
西安交通大學
XI'AN JIAOTONG UNIVERSITY


# 第7章 形态学图像处理

田智强

西安交通大学软件学院

# 7 形态学图像处理



## 7.1 预备知识

## 7.2 腐蚀与膨胀

## 7.3 开运算与闭运算

## 7.4 击中-击不中变换

## 7.5 一些基本的形态学算法

## 7.6 灰度级形态学

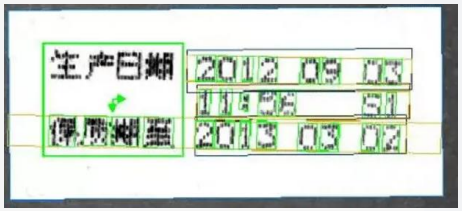
# 7.1 预备知识



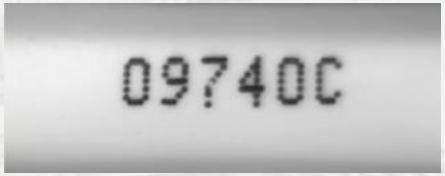
- 形态学：一般指生物学中研究动物和植物形状和结构的一个分支。
- 数学形态学的语言是集合论。
- 数学形态学（Mathematical Morphology，MM）经常用于图像的预处理和后处理，如：形态学滤波、细化和修剪等。

测螺丝齿深和角度

用形态学去除毛刺




字符识别




用形态学提取字符

# 7.1 预备知识



## 集合的基本定义

- 集合：具有某种性质的、确定的、有区别的事物的全体（它本身也是一个事物）。常用大写字母如$A$，$B$等表示。如果事物不存在，就称这种事物的全体是空集，记为$\varnothing$。
- 元素：构成集合的每个事物。
- 子集：当且仅当集合$A$的元素都属于集合$B$时，称$A$为$B$的子集。
- 并集：$A$、$B$所有的元素合并在一起组成的集合，叫做集合$A$与集合B的并集。
- 交集：由所有属于集合$A$且属于集合$B$的元素所组成的集合，叫做集合$A$与集合$B$的交集。

# 7.1 预备知识

## 集合的基本定义

- 补集：$A$是$S$的一个子集，由$S$中所有不属于$A$的元素组成的集合，称为$A$ 的补集$(A)^C$。
- 差集：由所有属于$A$且不属于$B$的元素组成的集合,叫做集合$A$减集合$B$ 。

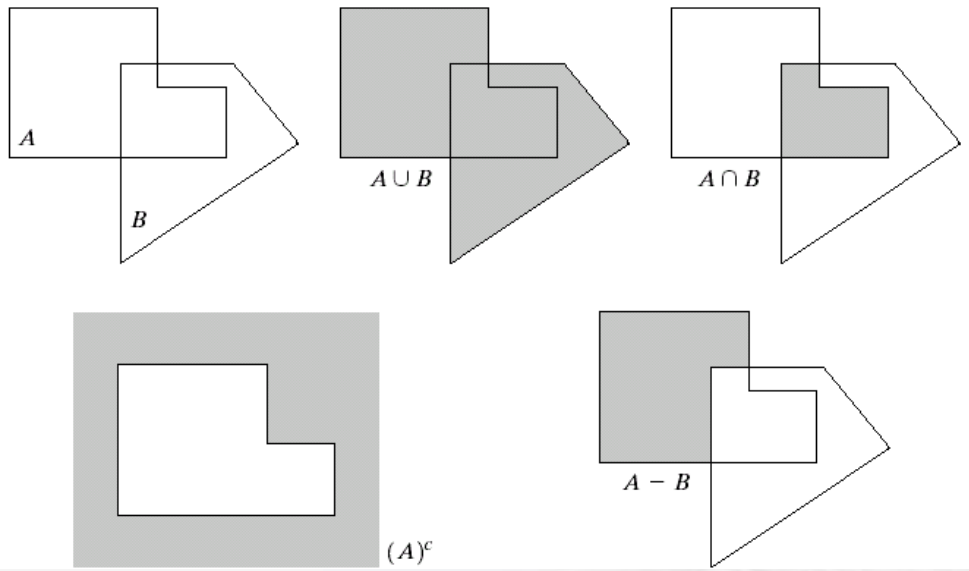



图7.1.1 集合关系图示
(a)集合$A$、$B$ ；(b)并集；(c)交集；
(d)补集；(e)差集

# 7.1 预备知识



子集：$A \subseteq B$

交集：$C = A \cap B$

并集：$C = A \cup B$

补集：$(A)^C = \{x | x \in C, x \notin A\}$

差集：$A - B = \{x | x \in A, x \notin B\}$

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

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

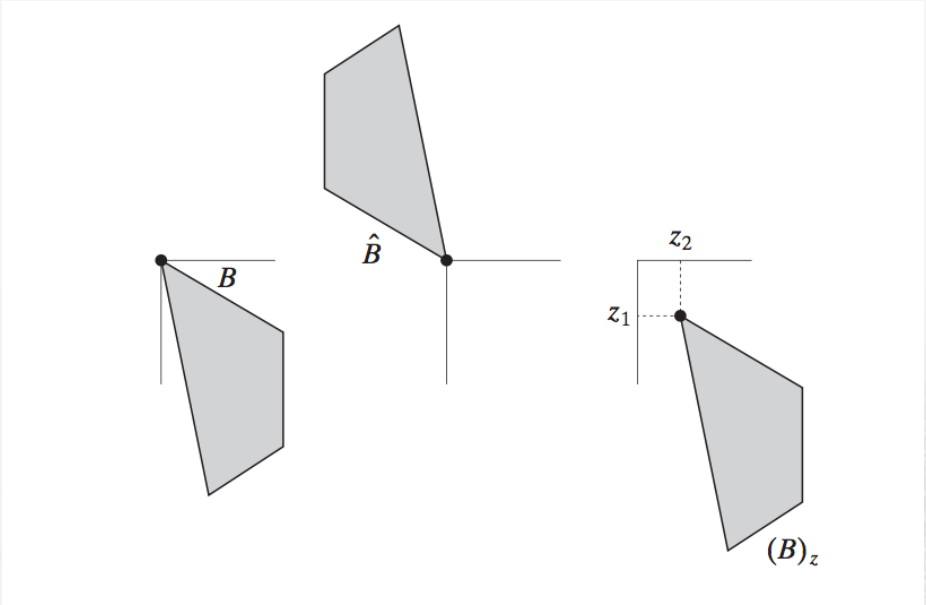



图7.1.2
(a)集合$B$；(b)集合$B$的反射；(c)集合$B$的平移

形态学中：集合的平移和反射广泛用于结构元（Structuring Element，SE）的操作：研究一幅图像中感兴趣特性所用的小集合或子图像。

# 7.1 预备知识



## 结构元（Structuring element, SE）

- 在形态学算法设计中，结构元的选择十分重要，其形状、尺寸的选择是能否有效提取信息的关键。
- 结构元必须在几何上比原图像简单，且有界。
- 当对图像操作时，要求结构元是矩形阵列。这是通过添加最小可能数量的背景元素形成一个矩形阵列来实现的。

# 7.1 预备知识



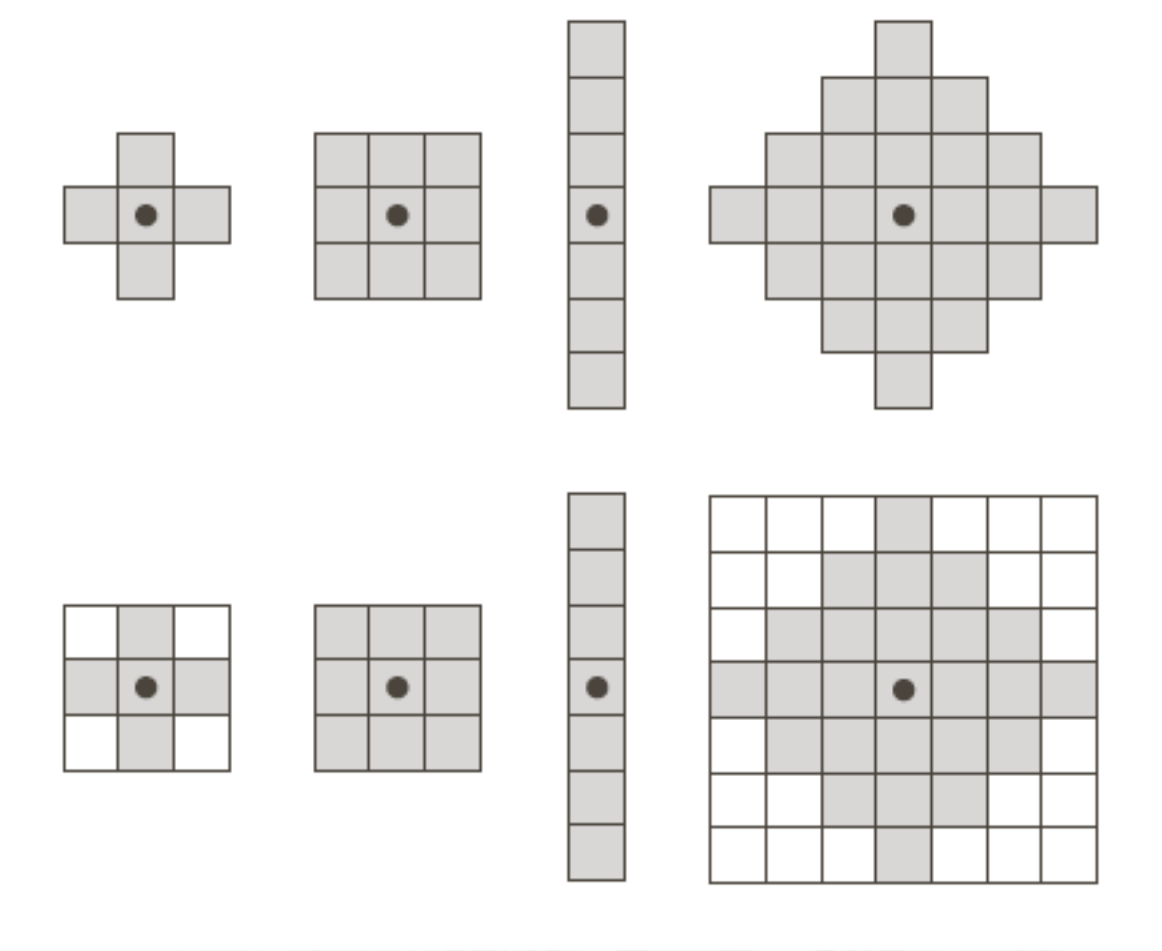


图7.1.3 第一行：结构元的例子；第二行：转换为矩形阵列的结构元；点表示结构元的中心点（原点）。

# 7.1 预备知识



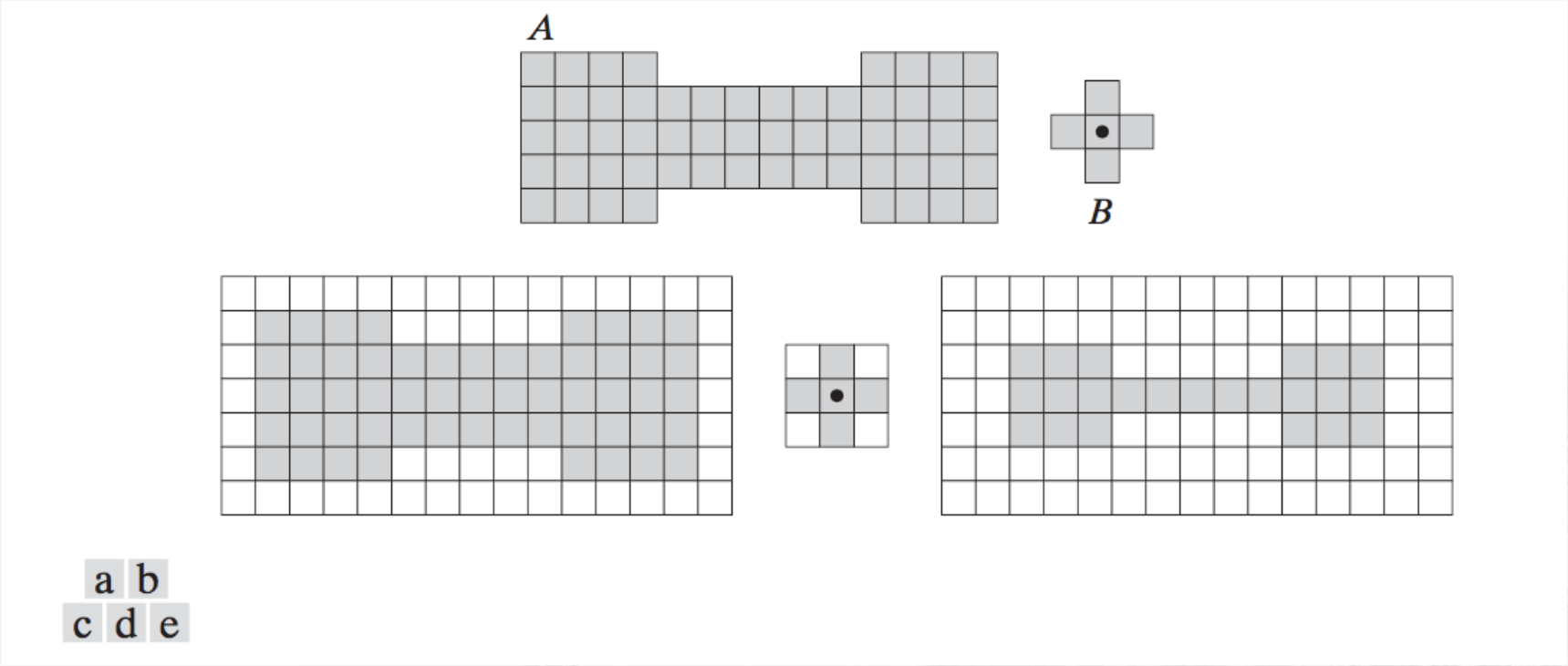



图7.1.4 (a)一个集合（每个阴影方块都是集合的一个元素）；(b)结构元；(c)使用背景元素填充集合形成的一个矩形阵列，并且提供了一个背景边界；(d)矩形阵列形式的结构元；(e)由结构元处理过的集合。

