数字图像处理

第八讲

形态学处理

提纲

- 预备知识
- 腐蚀和膨胀
- 开操作和闭操作
- 击中或击不中变换
- 基本形态学算法
  - 边界提取、孔洞填充
  - 连通分量提取、凸包
  - 细化、粗化
  - 骨架、裁剪

引言

- 形态学（morphology）
  - 生物学的一个分支
  - 研究动植物的形态和结构

- 数学形态学（mathematical morphology）
  - 提取表示区域形状的图像成分
    - 边界、凸包、骨架
  - 输入：图像
  - 输出：图像中提取的属性

预备知识

- 集合论
  - 描述形态学的数学语言
  - 集合：表示图像中的对象
  - 例如，二值图像中的所有白色像素
- 二值图像
  - 集合：属于2维整数空间Z^2
  - 元素：二元组(x, y)
    - 表示白色像素的坐标
- 灰度图像、Z^3

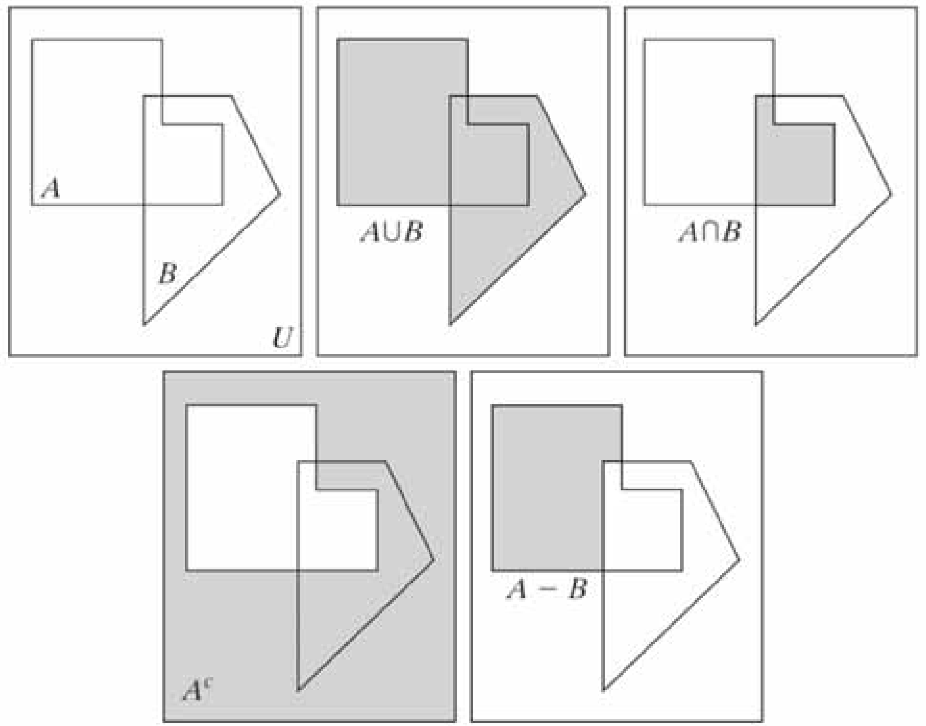
基本集合操作

- $a = (a_1, a_2)$是A的元素：$a \in A$
- a不是A的元素：$a \notin A$
- 空集：$\emptyset$
- 全集：U
- A是B的子集：$A \subseteq B$
- 集合A和B的并集：$A \cup B$
- 集合A和B的交集：$A \cap B$
- 集合A和B互斥：$A \cap B = \emptyset$

基本集合操作

- 集合A的补集：$A^c = \{w|w \notin A\} = U - A$
- 集合A和B的差：

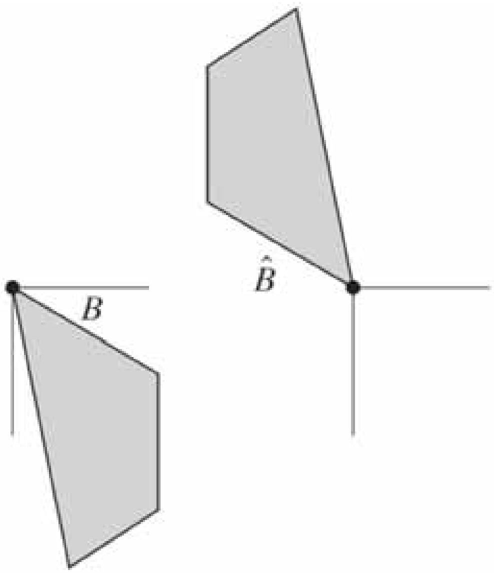
$$A - B = \{w|w \in A, w \notin B\} = A \cap B^c$$



集合操作

- 集合的反射

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



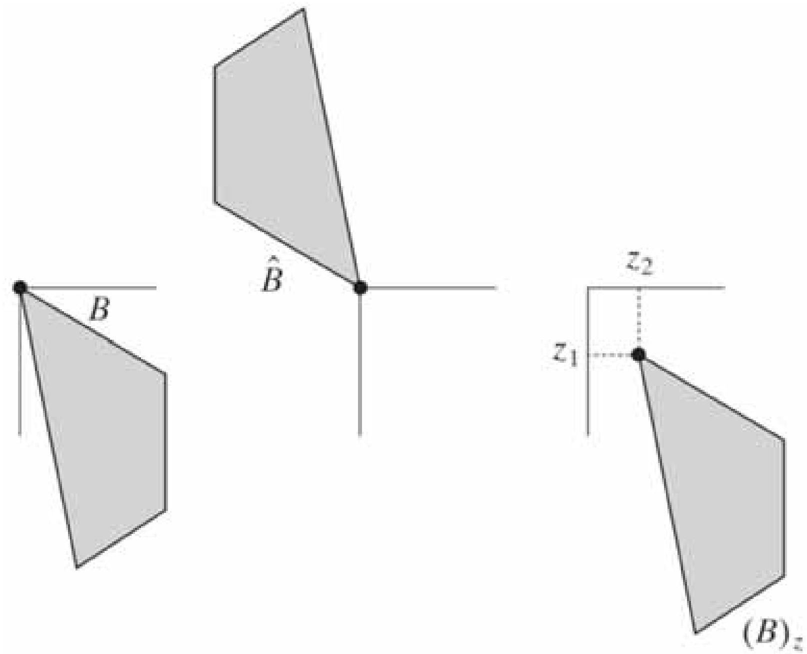
集合操作

- 集合的反射

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

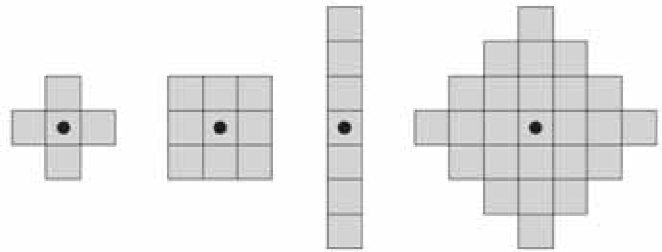
- 集合的平移

$$(B)_z = \{c \mid c = b + z, \quad \text{for} \quad b \in B\}$$

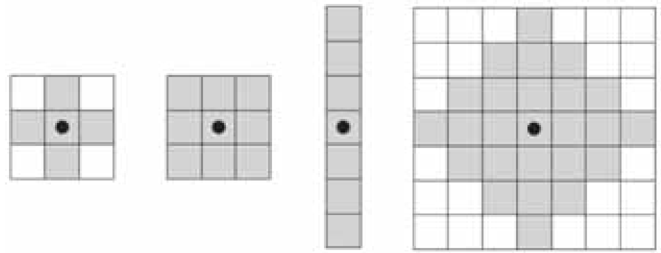


基于结构元的操作

- 结构元（structuring elements）
  - 用于研究图像性质的小集合或子图像

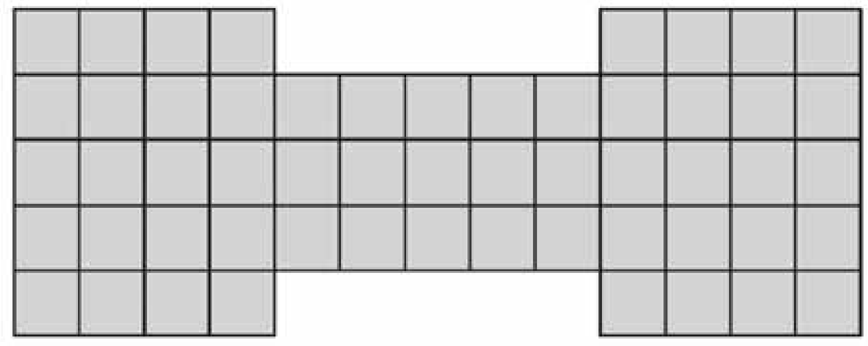


  - 黑点表示结构元的原点
  - 通常用矩形表示
    - 填充背景

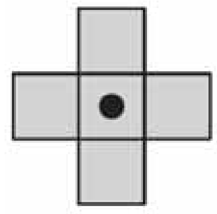


基于结构元的操作

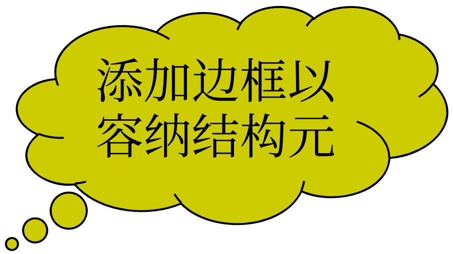
集合A

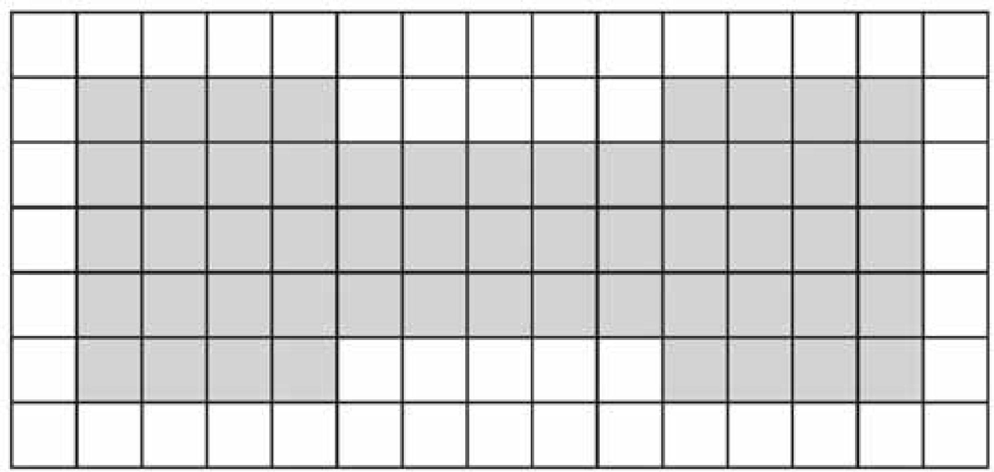


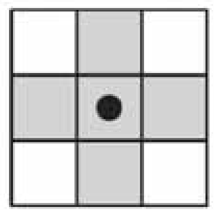
结构元B



- 填充成矩形

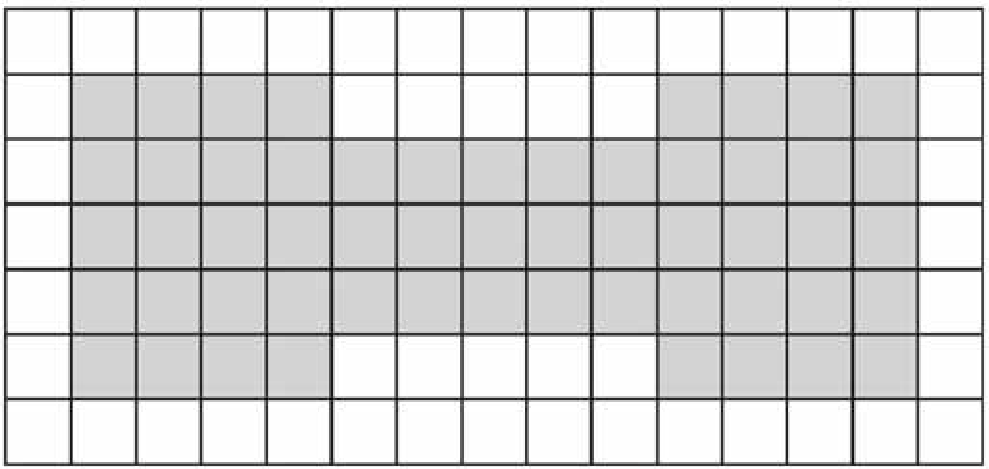


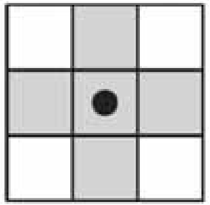




基于结构元的操作

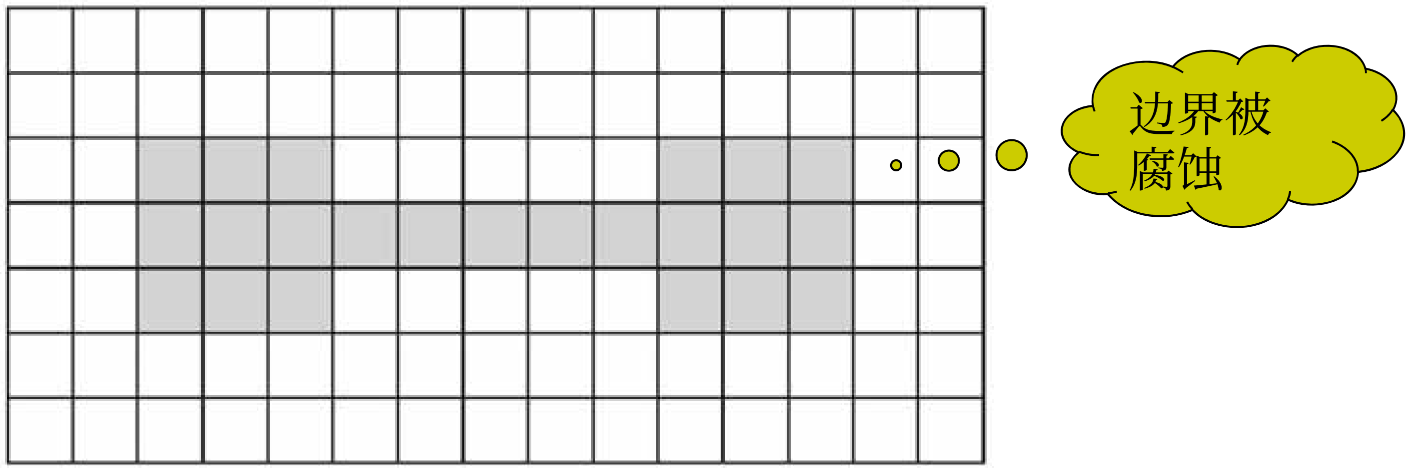
- 利用结构元构造一个新集合C
  1. 用结构元B覆盖集合A
  2. 在当前位置（B的原点），如果A完全包含B，则当前位置属于C





基于结构元的操作

- 利用结构元构造一个新集合C
  1. 用结构元B覆盖集合A
  2. 在当前位置（B的原点），如果A完全包含B，则当前位置属于C
  3. 移动结构元B，使其原点访问A中的所有元素



提纲

- 预备知识
- 腐蚀和膨胀
- 开操作和闭操作
- 击中或击不中变换
- 基本形态学算法
  - 边界提取、孔洞填充
  - 连通分量提取、凸包
  - 细化、粗化
  - 骨架、裁剪

腐蚀

- 集合B对集合A的腐蚀（erosion）

$$A \ominus B = \{z \mid (B)_z \subseteq A\}$$

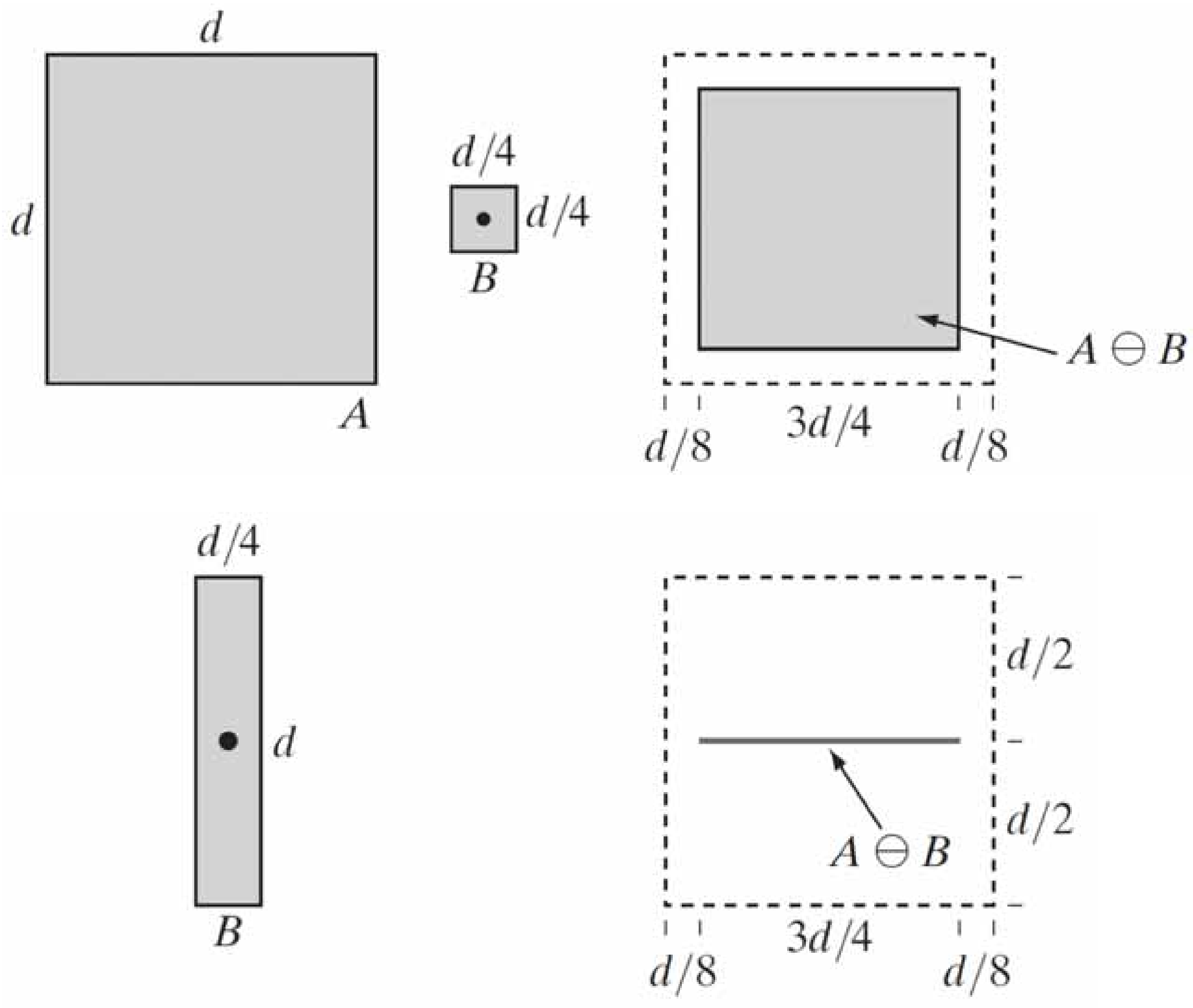
  - $(B)_z$表示把集合B平移到坐标z
  - 通常假设集合B为结构元
  - $(B)_z$意味着把B的原点平移到z

