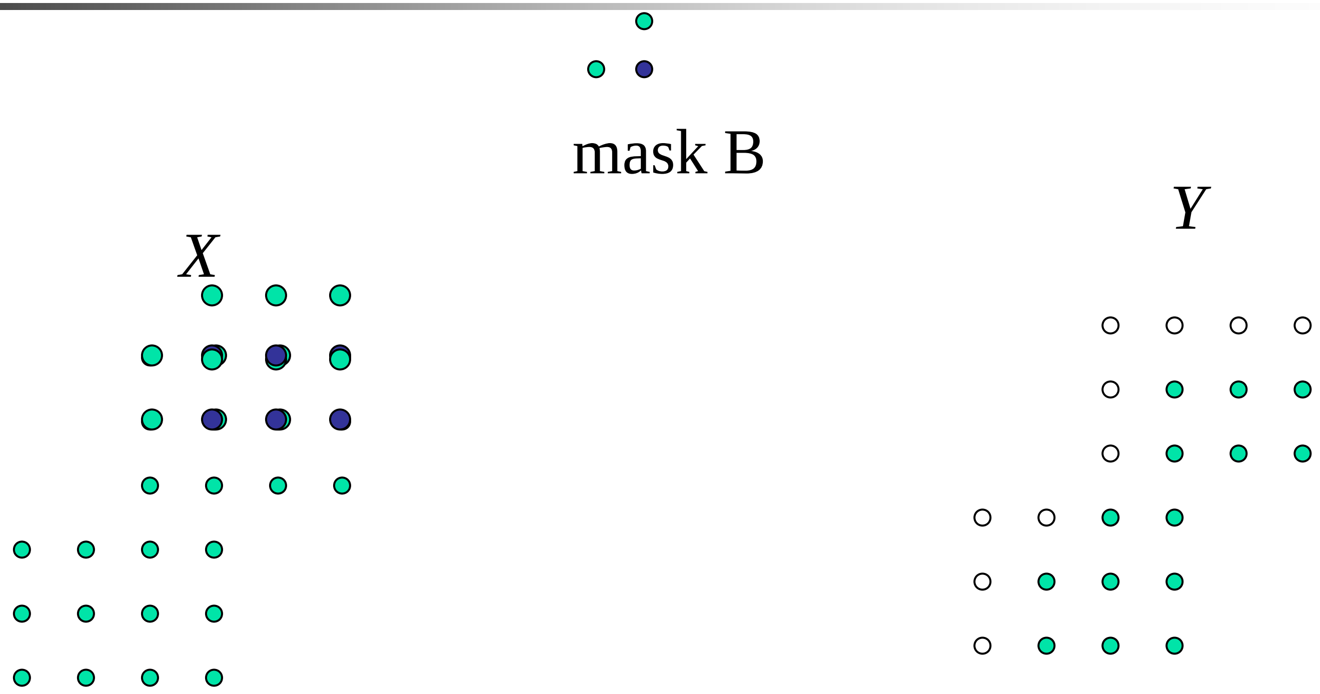


Illustration By Animation



Duality Property*

$$\boxed{(X \ominus B)^c = X^c \oplus \hat{B}}$$

Proof:

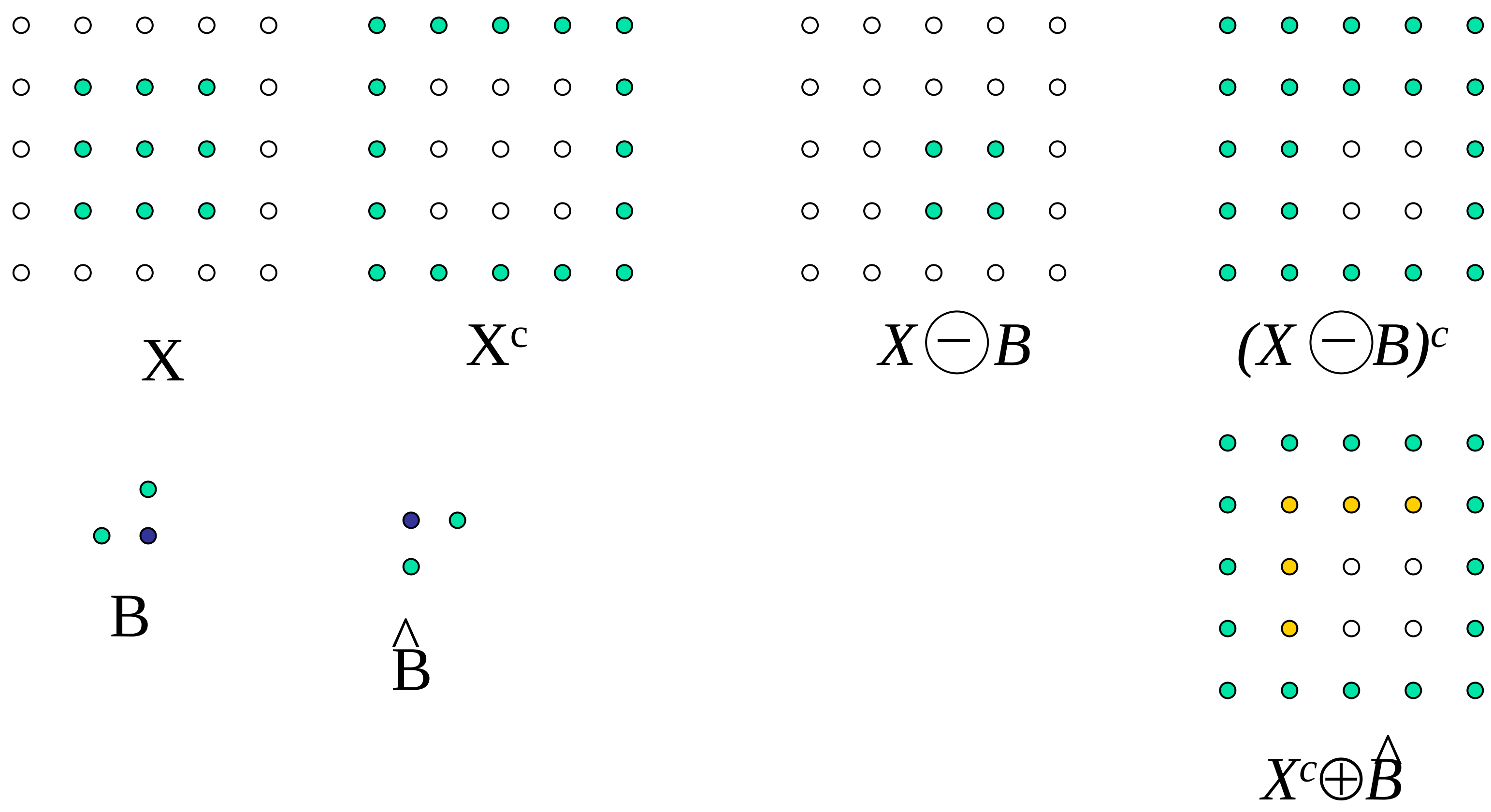
$$(X \ominus B)^c = \{z \mid B_z \subseteq A\}^c$$

$$= \{z \mid B_z \cap A^c = \phi\}^c$$

$$= \{z \mid B_z \cap A^c \neq \phi\}$$

$$= X^c \oplus \hat{B}$$

Example

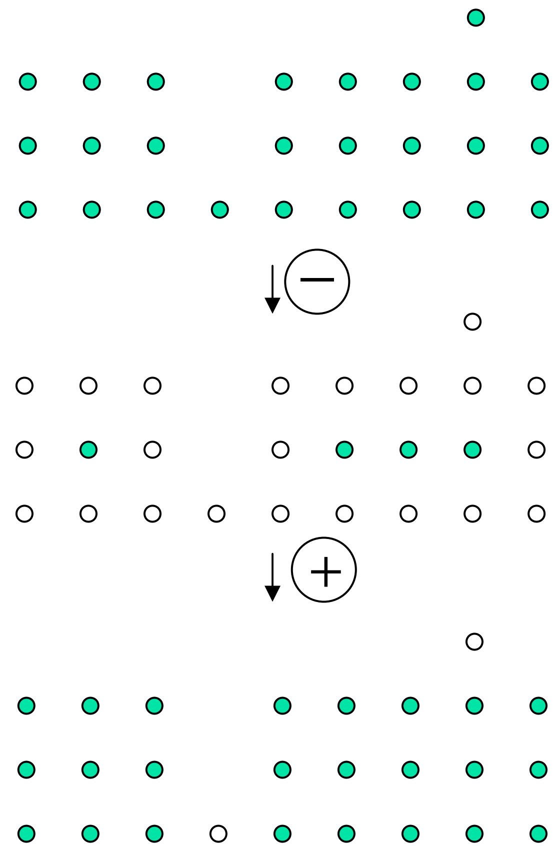


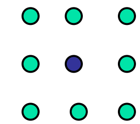
Opening Operator

Definition $X \circ B = (X \ominus B) \oplus B$

<u>Example</u>

X

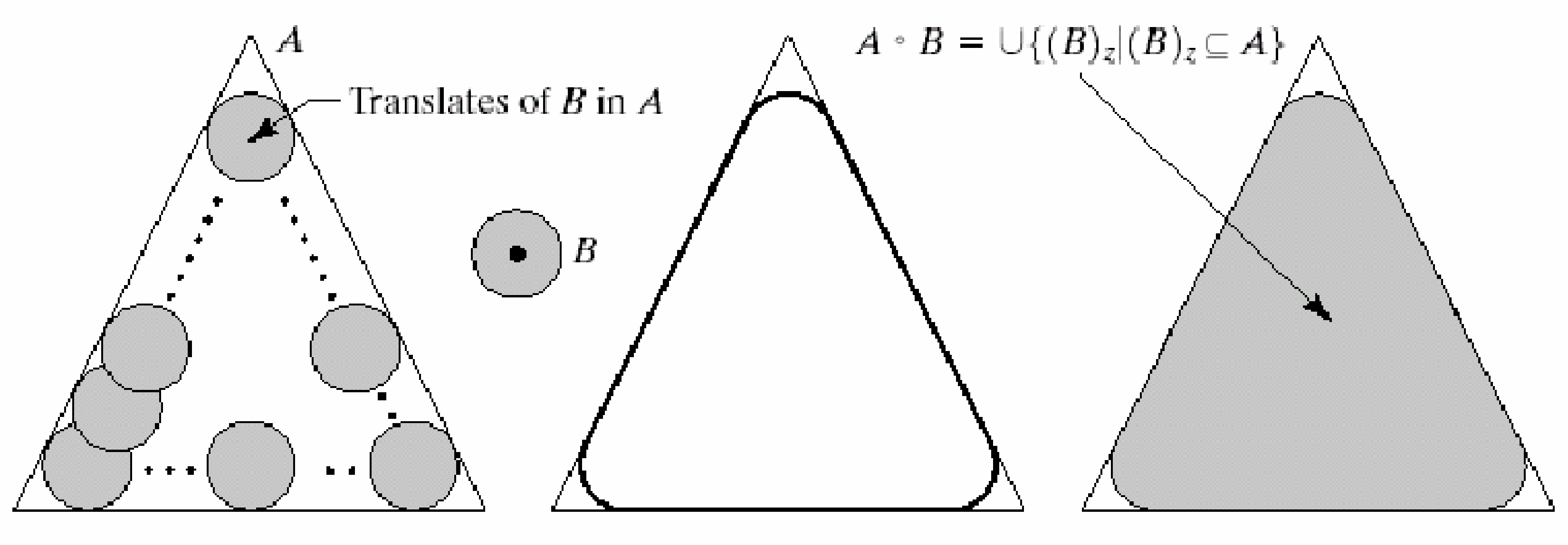




mask B

$X \circ B$

Geometric Interpretation of Opening Operator

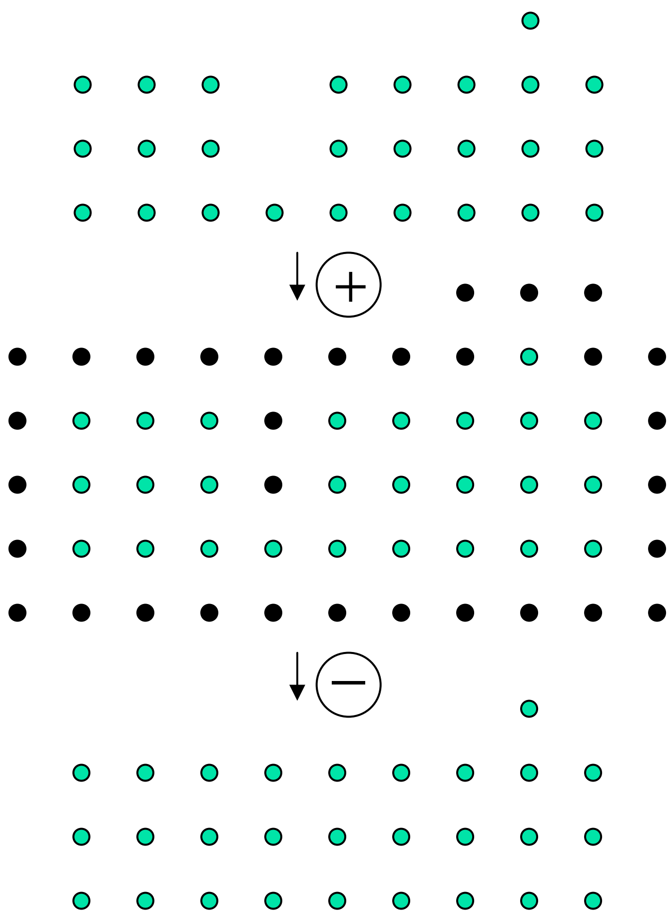


Closing Operator

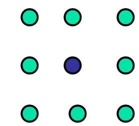
Definition $X \bullet B = (X \oplus B) \ominus B$