# 7 形态学图像处理



7.1 预备知识

**7.2 腐蚀与膨胀**

7.3 开运算与闭运算

7.4 击中-击不中变换

7.5 一些基本的形态学算法

7.6 灰度级形态学

# 7.2 腐蚀与膨胀



## 7.2.1 腐蚀

## 7.2.2 膨胀

# 7.2.1 腐蚀



假定$A$和$B$是$Z^2$中的两个集合，$A$被$B$腐蚀可定义为：

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

该式指出$B$对$A$的腐蚀是，$B$通过$z$平移后，仍包含在$A$中的所有点*的集合。在下面的讨论中，假定集合$B$是一个结构元。因为$B$必须包含在$A$中这一陈述等价于$B$不与背景共享任何公共元素，故可以将腐蚀表达为如下的等价形式：

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

*此处所有点指的是B的中心点。

# 7.2.1 腐蚀



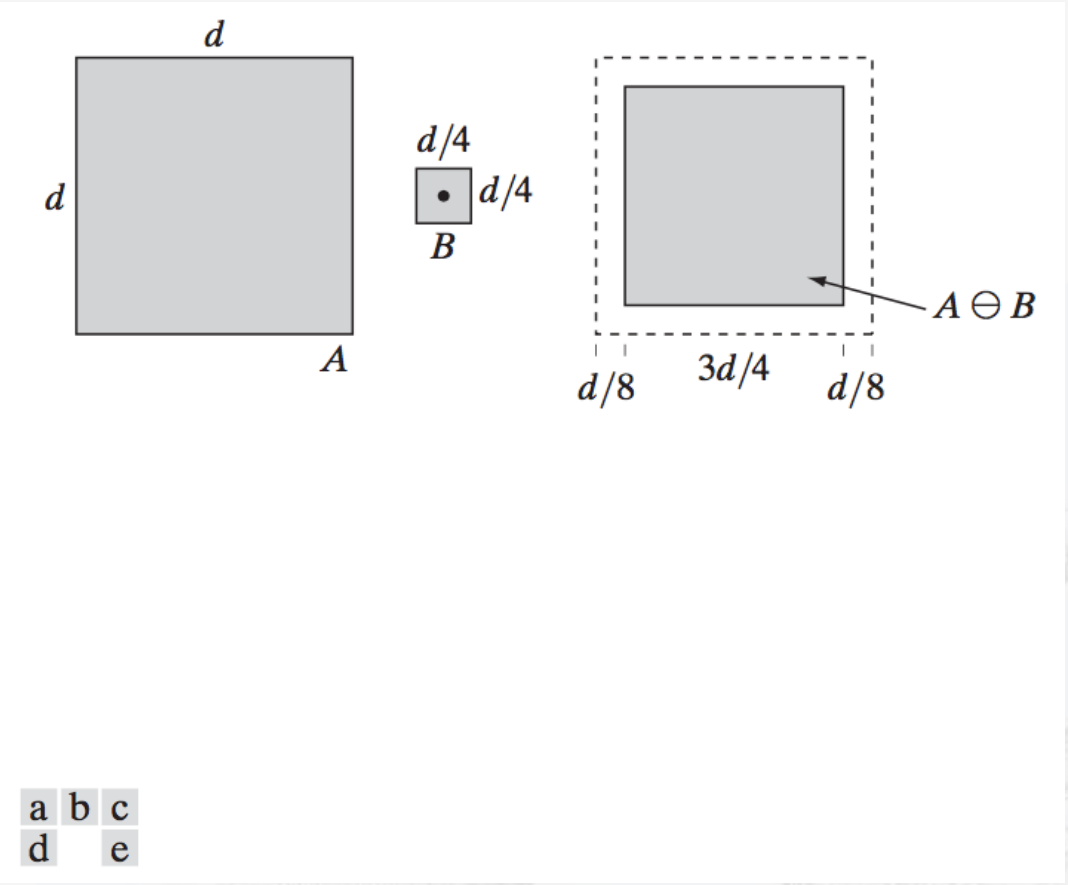



图7.2.1 (a)集合$A$；(b)方形结构元；(c)$B$对$A$的腐蚀，如阴影部分所示；(d)拉长的结构元；(e)用拉长后结构元$B$对$A$的腐蚀。(c)和(e)中的虚线边界是集合$A$的边界。

# 7.2.1 腐蚀



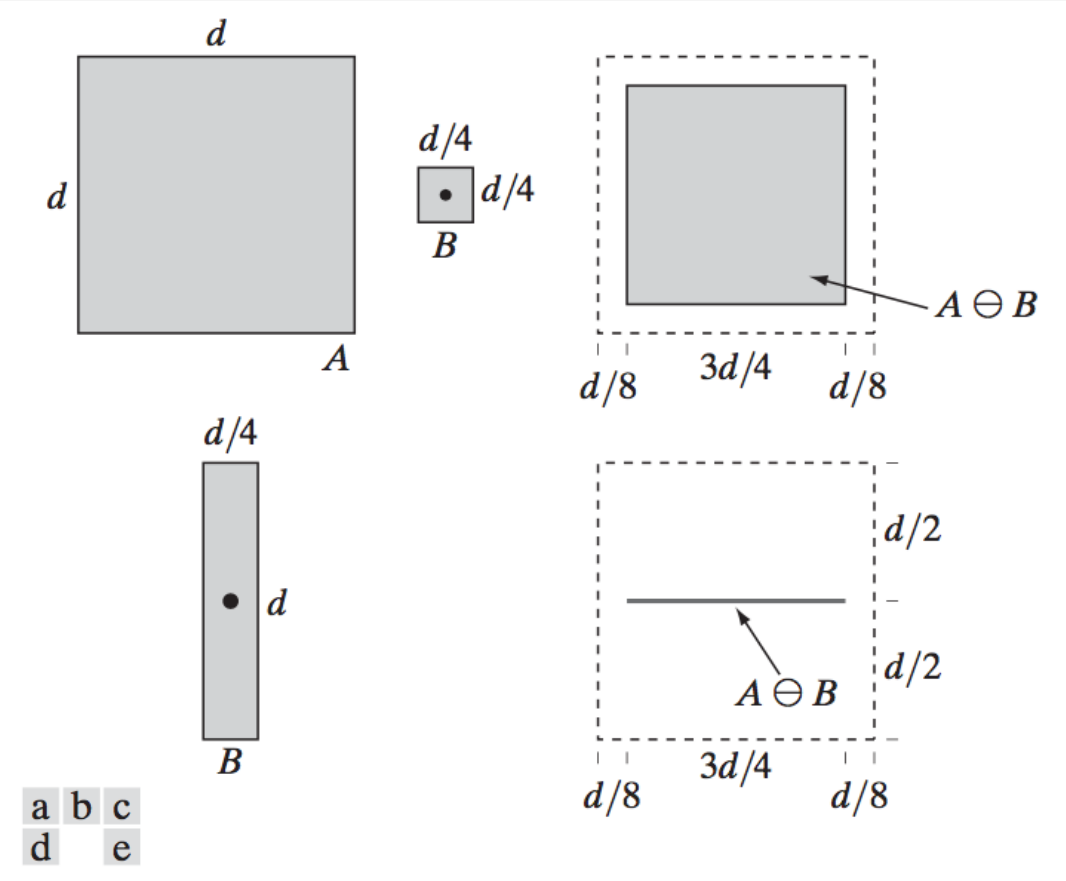



图7.2.1 (a)集合$A$；(b)方形结构元；(c)$B$对$A$的腐蚀，如阴影部分所示；(d)拉长的结构元；(e)用拉长后结构元$B$对$A$的腐蚀。(c)和(e)中的虚线边界是集合$A$的边界。

# 7.2.1 腐蚀



腐蚀的步骤：

1. 用结构元$B$（B中元素值为1）扫描$A$的每一个元素（A中元素值为1，非A元素为0）；
2. 用结构元$B$中的所有元素与其覆盖的A做“与”操作，如果都为1，结果图像的该像素（结构元当前原点所在位置）为1；如果不都为1，则结果图像该像素为0，即发生腐蚀操作。