- 等价定义

$$A \ominus B = \{z \mid (B)_z \cap A^c = \varnothing\}$$

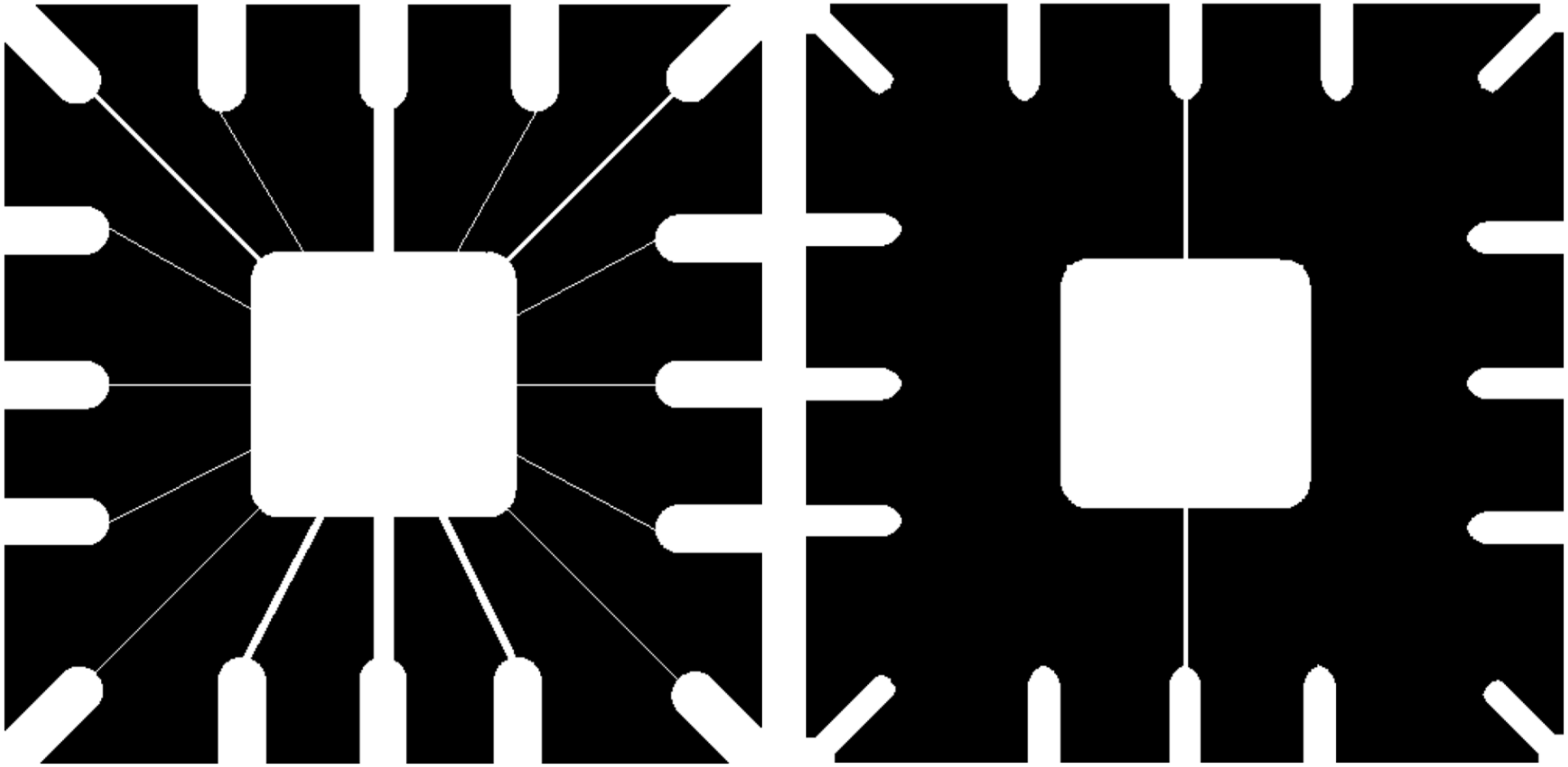
  - A^c表示集合A的补集

举例



举例

- 去掉连接线

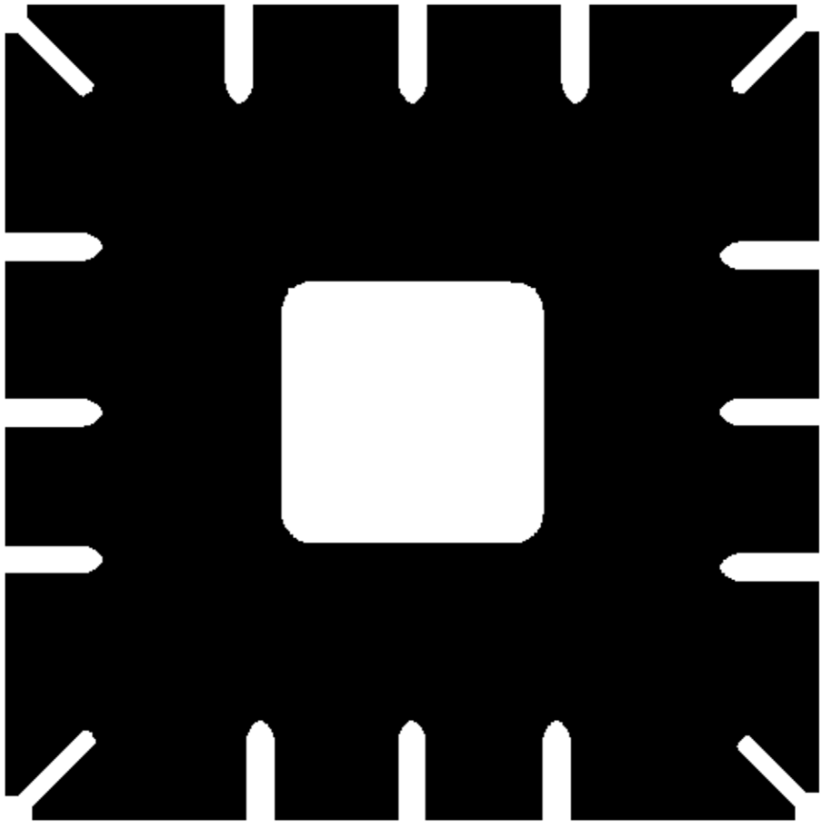


用11 × 11的方框腐蚀

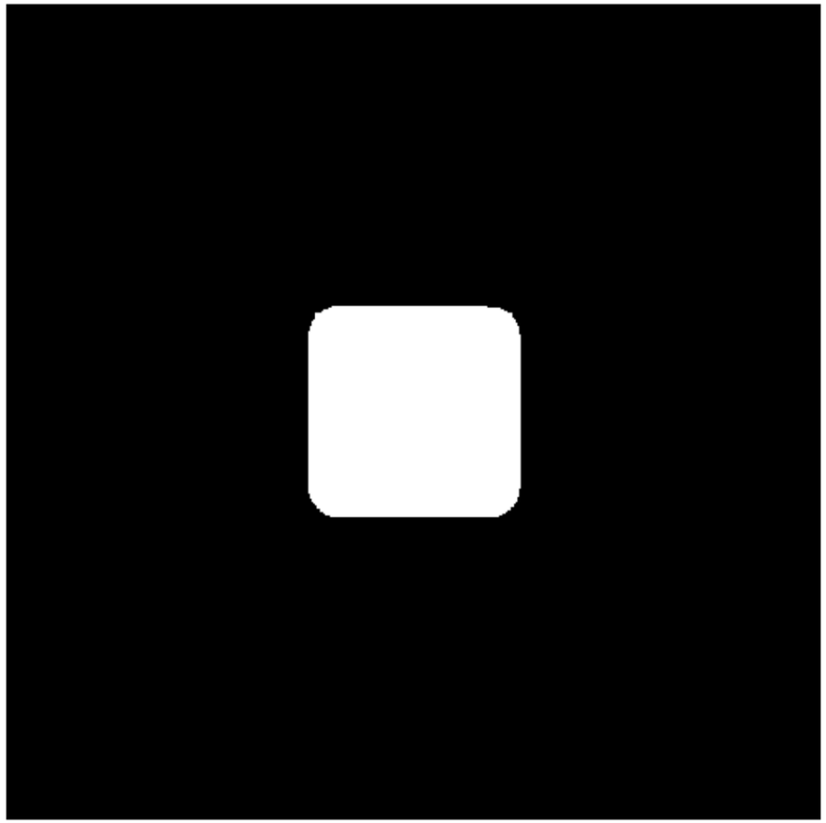
举例

腐蚀，可以被理解为形态学滤波

- 去掉连接线



用15 × 15的方框腐蚀



用45 × 45的方框腐蚀

膨胀

- 集合B对集合A的膨胀（dilation）

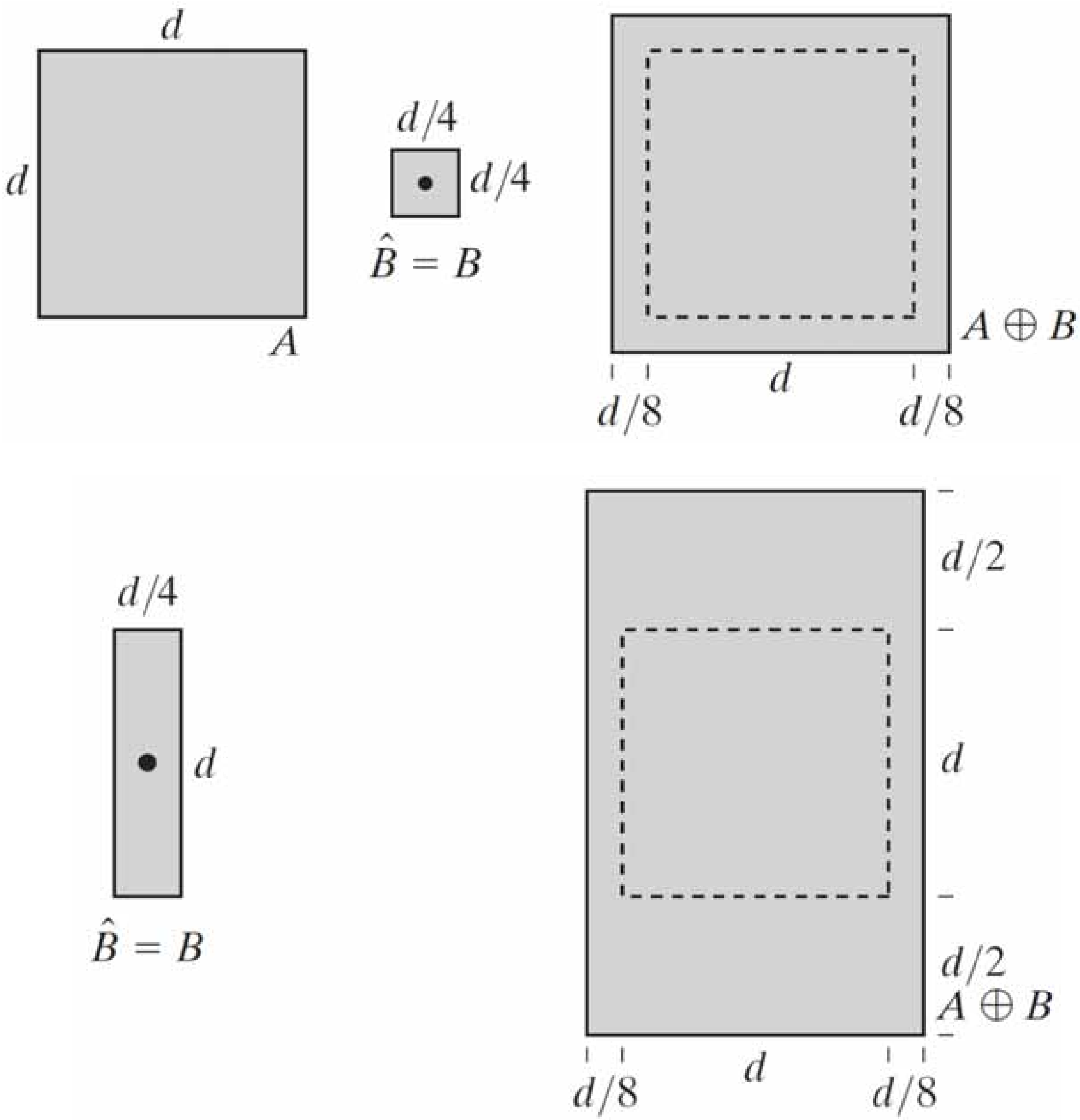
$$A \oplus B = \left\{ z \mid (\hat{B})_z \cap A \neq \varnothing \right\}$$

  - $\hat{B}$表示集合B的反射
  - $(\hat{B})_z$表示把集合$\hat{B}$平移到坐标z
  - 通常假设集合B为结构元

- 等价定义

$$A \oplus B = \bigcup_{b \in B} (A)_b$$

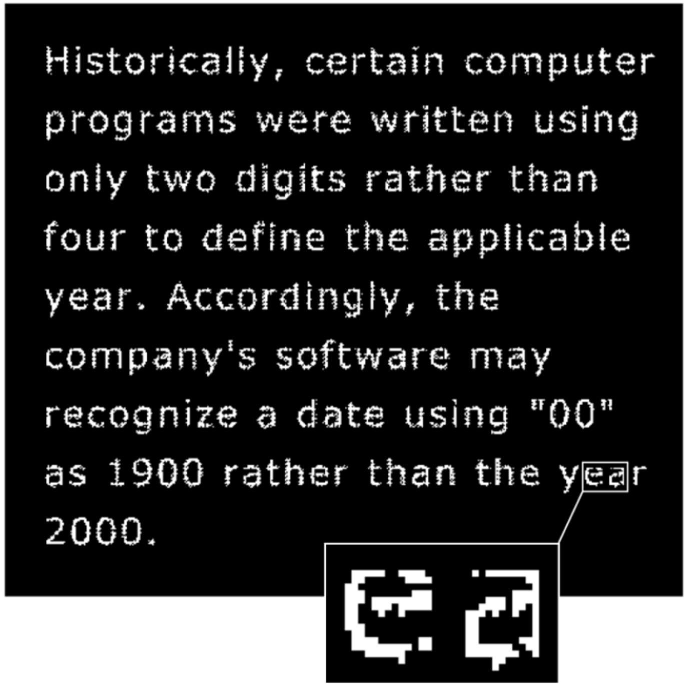
举例



举例

比低通滤波器
更简单、直接

输出仍然
是二值图
像


Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.

0	1	0
1	1	1
0	1	0


Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.

最长间距是2个像素

对偶性

- 公式

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

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

- 证明

$$(A \ominus B)^c = \left\{ z \mid (B)_z \subseteq A \right\}^c$$

$$= \left\{ z \mid (B)_z \cap A^c = \varnothing \right\}^c$$

$$= \left\{ z \mid (B)_z \cap A^c \neq \varnothing \right\}$$

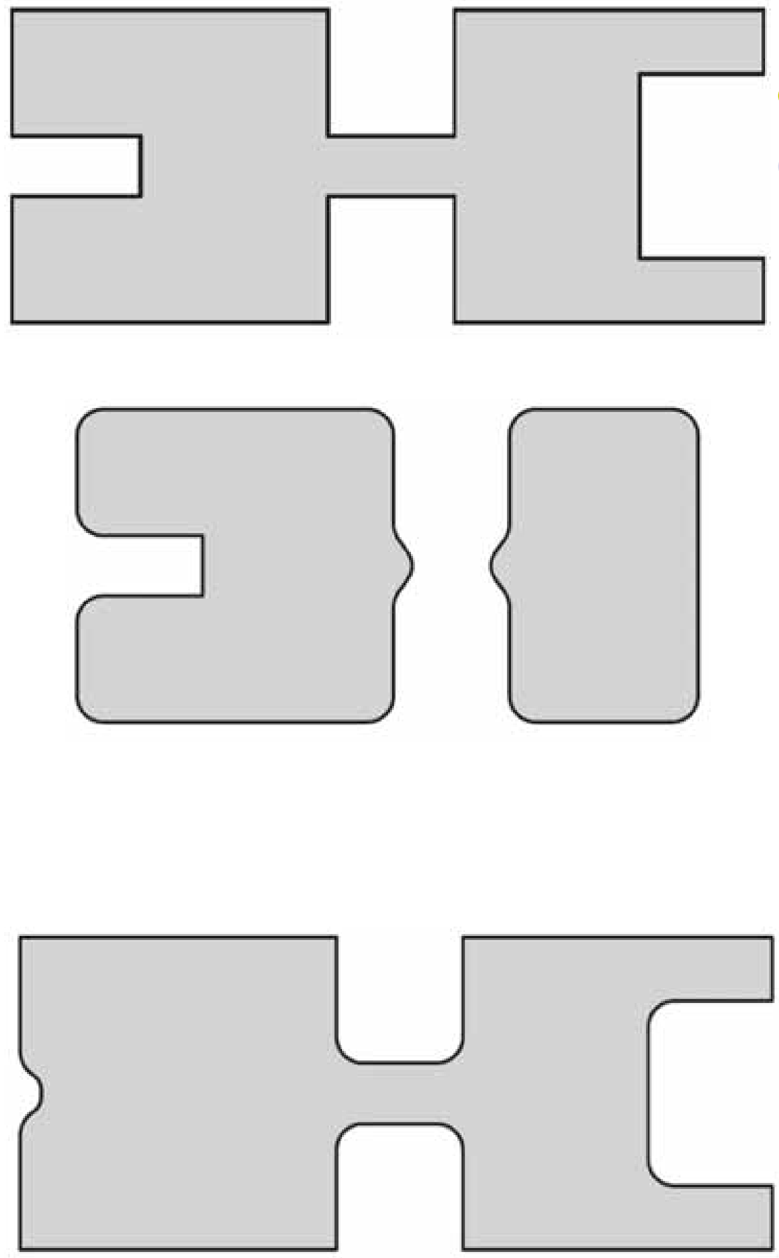
$$= A^c \oplus \hat{B}$$

提纲

- 预备知识
- 腐蚀和膨胀
- 开操作和闭操作
- 击中或击不中变换
- 基本形态学算法
  - 边界提取、孔洞填充
  - 连通分量提取、凸包
  - 细化、粗化
  - 骨架、裁剪

开操作和闭操作

- 开操作（opening）
  - 平滑物体的轮廓
  - 断开窄的连接
  - 消除细的突出
- 闭操作（closing）
  - 平滑部分轮廓
  - 熔合窄的间断和长沟壑
  - 消除小孔洞
  - 填补轮廓中的缝隙