Example

X

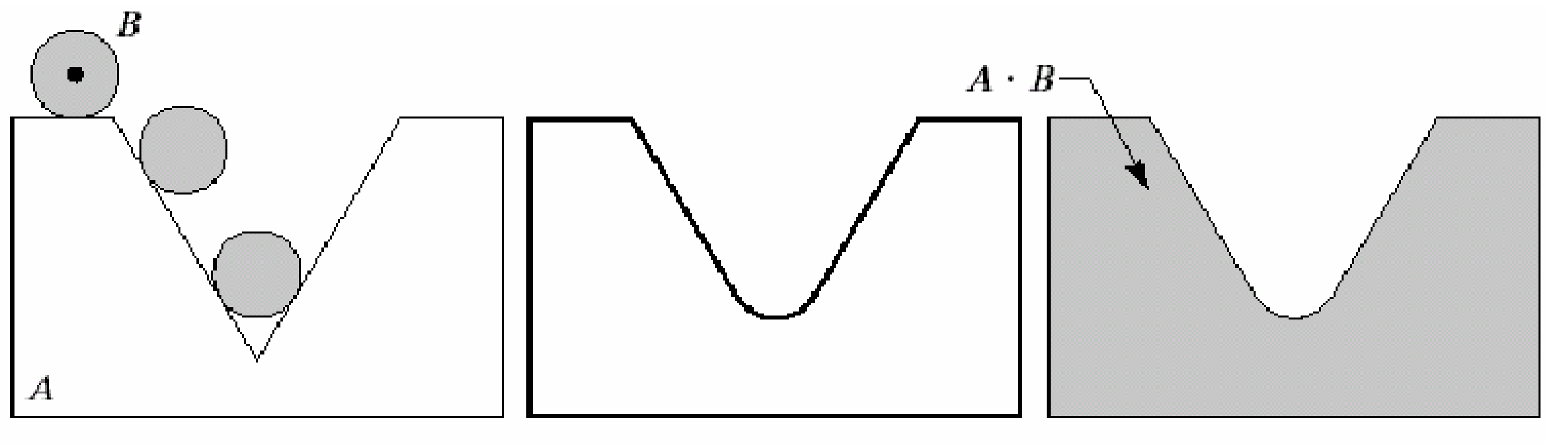


$X \bullet B$



mask B

Geometric Interpretation of Closing Operator



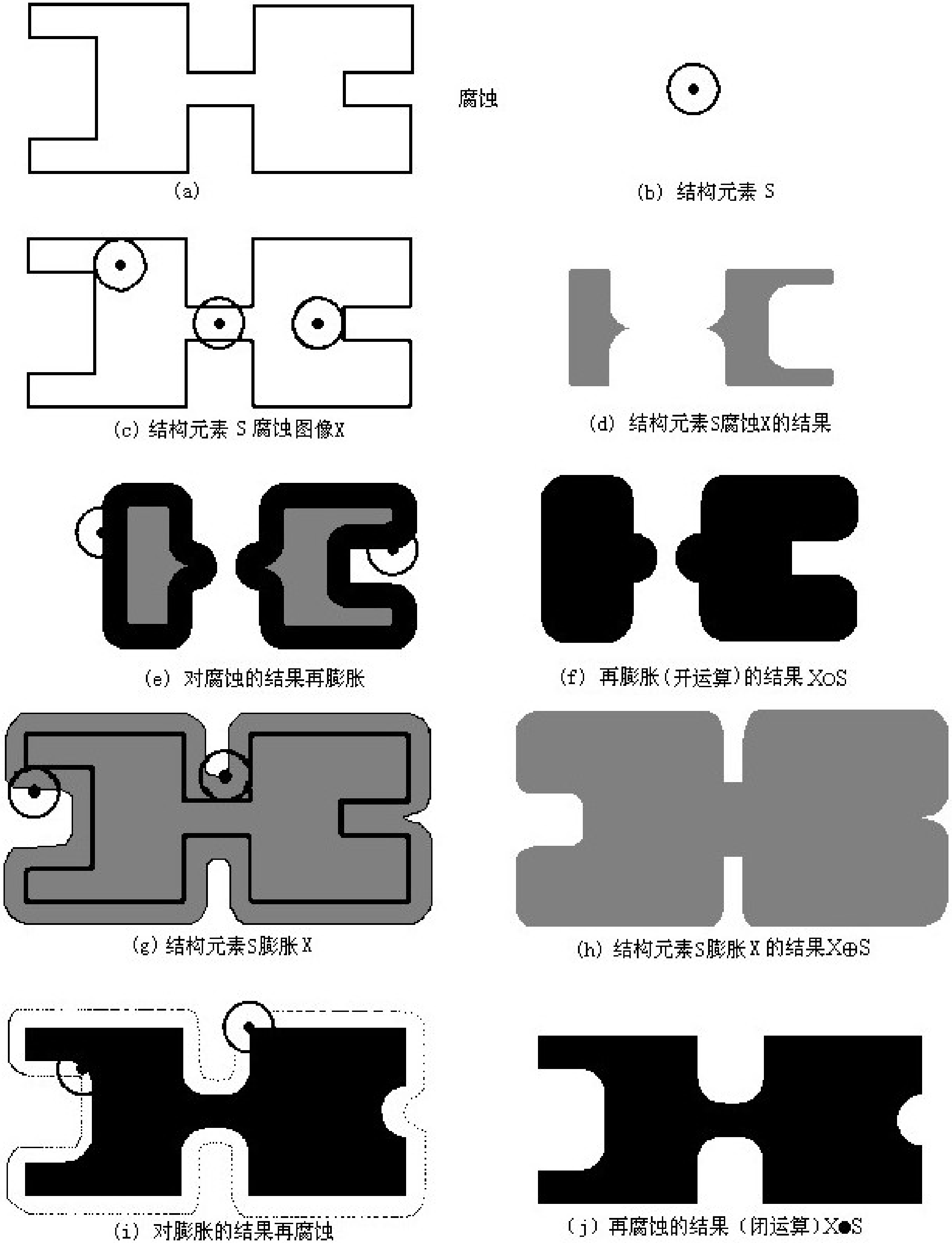


图8-12 开、 闭运算示例

(a) 原图像；

(b) 结构元素S；

(c) 结构元素S腐蚀图像X；

(d) 结构元素S腐蚀X的结果；

(e) 对腐蚀的结构再膨胀；

(f) 再膨胀（开运算）的结果X○S；

(g) 结构元素S膨胀X；

(h) 结构元素S膨胀X的结果X ⊕S；

(i) 对膨胀的结果再腐蚀；

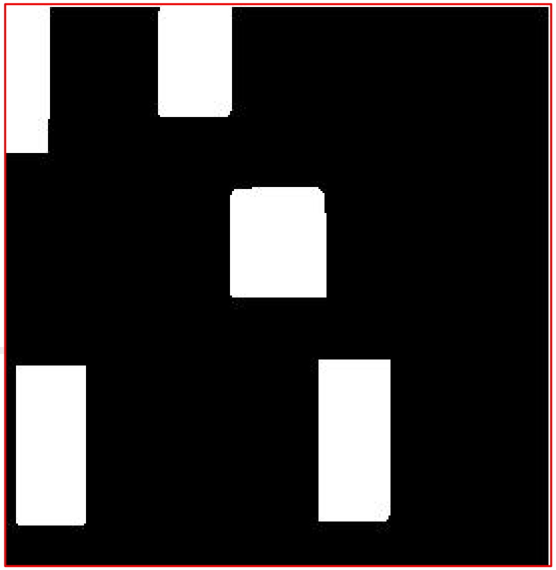
(j)再腐蚀的结果（闭运算）X●S

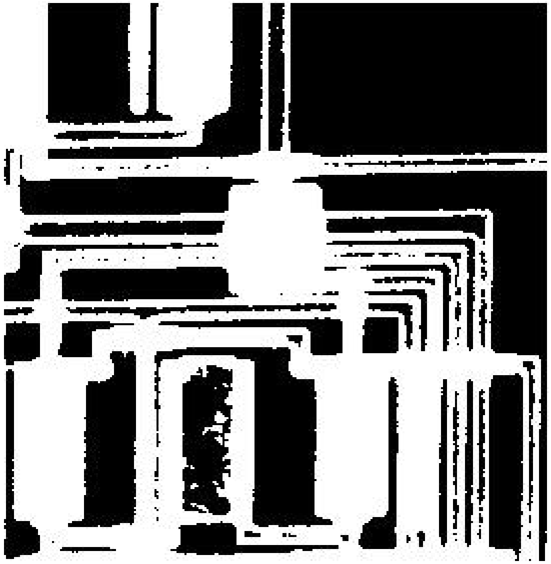
Opening and Closing operations

- 开操作（Opening）：使图像的轮廓变得光滑，去除孤立噪声，断开狭窄的间断和消除细的突出物。
- 闭操作（Closing）：同样使图像的轮廓变得光滑，消除狭窄的间断和长细的鸿沟，消除小的孔洞，并填补轮廓线中的裂痕。

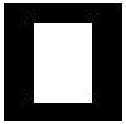
Matlab demo:

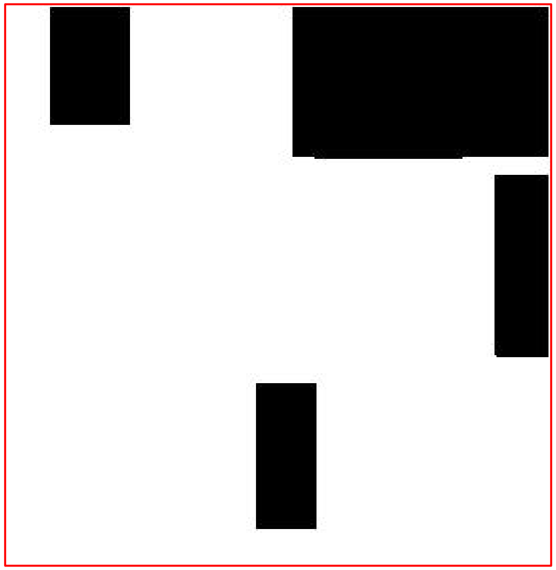
Opening and Closing





```
clear all;close all;
iptsetpref('ImshowBorder','tight');
BW1 = imread('circbw.tif');
figure,imshow(BW1);
mask=ones(40,30);
figure,imshow(mask);
SE = strel('rectangle',[40 30]);
%open operation equals erode_dilate
BW3 = imopen(BW1,SE);
figure,imshow(BW3);
%close operation equals dilate_erode
BW4 = imclose(BW1,SE);
figure,imshow(BW4);
```





A Review of Homework IV

- Try build a skin color model and detect skin regions in the following picture. Submit your code and the result as a binary image.

```
temp=bwareaopen(temp,22); %clear up isolated blocks
```

Properties of Opening and Closing Operators*

Opening

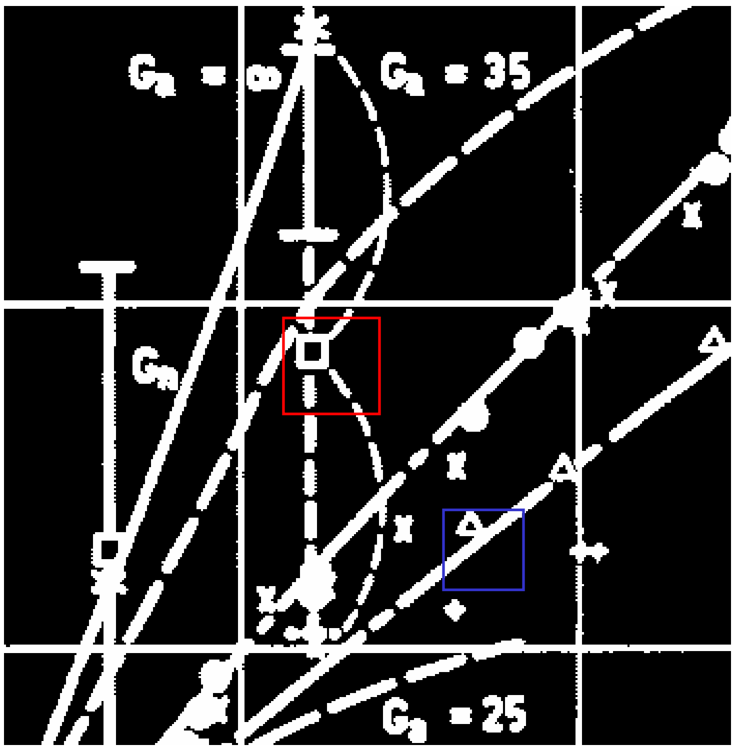
- $X \circ B \subseteq X$
- $X \subseteq Y \Rightarrow X \circ B \subseteq Y \circ B$
- $(X \circ B) \circ B = X \circ B$

Closing

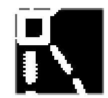
- $X \subseteq X \bullet B$
- $X \subseteq Y \Rightarrow X \bullet B \subseteq Y \bullet B$
- $(X \bullet B) \bullet B = X \bullet B$

$$(X \bullet B)^C = X^C \circ \hat{B} \qquad (X \circ B)^C = X^C \bullet \hat{B}$$

A Little Game of Matching



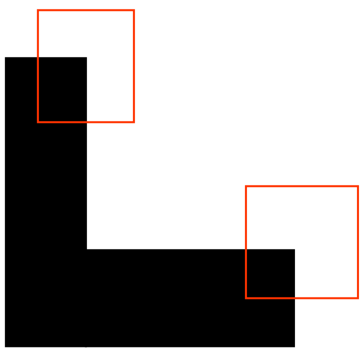
Templates



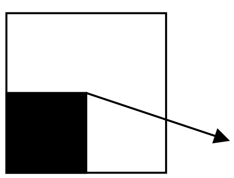
A

B

Illustration by a Simpler Case



X



origin

Template B

How to find the match of A in X using a computer?

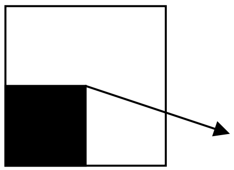
Hit: the southwest quadrant must be black in X

Hit: the other three quadrants must be white in X

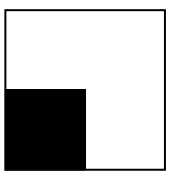
Hit: the southwest quadrant must be black in X

Hit: the other three quadrants must be black in X^c

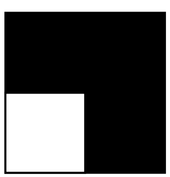
Matching via Hit-or-Miss


origin

Template B



Template B_1



Template B_2

Hit: the southwest quadrant must be black in X

$$X_1 = X \ominus B_1$$

Hit: the other three quadrants must be black in X^c

$$X_2 = X^c \ominus B_2$$

To satisfy both conditions, we need to take the Intersection of X_1 and X_2

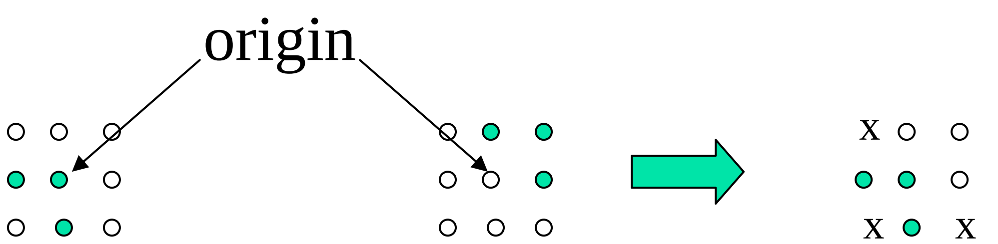
Hit-or-Miss Operator

Definition $X \circledast B = \boxed{(X \ominus B_1) \cap (X^c \ominus B_2)}$

Hit Miss

Why $\ominus$?
Why complement?
Why intersection?