腐蚀处理的结果是使原来的A减小一圈。

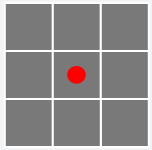
# 7.2.1 腐蚀



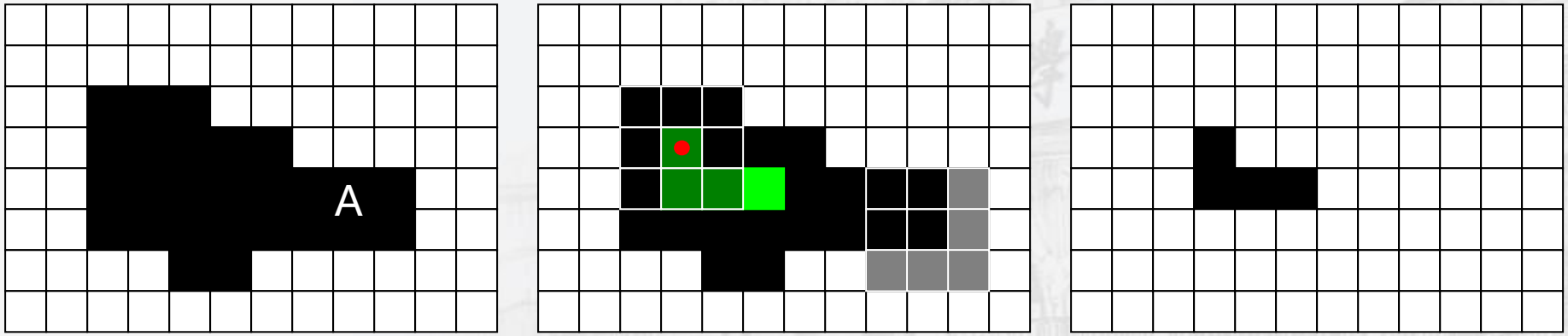
腐蚀举例：

将结构元$B$移动到图像$A$中每个像素的位置，并且结构元$B$需要完全包含在图像$A$中。

采用的结构元



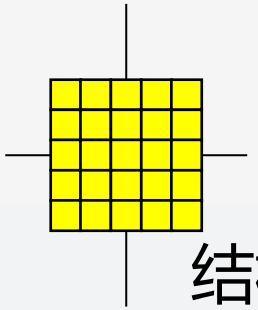
结构元$B$



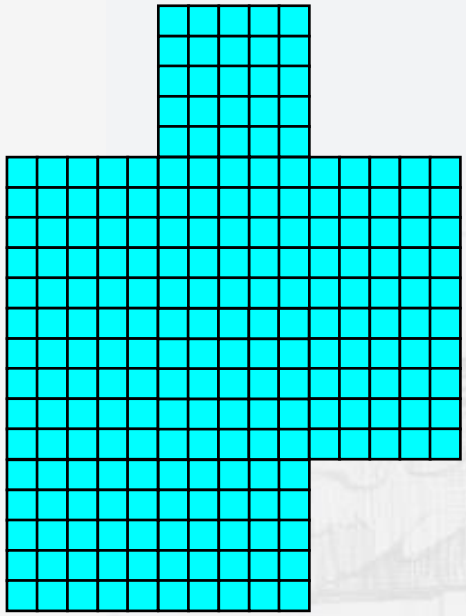
# 7.2.1 腐蚀



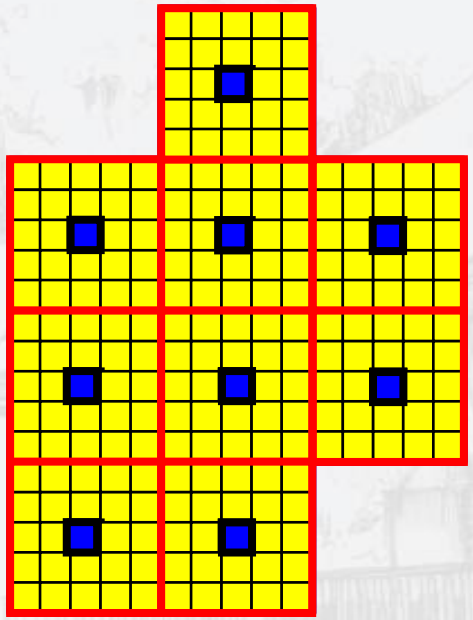
腐蚀过程：



结构元



原始图像

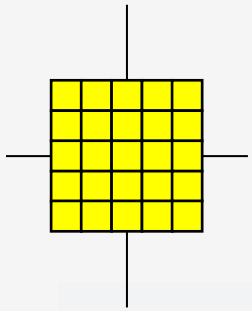


相交的像素

中心点（原点）

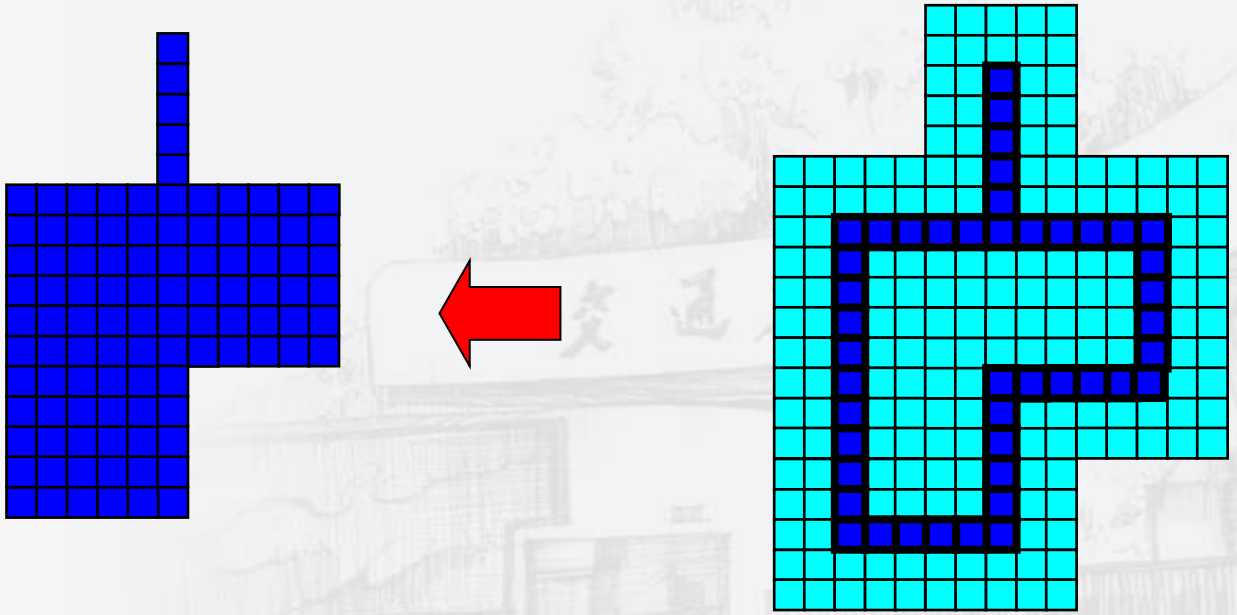
# 7.2.1 腐蚀





结构元

注：原点可以不在结构元的几何中心



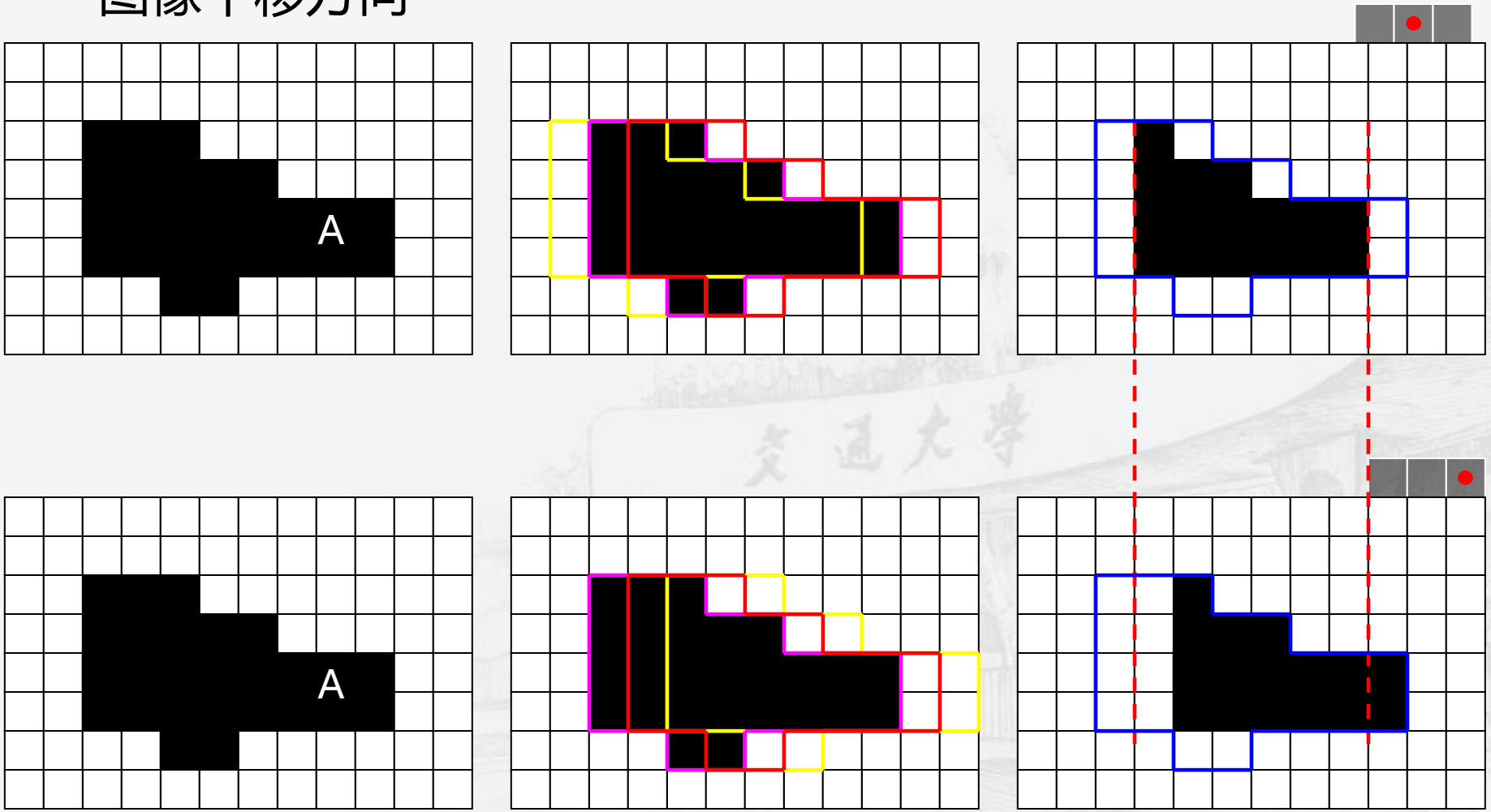
腐蚀结果

结构元在原图内部的原点组成的边界

# 7.2.1 腐蚀



从平移的角度定义腐蚀：先平移再求交集，结构元形状决定图像平移方向



# 7.2.1 腐蚀



腐蚀操作应用举例：消除二值图像中的不相关细节

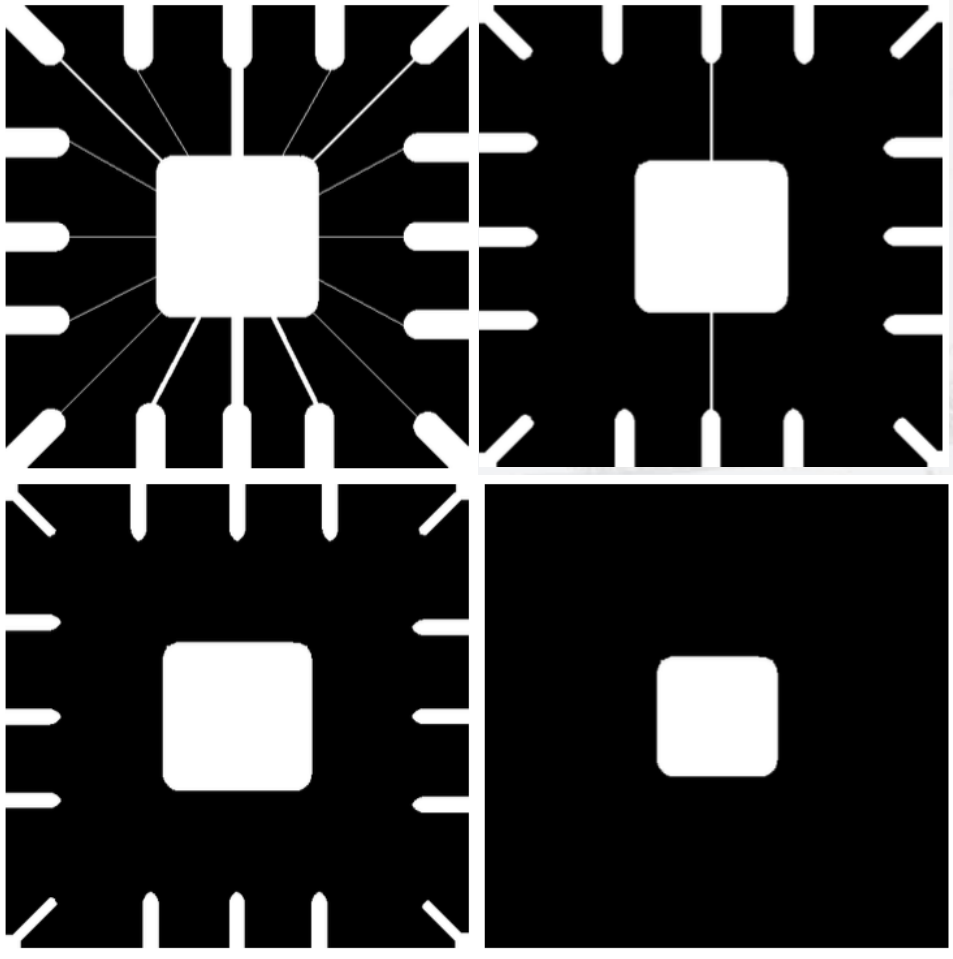



图7.2.2 使用腐蚀去除图像中不相关的部件。(a) 一幅大小为486×486的连线模板二值图像；(b)~(d)分别使用大小为11x11，15×15和45×45的结构元腐蚀图像。结构元的元素值为1。

# 7.2 腐蚀与膨胀



7.2.1 腐蚀

7.2.2 膨胀

# 7.2.2 膨胀



假定$A$和$B$是$Z^2$中的两个集合，$A$被$B$膨胀可定义为：

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

这个公式是以$B$关于它的原点的反射，并且以$z$对反射进行平移为基础的。$B$对$A$的膨胀是所有位移$z$ 的集合，这样$B$的反射和$A$至少有一个元素是重叠的。

根据这种解释，式子可以等价地写为：

$$A \oplus B = \left\{ z \mid \left[ (\hat{B})_z \cap A \right] \subseteq A \right\}$$