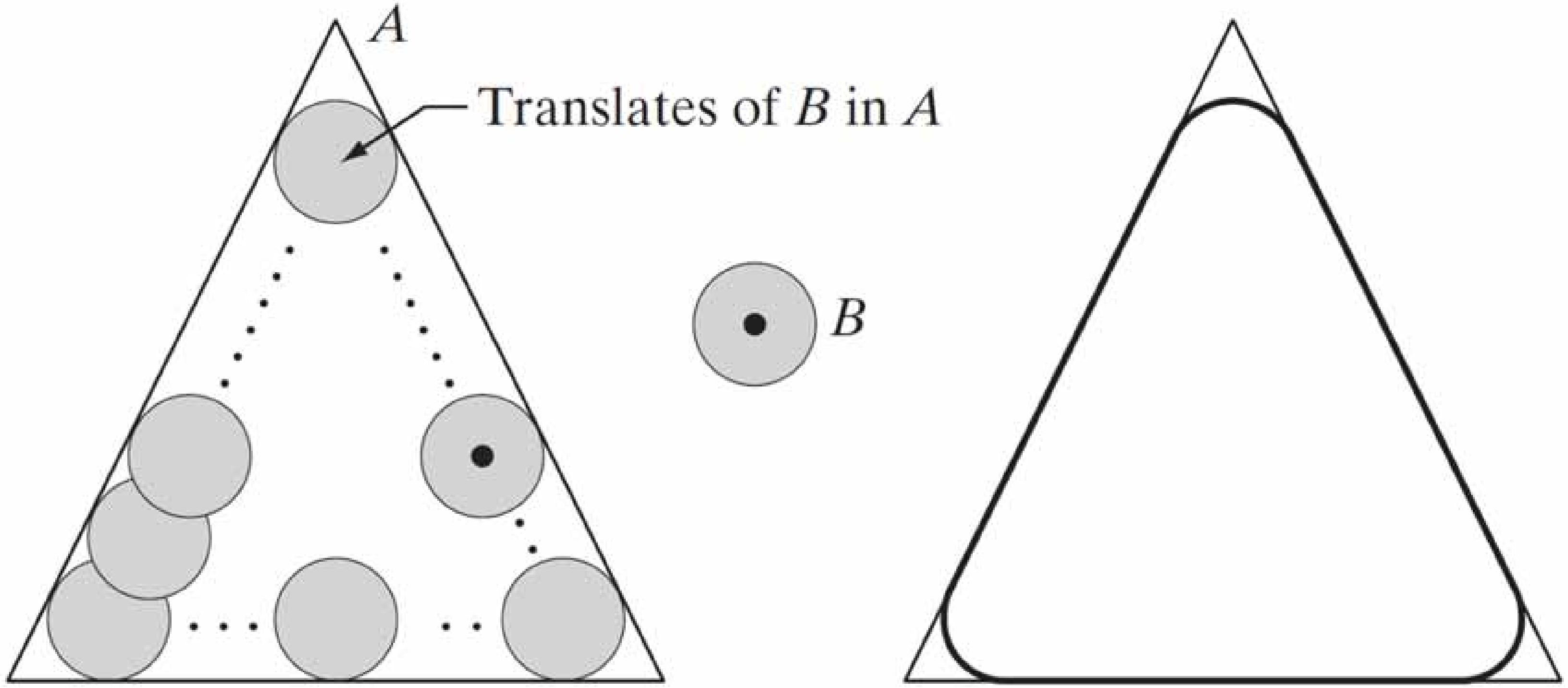
开操作

在A的边界内侧滚动B，B的最远点决定了轮廓

- 结构元B对集合A的开操作

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

  - 先用B腐蚀A，然后再用B对结果进行膨胀

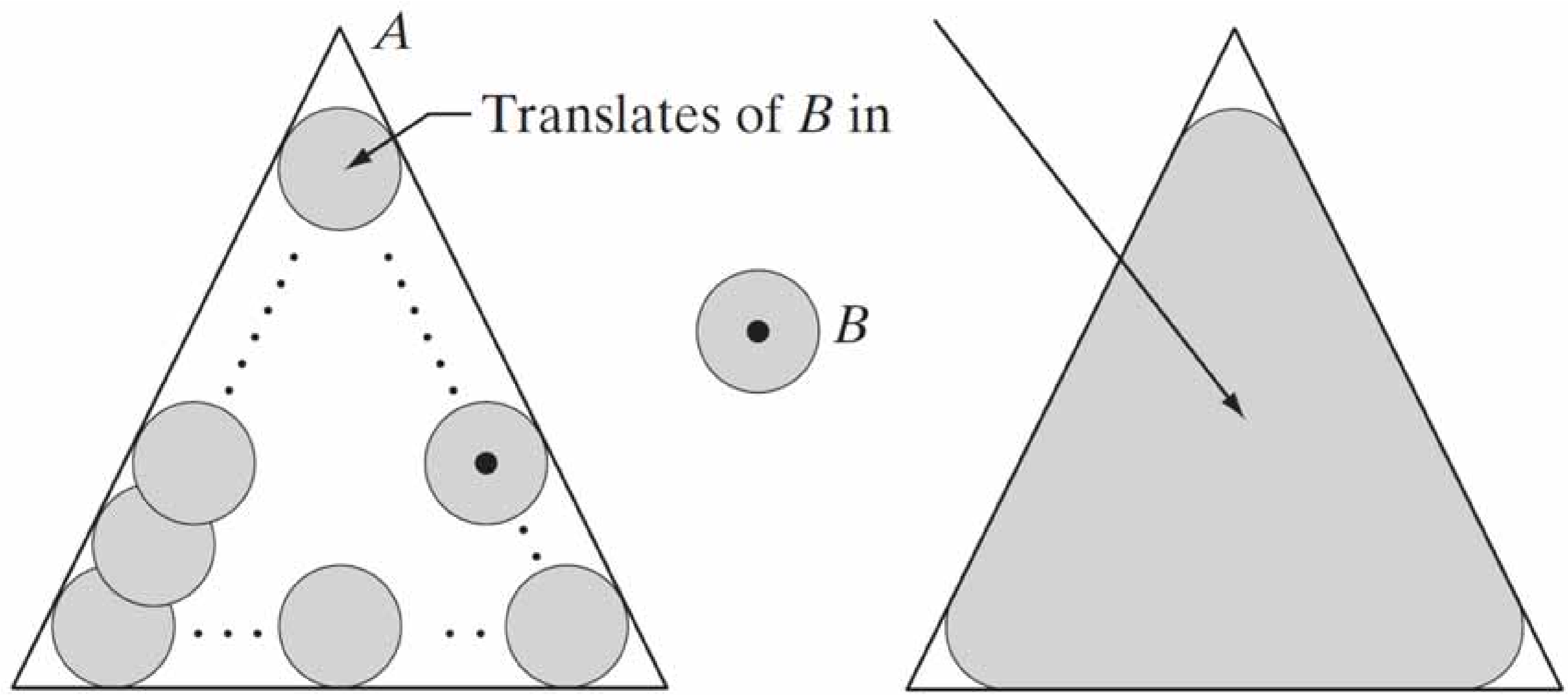


开操作

- 结构元B对集合A的开操作

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

$$A \circ B = \bigcup \left\{ (B)_z \middle| (B)_z \subseteq A \right\}$$



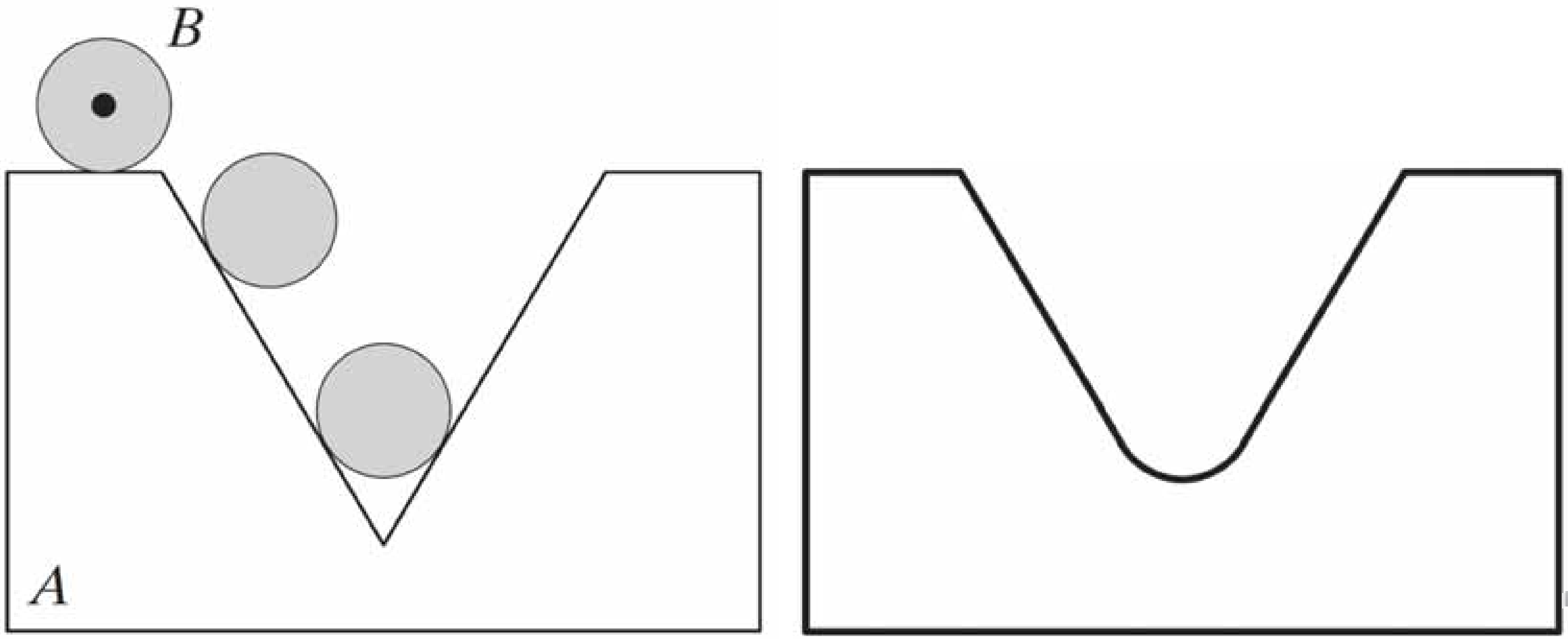
闭操作

在A的边界外侧滚动B，B的最近点决定了轮廓

- 结构元B对集合A的闭操作

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

  - 先用B膨胀A，然后再用B对结果进行腐蚀



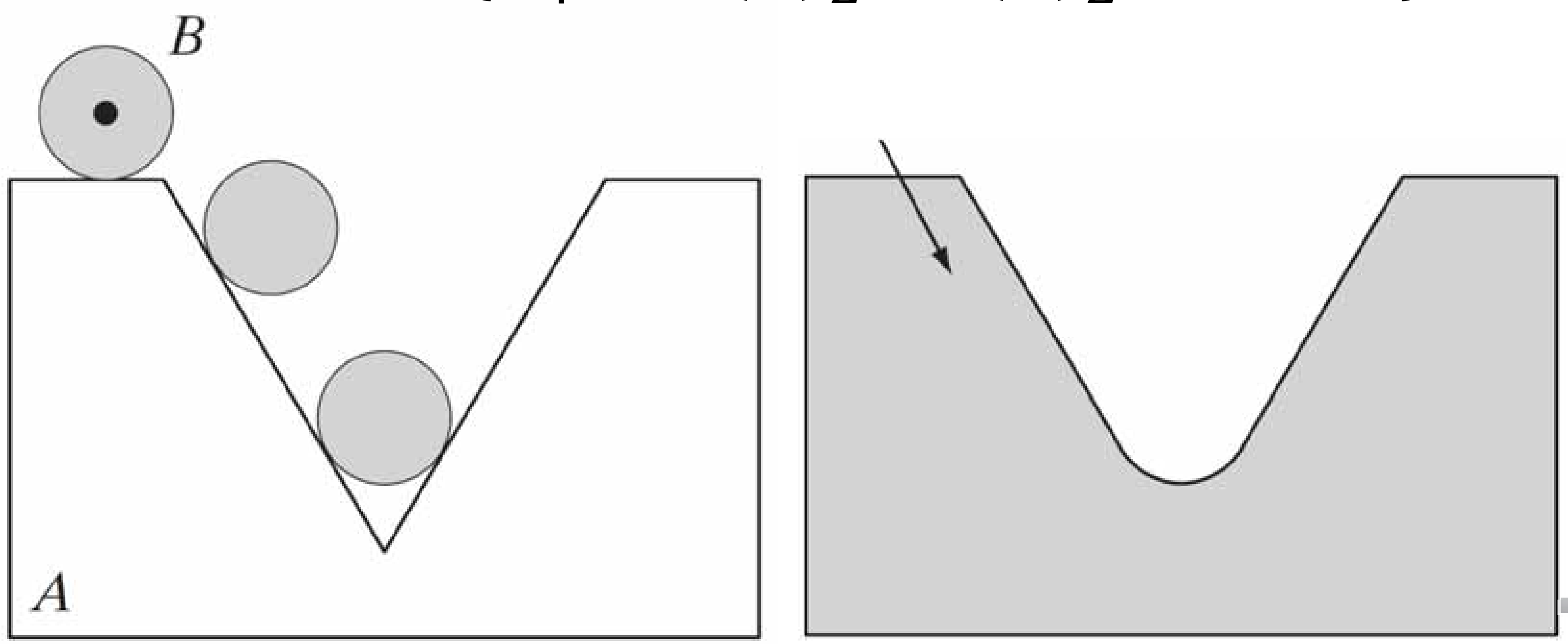
闭操作

- 结构元B对集合A的闭操作

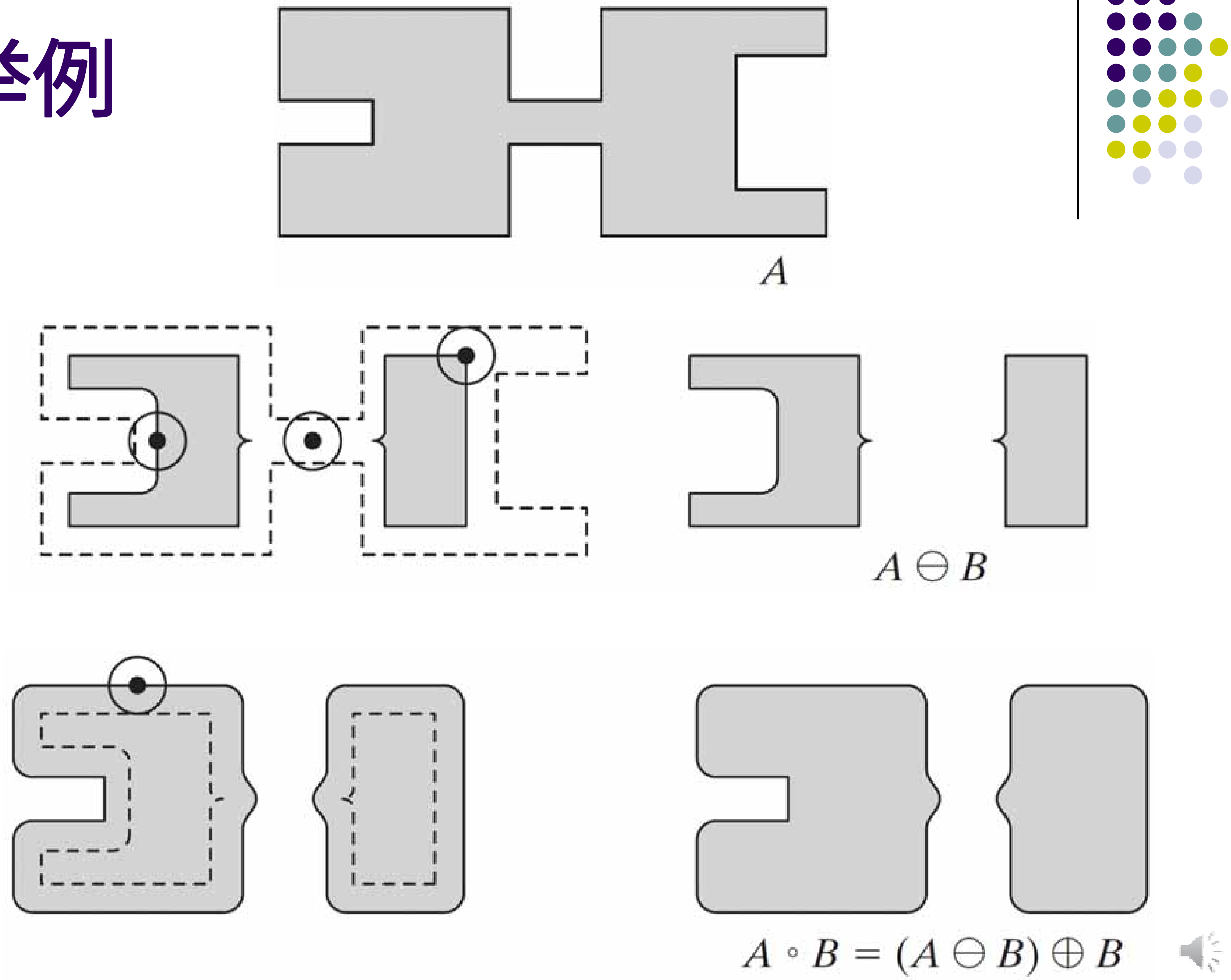
$$A \bullet B = (A \oplus B) \ominus B$$

  - 先用B膨胀A，然后再用B对结果进行腐蚀

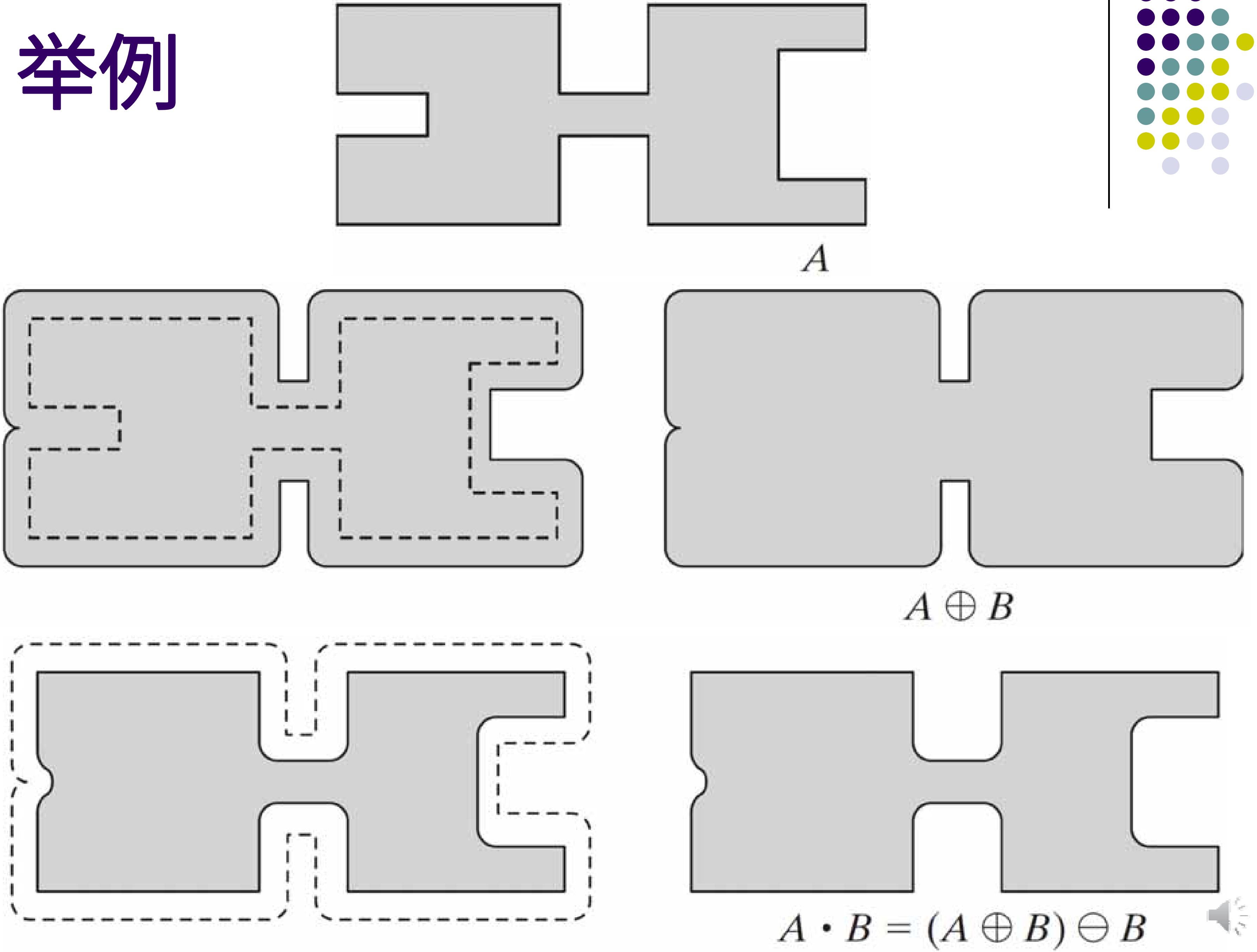
$$A \cdot B = \{w | w \in (B)_Z \Rightarrow (B)_Z \cap A \neq \emptyset\}$$