<u>Structuring element example</u>



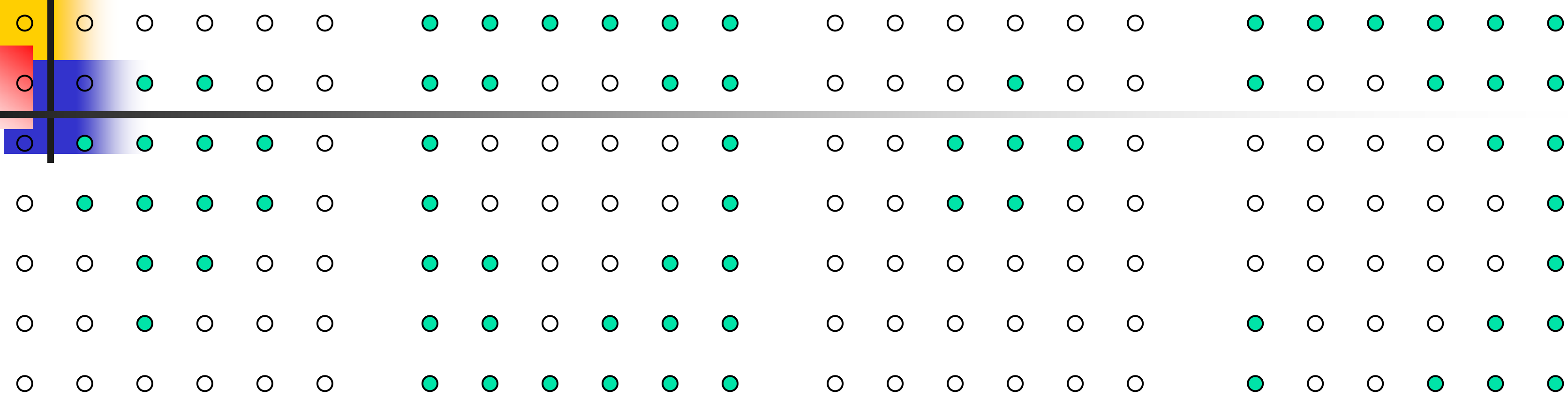
mask B_1 mask B_2 mask B

$$\begin{matrix} 0 & -1 & -1 \\ 1 & 1 & -1 \\ 0 & 1 & 0 \end{matrix}$$

MATLAB

(MATLAB function: bwhitmiss)

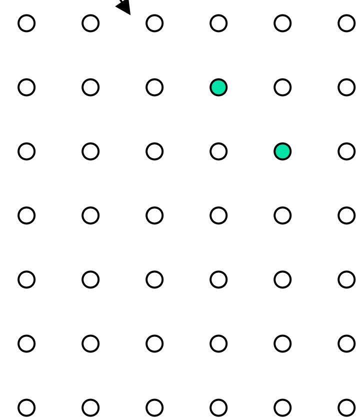
Example 1



X $\qquad$ X^c $\qquad$ $X \ominus B_1$ $\qquad$ $X^c \ominus B_2$

origin

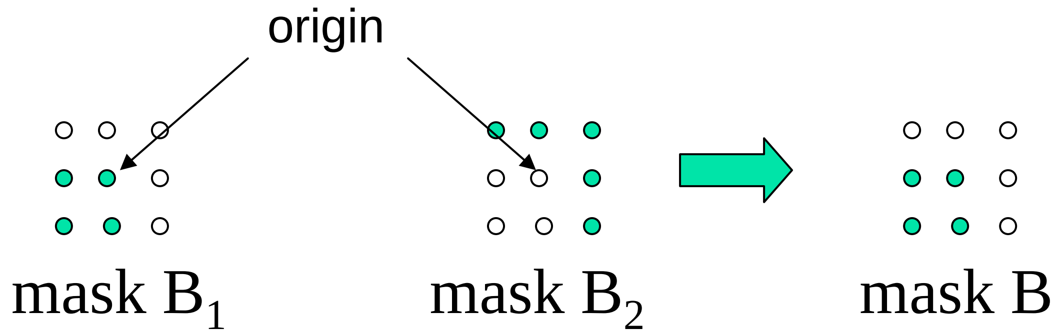
mask B_1 $\qquad$ mask B_2

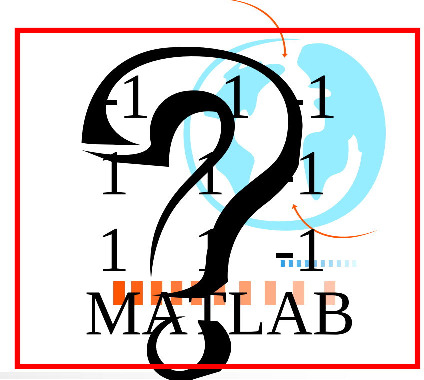


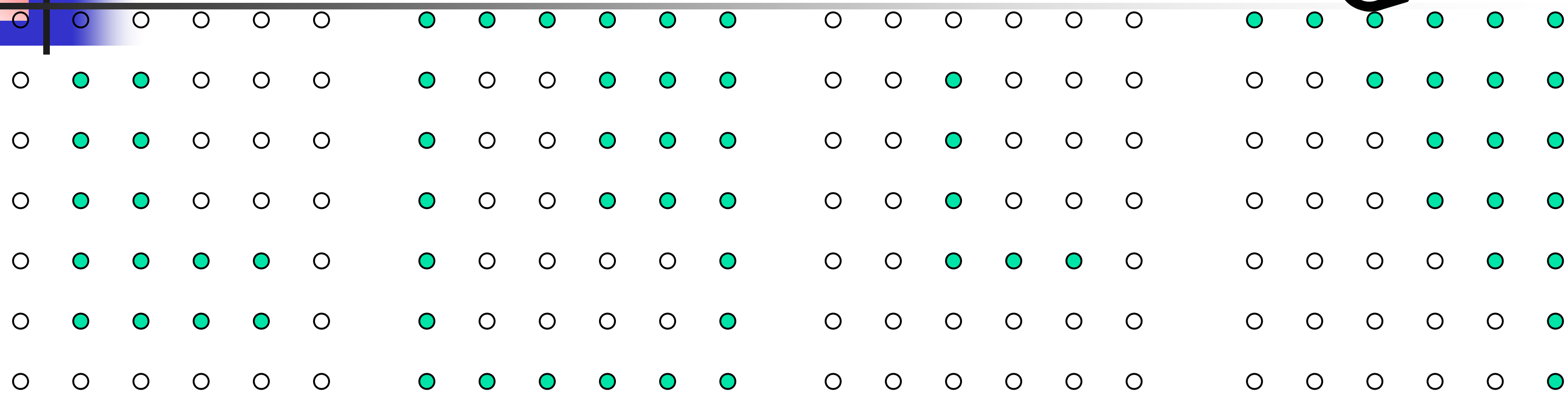
$X \circledast B$

$$X \circledast B = (X \ominus B_1) \cap (X^c \ominus B_2)$$

Example 2

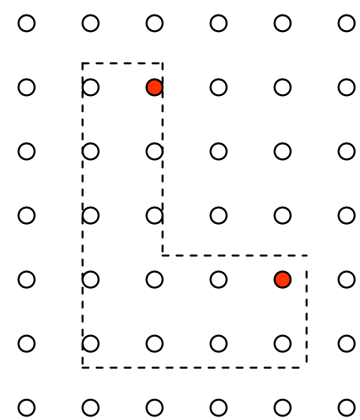






X X^c $X \ominus B_1$ $X^c \ominus B_2$

Now, take the intersection
and note that only two points
Remain (highlighted by red)



$X \circledast B$

Roadmap of Morphological Filtering

basic set operators (complement, union, intersection, difference)
dilation/erosion operators

↓

closing/opening, Hit-or-Miss operators

↓

morphological filters/algorithms
(boundary, region filling, thinning)

1. Boundary Extraction

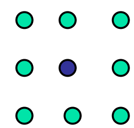
Definition

$$\partial X = X - (X \ominus B)$$

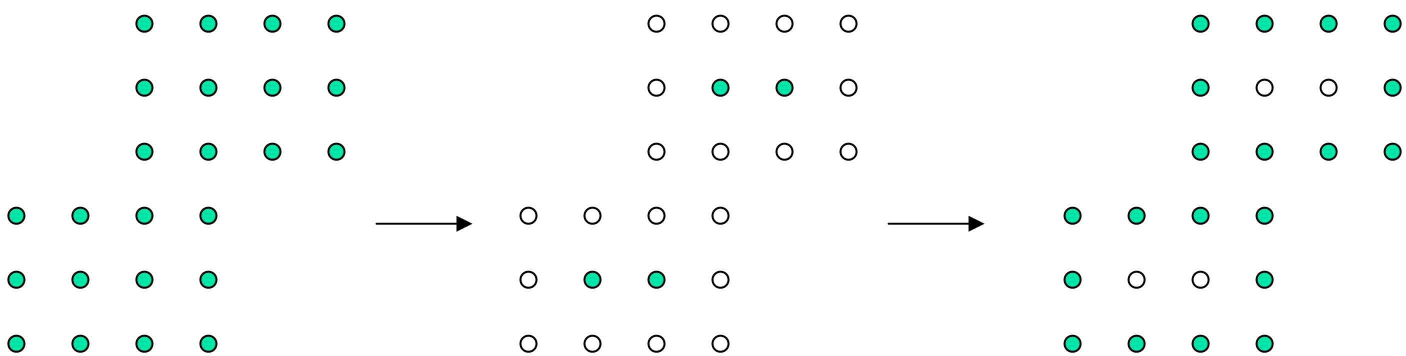
How about

$$\partial X = (X \oplus B) - X?$$

Example



mask B



X $\qquad$ $X \ominus B$ $\qquad$ $X-(X \ominus B)$

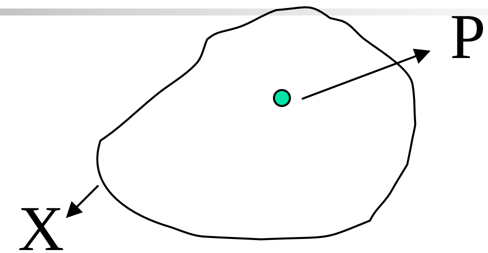
MATLAB function: bwmorph(…,'remove')

Image Example

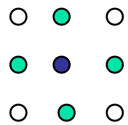
X

∂X

2. Region Filling



Idea: recursively expand the region around the seed P
but stop the expansion at the boundary X



mask B

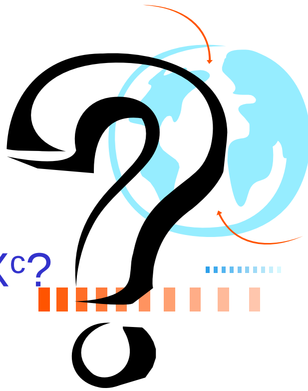
Iterations:

expansion stop at the boundary

$Y_0 = P$

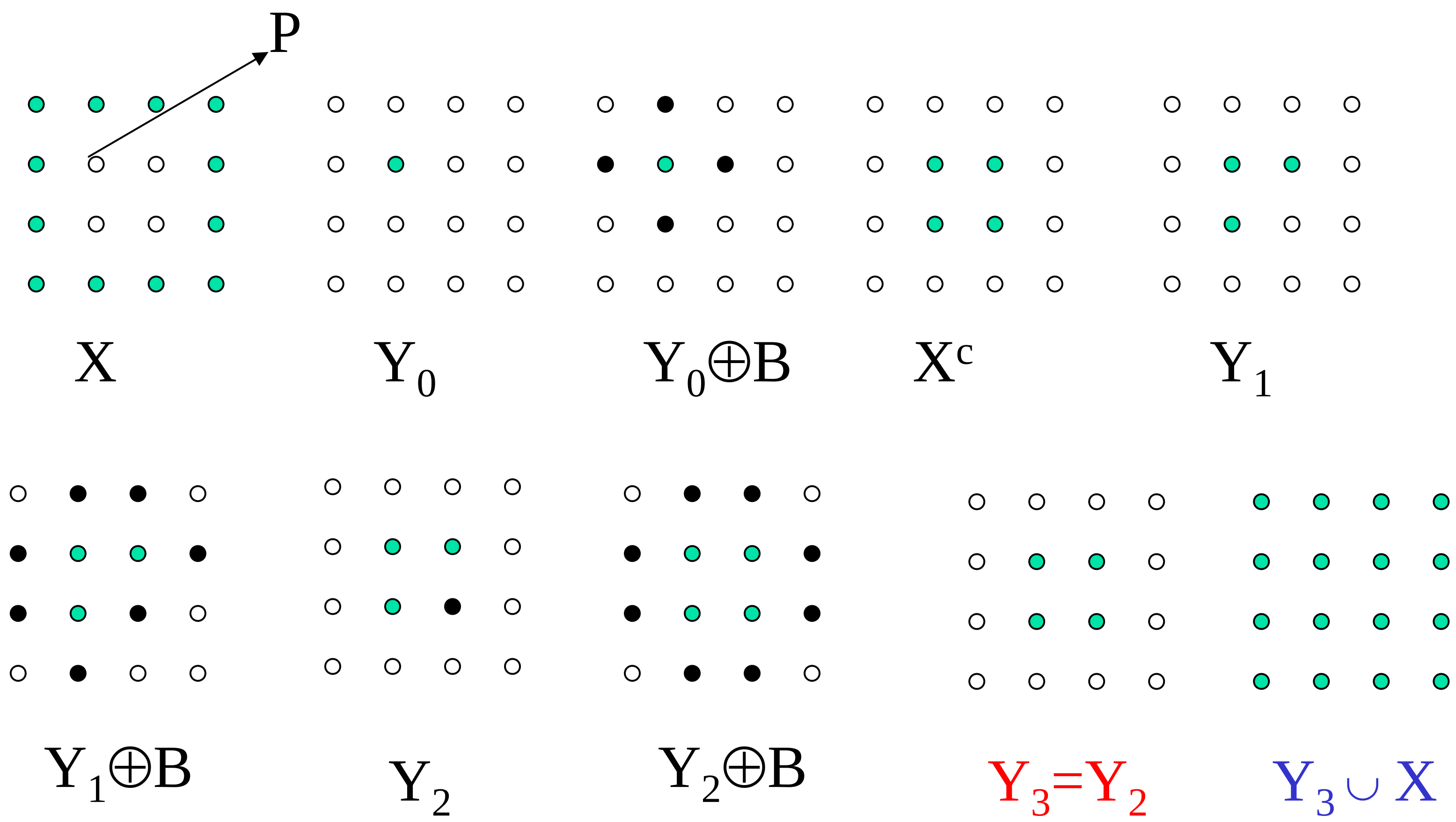
$$\boxed{Y_k = (Y_{k-1} \oplus B) \cap X^c,\ k = 1, 2, 3\ldots}$$

Why dilation?
Why intersection with X^c?