# 7.2.2 膨胀



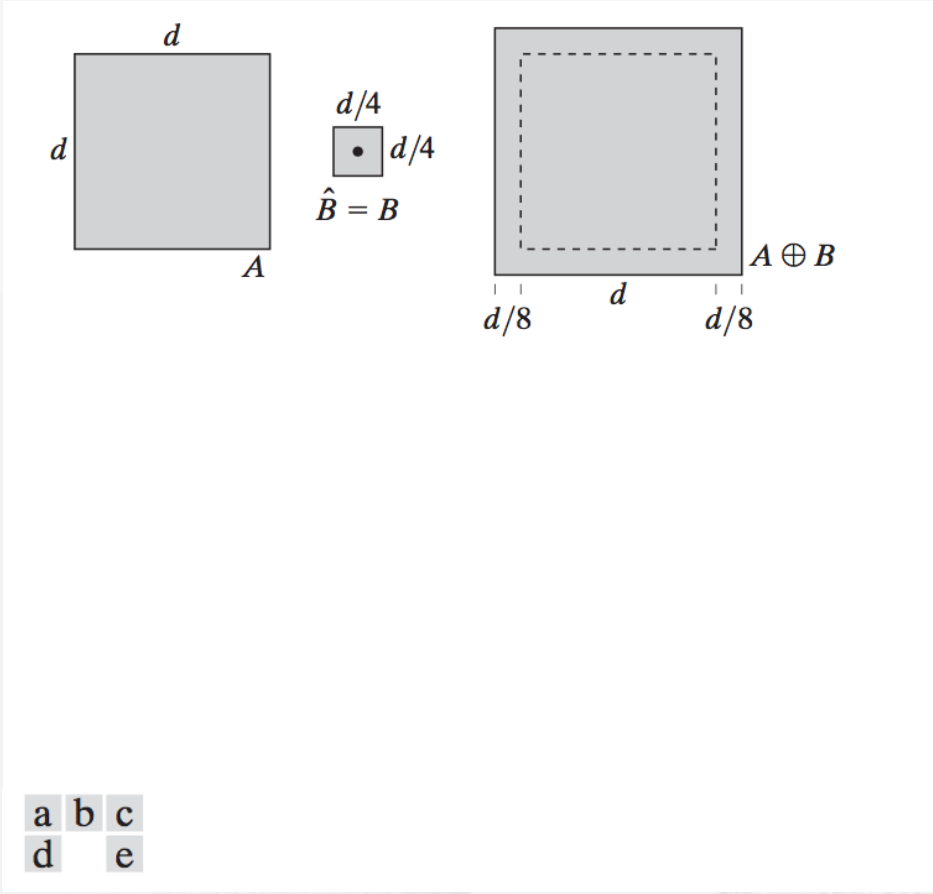



图7.2.3 (a)集合$A$；(b)方形结构元；(c)$B$对$A$的膨胀，如阴影部分所示；（d）拉长的结构元；(e)用拉长后结构元$B$对$A$的膨胀。(c)和(e)中的虚线边界是集合$A$的边界。

# 7.2.2 膨胀



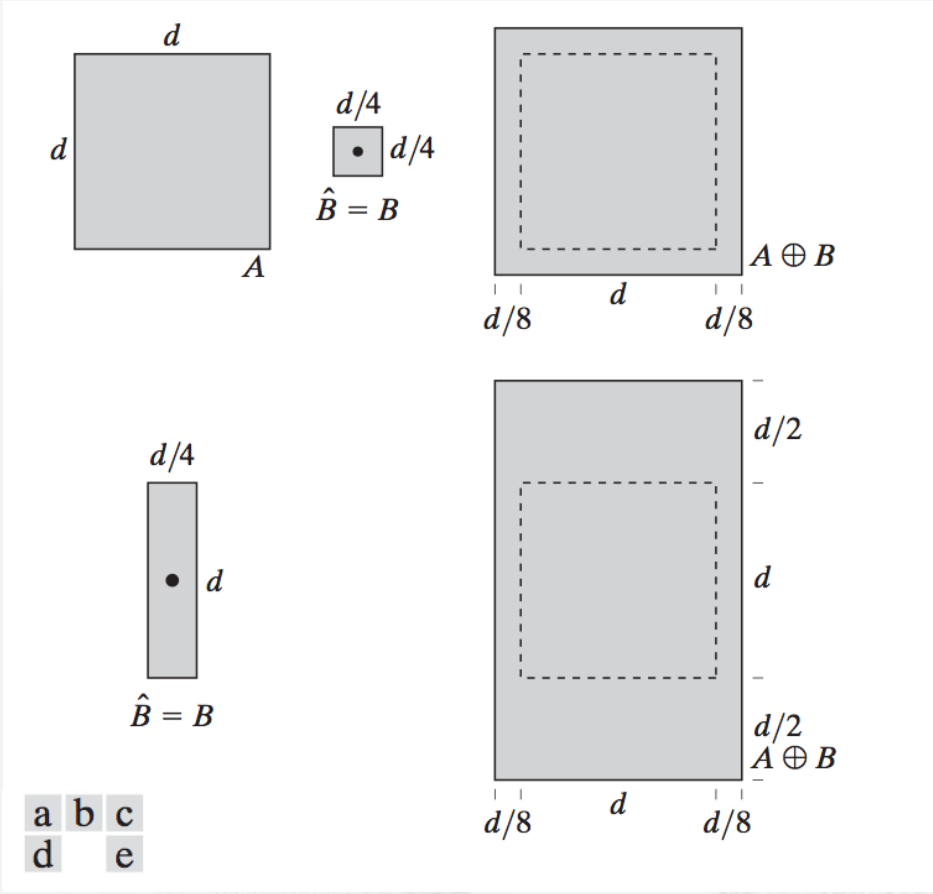



图7.2.3 (a)集合$A$；(b)方形结构元；(c)$B$对$A$的膨胀，如阴影部分所示；（d）拉长的结构元；(e)用拉长后结构元$B$对$A$的膨胀。(c)和(e)中的虚线边界是集合$A$的边界。

# 7.2.2 膨胀



膨胀的步骤：

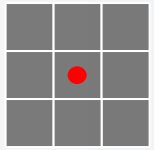
1. 用结构元$B$的反射（其原点）扫描图像$A$的每一个像素；
2. 用结构元的反射与其覆盖的二值图像做“或”操作，即有一个元素为1，结果图像的该像素为1，否则为0。

膨胀处理的结果是使原来的二值图像增大一圈。

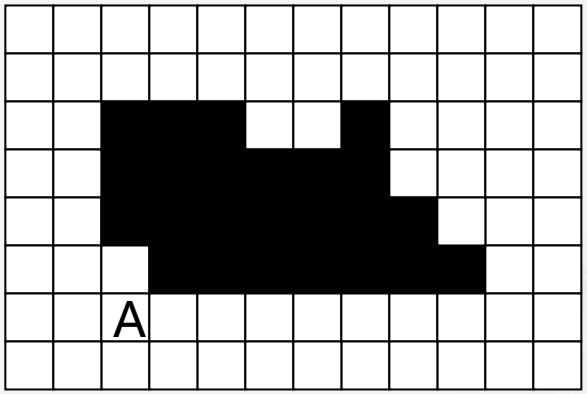
# 7.2.2 膨胀

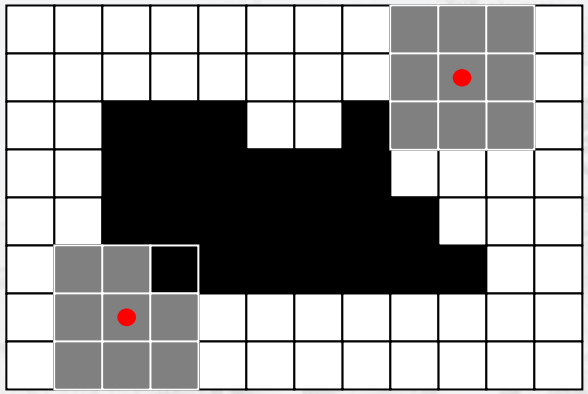


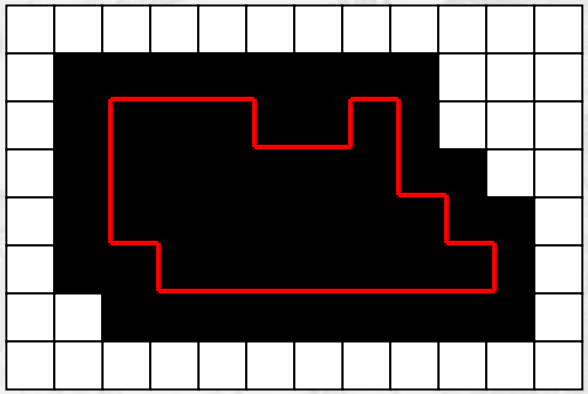
将结构元$B$的反射移动到图像$A$中每个像素的位置，并且结构元$B$的反射和图像$A$的交集不为空集。