举例



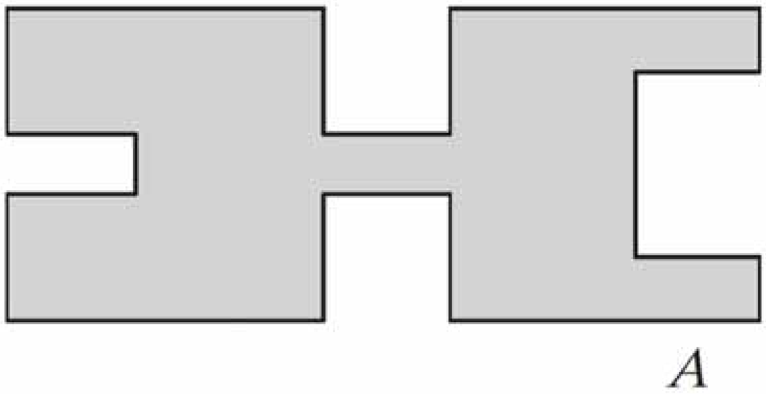
举例



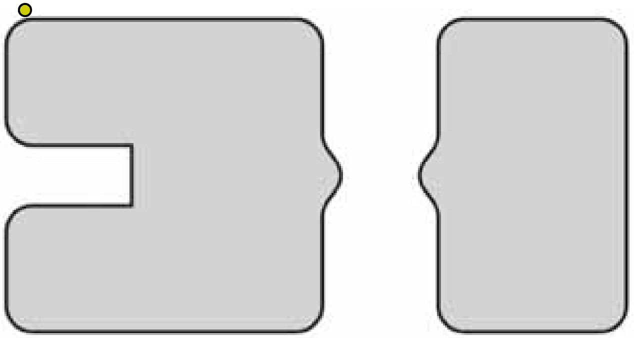
举例

- 对比

方向向外
的角变圆

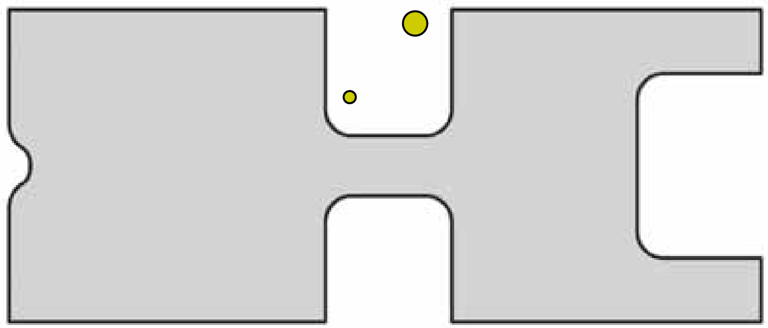


方向向内
的角变圆



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

开操作



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

闭操作

性质

- 对偶性

$$(A \bullet B)^c = (A^c \circ \hat{B}) \quad (A \circ B)^c = (A^c \bullet \hat{B})$$

- 开操作
  1. $A \circ B$是A的子集
  2. 如果C是D的子集，那么$C \circ B$是$D \circ B$的子集
  3. $(A \circ B) \circ B = A \circ B$
- 闭操作
  1. A是$A \cdot B$的子集
  2. 如果C是D的子集，那么$C \cdot B$是$D \cdot B$的子集
  3. $(A \cdot B) \cdot B = A \cdot B$

举例

- 去噪

结构元

1	1	1
1	1	1
1	1	1

B

1. 黑色背景中的白噪音被去除
2. 白色指纹中的黑噪声被加强

A

含噪声的指纹

$A \ominus B$

腐蚀

举例

- 去噪

1. 白色指纹中的黑噪声被削弱
2. 指纹纹路产生了断裂

1. 纹路中的大部分断裂被修复
2. 纹路变得更粗

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

开操作

$(A \circ B) \oplus B$

开操作的膨胀

举例

- 去噪

1. 纹路变细
2. 噪声被消除
3. 存在部分断裂

$$[(A \circ B) \oplus B] \ominus B = (A \circ B) \bullet B$$

开操作的闭操作

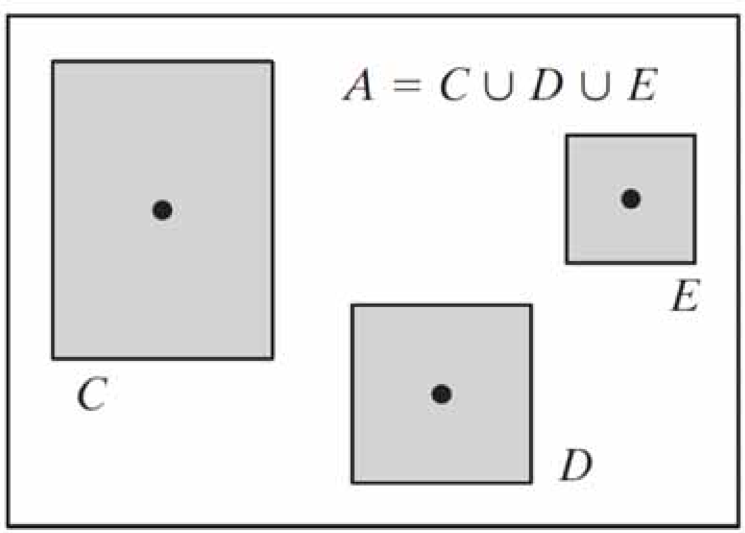
含噪声的指纹

提纲

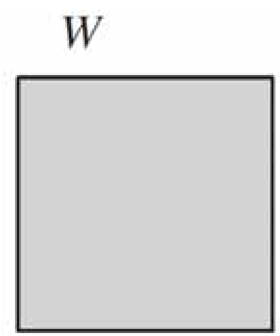
- 预备知识
- 腐蚀和膨胀
- 开操作和闭操作
- 击中或击不中变换
- 基本形态学算法
  - 边界提取、孔洞填充
  - 连通分量提取、凸包
  - 细化、粗化
  - 骨架、裁剪

击中或击不中变换

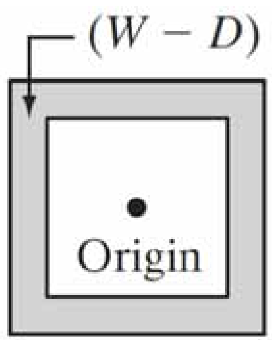
- 击中或击不中变换（hit-or-miss transform）
  - 用于检测图像中的形状
- 检测形状D



包含三个形状的集合A



包含D的小窗口W

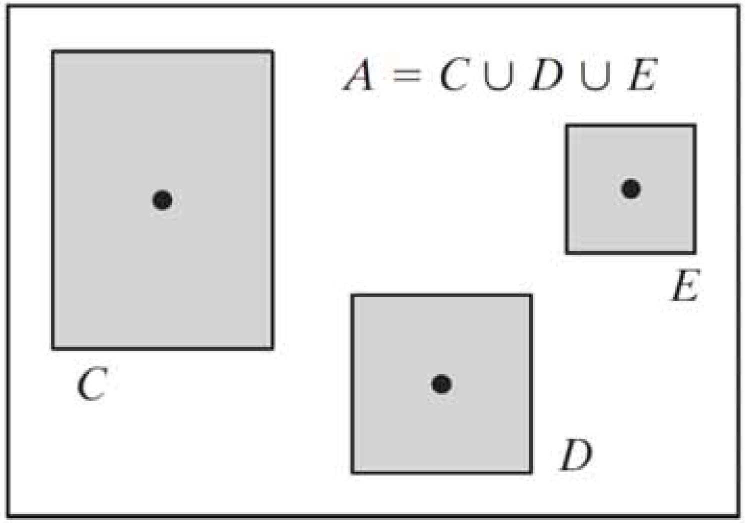


相对W而言，D的局部背景

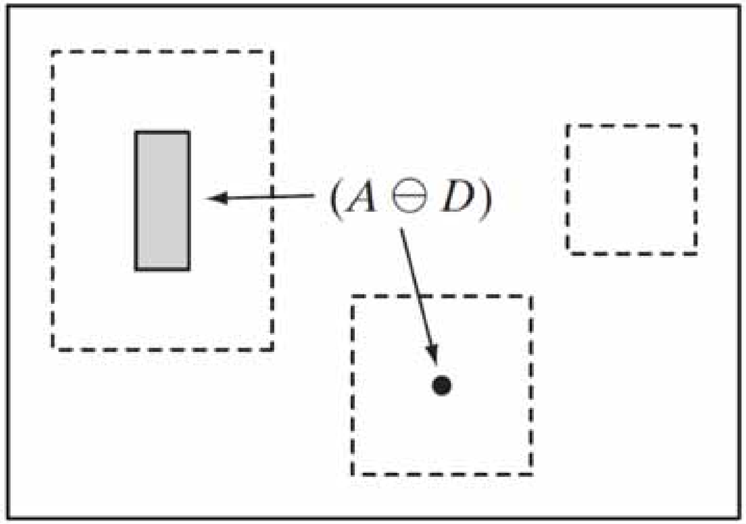
击中或击不中变换

- 击中或击不中变换（hit-or-miss transform）
  - 用于检测图像中的形状
- 检测形状D

表示D的匹配（击中）



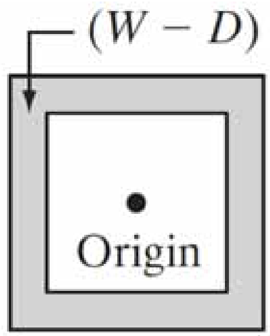
包含三个形状的集合A



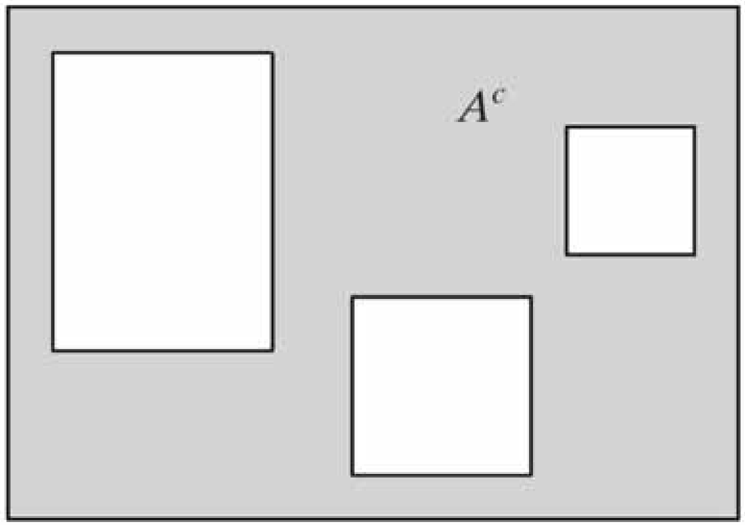
D对A的腐蚀

击中或击不中变换

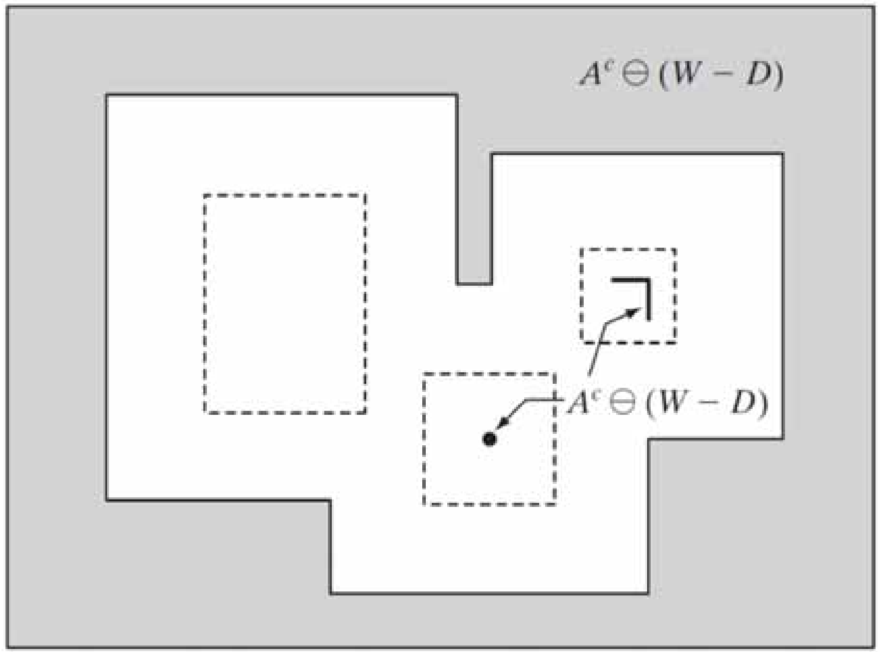
- 击中或击不中变换
  - 用于检测图像中的形状
- 检测形状D



相对W而言，D的局部背景



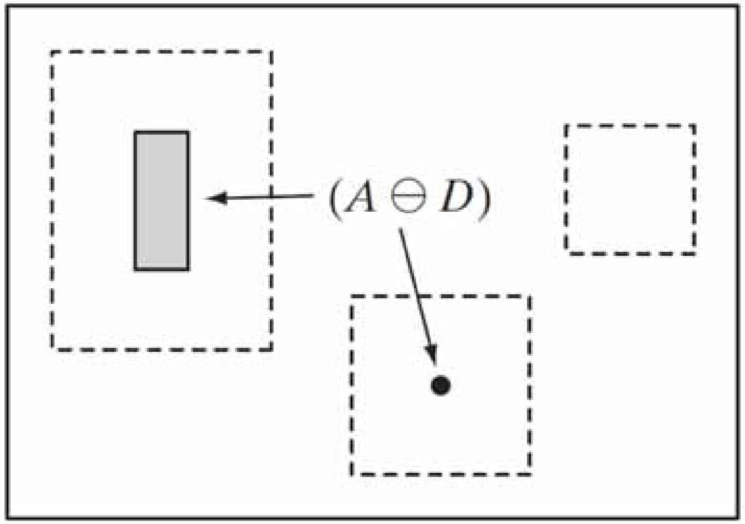
集合A的补集A^c



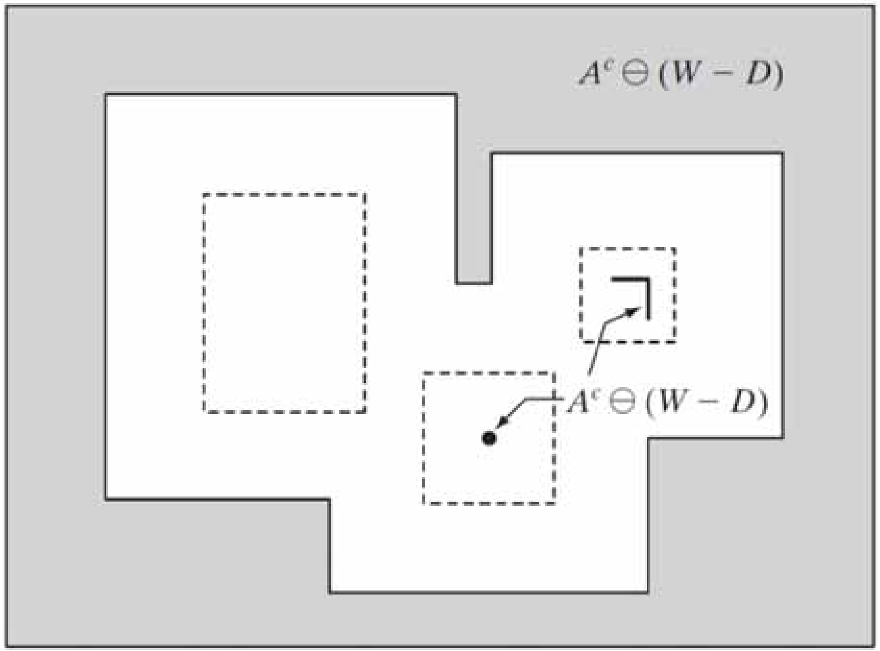
$W - D$对A^c的腐蚀

击中或击不中变换

- 击中或击不中变换（hit-or-miss transform）
  - 用于检测图像中的形状
- 检测形状D



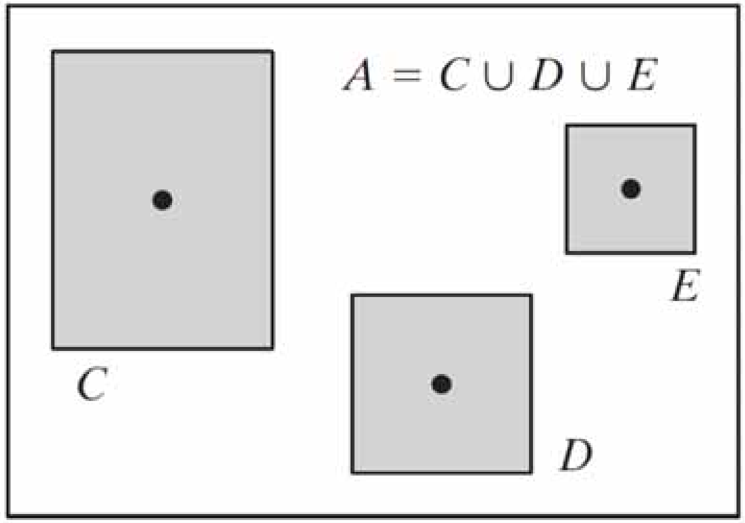
D对A的腐蚀



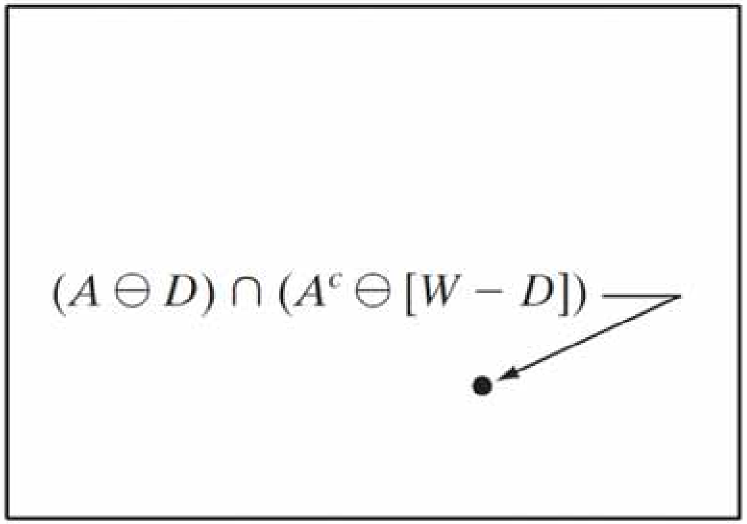
$W - D$对A^c的腐蚀

击中或击不中变换

- 击中或击不中变换（hit-or-miss transform）
  - 用于检测图像中的形状
- 检测形状D



包含三个形状的集合A



交集确定D的位置

击中或击不中变换

- 击中或击不中变换（hit-or-miss transform）
  - 用于检测图像中的形状
- 集合B在A中的匹配

$$A \circledast B = (A \ominus D) \cap \left[A^c \ominus (W - D)\right]$$

  - B表示集合D及其背景
  - 令$B = (B_1, B_2)$
    - $B_1 = D$表示物体，$B_2 = W - D$表示背景

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

    - B_1在A中匹配，B_2在A^c中匹配

- 背景使物体独立出现
- 无背景时变成腐蚀

击中或击不中变换

- 击中或击不中变换（hit-or-miss transform）
  - 用于检测图像中的形状
- 集合B在A中的匹配

$$A \circledast B = (A \ominus D) \cap \left[A^c \ominus (W - D) \right]$$

  - B表示集合D及其背景
  - 令$B = (B_1, B_2)$

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

  - 等价形式

$$A \circledast B = (A \ominus B_1) - (A \oplus \hat{B}_2)$$

提纲

- 预备知识
- 腐蚀和膨胀
- 开操作和闭操作
- 击中或击不中变换
- 基本形态学算法
  - 边界提取、孔洞填充
  - 连通分量提取、凸包
  - 细化、粗化
  - 骨架、裁剪

基本的形态学算法

- 提取表示区域形状的图像成分
  - 边界
  - 连通分量
  - 凸包
  - 骨架
- 配合上述算法的预处理或后处理
  - 区域填充
  - 细化、粗化
  - 裁剪