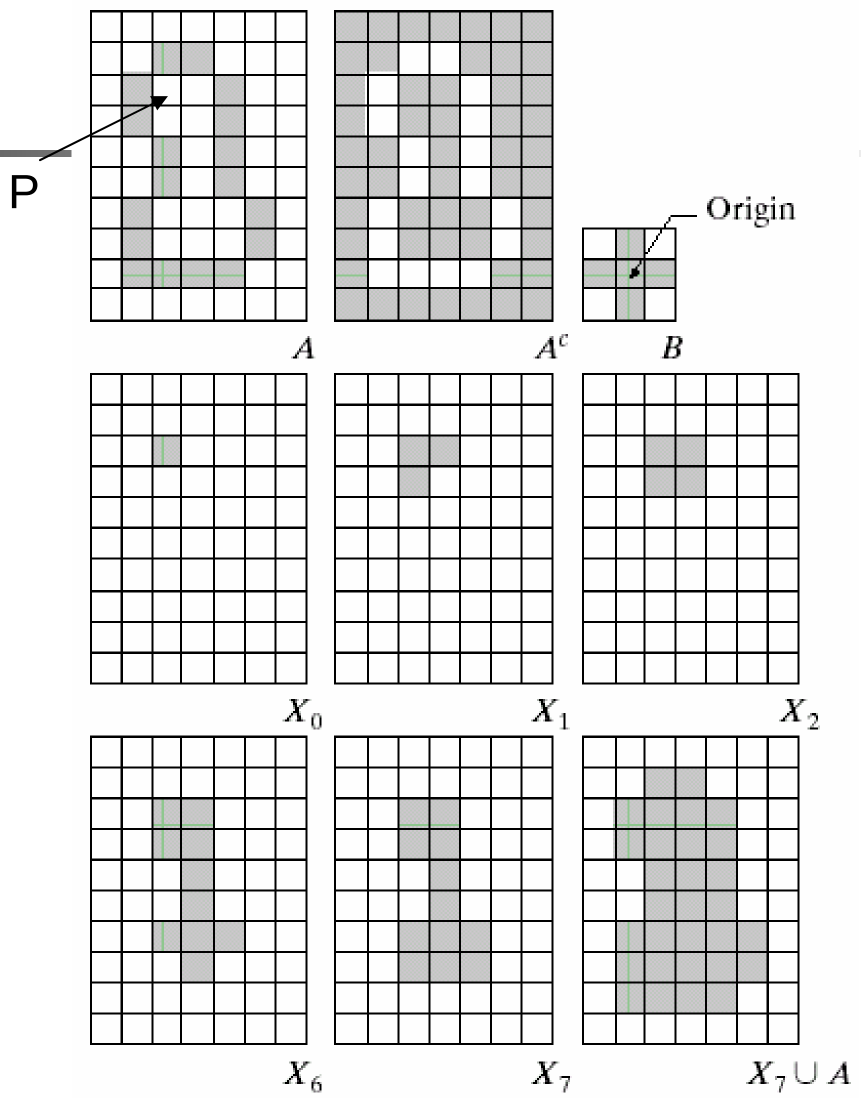


Terminate when $Y_k = Y_{k-1}$, output $Y_k \cup X$

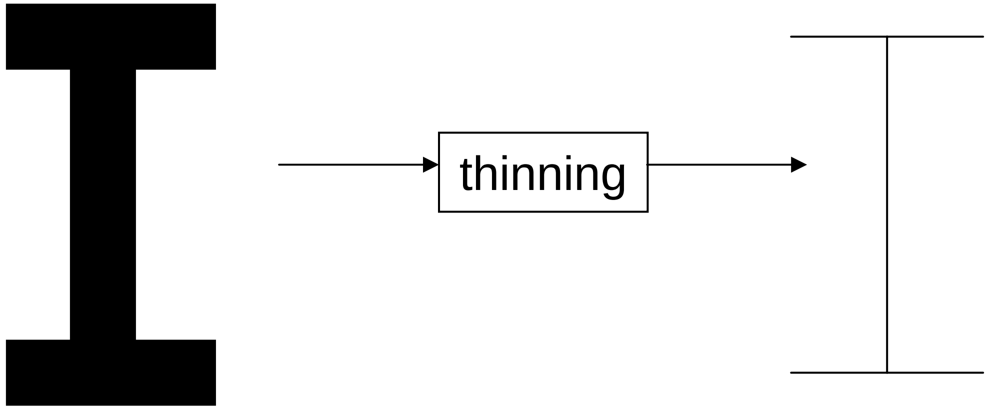
Image Example



Additional Example



3. Thinning

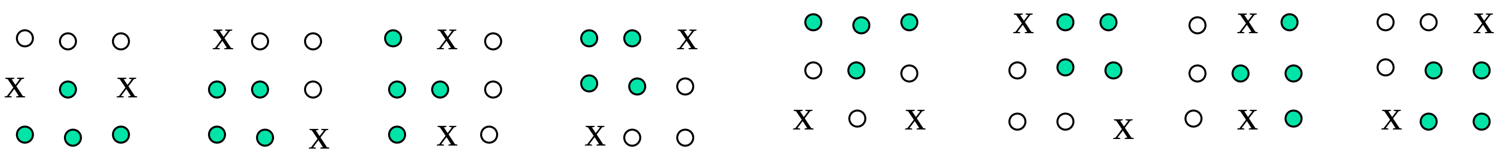


Intuitively, thinning finds the skeleton of a binary image (you will learn a different way of finding skeleton by distance transform later)

Thinning Algorithm*

Basic idea:

- Use Hit-or-Miss operator as a sifter
- Use multiple masks to characterize different patterns



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

$$X_0 = X$$

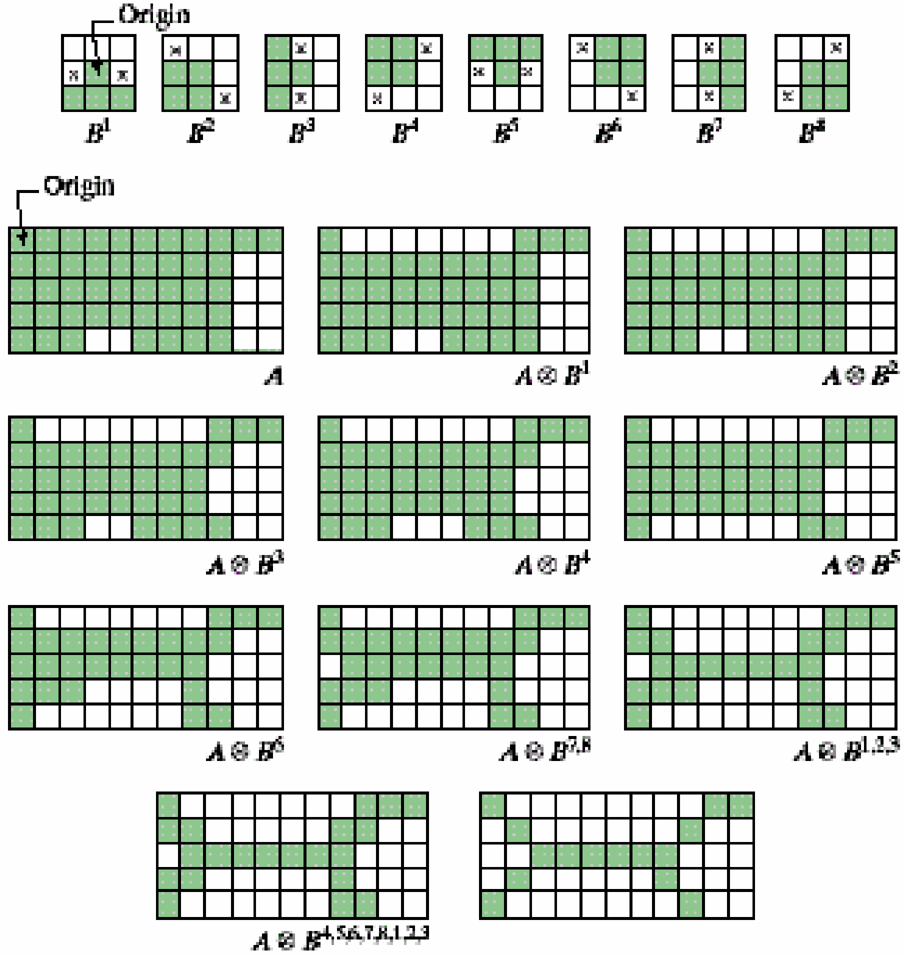
$$\boxed{X_k = (\ldots(\,(X_{k-1} \otimes B^1) \otimes B^2 \,\ldots \otimes B^8)}$$

where $X \otimes B = X - X \circledast B$

Stop the iteration when $X_k = X_{k-1}$

Why eight different B's?
Why hit-or-miss?

Image Example



Binary Image Processing

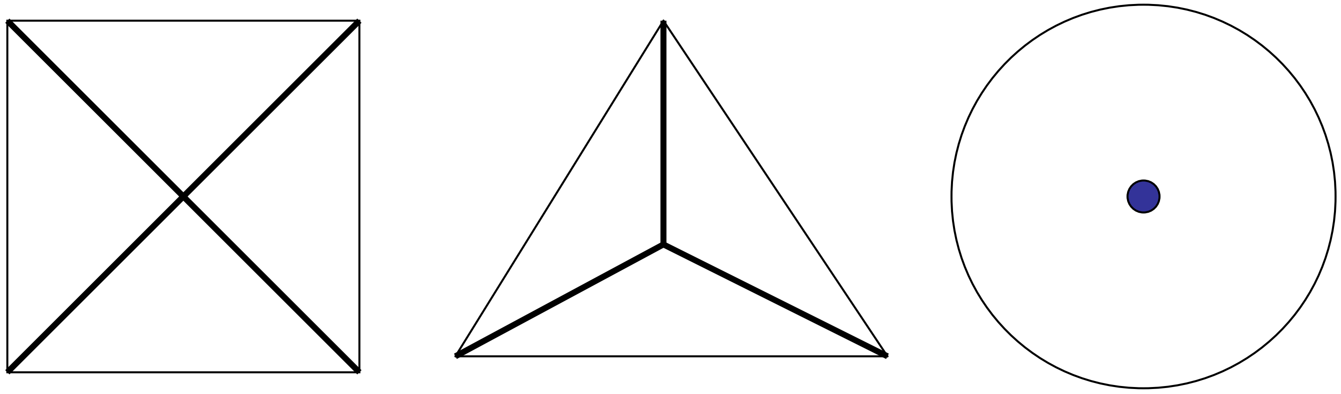
- Introduction
- Set theory review
- Morphological filtering
  - Erosion and dilation
  - Opening and closing
  - Hit-or-miss, boundary extraction, …
- Skeleton via morphological filtering

Medial Axis (Skeleton)

- Definition

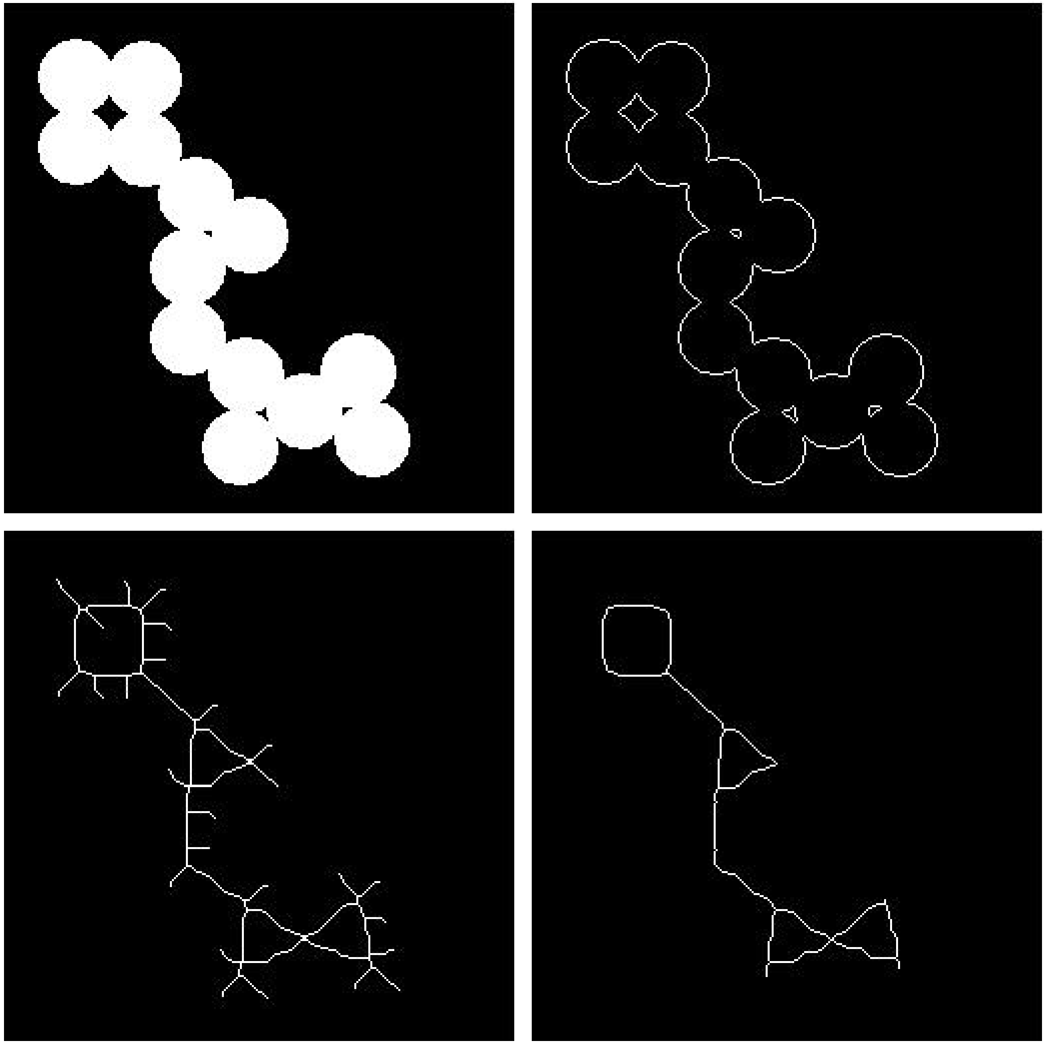
  Suppose that a fire line propagates with constant speed from the contour of a connected object towards its inside, then all those points lying in positions where at least two wave fronts of the fire line meet during the propagation will constitute a form of a **skeleton**