结构元$B$

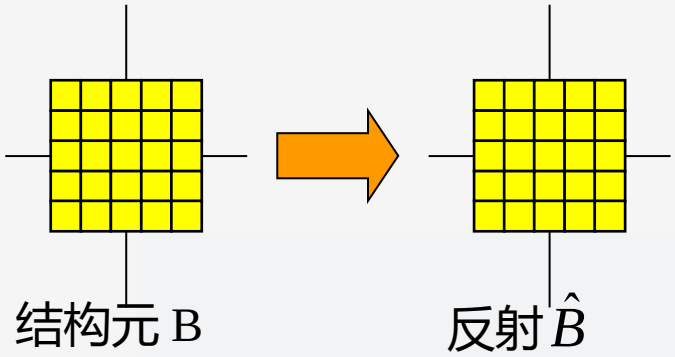


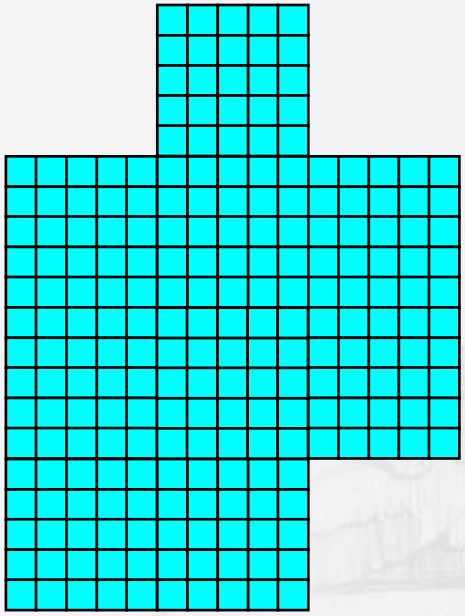



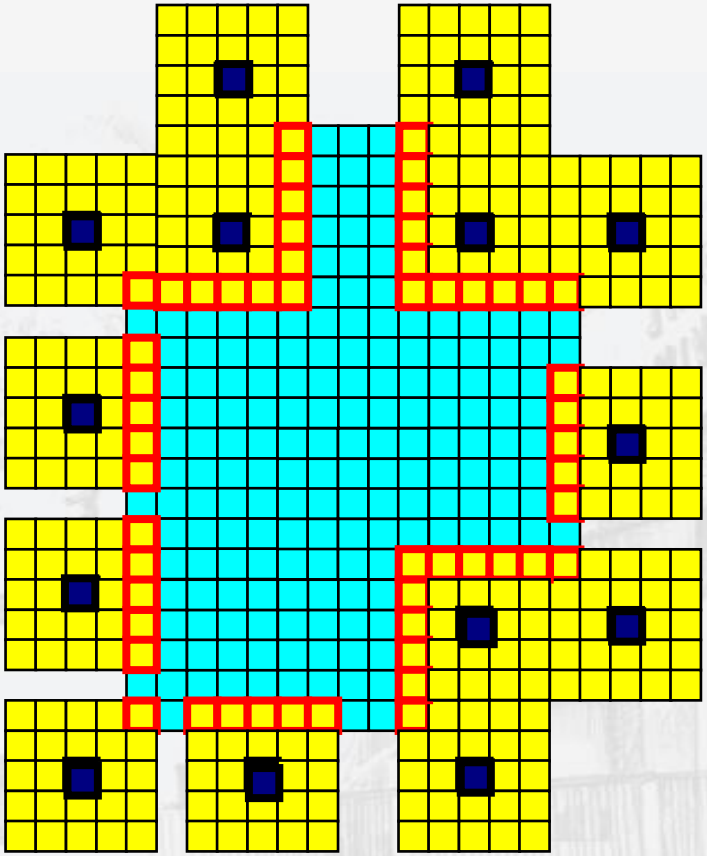


# 7.2.2 膨胀





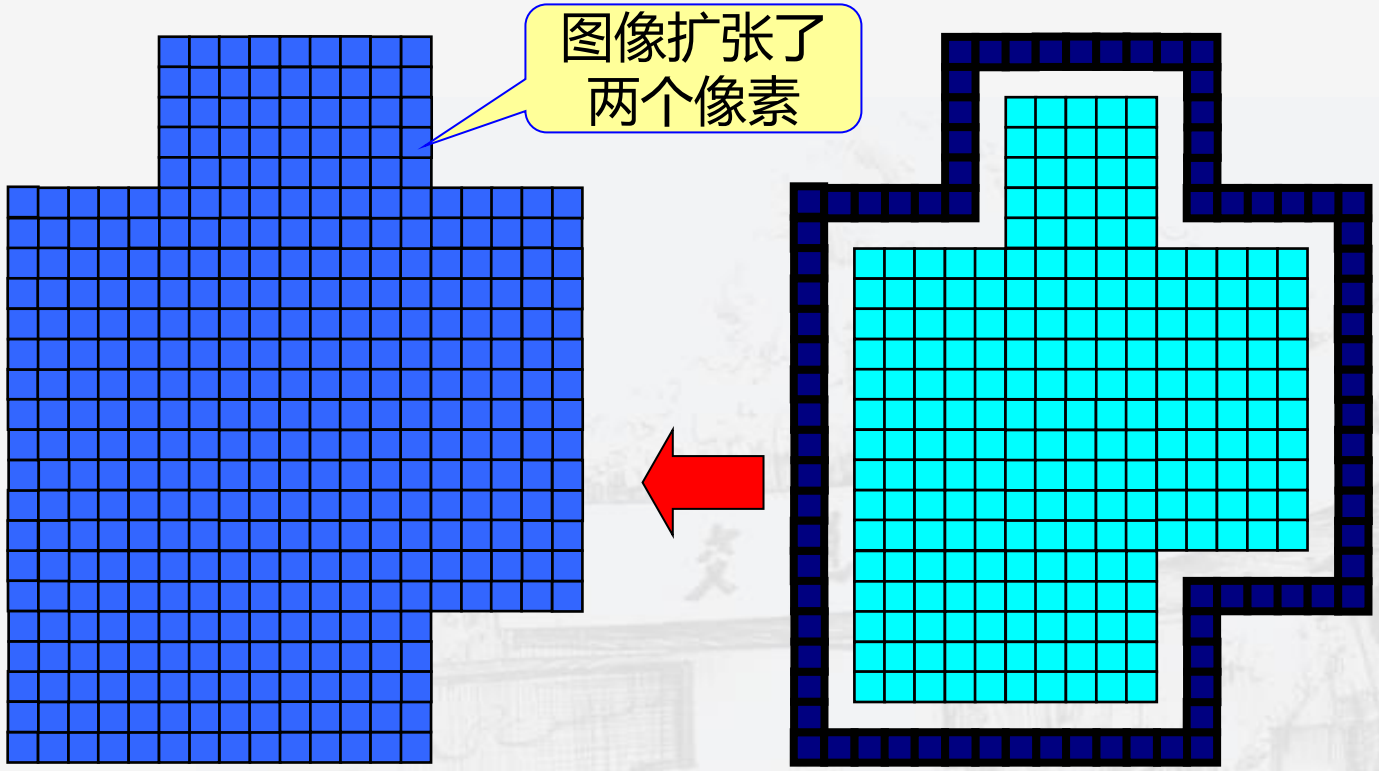



相交的像素　　中心点

# 7.2.2 膨胀

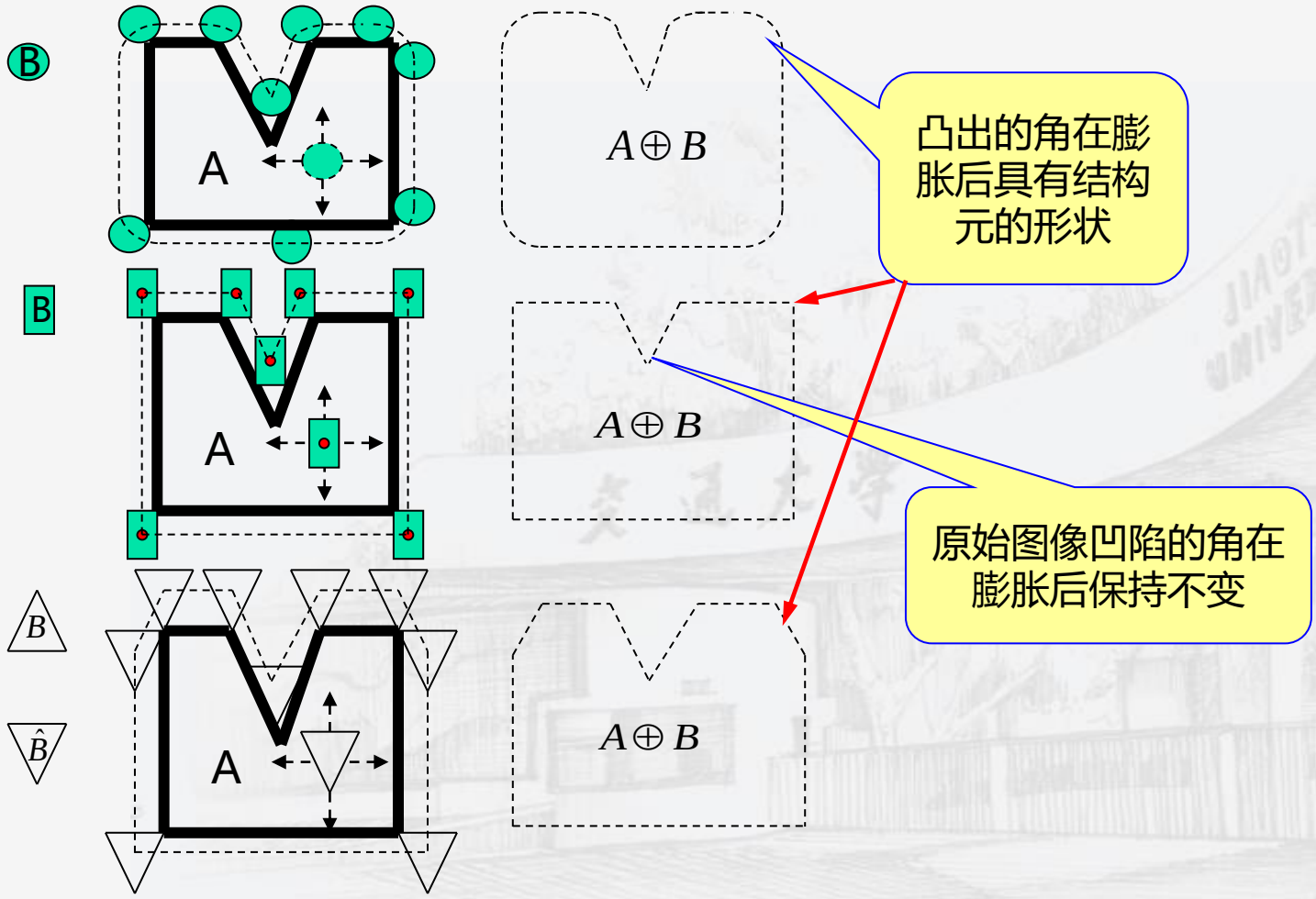





视频演示

# 7.2.2 膨胀

结构元形状对膨胀结果的影响



# 7.2.2 膨胀



应用举例：桥接断裂文字间的间隙

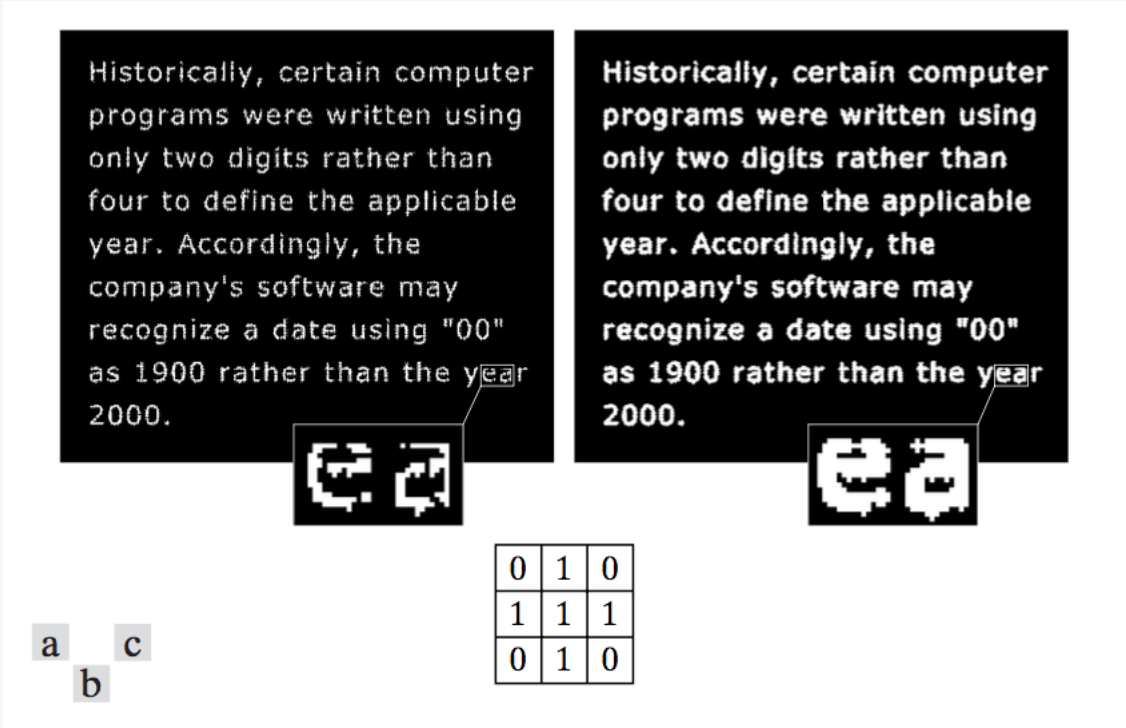



图7.2.4 (a)低分辨率样本文本与断裂的字符(参见放大视图)；(b)结构元；(c) (b)结构元对(a)进行膨胀。断裂线段连接。

# 7.2.2 膨胀

## 腐蚀与膨胀的对偶性

腐蚀与膨胀彼此关于集合的求补运算和反射运算是对偶的，即：

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

$B$对$A$的腐蚀是对$A^C$的膨胀的补，反之亦然。当结构元关于其原点对称时（通常如此），因为 $\hat{B} = B$，为对偶性决定。这样即可以用相同的结构元简单地使用$B$膨胀图像的背景（即膨胀），并对该结果求补就可得到$B$对该幅图像的腐蚀。

# 7 形态学图像处理



# 7.3 开运算与闭运算



## 7.3.1 开运算

## 7.3.2 闭运算

# 7.3.1 开运算



开运算：相当于先用结构元$B$对$A$腐蚀，再对腐蚀结果用同样的结构元进行膨胀操作。公式如下：

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

经过开运算的图像更加规则化，相比原图有较少的细节，一般会平滑物体的轮廓、断开较窄的狭颈并消除细的突出物。

开操作有一个简单的几何解释。假设把结构元$B$视为“转球"。然后，$A \circ B$的边界由$B$中的点建立：当$B$在$A$的边界内侧滚动时，$B$所能到达边界的最远点。公式如下：

$$A \circ B = \bigcup \{(B)_z \mid (B)_z \subseteq A\}$$

# 7.3.1 开运算



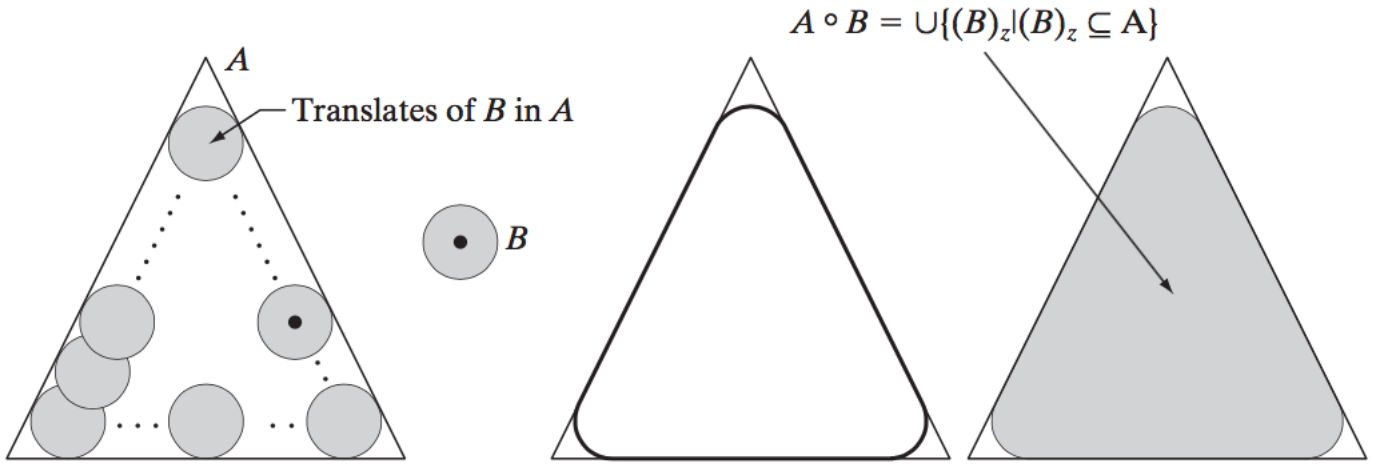



图7.3.1 (a)结构元$B$沿集合$A$的内侧边界滚动（黑点表示$B$的原点）；(b)结构元；(c)粗线是开操作的外部边界；(d)完全的开操作（阴影部分）。为清楚起见，在图(a)中未加阴影。