提纲

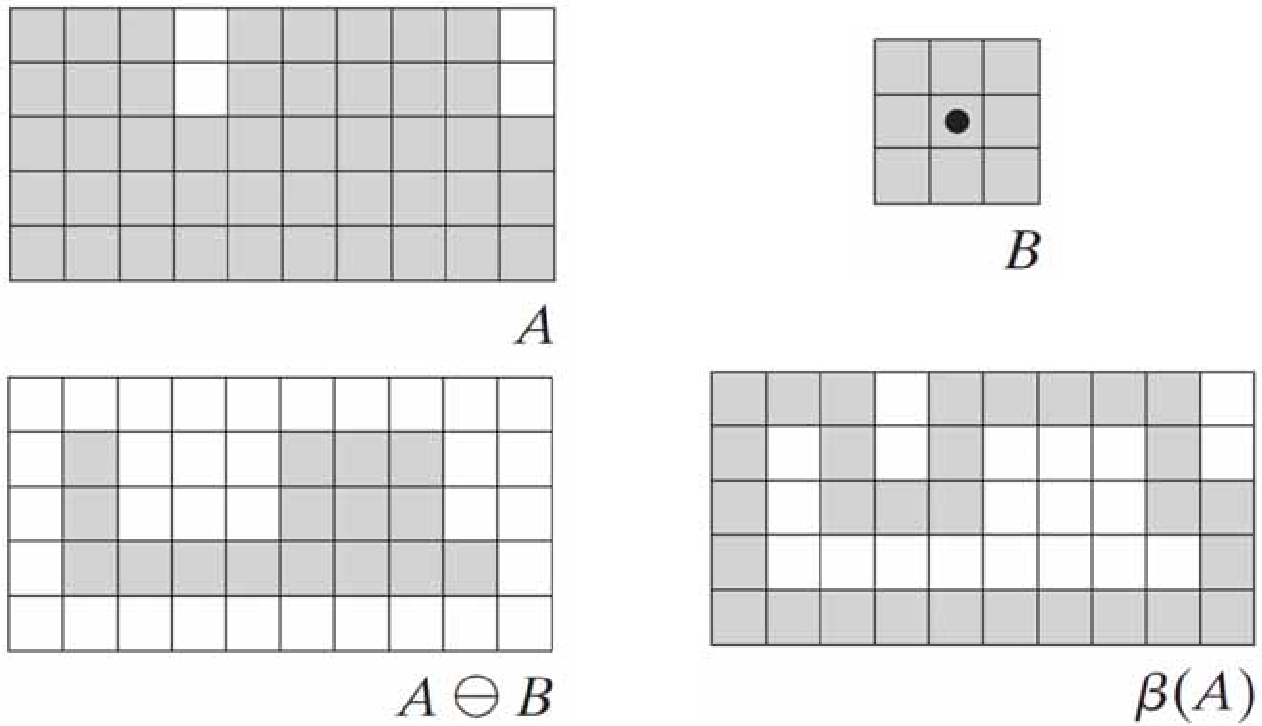
- 预备知识
- 腐蚀和膨胀
- 开操作和闭操作
- 击中或击不中变换
- 基本形态学算法
  - 边界提取、孔洞填充
  - 连通分量提取、凸包
  - 细化、粗化
  - 骨架、裁剪

边界提取

- 集合A的边界

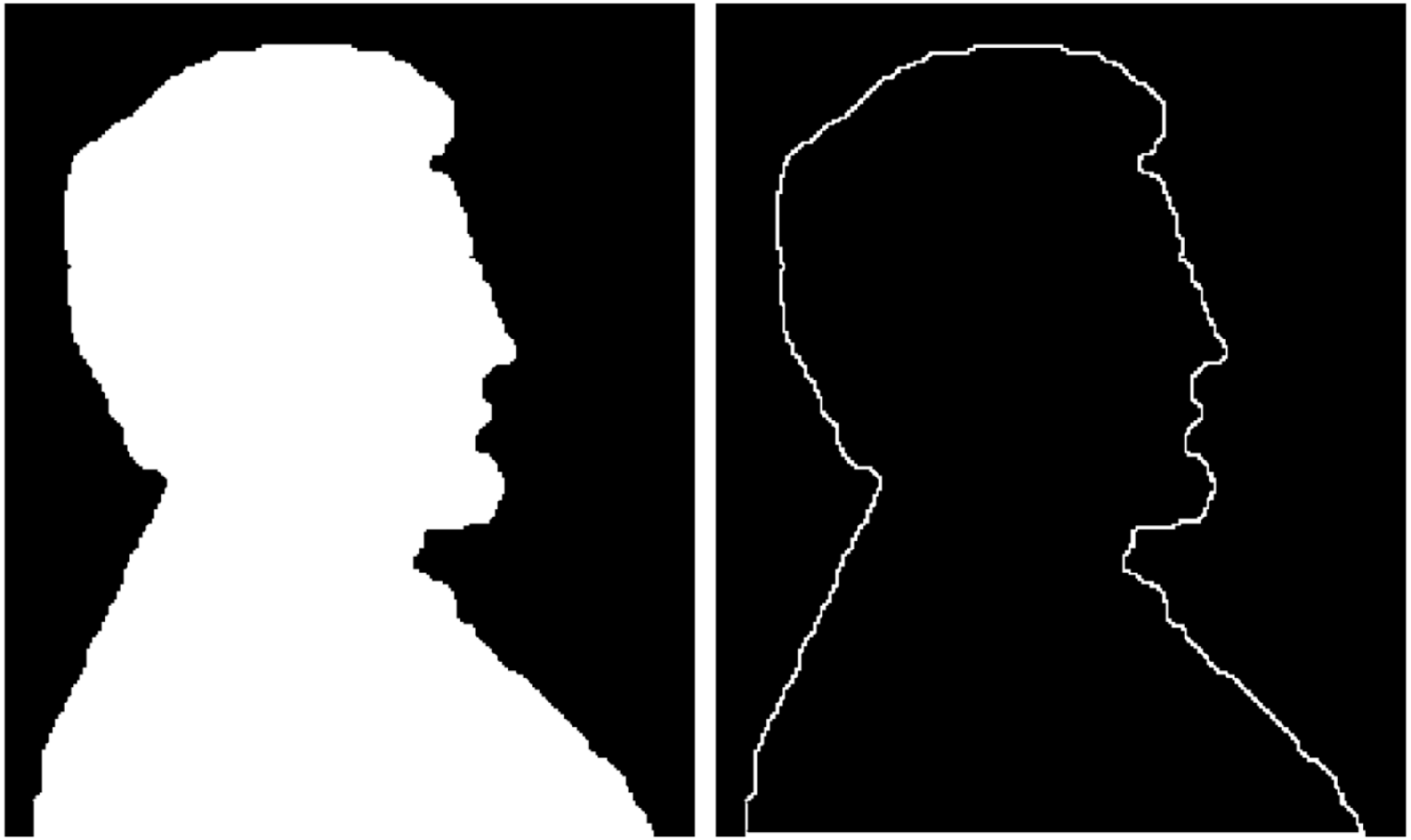
$$\beta(A) = A - (A \ominus B)$$

  - B是一个合适的结构元



举例

- 白色代表1



孔洞填充

- 孔洞（hole）
  - 由前景像素连成的边界包围的背景区域
- 孔洞填充
  - 利用膨胀、求补、交集等操作

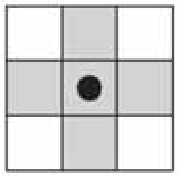
- A表示一个集合
  - 元素为8连通的边界
  - 每个边界包含一个孔洞（背景区域）
  - 给定每个孔洞内1个点

填充算法

1. 构造初始X_0
   - 给定的孔洞内初始点设为1，其他为0
2. 按照下面的公式更新

$$X_k = (X_{k-1} \oplus B) \cap A^c \quad k = 1, 2, 3, \ldots$$

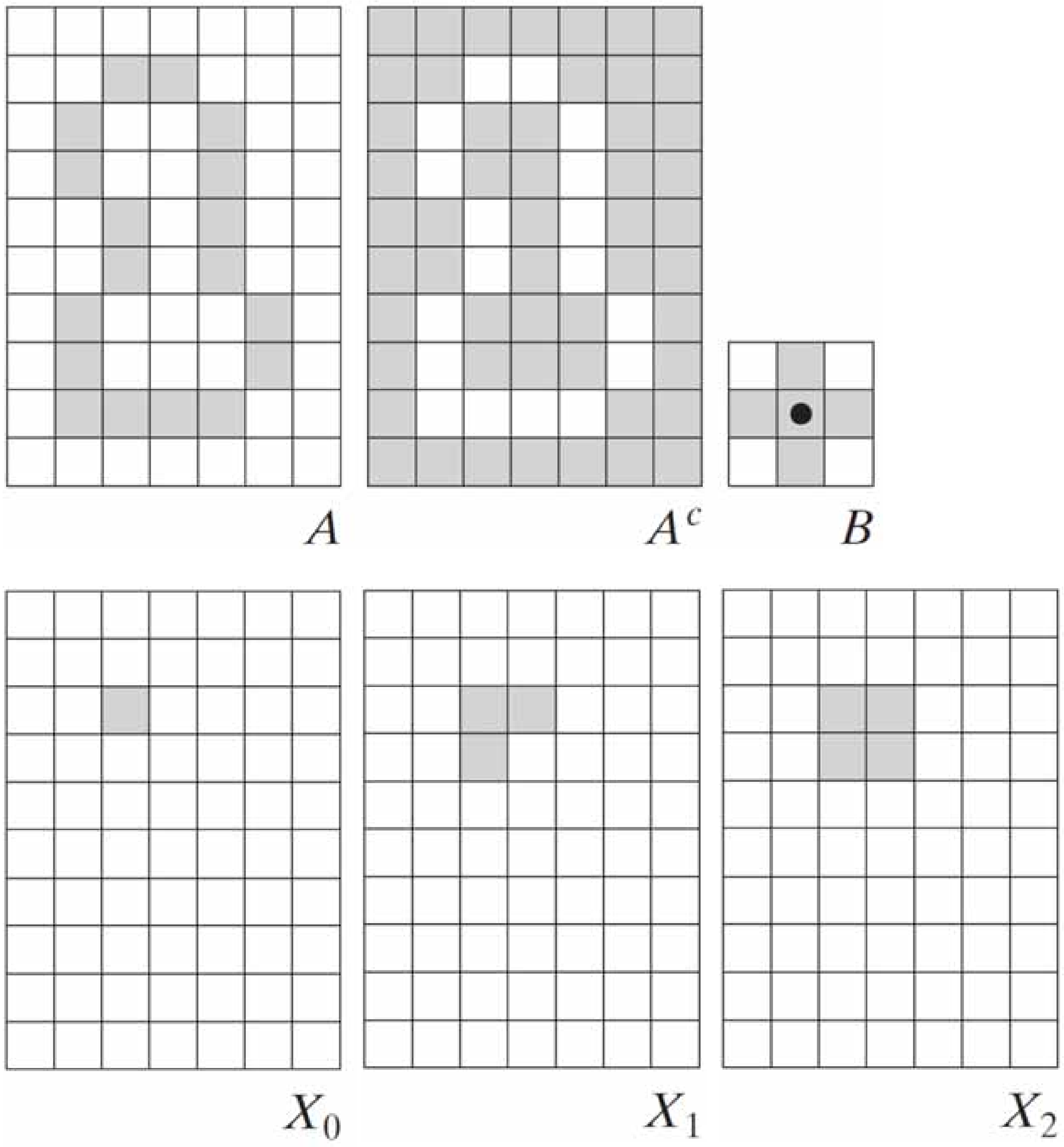
   - 其中B为结构元



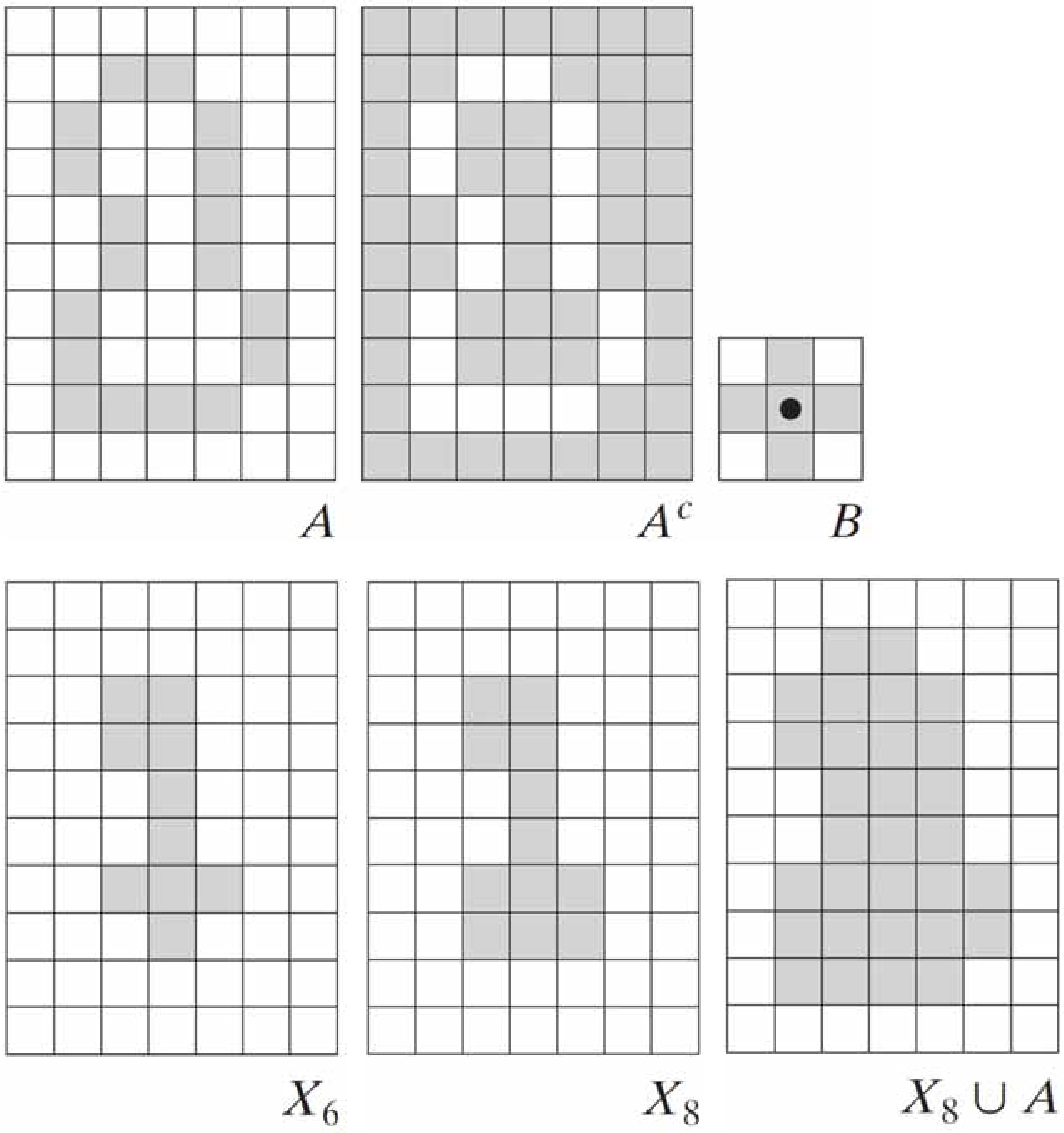
条件膨胀,
否则膨胀
会填充整
个空间

3. 重复上述公式，直到$X_k = X_{k-1}$
   - X_k包含填充后的孔洞
   - $A \cup X_k$为填充后的图像

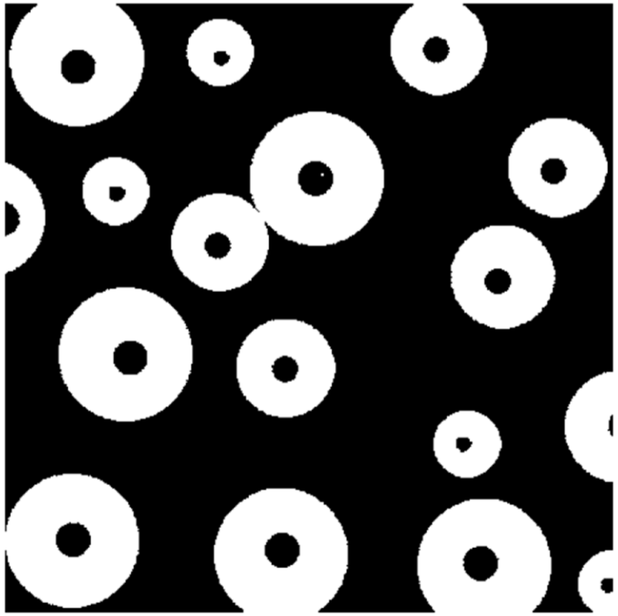
举例



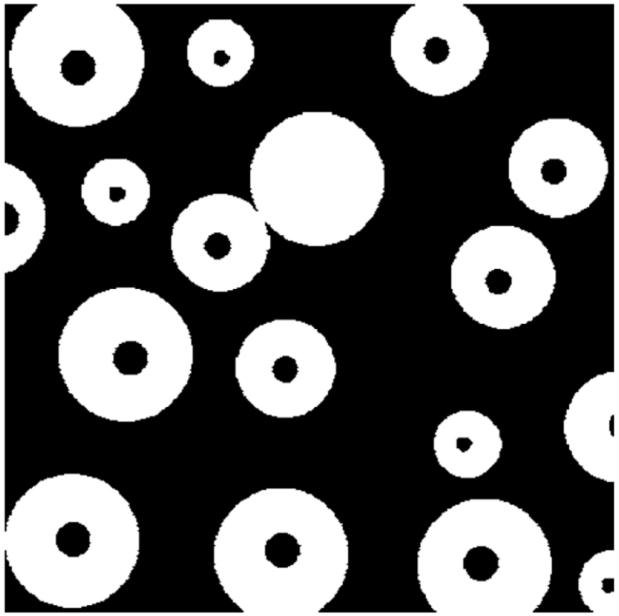
举例



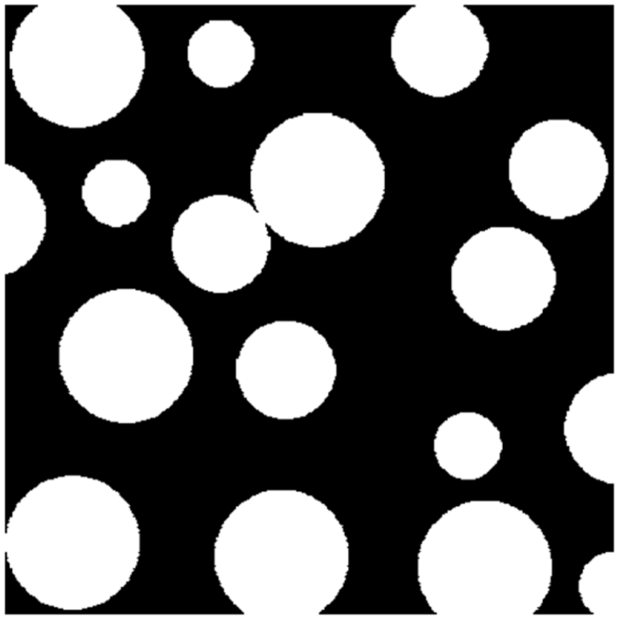
举例



包含一个初始点的原图



填充1个孔



全部填充

提纲

- 预备知识
- 腐蚀和膨胀
- 开操作和闭操作
- 击中或击不中变换
- 基本形态学算法
  - 边界提取、孔洞填充
  - 连通分量提取、凸包
  - 细化、粗化
  - 骨架、裁剪

邻接性

- 令V是用于定义邻接性的灰度值集合
  - 对于二值图像，$V = \{1\}$或$V = \{0\}$
  - 对于非二值图像，V是灰度级任意一个子集，比如$V = \{128,129, \dots, 255\}$

1. 4邻接（4-adjacency）
  - p和q的灰度值均属于集合V
  - q属于p的4邻域，即$q \in N_4(p)$

邻接性

2. 8邻接（8-adjacency）
- p和q的灰度值均属于集合V
- q属于p的8邻域，即$q \in N_8(p)$

3. m邻接（m-adjacency）

消除歧义

- p和q的灰度值均属于集合V

a) q属于p的4邻域，即$q \in N_4(p)$

a‘) q属于p的4对角邻域，即$q \in N_D(p)$，并且$N_4(p) \cap N_4(q)$中没有元素的灰度属于V

连通分量提取

- 连通分量
  - 连接在一起的像素集合
    - 4邻接、8邻接、m邻接

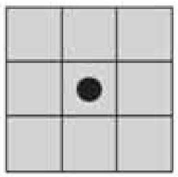
- A表示一个集合
  - 元素为若干连通分量
  - 给定每个连通分量内1个点