- Examples



Matlab demos:

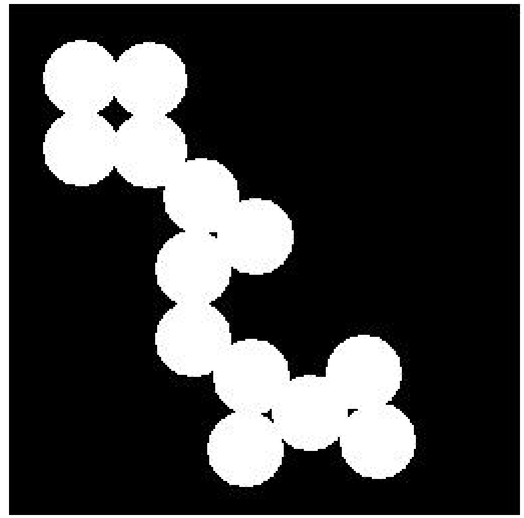
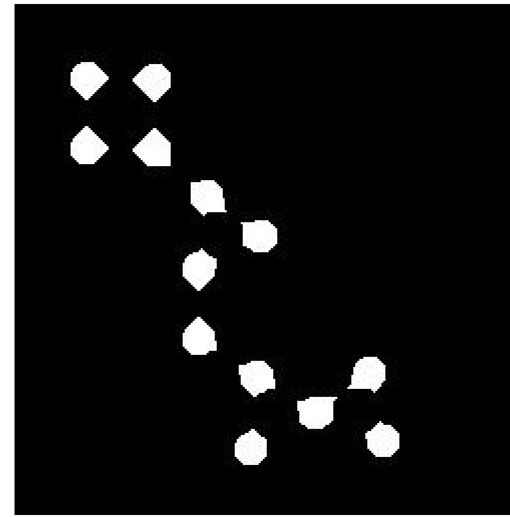
```
clear all;close all;
iptsetpref('ImshowBorder','tight');
BW = imread('circles.png');
figure,imshow(BW);
BW2 = bwmorph(BW,'remove');
figure, imshow(BW2);
BW3 = bwmorph(BW,'skel',Inf);
figure, imshow(BW3);
BW4 = bwmorph(BW,'thin',Inf);
figure, imshow(BW4);
```



Grayscale image demos

```
clear all;close all;
iptsetpref('ImshowBorder','tight');
originalBW = imread('circles.png');
figure,imshow(originalBW);
se = strel('disk',11);
erodedBW = imerode(originalBW,se);
figure, imshow(erodedBW);

I = imread('cameraman.tif');
figure,imshow(I);
se = strel('ball',5,5);
I2 = imerode(I,se);
figure, imshow(I2);
```

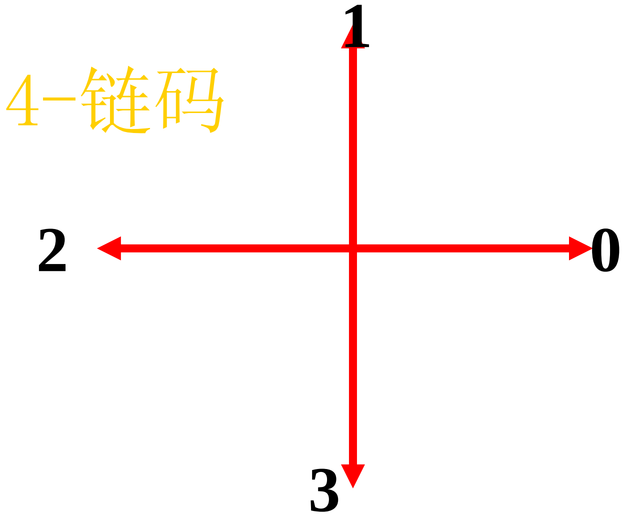



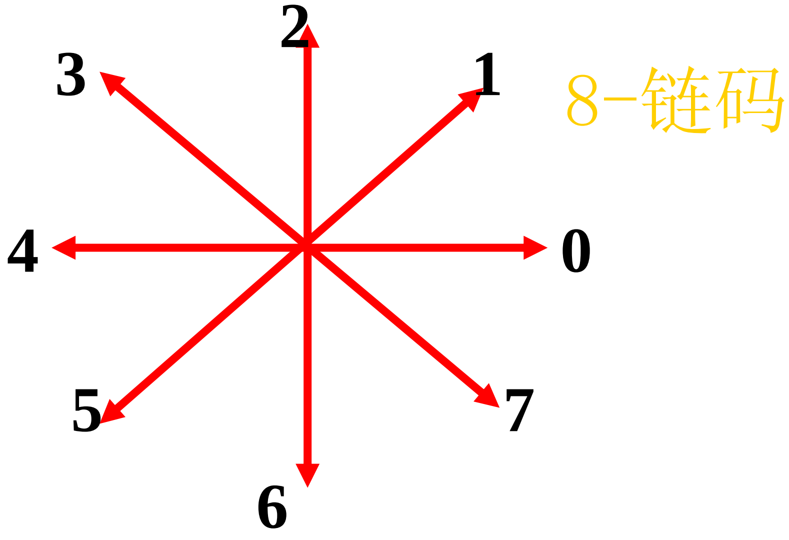
Digital Image Processing

Representation and Description

表示与描述:表示法设计

- 链码
  - 定义：1）链码是一种<u>边界的编码表示法</u>。
    2）<u>用边界的方向作为编码依据</u>。为简化边界的描述。<u>一般描述的是边界点集</u>。



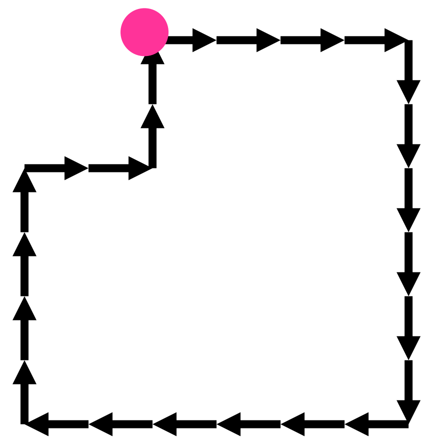


表示与描述:表示法设计

- 链码
  - 算法：
    - 给每一个线段一个方向编码。
    - 有4-链码和8-链码两种编码方法。
    - 从起点开始，沿边界编码，至起点被重新碰到，结束一个对象的编码。

表示与描述:表示法设计

- 链码举例：



4-链码：0000333333222222111110011

表示与描述:表示法设计

- 链码
  - 问题1：

    1）链码相当长。

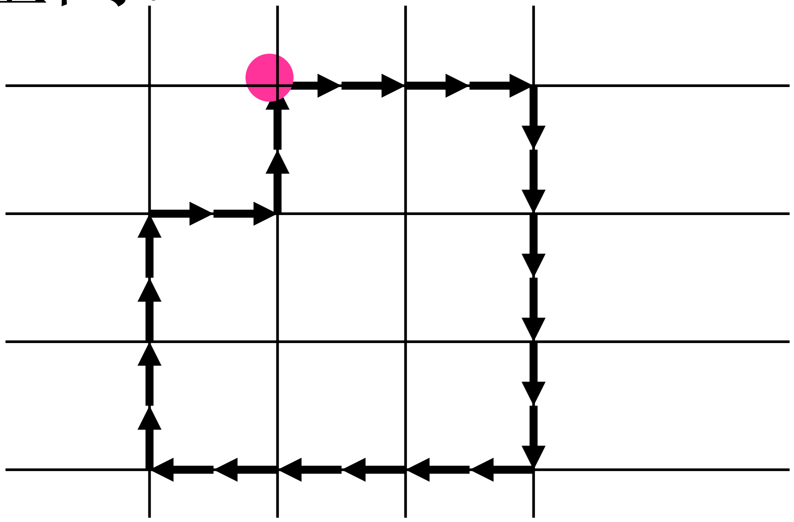
    2）噪音会产生不必要的链码。
  - 改进1：

    1）加大网格空间。

    2）依据原始边界与结果的接近程度，来确定新点的位置。

表示与描述:表示法设计

- 加大网格空间：



4-链码：003332221101

表示与描述:表示法设计

- 链码
  - 问题2：

    1）由于起点的不同，造成编码的不同

    2）由于角度的不同，造成编码的不同
  - 改进2：

    1）从固定位置作为起点(最左最上)开始编码

    2）通过使用链码的首差代替码子本身的方式

表示与描述:表示法设计

- 链码
  - 循环首差链码：用相邻链码的差代替链码

例如：4-链码 10103322 循环首差为：33133030

循环首差：1 - 2 = -1(3)　　3 - 0 = 3

0 - 1 = -1(3)　　3 - 3 = 0

1 - 0 = 1　　2 - 3 = -1(3)

0 - 1 = -1(3)　　2 - 2 = 0

表示与描述:表示法设计

- 链码
  - 应用背景:
    - 如果边界的本身对于旋转和比例修改来说是无变化的，使用链码才是正确的。一般来说这是不可能的，实际应用时还需要改进。
    - 用链码后，对象只要用1)起点坐标，2)周长（边界点数）3)链码，4)对象编号，就可以描述。

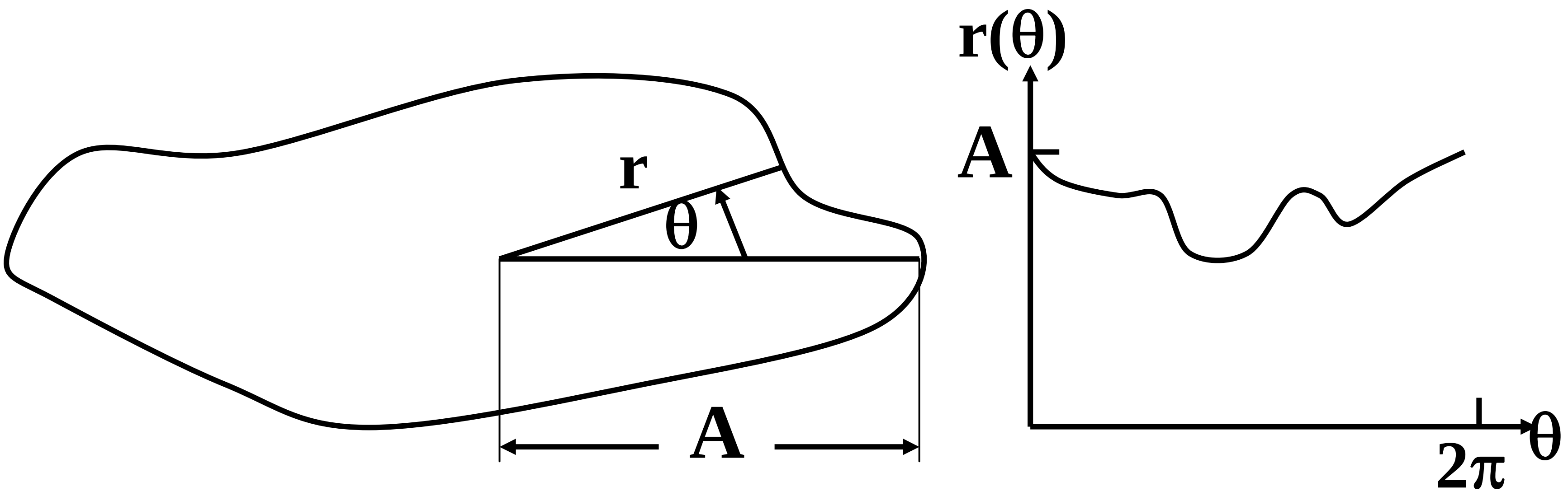
表示与描述:表示法设计

- 外形特征
  - 基本思想:

    外形特征是一种用一维函数表达边界的方法。基本思想是把边界的表示降到一维函数

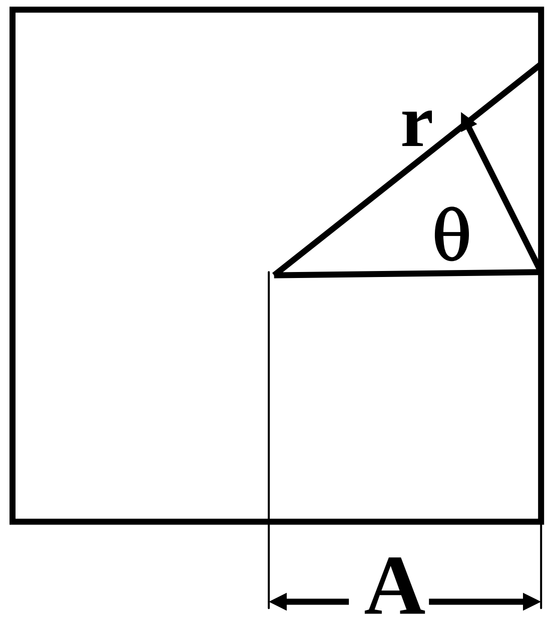
表示与描述:表示法设计

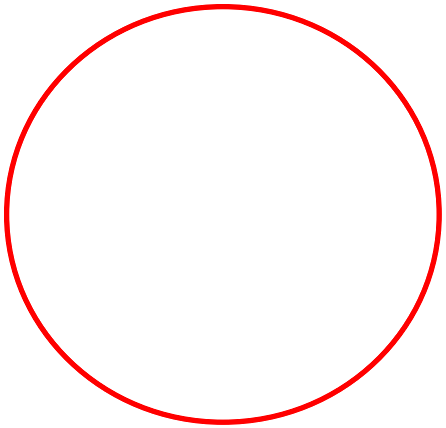
- 外形特征
  - 函数定义——质心角函数：边上的点到质心的距离r，作为夹角的θ的函数。