# 7.3.1 开运算



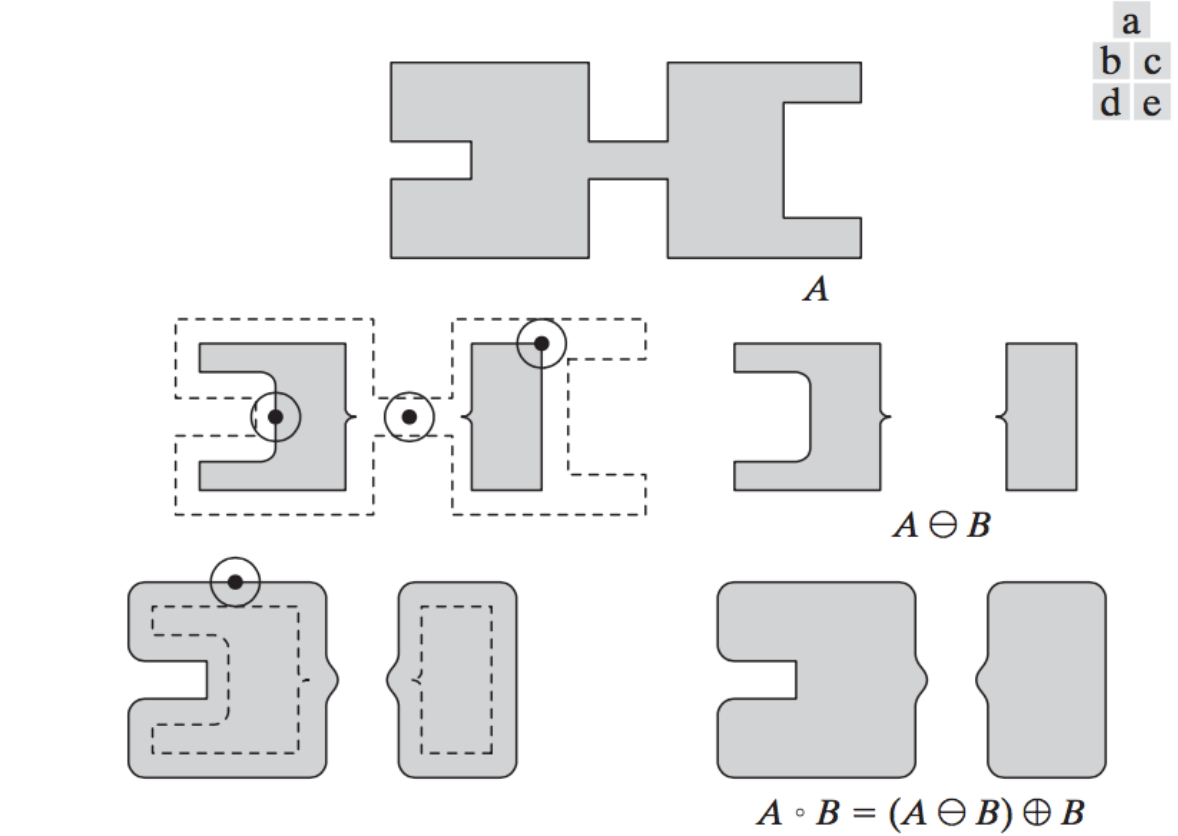
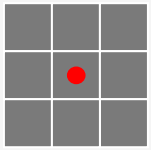


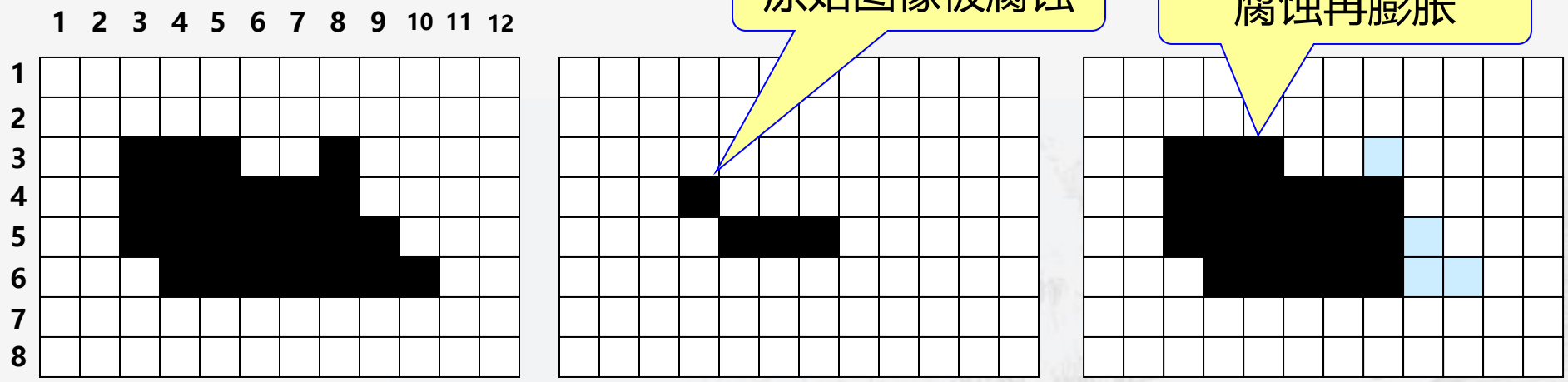

图7.3.2 (a)为集合 $A$ ； (b)腐蚀期间结构元各个位置；(c)腐蚀后的结果；(d)膨胀处理；(e)开操作最后结果。

# 7.3.1 开运算

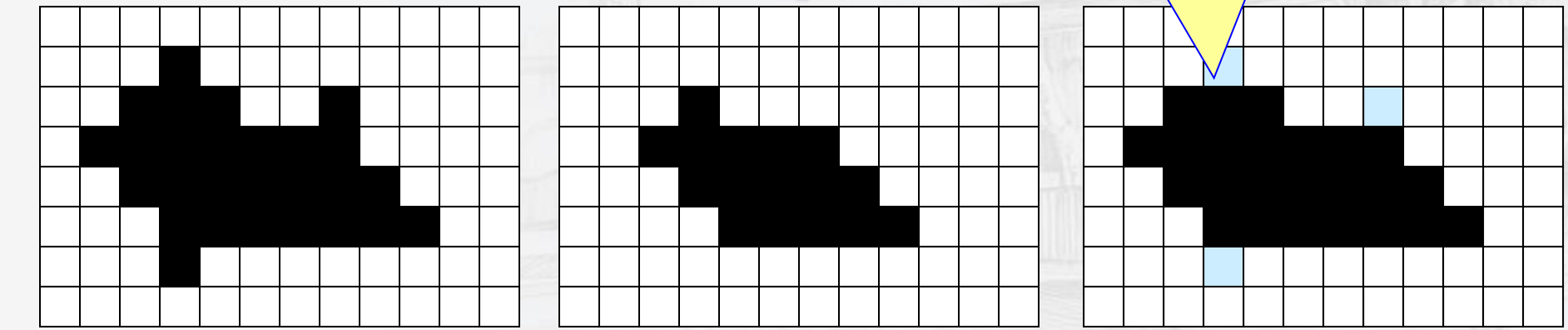


结构元:





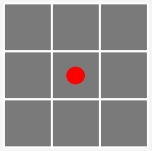

结构元:

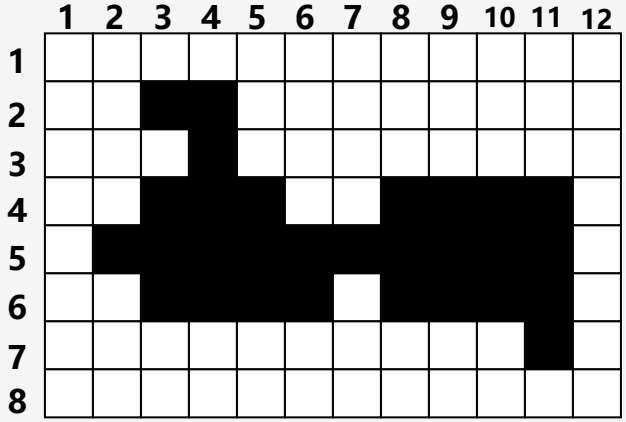


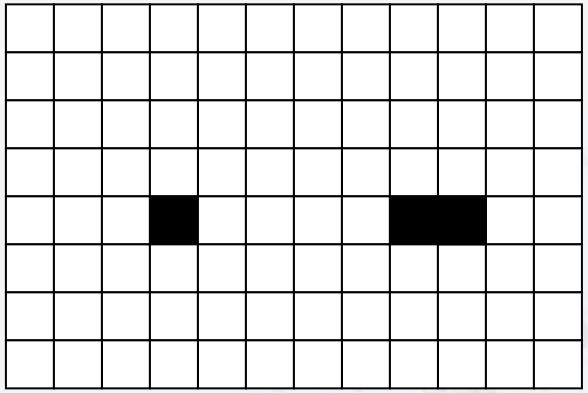
# 7.3.1 开运算

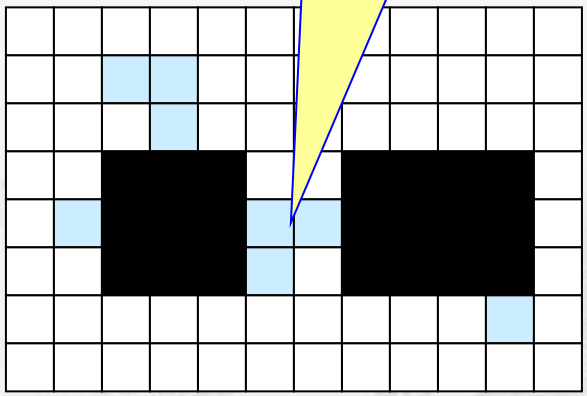


结构元:





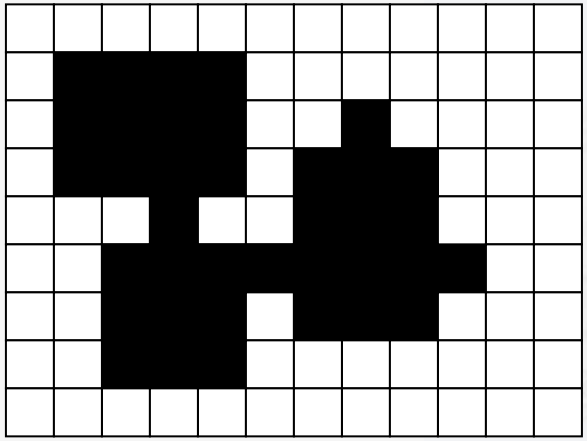



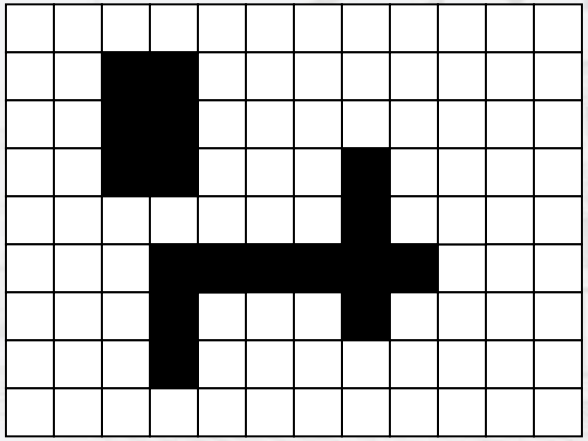


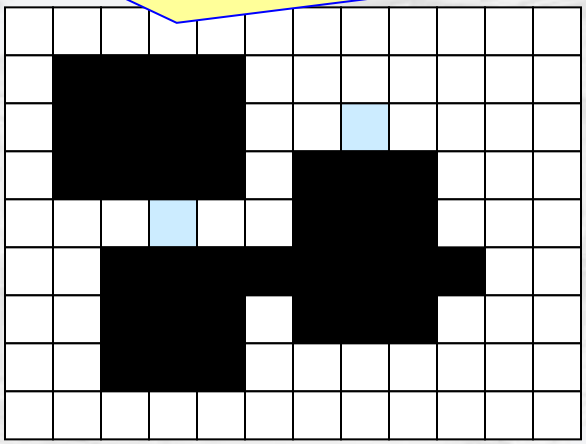
细的连接被断开后无法恢复

结构元:

横的细的连接被断开后可以恢复
纵的细的连接被断开后无法恢复



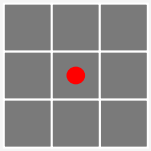




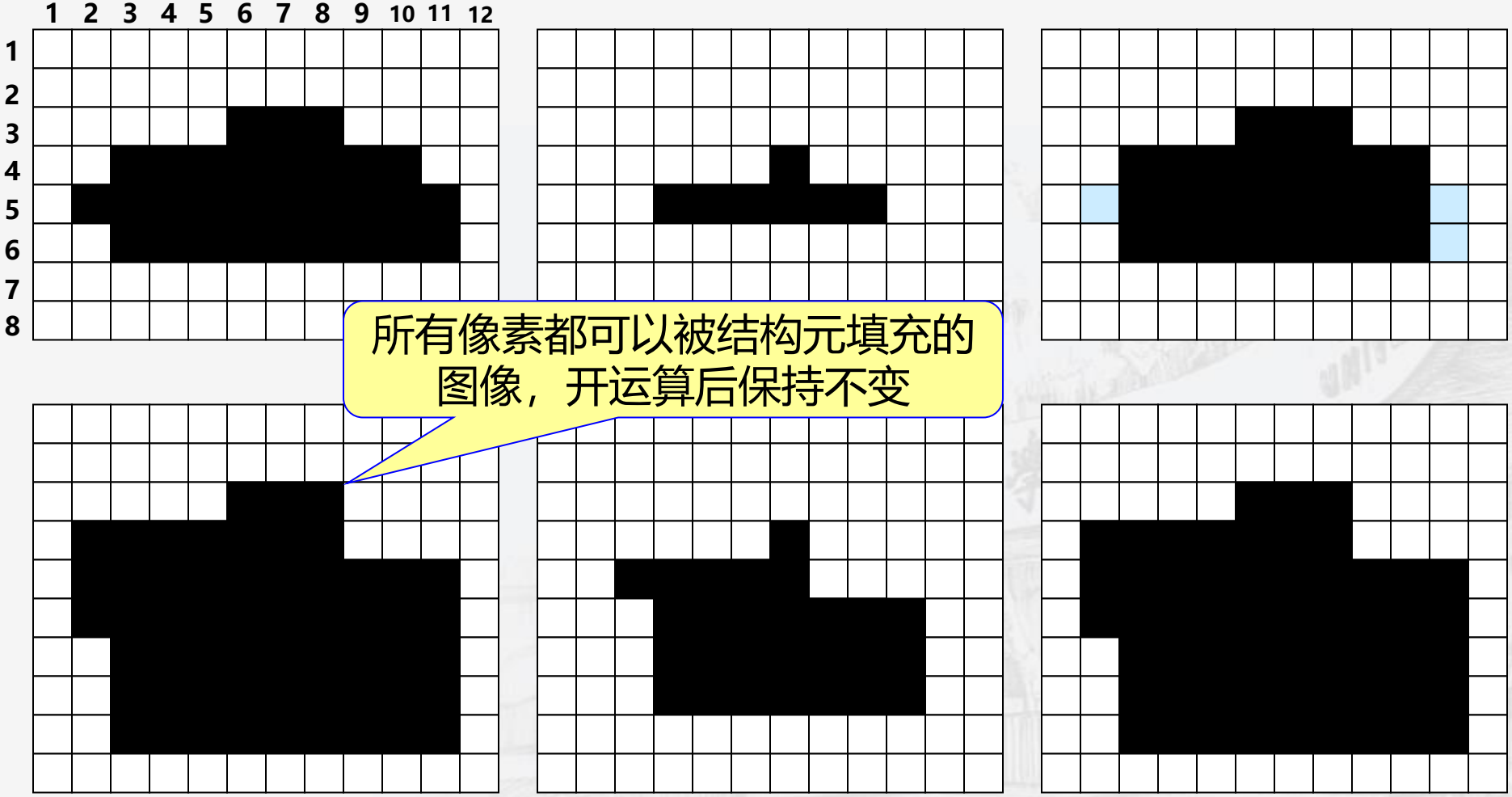
# 7.3.1 开运算



结构元:



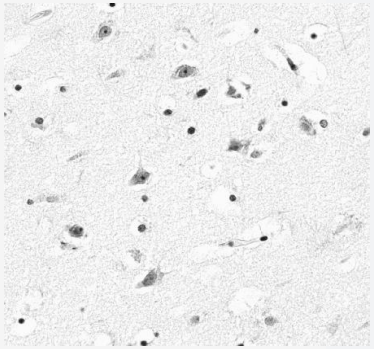
是否有开运算后保持不变的图像呢?
——取决于结构元的选择



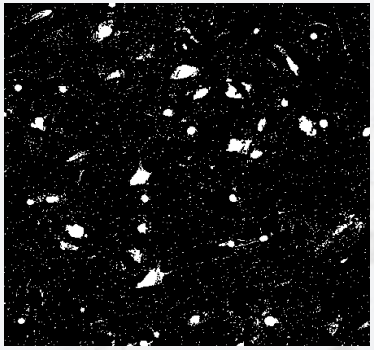
# 7.3.1 开运算



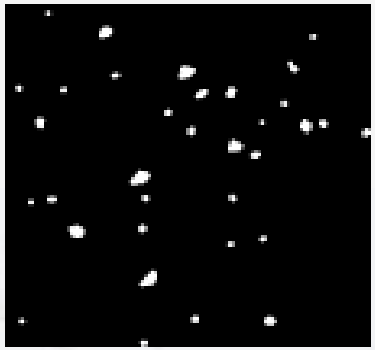
不同尺寸结构元提取特征



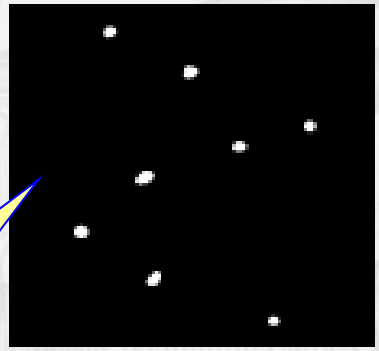
原始图像



阈值处理后的二值图像



使用小的结构元开运算



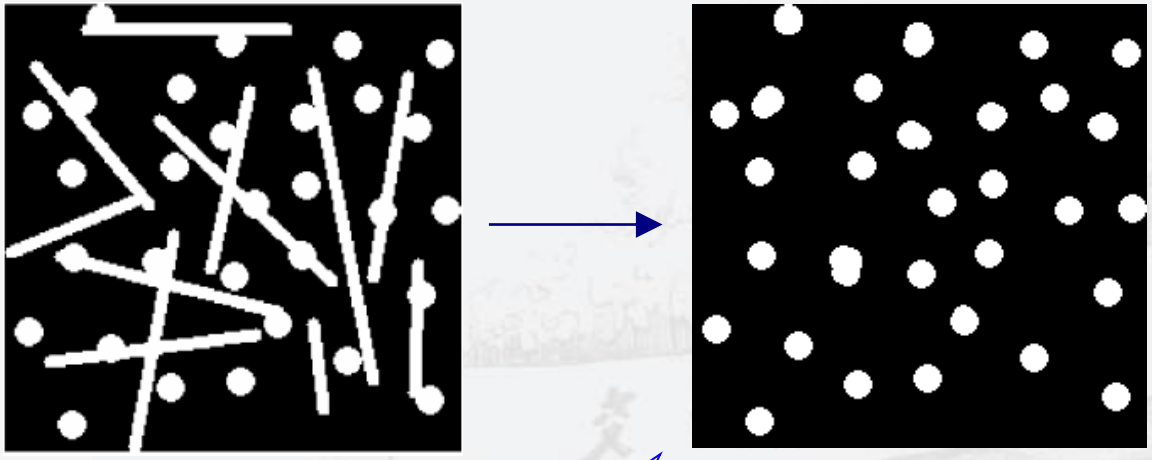
使用大的结构元开运算

更多的细节特征被抑制

# 7.3.1 开运算



通过结构元的形状选择特征



输入图像

使用圆形结构元
进行开运算

图像中只有与结构元形态相匹配的特征被保留

# 7.3 开运算与闭运算



## 7.3.1 开运算

## 7.3.2 闭运算

# 7.3.2 闭运算



闭运算：先执行膨胀操作，再腐蚀操作。闭运算是开运算的对偶运算。闭运算公式如下：

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

闭操作同样也会平滑轮廓的一部分，但与开操作相反，它的作用通常是：

1. 弥合较窄的间断和细长的沟壑；
2. 消除小的孔洞；
3. 填补轮廓线中的断裂。

# 7.3.2 闭运算



闭运算在边界外侧滚动球体，从几何上讲，当且仅当对包含$w$的$(B)_z$进行的任何平移都有$(B)_z \cap A \neq \emptyset$时，点$w$才是$A \cdot B$的一个元素。下图说明了闭操作这一基本的几何性质。