连通分量提取算法

1. 构造初始X_0
   - 给定的连通分量内初始点设为1，其他为0
2. 按照下面的公式更新

$$X_k = (X_{k-1} \oplus B) \cap A \quad , \quad k = 1, 2, 3, \ldots$$

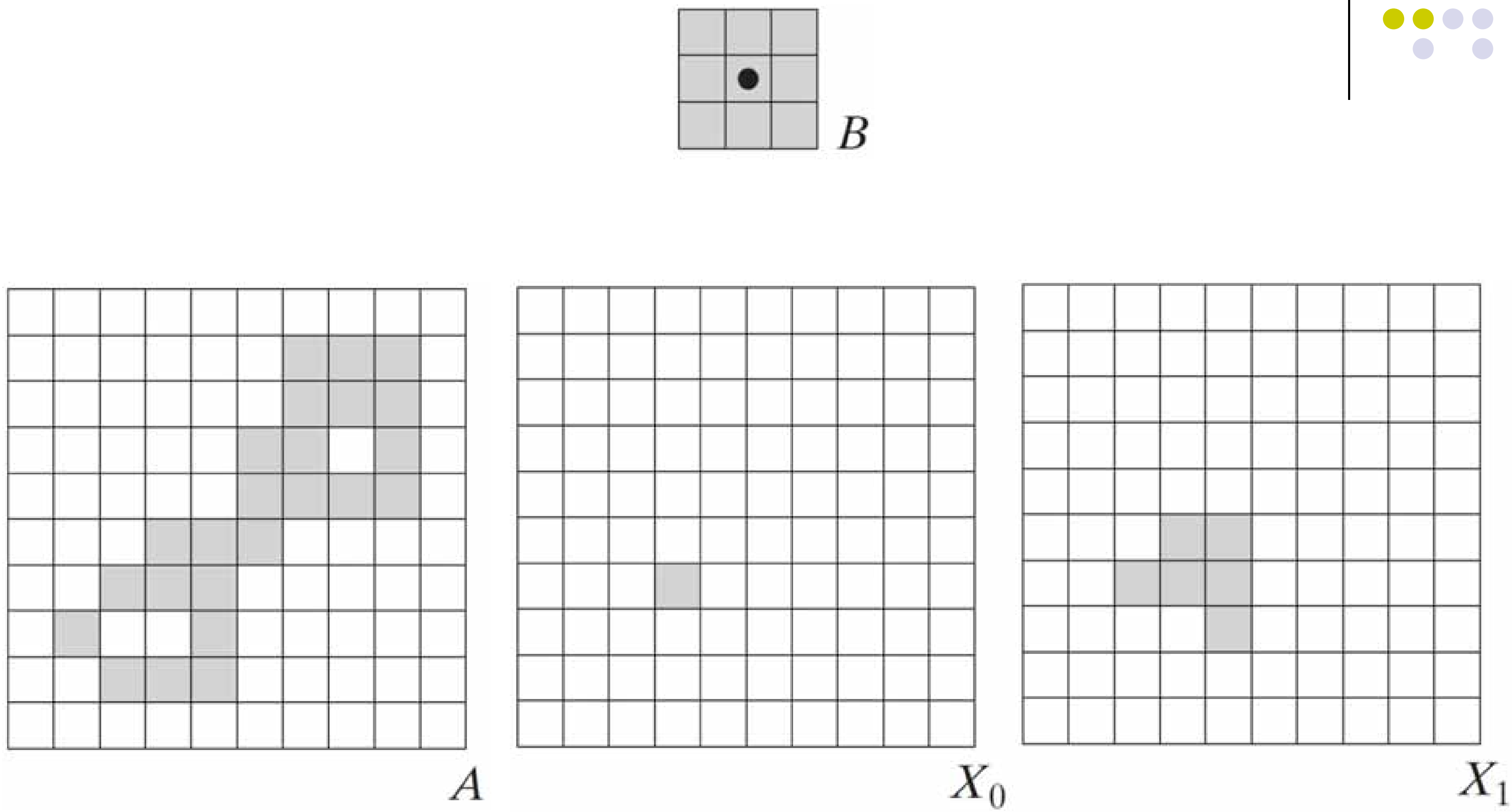
   - 其中B为结构元
     - 考虑8连通



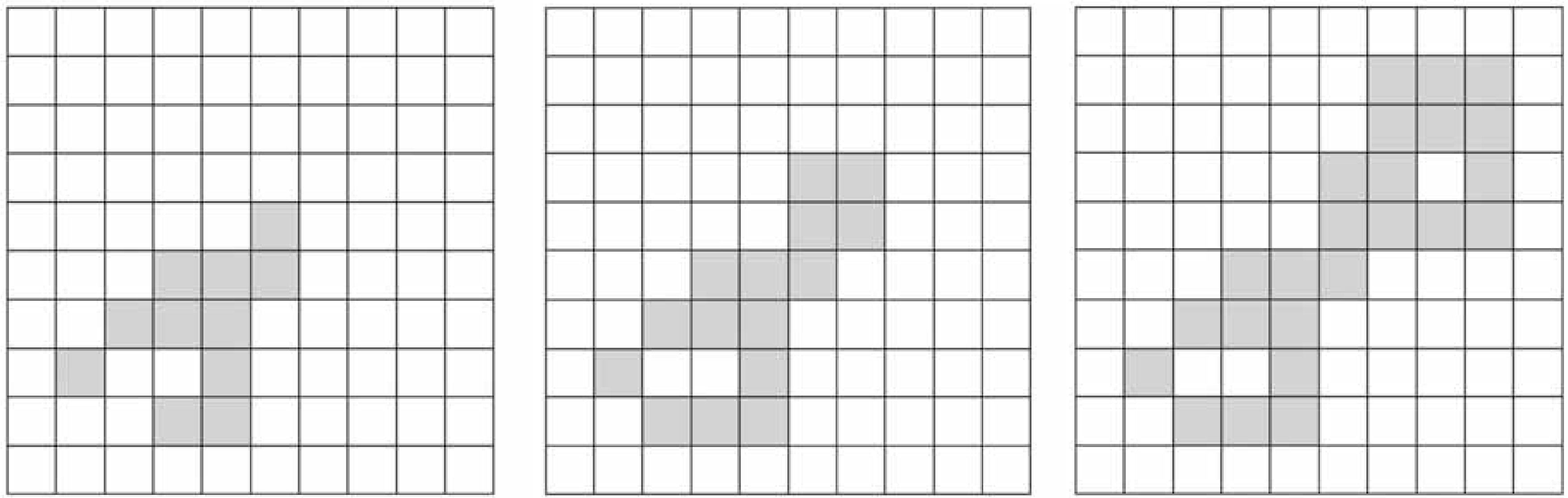
条件膨胀，否则膨胀会填充整个空间

3. 重复上述公式，直到$X_k = X_{k-1}$
   - X_k包含提取的连通分量

举例



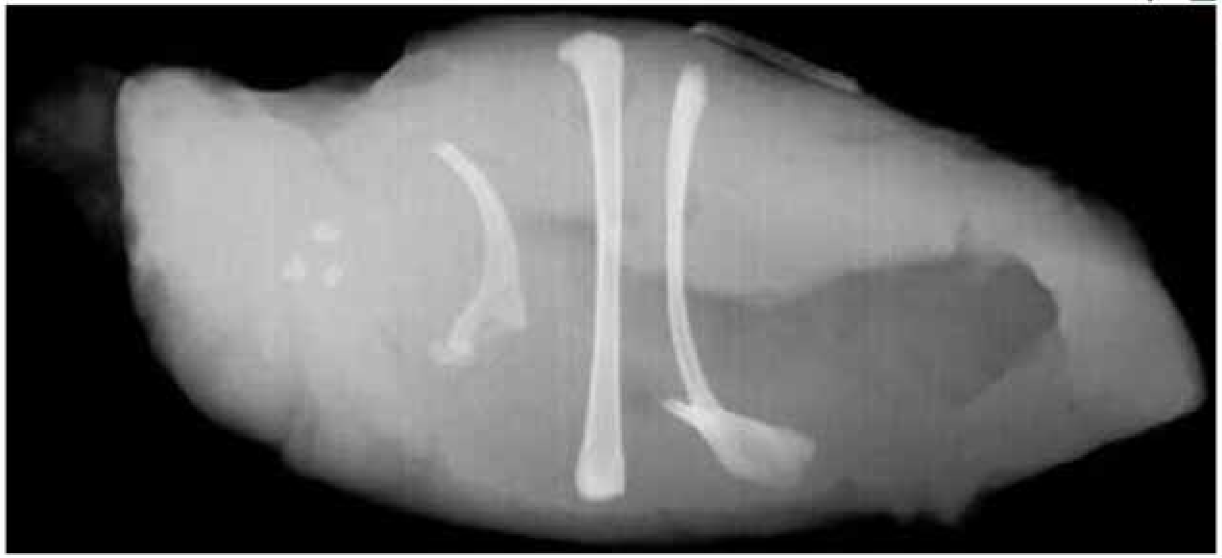
举例



X_2　X_3　X_6

举例

包含碎骨头
的鸡胸肉



阈值化

举例

4个尺寸较大，
证明有碎骨头

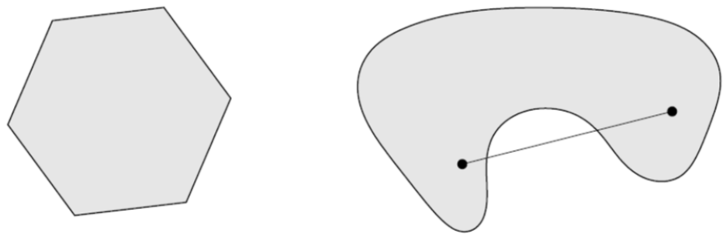
利用5 × 5结构元腐蚀（去掉小区域）

Connected component	No. of pixels in connected comp
01	11
02	9
03	9
04	39
05	133
06	1
07	1
08	743
09	7
10	11
11	11
12	9
13	9
14	674
15	85

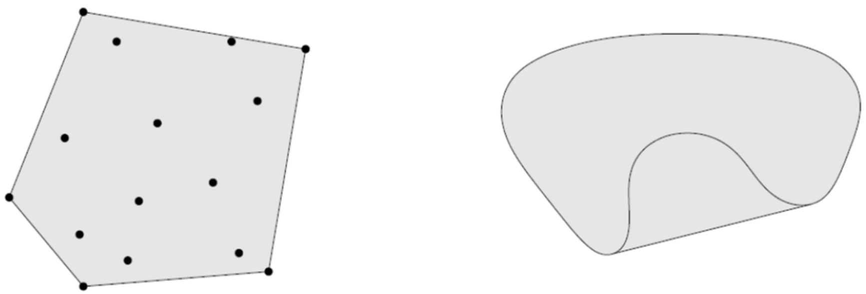
连通分量提取

凸包

- 凸集合（convex set）
  - 集合内任意两点的连线属于该集合



- 集合S的凸包（convex hull）H
  - 包含S的最小凸集合

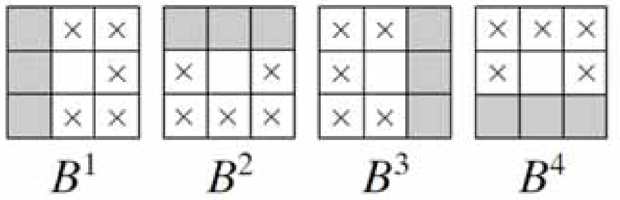


- 凸缺（convex deficiency）：$H - S$

构建凸包算法

相差90°

- 四个结构元
  - 黑色表示1
  - 白色表示0，×表示任意值



$\circledast$不用考虑外围背景

1. 按照下面的公式更新

$$X_k^i = (X_{k-1}^i \circledast B^i) \cup A \quad i = 1, 2, 3, 4 \quad \text{and} \quad k = 1, 2, 3, \ldots$$