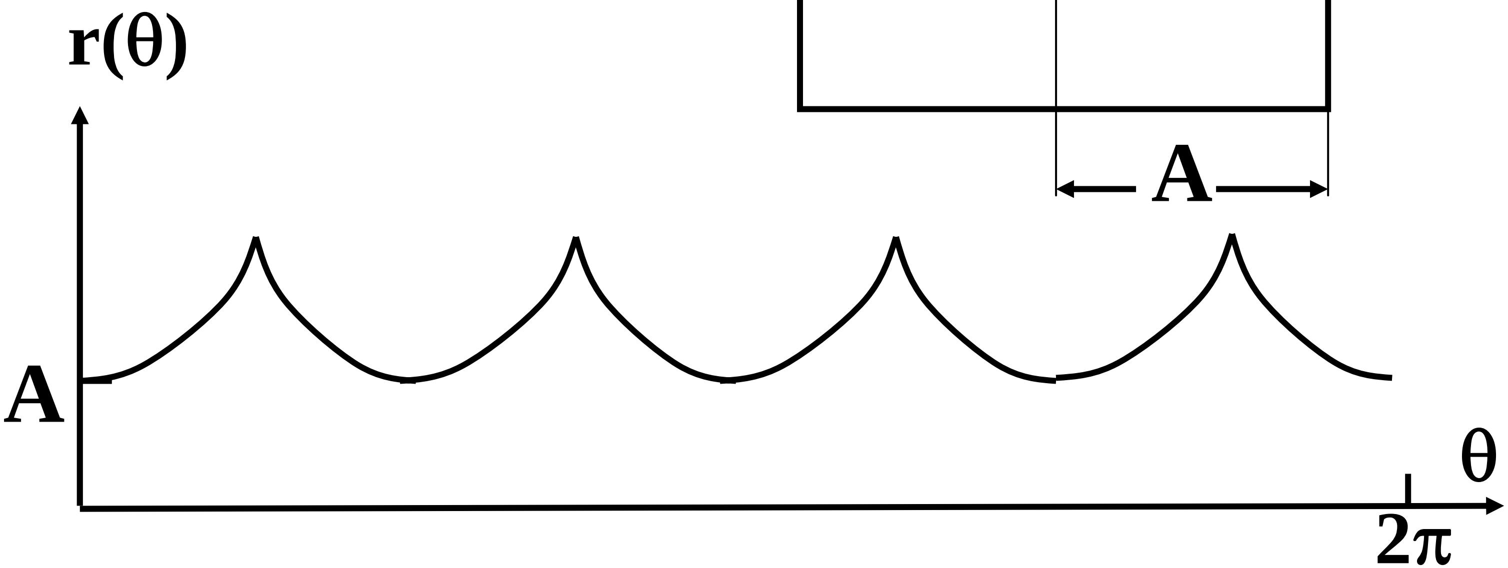


表示与描述:表示法设计

- 外形特征
  - 举例:







表示与描述:表示法设计

- 外形特征
  - 问题：函数过分依赖于旋转和比例的变化
  - 改进：
    - 对于旋转——两种改进：

      a.选择离质心最远的点作为起点

      b.选择从质心到主轴最远的点作为起点

    - 对于比例变换：

      对函数进行正则化，使函数值总是分布在相同的值域里，比如说[0，1]

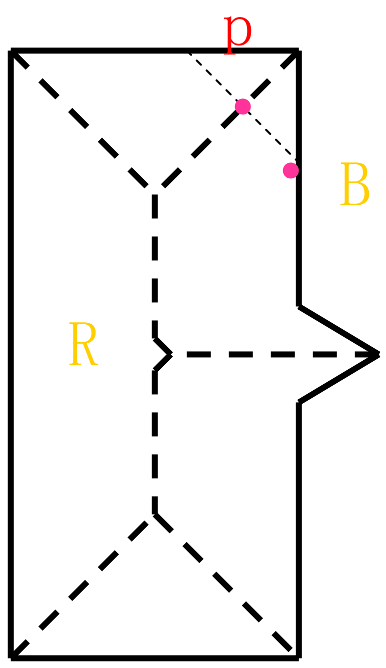
表示与描述:表示法设计

- 区域骨架
  - 基本思想
    - 表示一个平面区域结构形状的重要方法是把它削减成图形。这种削减可以通过细化（也称为抽骨架）算法，获取区域的骨架来实现
    - Blum的中轴变换方法（MAT）

    设:R是一个区域，B为R的边界点，对于R中的点p，找p在B上“最近”的邻居。如果p有多于一个的邻居，称它属于R的中轴（骨架）

表示与描述:表示法设计

- 区域骨架
  - 基本思想
  - 问题：计算量大



表示与描述:表示法设计

- 区域骨架
  - 算法改进思想
    - 在保证产生正确的骨架的同时，改进算法的效率。比较典型的是一类细化算法，它们不断删去边缘，但保证删除满足：

（1）不移去端点

（2）不破坏连通性

（3）不引起区域的过度腐蚀

数学形态学图像处理

- 细化
  - 思想：在不破坏连通性的前提下，细化图像。
  - 算法实现：

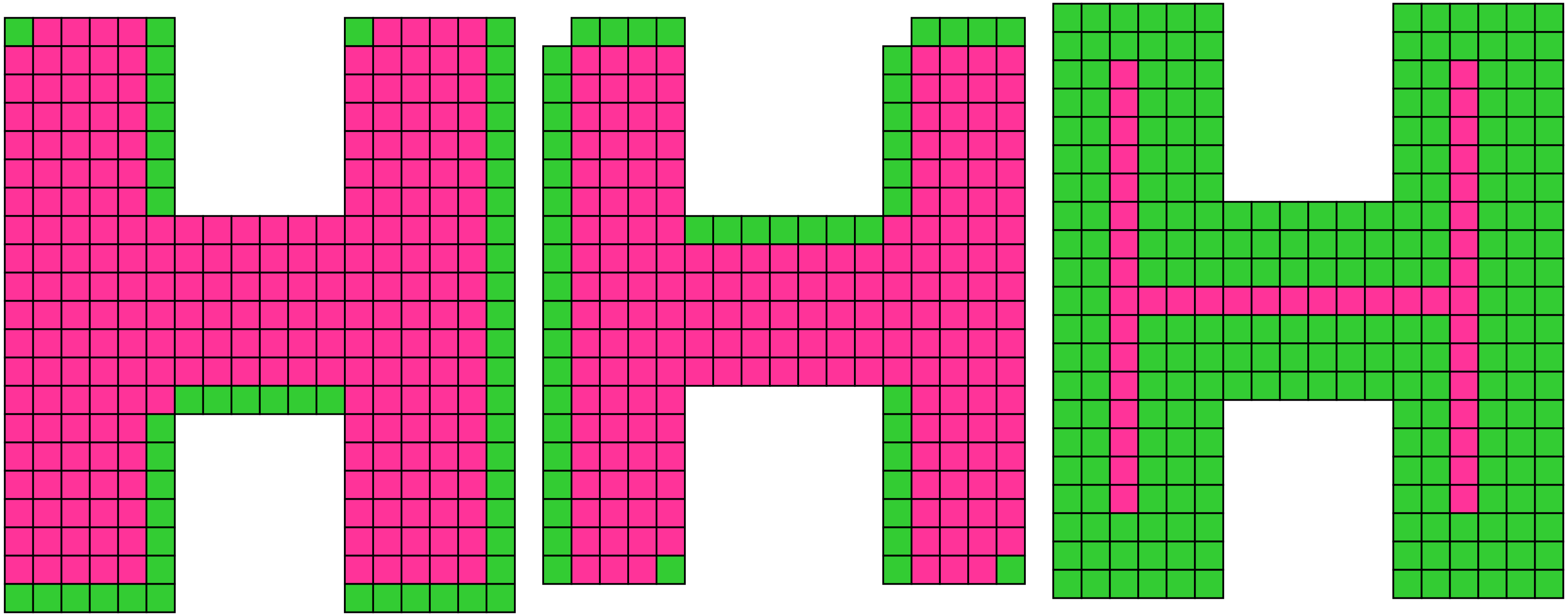
    1）做腐蚀操作，但不立刻删除像素，只打标记

    2）将不破坏连通性的标记点删掉。

    3）重复执行，将产生细化结果

3.3.2 表示与描述:表示法设计

- 区域骨架
  - 例:



表示与描述：边界描述子

- 边界描述子
  - 简单描述子
  - 形状数
  - 矩量

表示与描述：边界描述子

- 简单描述子
  - 边界的周长：

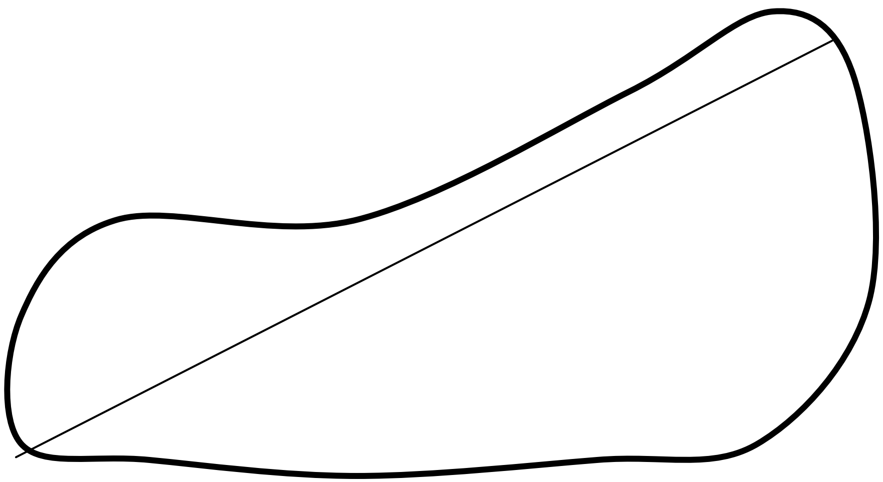
    是最简单的描述符之一。沿轮廓线计算象素的个数，给出了一个长度的近似估计
  - 边界的直径：边界B的直径是：

$$\mathrm{Diam}(B) = \max[D(p_i, p_j)]$$

D是欧氏距离或几何距离，p_i，p_j是边界上的点。直径的长度和直径的两个端点连线（这条线被称为边界的主轴）的方向，是关于边界的有用的描述符。

表示与描述：边界描述子

- 简单描述子
  - 边界的直径举例



表示与描述：边界描述子

- 形状数
  - 形状数定义：最小循环首差链码。
    - 循环首差链码：用相邻链码的差代替链码

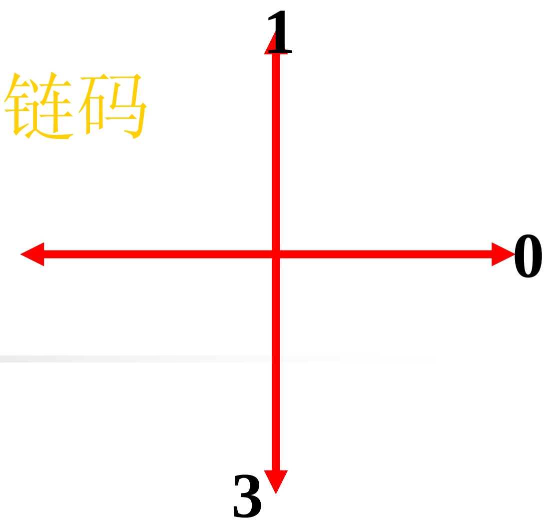
例如：4-链码　10103322

循环首差为：　33133030

循环首差:

1 - 2 = -1(3)	3 - 0 = 3
0 - 1 = -1(3)	3 - 3 = 0
1 - 0 = 1	2 - 3 = -1(3)
0 - 1 = -1(3)	2 - 2 = 0

表示与描述：边界描述子 4-链码

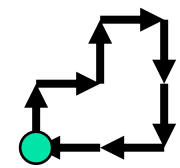


- 形状数
  - 形状数定义：<u>最小循环首差链码</u>。

    例如：　4-链码　：10103322

    　　　　循环首差　：33133030

    　　　　形状数　：03033133



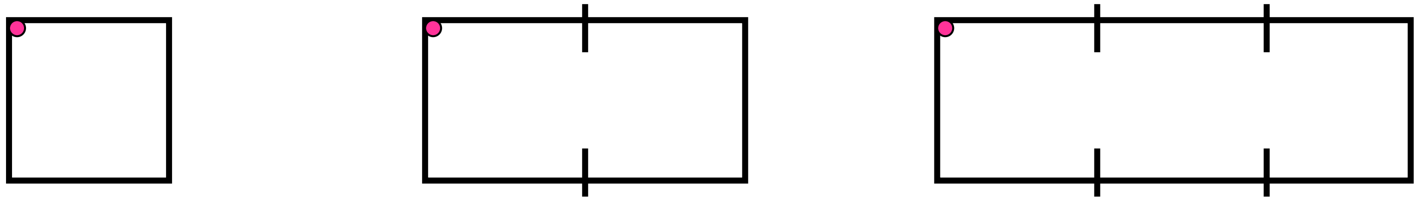
  - <u>形状数阶数</u>n的定义：

    形状数中阿拉伯数字的个数，4邻域封闭边界阶数是偶数。如order4、6、8。

表示与描述：边界描述子

- 形状数
  - 形状数例如：

序号4　　序号6　　序号8

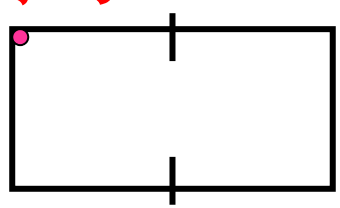


链码：0321	链码：003221	链码：00032221
首差：3333	首差：303303	首差：30033003
形状：3333	形状：033033	形状：00330033

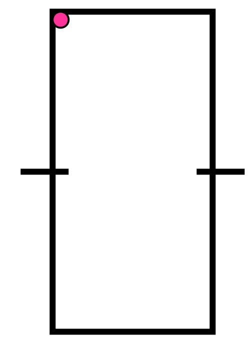
表示与描述：边界描述子

- 形状数
  - 形状数例如：

序号6



序号6



链码：003221	链码：033211
首差：303303	首差：330330
形状：033033	形状：033033

表示与描述：边界描述子

- 形状数
  - 问题：
    虽然链码的首差是不依赖于旋转的，但一般情况下边界的编码依赖于网格的方向。
  - 改进：
    规整化网格方向，具体方法如下：

表示与描述：边界描述子

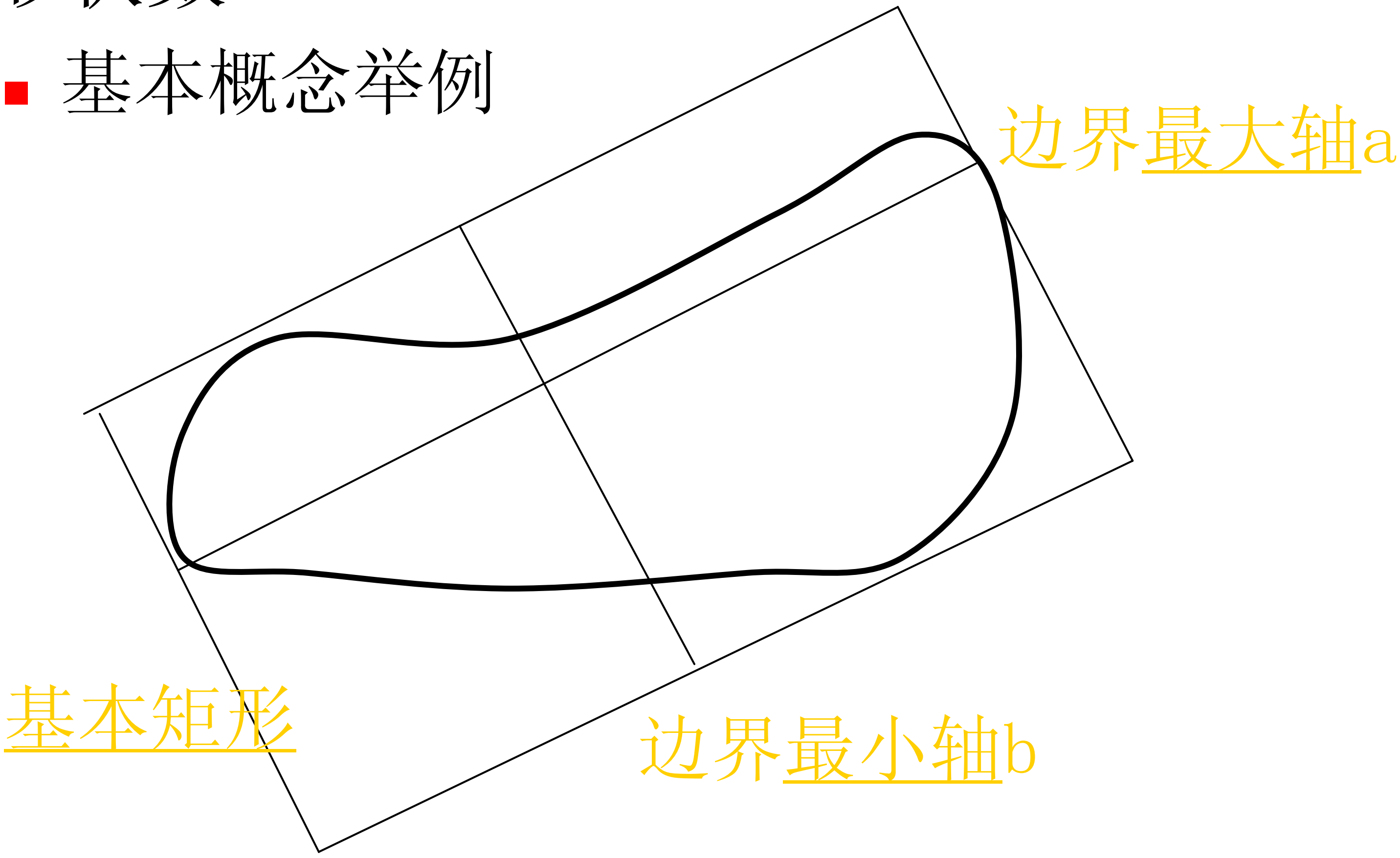
- 形状数
  - 几个基本概念：
    - 边界<u>最大轴</u>a:是连接距离最远的两个点的线段
    - 边界<u>最小轴</u>b:与最大轴垂直，且其长度确定的包围盒刚好包围边界。
    - 边界<u>离心率</u>c：最大轴长度与最小轴长度的比

$$c = a / b$$

    - <u>基本矩形</u>: 包围边界的矩形。

表示与描述：边界描述子

- 形状数
  - 基本概念举例



表示与描述：边界描述子

- 形状数
  - 规整化网格方向算法的思想：

    大多数情况下，将链码网格与基本矩形对齐，即可得到一个唯一的形状数。

    对一个给定的形状阶数，处理步骤如下：

    （1）我们找出一个阶数为n的矩形，<u>它的离心率最接近于给定形状的基本矩形的离心率</u>。

表示与描述：边界描述子

- 形状数

（2）然后再用这个矩形构造网格。

例如：如果n=12，所有阶数为12的矩形（即周长为12）为2*4，3*3，1*5。如果2*4矩形的离心率最接近于给定边界的基本矩形的离心率，我们建立一个2*4的网格。

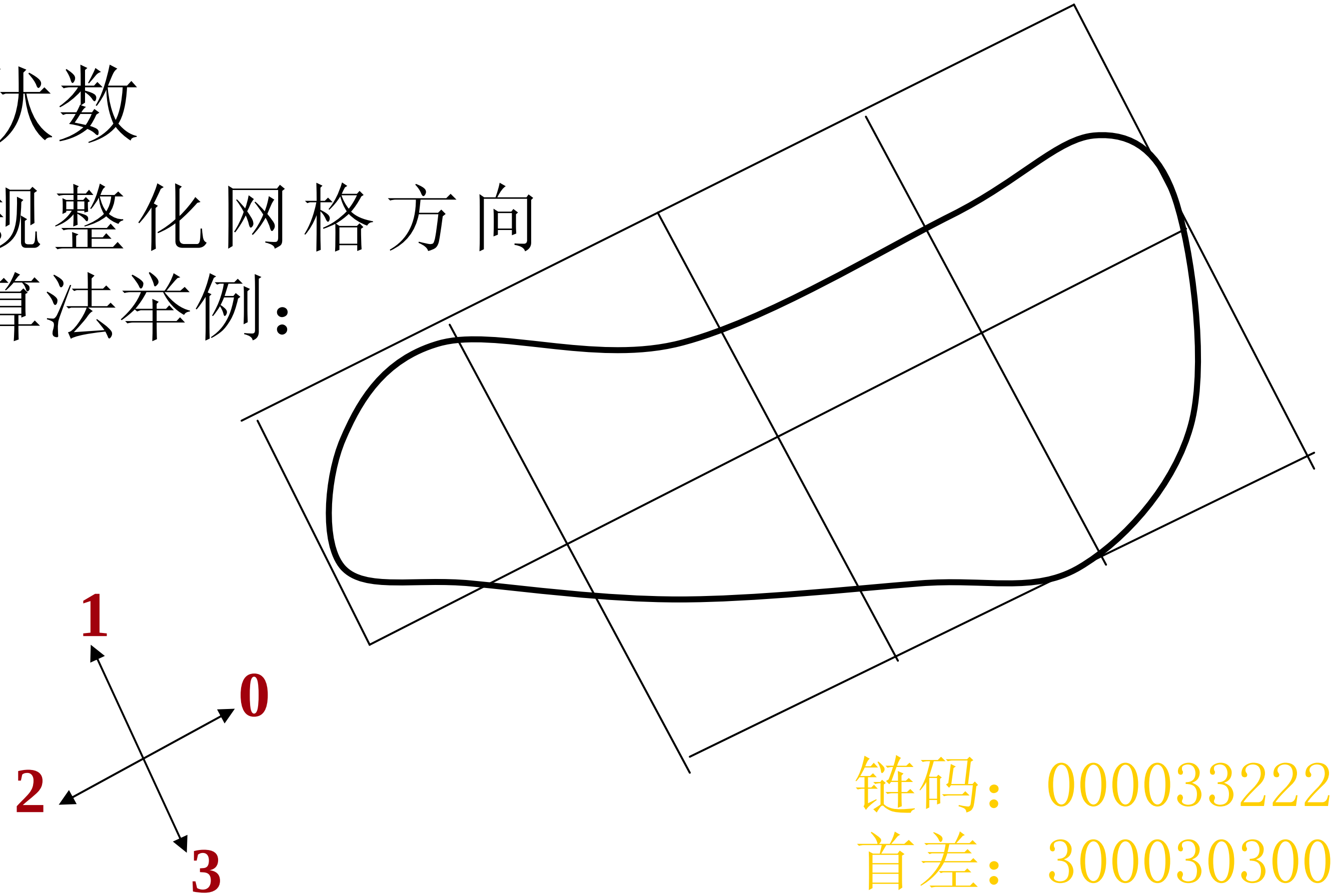
（3）再得到链码。

（4）最后，再得到循环首差。

（5）首差中的最小循环数即为形状数。

表示与描述：边界描述子

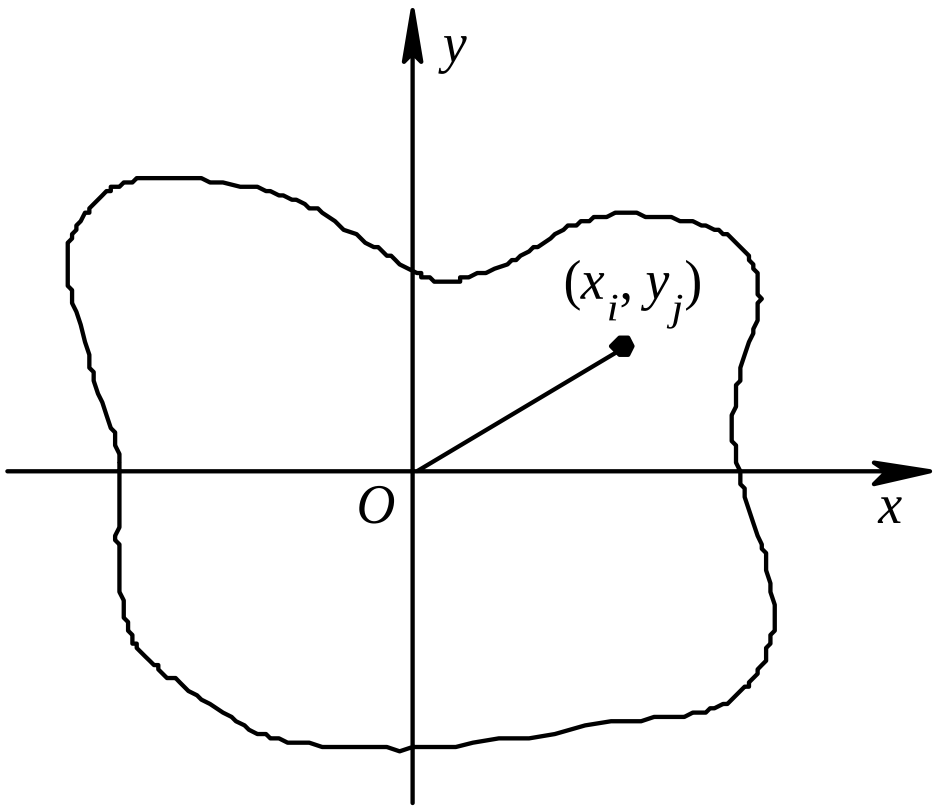
- 形状数
  - 规整化网格方向算法举例：



链码：000033222121
首差：300030300313
形状：003030031 33

图像的几何特征

位置与方向



物体位置由质心表示

图像中的物体通常并不是一个点，因此，用物体的面积的中心点作为物体的位置。面积中心就是单位面积质量恒定的相同形状图形的质心O（见图9-1）。因二值图像质量分布是均匀的， 故质心和形心重合。若图像中的物体对应的像素位置坐标为(x_i, y_j) $(i=0, 1, ..., n-1;\ j=0, 1, \ldots, m-1)$，则可用下式计算质心位置坐标：

$$\bar{x} = \frac{1}{mn}\sum_{i=0}^{n-1}\sum_{j=0}^{m-1} x_i, \quad \bar{y} = \frac{1}{mn}\sum_{i=0}^{n-1}\sum_{j=0}^{m-1} y_j \qquad (9\text{-}1)$$

面积

面积是物体的总尺寸的一个方便的度量。面积只与该物体的边界有关， 而与其内部灰度级的变化无关。一个形状简单的物体可用相对较短的周长来包围它所占有的面积。

1. 像素计数面积

最简单的(未校准的)面积计算方法是统计边界内部(也包括边界上)的像素的数目。在这个定义下面积的计算非常简单， 求出域边界内像素点的总和即可，计算公式如下：

$$A=\sum_{x=1}^{N}\sum_{y=1}^{M}f(x,y)$$

对二值图像而言，若用1表示物体，用0表示背景，其面积就是统计$f(x,y)=1$的个数。

距离

图像中两点P(x， y)和Q(u，v)之间的距离是重要的几何性质，常用如下三种方法测量：

（1） 欧几里德距离：

$$d_e(P,Q)=\sqrt{(x-u)^2+(y-v)^2} \qquad (9\text{-}10)$$

（2） 市区距离：

$$d_4(P,Q)=|x-u|+|y-v| \qquad (9\text{-}11)$$

（3）棋盘距离：

$$d_8(P,Q) = \max(|x-u|,|y-v|) \quad (9\text{-}12)$$

显然，以P为起点的市区距离小于等于t（t=1，2，...）的点形成以P为中心的菱形。图9-5(a)为t≤2时用点的距离表示的这些点。可见，d 4(P ， Q)是从P到Q最短的4路径的长度。同样，以P为起点的棋盘距离小于等于t(t=1, 2, …)的点形成以P为中心的正方形。例如， 当t≤2，用点的距离表示这些点时，如图9-5（b）所示。同样由图可见，$d_8(P, Q)$是从P到Q最短的8路径的长度。