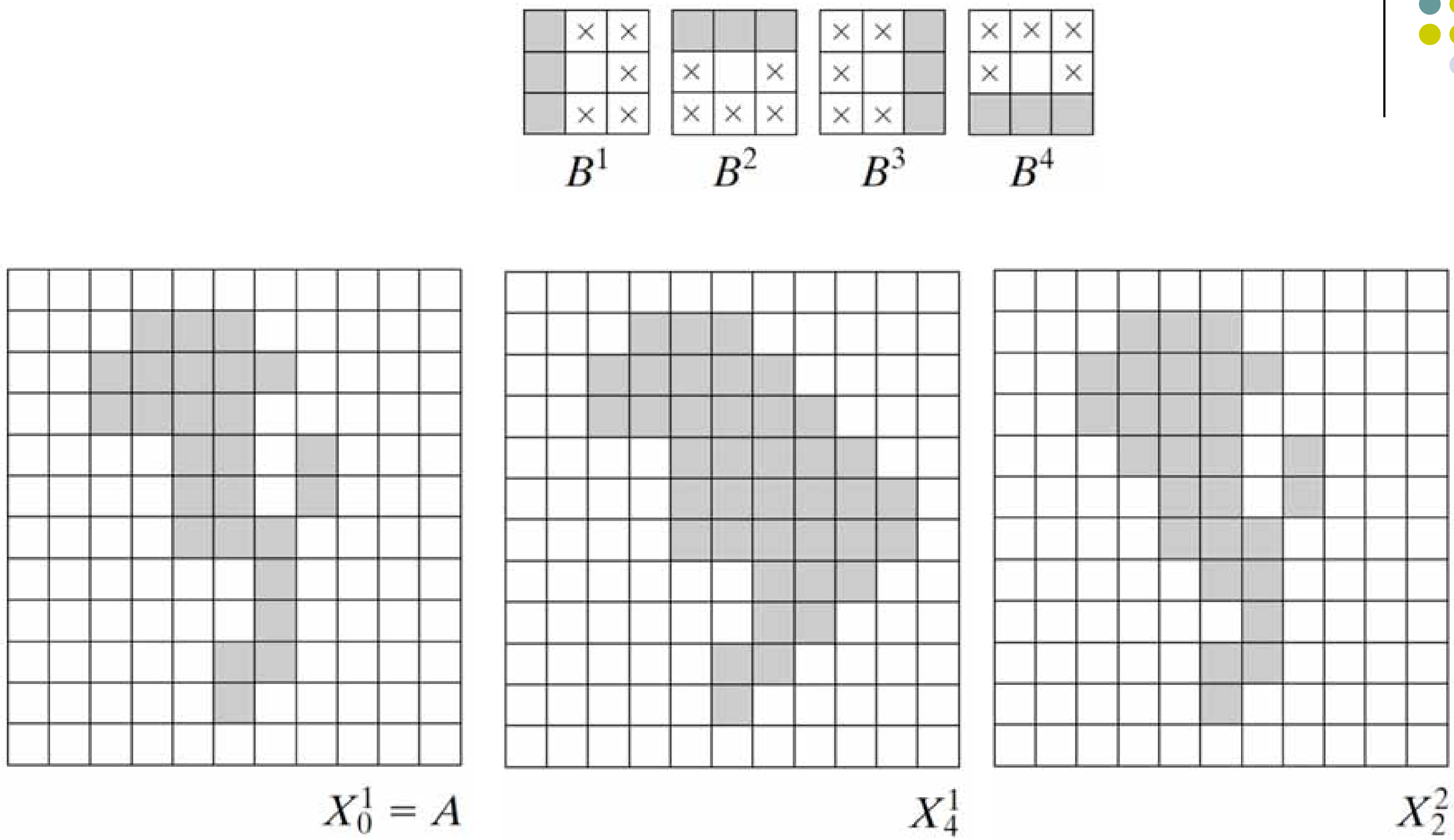
   - 其中$X_0^i = A$

2. 重复上述公式，直到$X_k^i = X_{k-1}^i$
3. 集合A的凸包

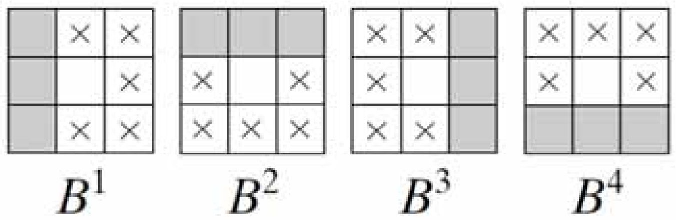
$$C(A) = \bigcup_{i=1}^{4} D^i$$

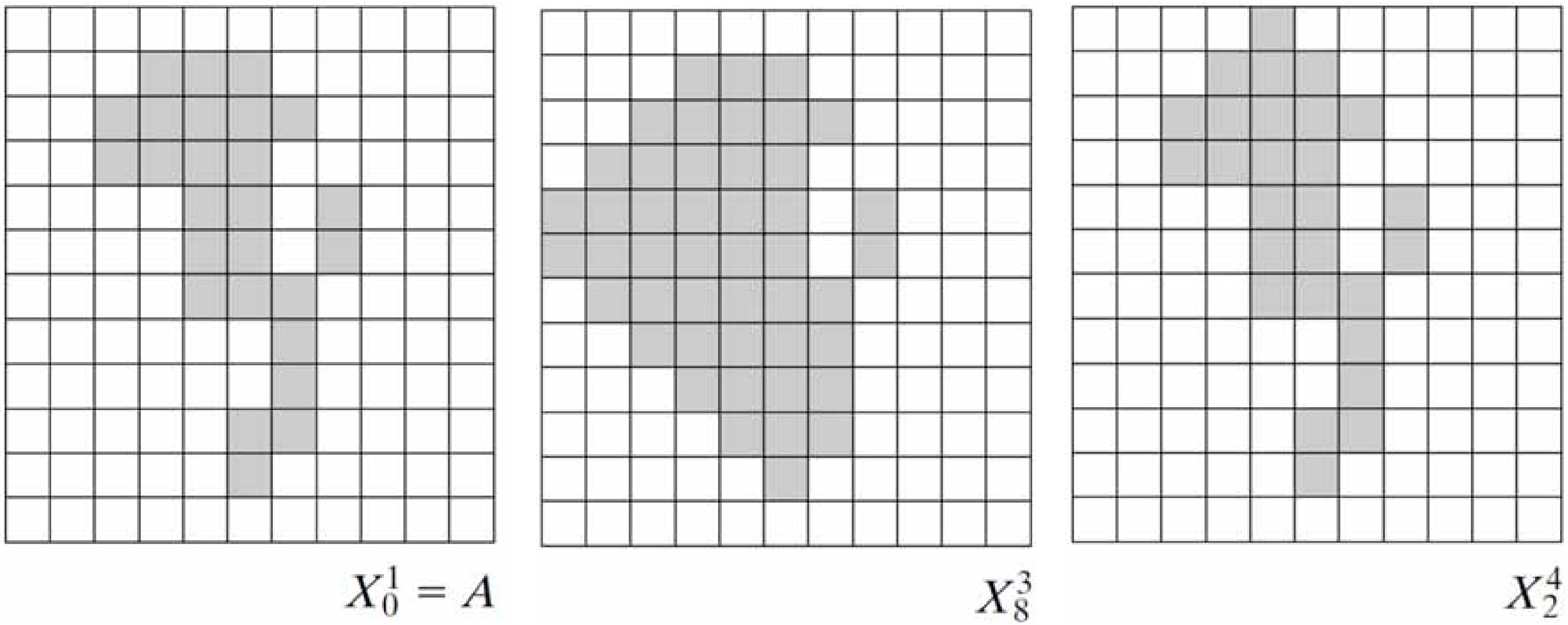
   - 其中$D^i = X_k^i$

举例

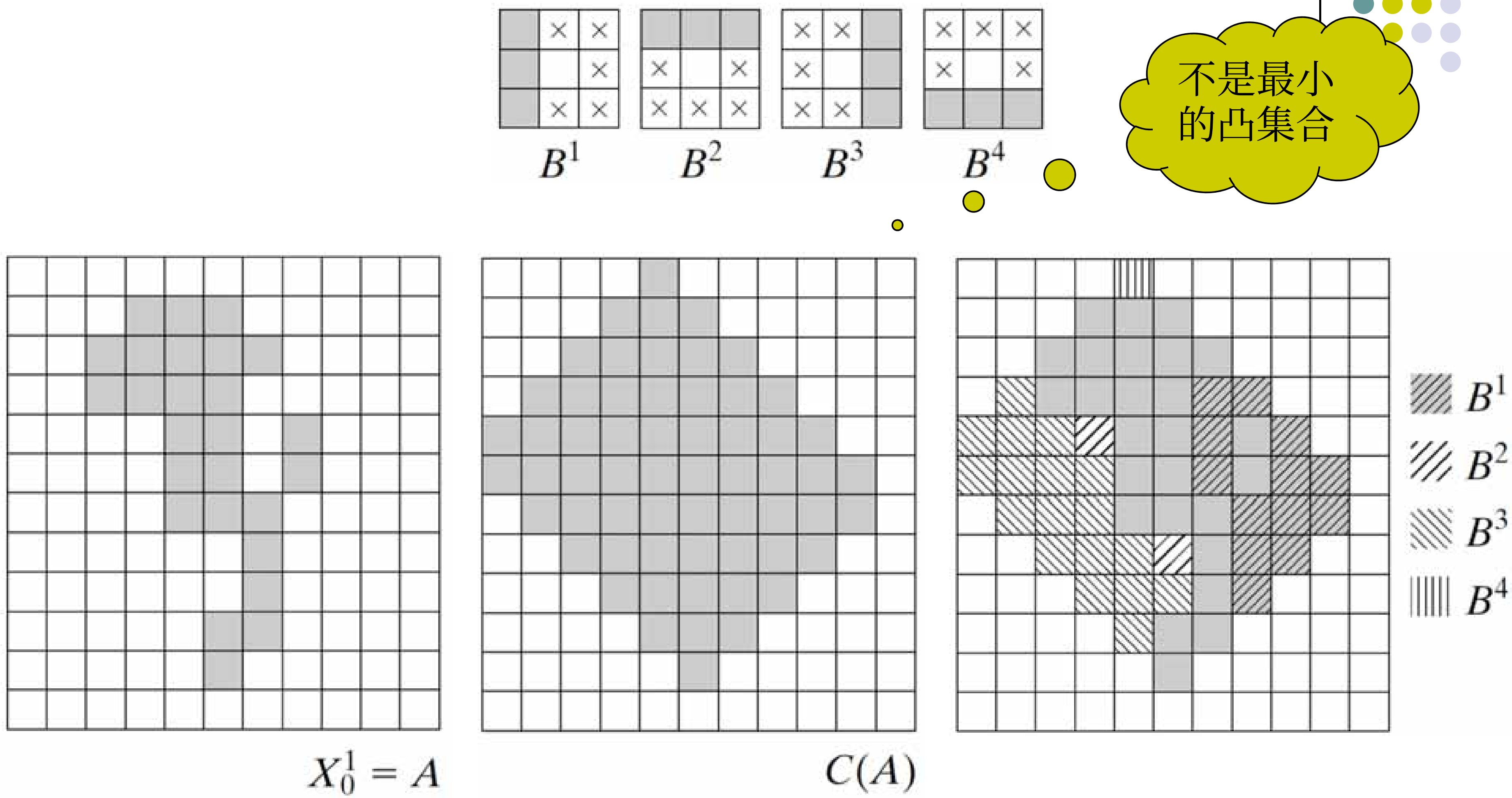


举例



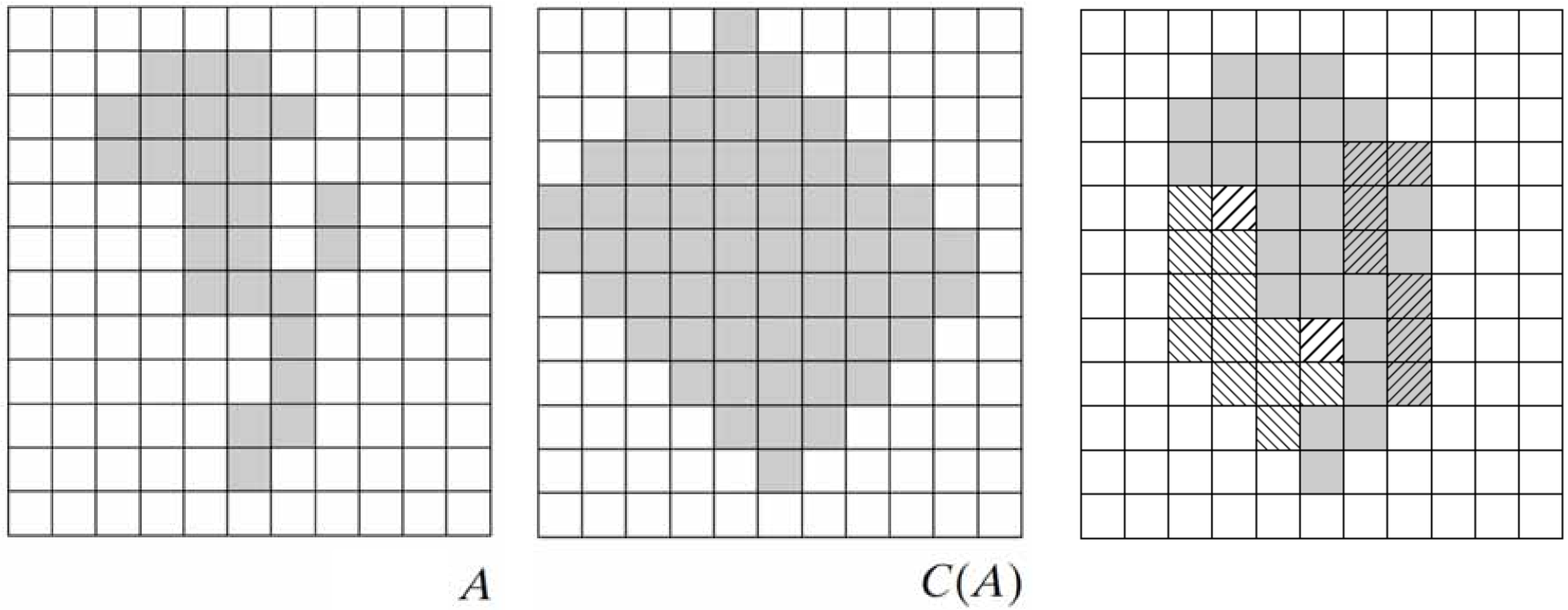


举例



举例

- 不能超过原图像的垂直和水平范围



- 还可以添加更复杂的约束

提纲

- 预备知识
- 腐蚀和膨胀
- 开操作和闭操作
- 击中或击不中变换
- 基本形态学算法
  - 边界提取、孔洞填充
  - 连通分量提取、凸包
  - 细化、粗化
  - 骨架、裁剪

细化

- 结构元B对集合A的细化（thinning）

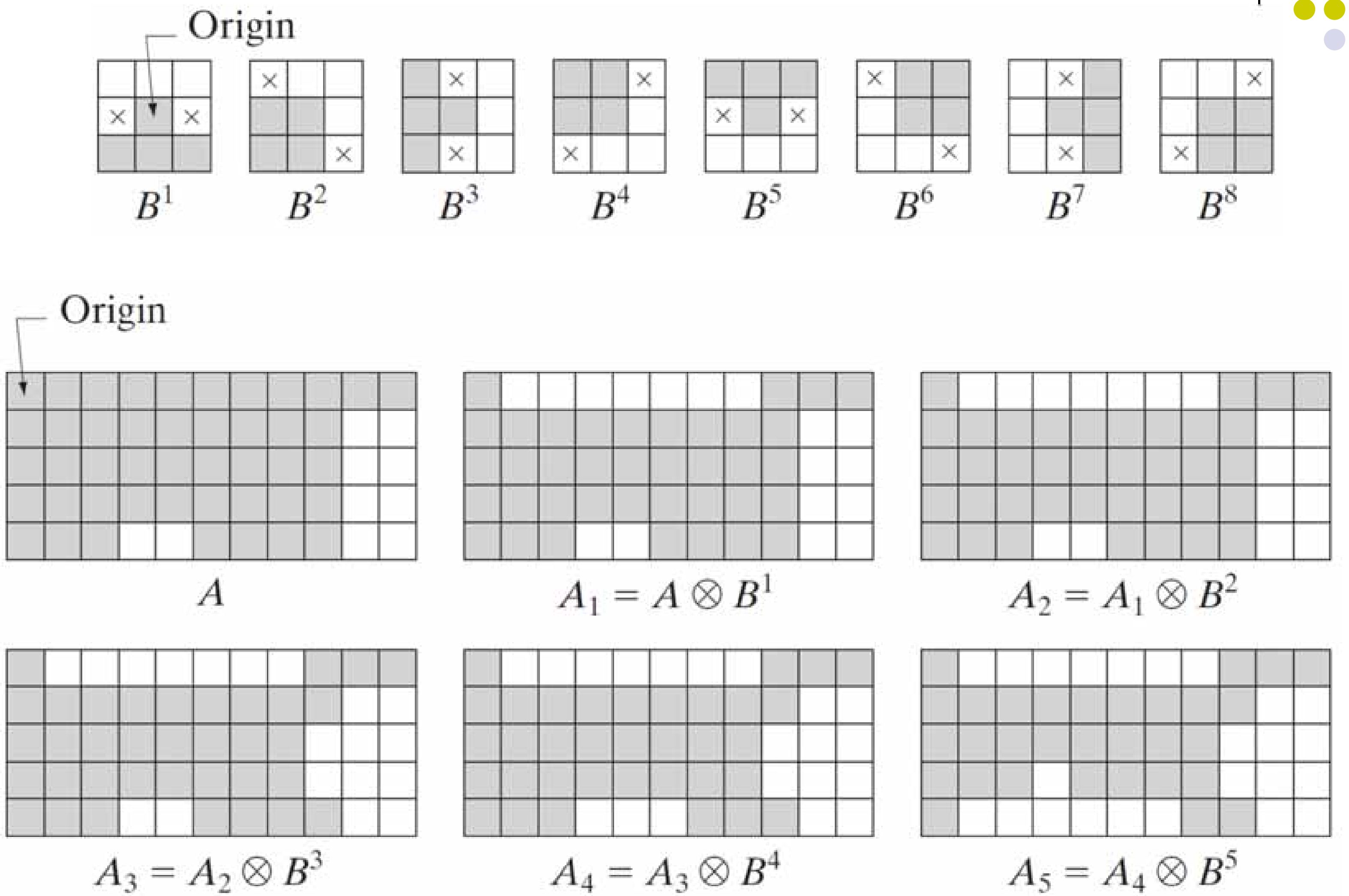
$$A \otimes B = A - (A \circledast B) = A \cap (A \circledast B)^c$$

  - $\circledast$不用考虑外围背景
  - 与边界提取很类似
- 结构元序列$\{B\} = \{B^1, B^2, \ldots, B^n\}$对集合$A$的细化

$$A \otimes \{B\} = ((\ldots((A \otimes B^1) \otimes B^2) \ldots) \otimes B^n)$$

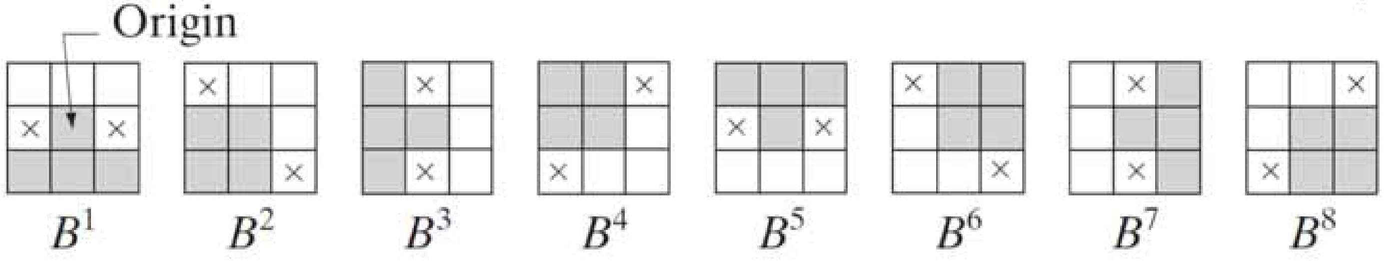
  - B^i是B^{i-1}的旋转版本
  - 重复上述过程，直至结果不发生变化

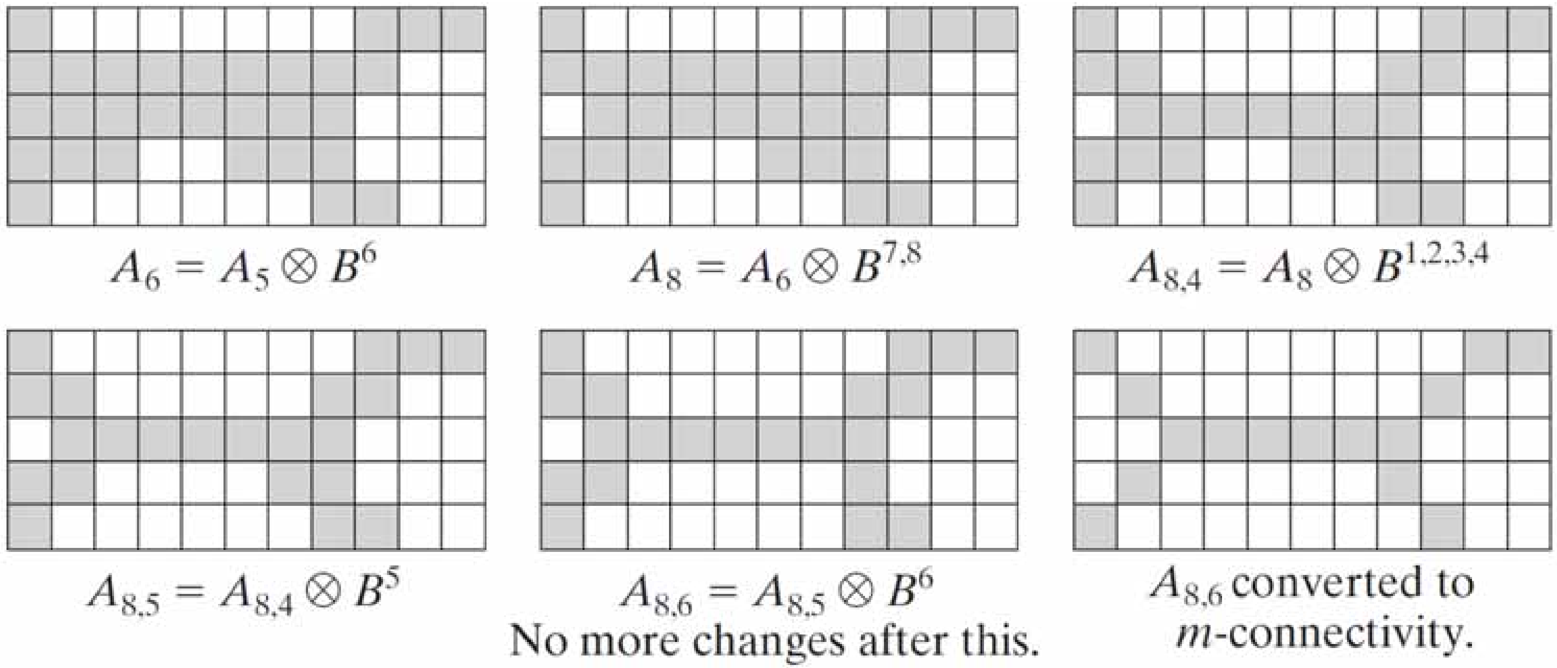
举例



Origin
B^1
B^2
B^3
B^4
B^5
B^6
B^7
B^8
Origin
A
$A_1 = A \otimes B^1$
$A_2 = A_1 \otimes B^2$
$A_3 = A_2 \otimes B^3$
$A_4 = A_3 \otimes B^4$
$A_5 = A_4 \otimes B^5$

举例


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


$A_6 = A_5 \otimes B^6$
$A_8 = A_6 \otimes B^{7,8}$
$A_{8,4} = A_8 \otimes B^{1,2,3,4}$
$A_{8,5} = A_{8,4} \otimes B^5$
$A_{8,6} = A_{8,5} \otimes B^6$
No more changes after this.
$A_{8,6}$ converted to
m-connectivity.

粗化

- 结构元B对集合A的粗化（thickening）

$$A \odot B = A \cup (A \circledast B)$$

  - $\circledast$不用考虑外围背景
  - 结构元和细化的相反
- 结构元序列$\{B\} = \{B^1, B^2, \ldots, B^n\}$对集合$A$的粗化

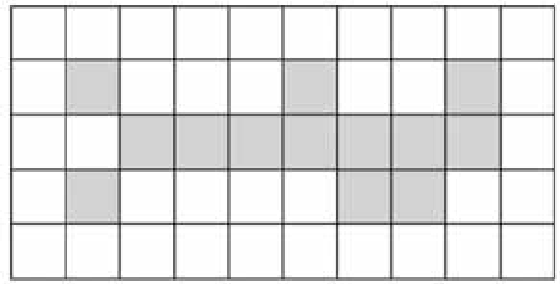
$$A \odot \{B\} = ((\ldots((A \odot B^1) \odot B^2) \ldots) \odot B^n)$$