		2		
	2	1	2	
2	1	0	1	2
	2	1	2	
		2		

(a)

2	2	2	2	2
2	1	1	1	2
2	1	0	1	2
2	1	1	1	2
2	2	2	2	2

(b)

图9-5 两种距离表示法

（a）$d_4(P, Q)\leqslant 2$; (b) $d_8(P, Q)\leqslant 2$

d_4、d_8计算简便，且为正整数，因此常用来测距离，而欧几里德距离很少被采用。

不变矩

1. 矩的定义

对于二元有界函数$f(x, y)$，它的$(j+k)$阶矩为

$$M_{jk} = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} x^j y^k f(x,y)dxdy \quad j,k = 0,1,2,\cdots \qquad （9\text{-}23）$$

由于j和k可取所有的非负整数值，因此形成了一个矩的无限集。而且，这个集合完全可以确定函数f $(x，y)$本身。换句话说，集合$\{M_{jk}\}$对于函数f $(x，y)$是惟一的，也只有$f(x，y)$才具有这种特定的矩集。

为了描述物体的形状，假设f (x，y)的目标物体取值为1，背景为0，即函数只反映了物体的形状而忽略其内部的灰度级细节。

参数$j+k$称为矩的阶。特别地，零阶矩是物体的面积，即

$$M_{00}=\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}f(x,y)dxdy \qquad (9\text{-}24)$$

对二维离散函数$f(x$，$y)$，零阶矩可表示为

$$M_{00}=\sum_{x=1}^{N}\sum_{y=1}^{M}f(x,y) \qquad (9\text{-}25)$$

所有的一阶矩和高阶矩除以M_{00}后，与物体的大小无关。

2. 质心坐标与中心矩

当j=1, k=0时，M_{10}对二值图像来讲就是物体上所有点的x坐标的总和，类似地，M_{01}就是物体上所有点的y坐标的总和，所以

$$\bar{x}=\frac{M_{10}}{M_{00}},\bar{y}=\frac{M_{01}}{M_{00}} \tag{9-26}$$

就是二值图像中一个物体的质心的坐标。

为了获得矩的不变特征，往往采用中心矩以及归一化的中心矩。中心矩的定义为

$$M'_{jk}=\sum_{x=1}^{N}\sum_{y=1}^{M}(x-\bar{x})^{j}(y-\bar{y})^{k}f(x,y) \tag{9-27}$$

3. 主轴

使二阶中心矩从 μ_{11}变得最小的旋转角 θ可以由下式得出：

$$\tan 2\theta = \frac{2\mu_{11}}{\mu_{20} - \mu_{02}} \qquad (9\text{-}28)$$

将x、y轴分别旋转 θ角得坐标轴x' 、y' ，称为该物体的主轴。式9-28中在 θ 为90° 时的不确定性可以通过如下条件限定解决：

$$\mu_{20} < \mu_{02}, \mu_{30} > 0$$

如果物体在计算矩之前旋转 θ角，或相对于x' 、 y' 轴计算矩，那么矩具有旋转不变性。

4. 不变矩

相对于主轴计算并用面积归一化的中心矩， 在物体放大、平移、 旋转时保持不变。只有三阶或更高阶的矩经过这样的规一化后不能保持不变性。

对于j+k ＝2, 3, 4…的高阶矩，可以定义归一化的中心矩为

$$\mu_{jk} = \frac{M'_{jk}}{(M_{00})^r}, r = \left(\frac{j+k}{2} + 1\right)$$

利用归一化的中心矩，可以获得六个不变矩组合，这些组合对于平移、旋转、尺度等变换都是不变的，它们是：

$$\varphi_1 = \mu_{20} + \mu_{02} \tag{9-30a}$$

$$\varphi_2 = (\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2 \tag{9-30b}$$

$$\varphi_3 = (\mu_{30} - 3\mu_{12})^2 + (\mu_{03} - 3\mu_{21})^2 \tag{9-30c}$$

$$\varphi_4 = (\mu_{30} + \mu_{12})^2 + (\mu_{03} + \mu_{21})^2 \tag{9-30d}$$

$$\begin{aligned}\varphi_5 = {} & (\mu_{30} - 3\mu_{12})(\mu_{03} + \mu_{12}) \times [(\mu_{30} + \mu_{12})^2 - 3(\mu_{21} + \mu_{03})^2] \\ & + (\mu_{03} - 3\mu_{21})(\mu_{30} + \mu_{21}) \times [(\mu_{03} + \mu_{21})^2 - 3(\mu_{12} + \mu_{30})^2]\end{aligned} \tag{9-30e}$$

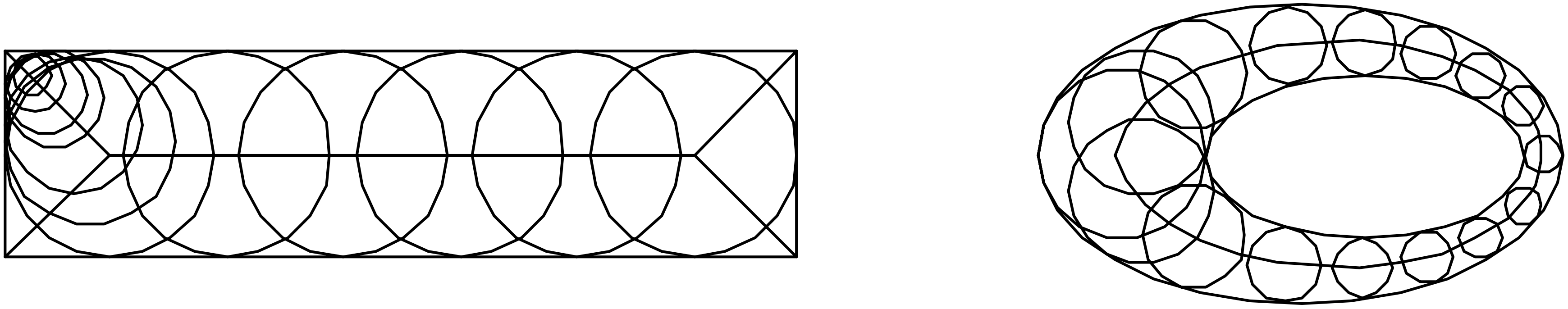
$$\varphi_6 = (\mu_{20} - \mu_{02})[(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2] + 4\mu_{11}(\mu_{30} + \mu_{21})(\mu_{03} + \mu_{21}) \tag{9-30f}$$

不变矩及其组合具备了好的形状特征应具有的某些性质，已经用于印刷体字符的识别、飞机形状区分、景物匹配和染色体分析中，但它们并不能确保在任意情况下都具有这些性质。一个物体形体的惟一性体现在一个矩的无限集中，因此，要区别相似的形体需要一个很大的特征集。这样所产生的高维分类器对噪声和类内变化十分敏感。在某些情况下，几个阶数相对较低的矩可以反映一个物体的显著形状特征。

中轴变换与骨架提取

把一个平面区域简化成图是一种重要的结构形状表示法。利用细化技术得到区域的骨架是常用的方法。中轴变换(Mdial Axis Transfonn，MAT)是一种用来确定物体骨架的细化技术。具有边界B的区域R的MAT是按如下方法确定的： 对每个R中的点P，在B中搜寻与它最近的点；如果对P能找到多于一个这样的点（即有两个或两个以上的B中的点与P同时最近），就可认为P属于R的中线或骨架， 或者说P是一个骨架点。

理论上讲，每个骨架点保持了其与边界点距离最小的性质， 因此用以每个骨架点为中心的圆的集合（利用合适的量度）， 就可恢复出原始的区域来。具体讲就是以每个骨架点为圆心， 以前述最小距离为半径作圆周， 它们的包络就构成了区域的边界，填充圆周就得到区域。或者以每个骨架点为圆心，以所有小于和等于最小距离的长度为半径作圆，这些圆的并集就覆盖了整个区域。中轴变换示意如图9-16所示。

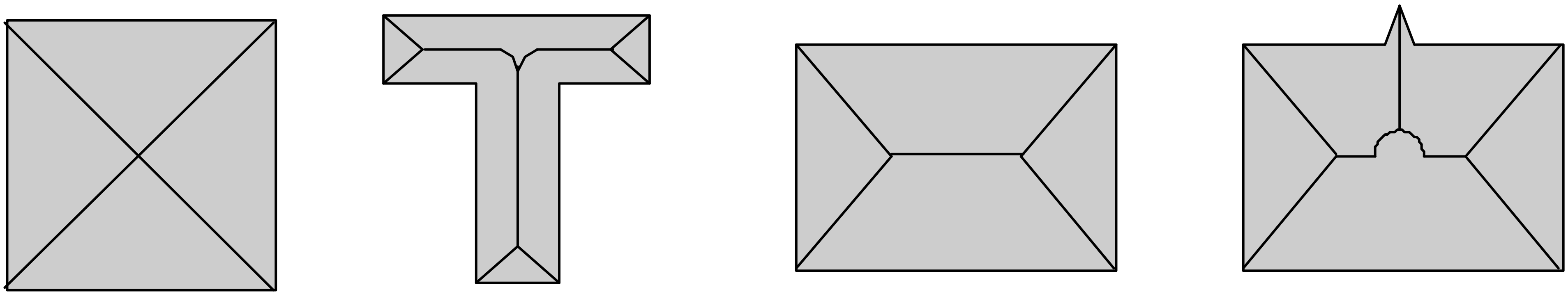


中轴变换示意图

由上述讨论可知，骨架是用一个点与一个点集的最小距离来定义的，可写成

$$d_s(p, B)=\inf\{d(p, z) \mid z \in B\} \qquad (9\text{-}54)$$

其中距离量度可以是欧几里德、市区或棋盘距离。因为最小距离取决于所用的距离量度，所以MAT的结果也和所用的距离量度有关。



(a)　(b)　(c)　(d)

一些区域和用欧氏距离算出的骨架示例

四叉树

四叉树表达表示图像是一个“金字塔”式的观察和处理过程。这种数据结构是一种有效的对空间占有数组的编码，可以很好地描述一幅图像。当图像是方形的， 且像素点的个数是2的整数次幂（即图像尺寸为$2^k \times 2^k$，k为正整数）时四叉树法最适用。如图9-23所示，在这种表达中，所有的节点可分成三类：目标节点(用白色表示)、 背景节点(用深色表示)和混合节点(用浅色表示)。四叉树的树根对应整幅图， 而树叶对应各单个像素或具有相同特性的像素组成的方阵。四叉树由多级构成， 数根在0级， 分一次叉多一级。对一个有n级的四叉树，其节点总数N最多为

$$N=\sum_{i=0}^{n}4^i=\frac{4^{n+1}-1}{3}\approx\frac{4}{3}4^n \qquad (9\text{-}70)$$

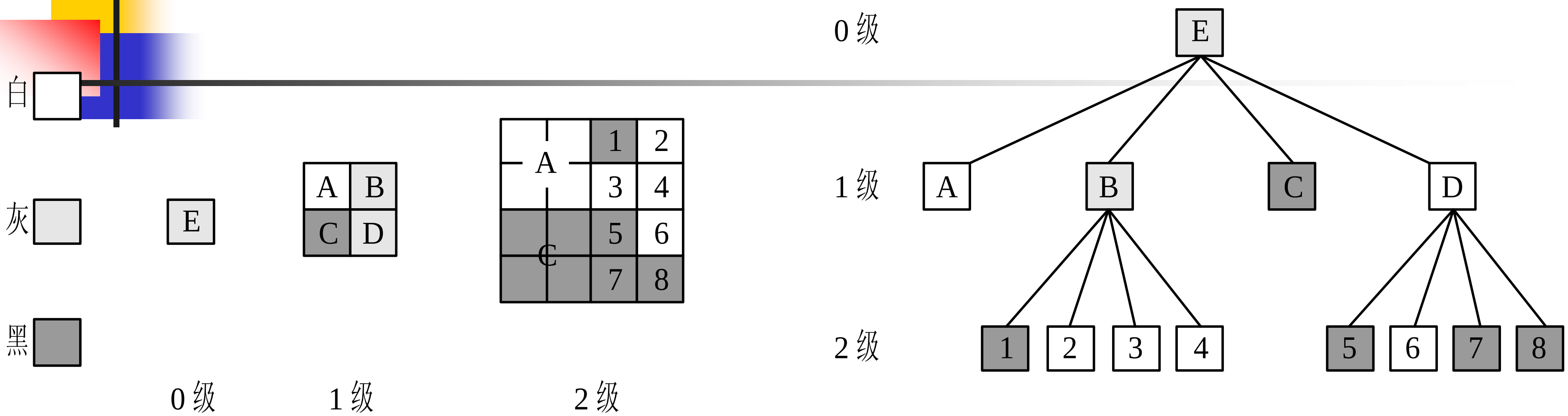


图9-23 四叉树表达图示

四叉树表示图像的具体做法是：树的根节点表示整幅图像，如果该图像只有一个值，就用那个值和终点标记根节点；否则， 在根节点上加上4个分支，产生新的节点，每个分支表示1／4图像。对每个新节点重复上述过程， 直到整个四叉树产生为止。通常，在h层上的节点(如果有的话)表示尺寸为$2^{k-h}\times 2^{k-h}$的块， 那些块的坐标位置是2^{k-h}的倍数。假如其中一块为同一值，它的节点即叶节点； 否则，会产生h+1层上的4个分支，将h层上的块4等分。在n层上的节点(假如有的话)全对应于单个像素的叶节点。

四叉树表达的优点是：四叉树容易生成得到， 根据它可方便地计算区域的多种特征；另外，四叉树本身的结构特点使得它常用在“粗略信息优先”的显示中。它的缺点是： 如果节点在树中的级确定后，分辨率就不可能进一步提高；另外，四叉树间的运算只能在同级的节点间进行。四叉树表达在三维空间的对应是八叉树(也叫八元树)表达。