  - B^i是B^{i-1}的旋转版本
  - 重复上述过程，直至结果不发生变化

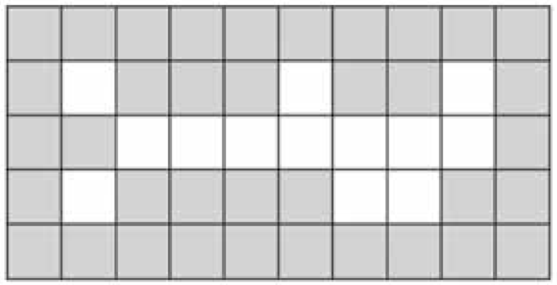
粗化算法

1. 计算集合A的补集C
2. 细化C
3. 计算上述结果的补集

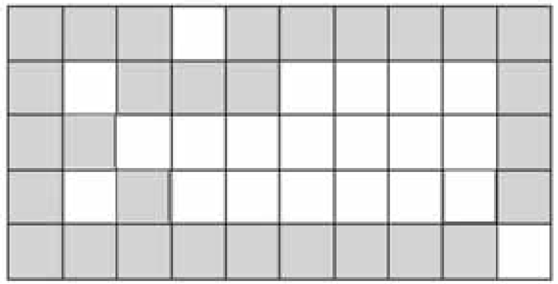
A



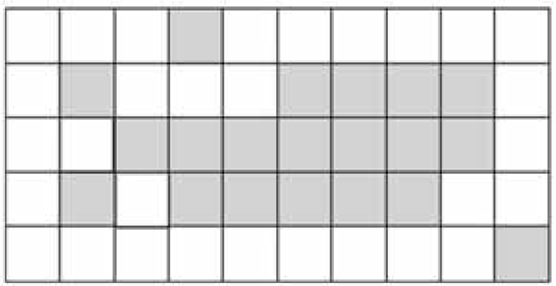
$C = A^c$

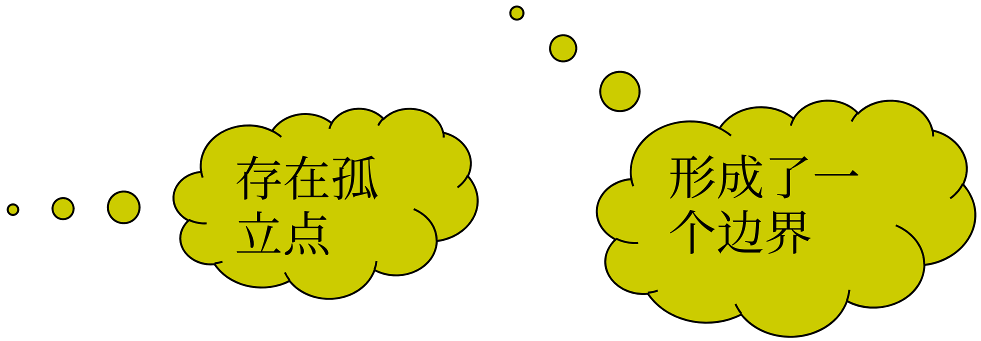


细化



求补

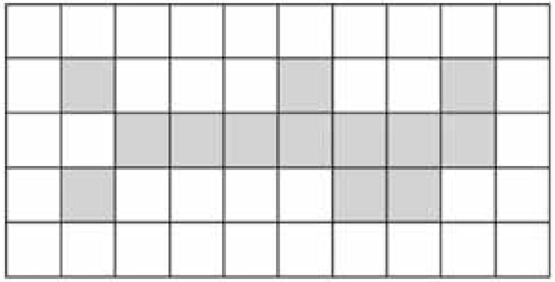




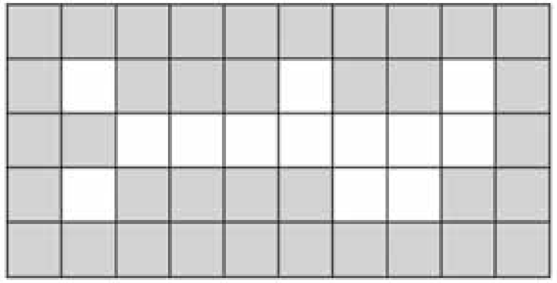
粗化算法

1. 计算集合A的补集C
2. 细化C
3. 计算上述结果的补集

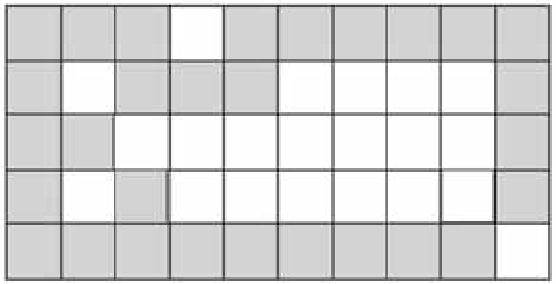
A



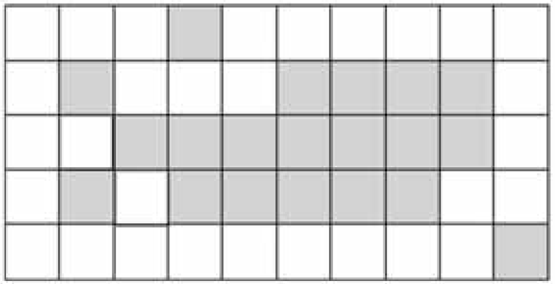
$C = A^c$



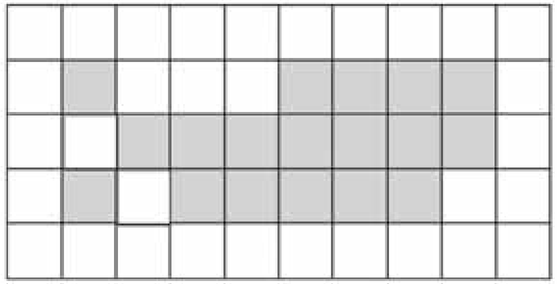
细化



求补



去掉孤立点

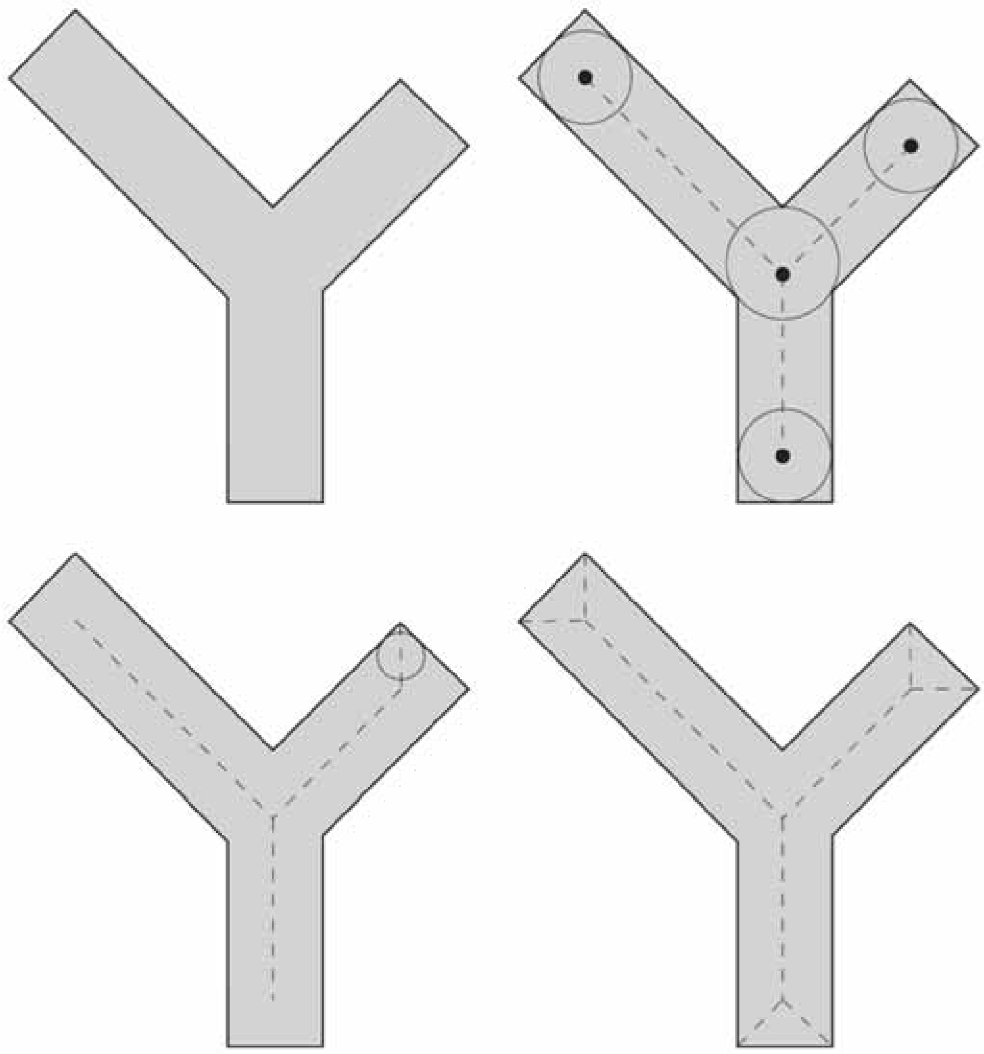


提纲

- 预备知识
- 腐蚀和膨胀
- 开操作和闭操作
- 击中或击不中变换
- 基本形态学算法
  - 边界提取、孔洞填充
  - 连通分量提取、凸包
  - 细化、粗化
  - 骨架、裁剪

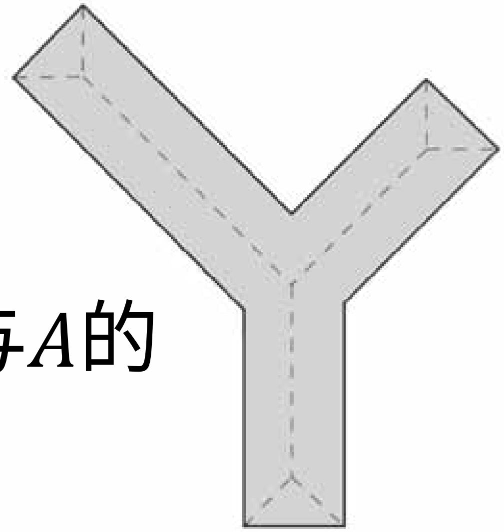
骨架

- 举例



骨架

- 集合A的骨架（skeleton）记为$S(A)$
  1. 如果$z \in S(A)$，并且$(D)_z$是A内以z为中心的最大圆盘，则不存在包含$(D)_z$且位于A内的更大圆盘。
     - $(D)_z$被称为最大圆盘
  2. $(D)_z$在两个或多个不同的位置与A的边界接触。



骨架

- 数学公式

$$S(A) = \bigcup_{k=0}^{K} S_k(A)$$

$$S_k(A) = (A \ominus kB) - (A \ominus kB) \circ B$$

  - $S_k(A)$是骨架子集
  - B是结构元
  - $A \ominus kB$表示对A进行k次连续腐蚀

$$(A \ominus kB) = ((\ldots((A \ominus B) \ominus B) \ominus \ldots) \ominus B)$$

  - K是A被腐蚀成空集的最后一次迭代

$$K = \max\{k | (A \ominus kB) \neq \varnothing\}$$

骨架

- 数学公式

$$S(A) = \bigcup_{k=0}^{K} S_k(A)$$

$$S_k(A) = (A \ominus kB) - (A \ominus kB) \circ B$$

$$(A \ominus kB) = ((\ldots((A \ominus B) \ominus B) \ominus \ldots) \ominus B)$$

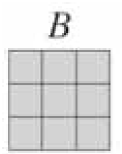
$$K = \max\{k \mid (A \ominus kB) \neq \varnothing\}$$

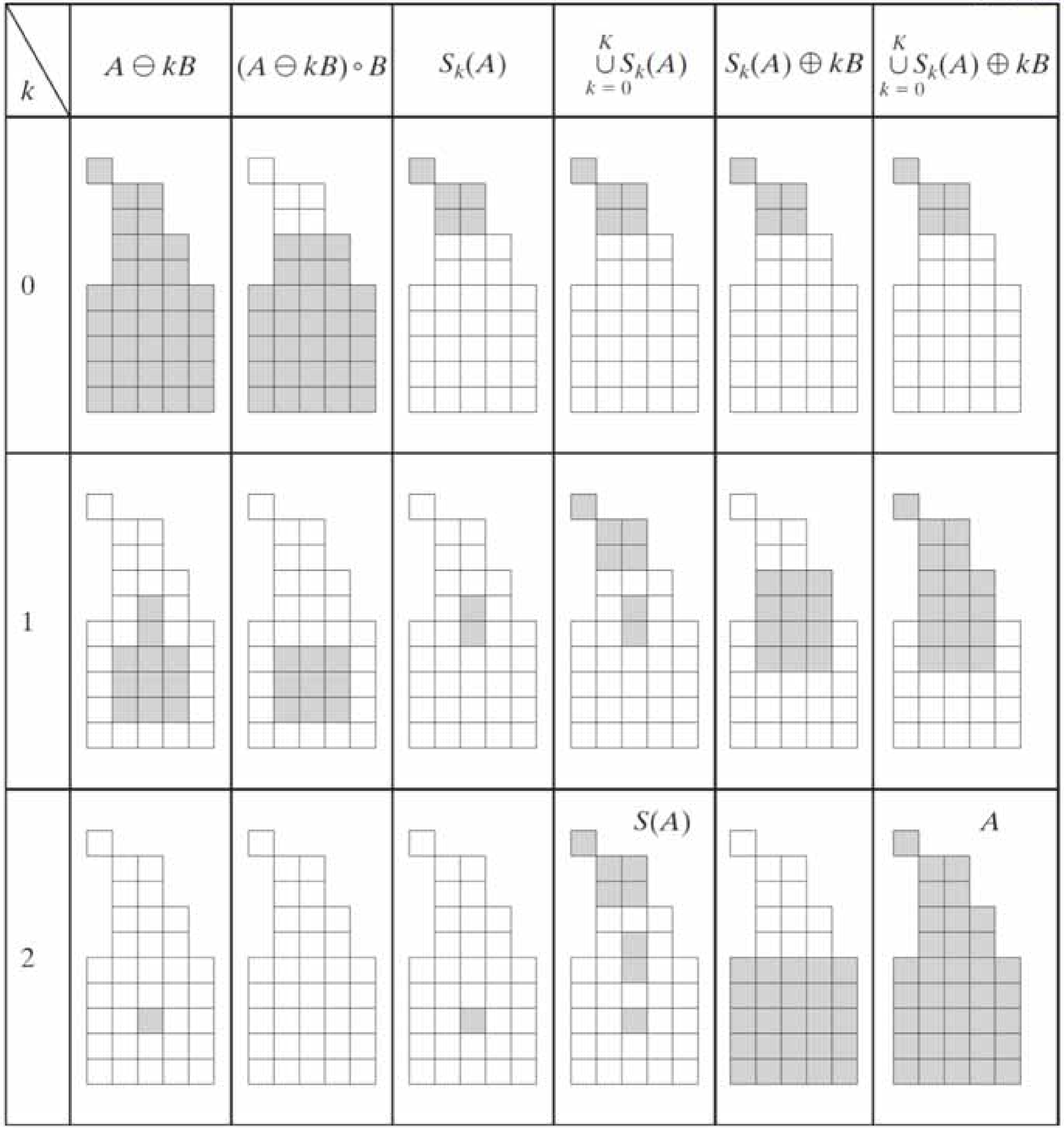
- 重构集合A

$$A = \bigcup_{k=0}^{K} (S_k(A) \oplus kB)$$

$$(S_k(A) \oplus kB) = ((\ldots((S_k(A) \oplus B) \oplus B) \oplus \ldots) \oplus B)$$

举例


B


k
$A \ominus kB$
$(A \ominus kB) \circ B$
$S_k(A)$
$\bigcup_{k=0}^{K} S_k(A)$
$S_k(A) \oplus kB$
$\bigcup_{k=0}^{K} S_k(A) \oplus kB$
0
1
2
$S(A)$
A

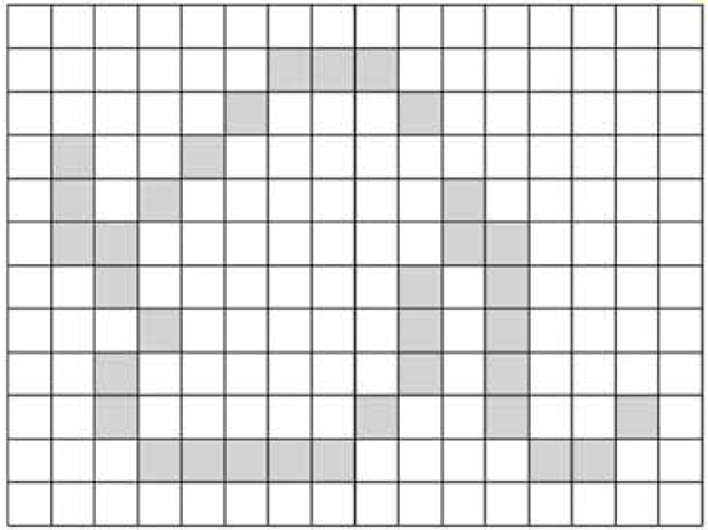
骨架粗，
且不连续

裁剪

- 裁剪（pruning）的作用
  - 对细化、骨架的补充
  - 上述操作易产生寄生分量，需要后处理去除
- 自动手写体识别
  - 通过需要分析字母的骨架形状
  - 但骨架往往带有许多“毛刺”（寄生分量）
  - “毛刺”是由笔画的不均匀造成
  - 假设寄生分量的长度较短

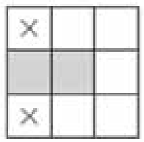
裁剪

- 字符a的骨架
  - 最左边存在毛刺
  - 通过删除端点去除
  - 删除长度≤3的分支

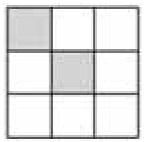


1. 使用检测端点的结构元对集合A细化

$$X_1 = A \otimes \{B\} = ((\ldots((A \otimes B^1) \otimes B^2) \ldots) \otimes B^n)$$



B^1, B^2, B^3, B^4 (rotated 90°)



B^5, B^6, B^7, B^8 (rotated 90°)

裁剪

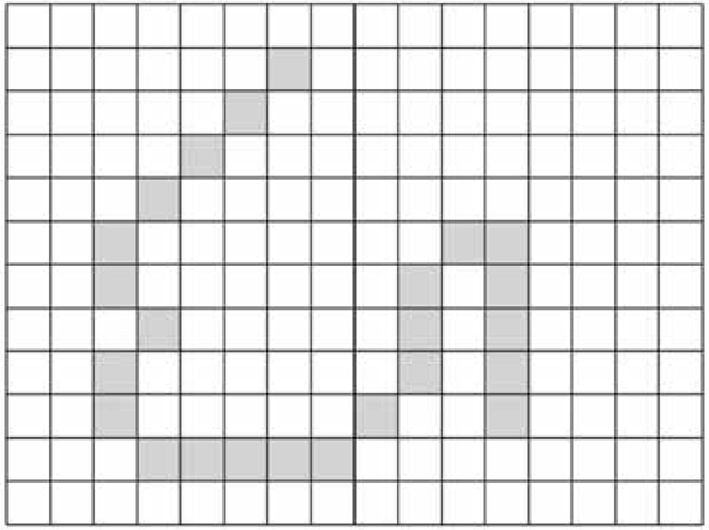
1. 使用检测端点的结构元对集合A细化

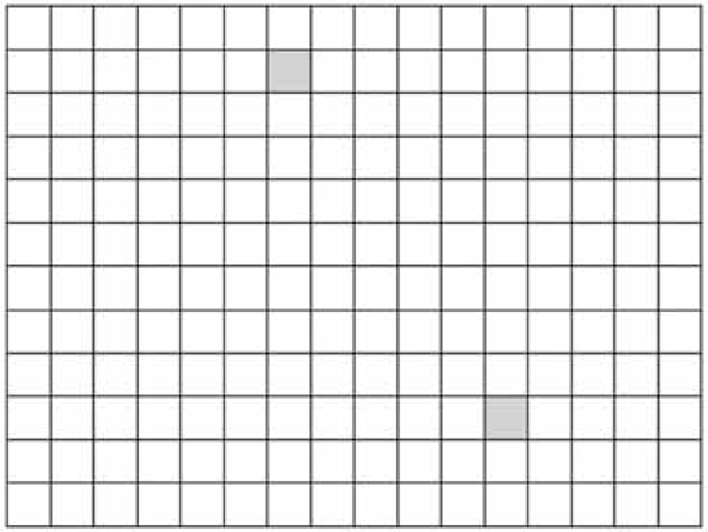
$$X_1 = A \otimes \{B\}$$

- 细化3次
- 复原形状

2. 计算X_1的端点

$$X_2 = \bigcup_{k=1}^{8} (X_1 \circledast B^k)$$



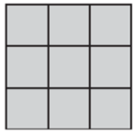


裁剪

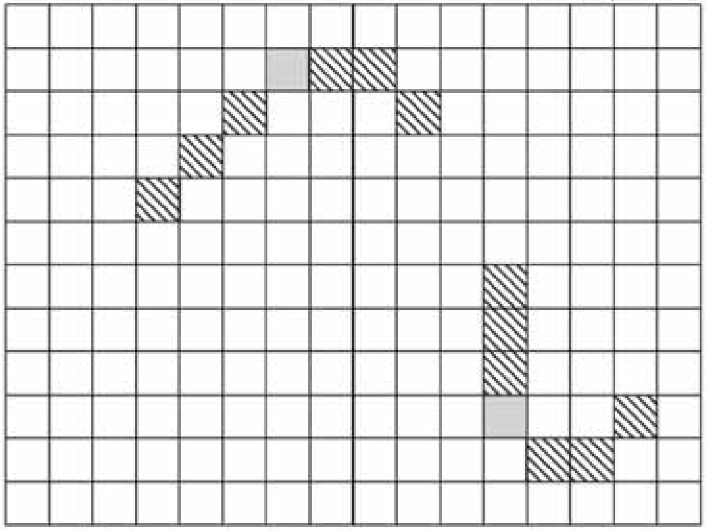
3. 对端点进行膨胀

$$X_3 = (X_2 \oplus H) \cap A$$

- 条件膨胀3次



H



4. 合并结果

$$X_4 = X_1 \cup X_3$$