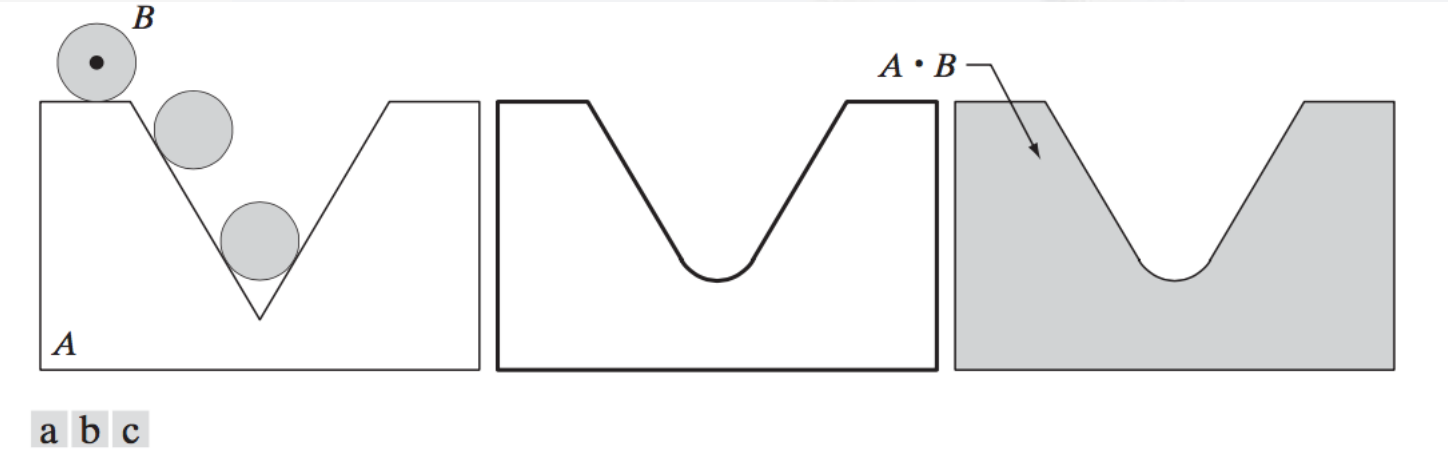



图7.3.3 (a)结构元$B$沿集合$A$的外侧边界滚动；(b)粗线是闭操作的外部边界；(c)完全的闭操作（阴影部分）。

# 7.3.2 闭运算



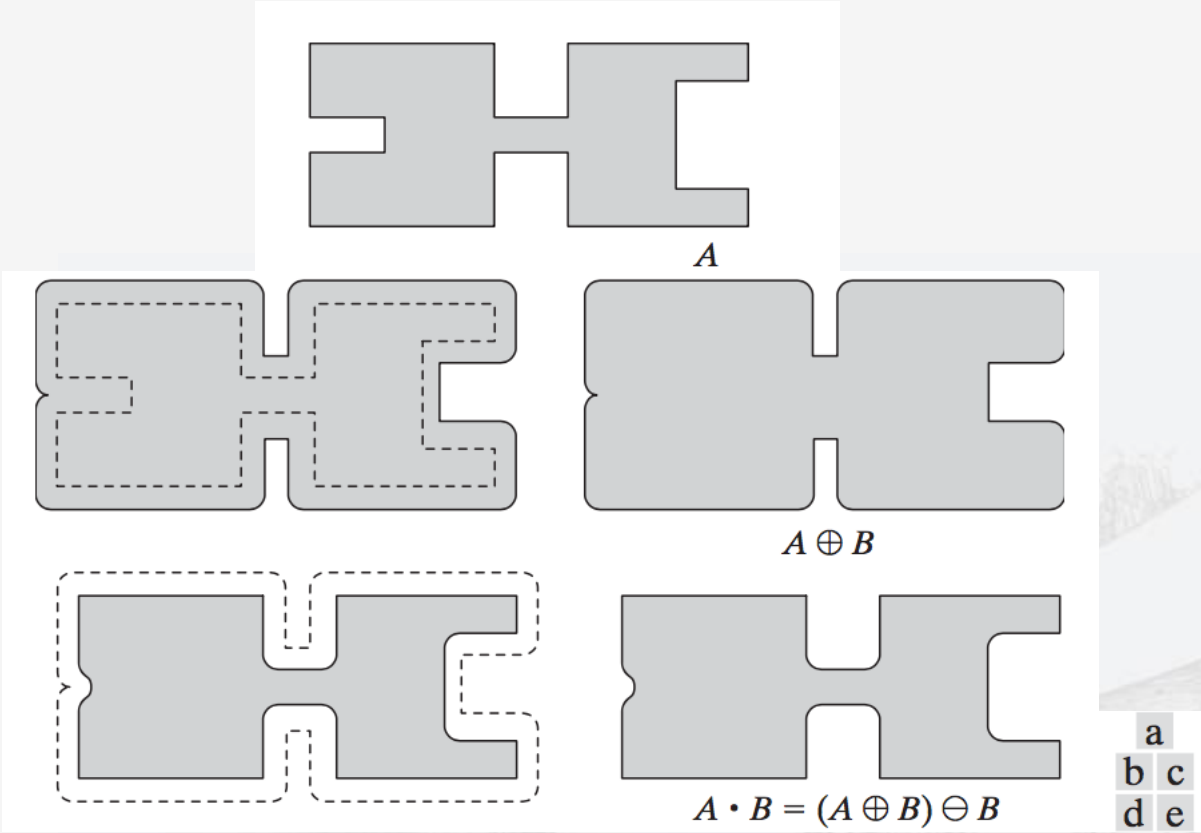


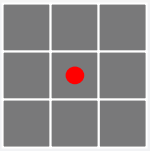

图7.3.4 (a)为集合$A$；(b)膨胀操作；(c)膨胀后的结果；(d)腐蚀处理；(e)闭操作最后结果。

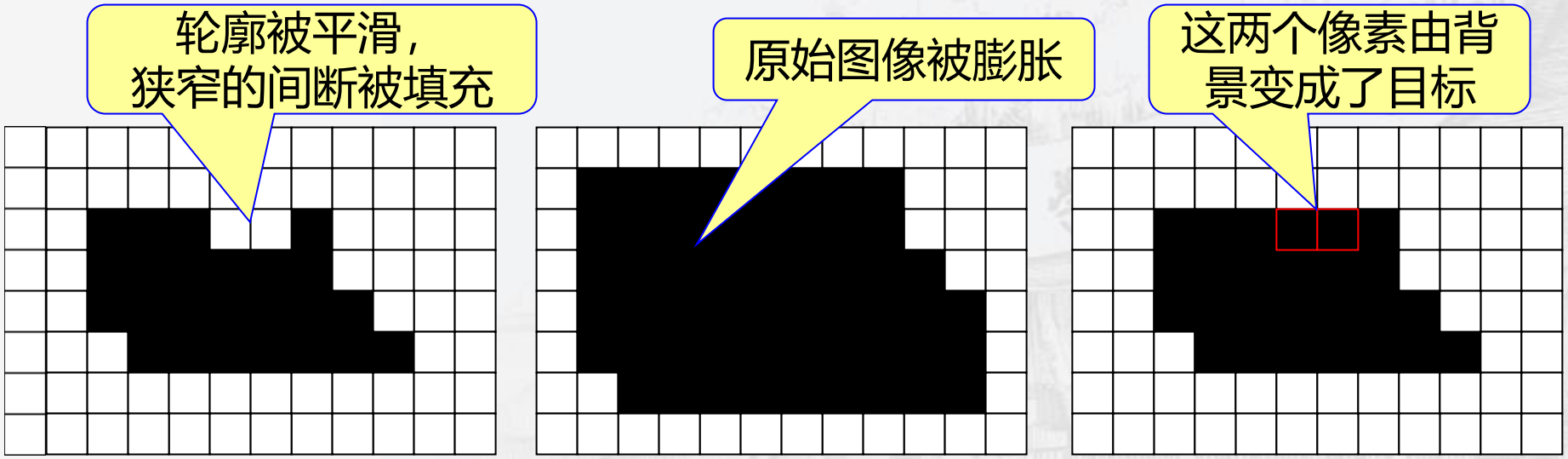
# 7.3.2 闭运算



闭运算过程

结构元:



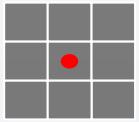


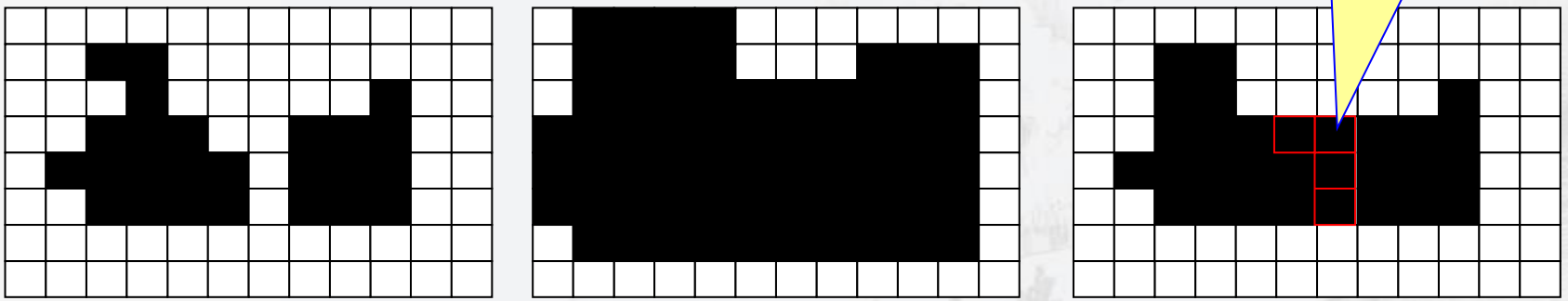
# 7.3.2 闭运算



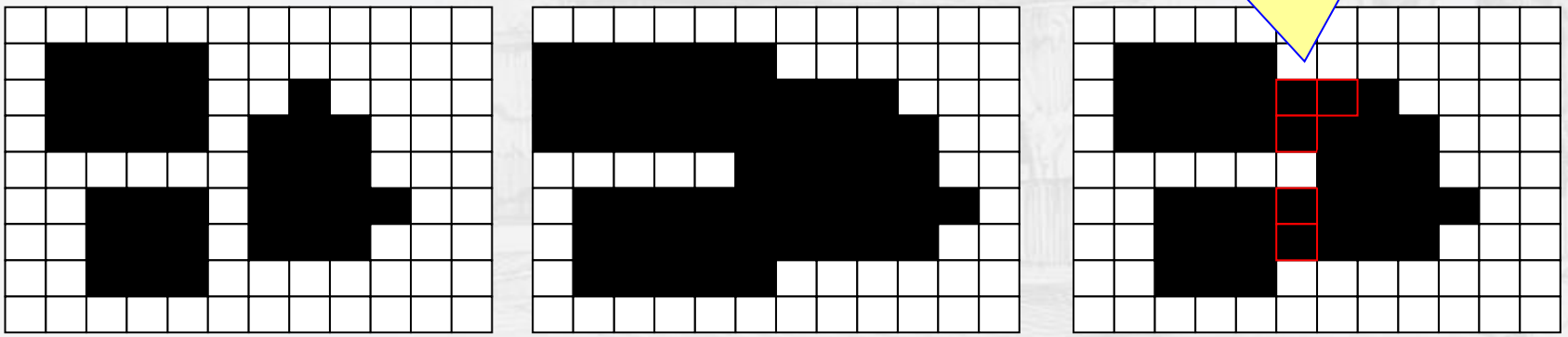
## 不同结构元的闭运算效果

结构元:






结构元:



# 7.3 开运算与闭运算



开、闭运算的性质：

- 对偶性：$(A \cdot B)^c = \left(A^c \circ \hat{B}\right)$　$(A \circ B)^c = \left(A^c \cdot \hat{B}\right)$

- 开运算：
  - $A \circ B$是$A$的一个子集；
  - 如果$C$是$D$的一个子集，则$C \circ B$是$D \circ B$的一个子集；
  - $(A \circ B) \circ B = A \circ B$

- 闭运算：
  - $A$是$A \cdot B$ 的一个子集；
  - 如果$C$是$D$的一个子集，则$C \cdot B$是$D \cdot B$的一个子集；
  - $(A \cdot B) \cdot B = A \cdot B$

# 7.3 开运算与闭运算



原图

a

b

1. 先膨胀 ->a
2. 膨胀后腐蚀 ->d
3. 先腐蚀 ->c
4. 腐蚀后膨胀 ->b

c

d

开运算使图像更加规则化，相比原图有较少的细节，一般会平滑物体的轮廓、断开较窄的狭颈并消除细的突出物。

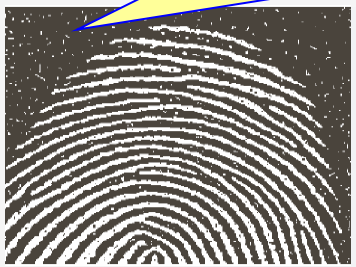
闭运算也会平滑轮廓，弥合较窄的间断和细长的沟壑，消除小的孔洞，填补轮廓线中的断裂。

# 7.3 开运算与闭运算



## 开、闭运算进行形态学滤波举例：指纹噪声消除

噪声为黑色背景上的亮元素和亮指纹部分的暗元素

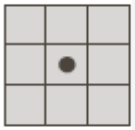


原始图像

开运算

闭运算

结构元



腐蚀

膨胀

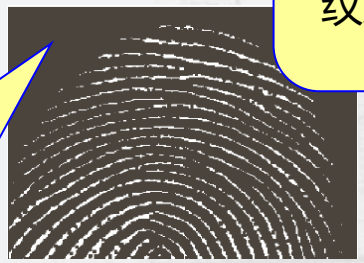
指纹中噪声增大部分被消除，但是纹路间产生间断

膨胀

腐蚀

消除了亮噪声和暗噪声，且纹路不受影响

黑色背景噪声消除，指纹中的噪声（黑点）尺寸增大



腐蚀后的图像

膨胀后的图像

大部分间断被恢复，指纹中噪声被去除，但是纹路变粗

# 7 形态学图像处理



# 7.4 击中-击不中变换



击中和击不中变换（Hit-or-Miss Transformation, HMT）是形态学形状检测的基本工具。令$I$是由前景$A$像素与背景像素组成的二值图像。不同于之前的形态学方法，HMT使用两个结构元：前景中检测形状用$B_1$，背景中检测形状用$B_2$。图像的HMT定义为：

$$I \circledast B_{1,2} = \{z \mid (B_1)_z \subseteq A \text{ and } (B_2)_z \subseteq A^c\}$$
$$= (A \ominus B_1) \cap (A^c \ominus B_2)$$

# 7.4 击中-击不中变换



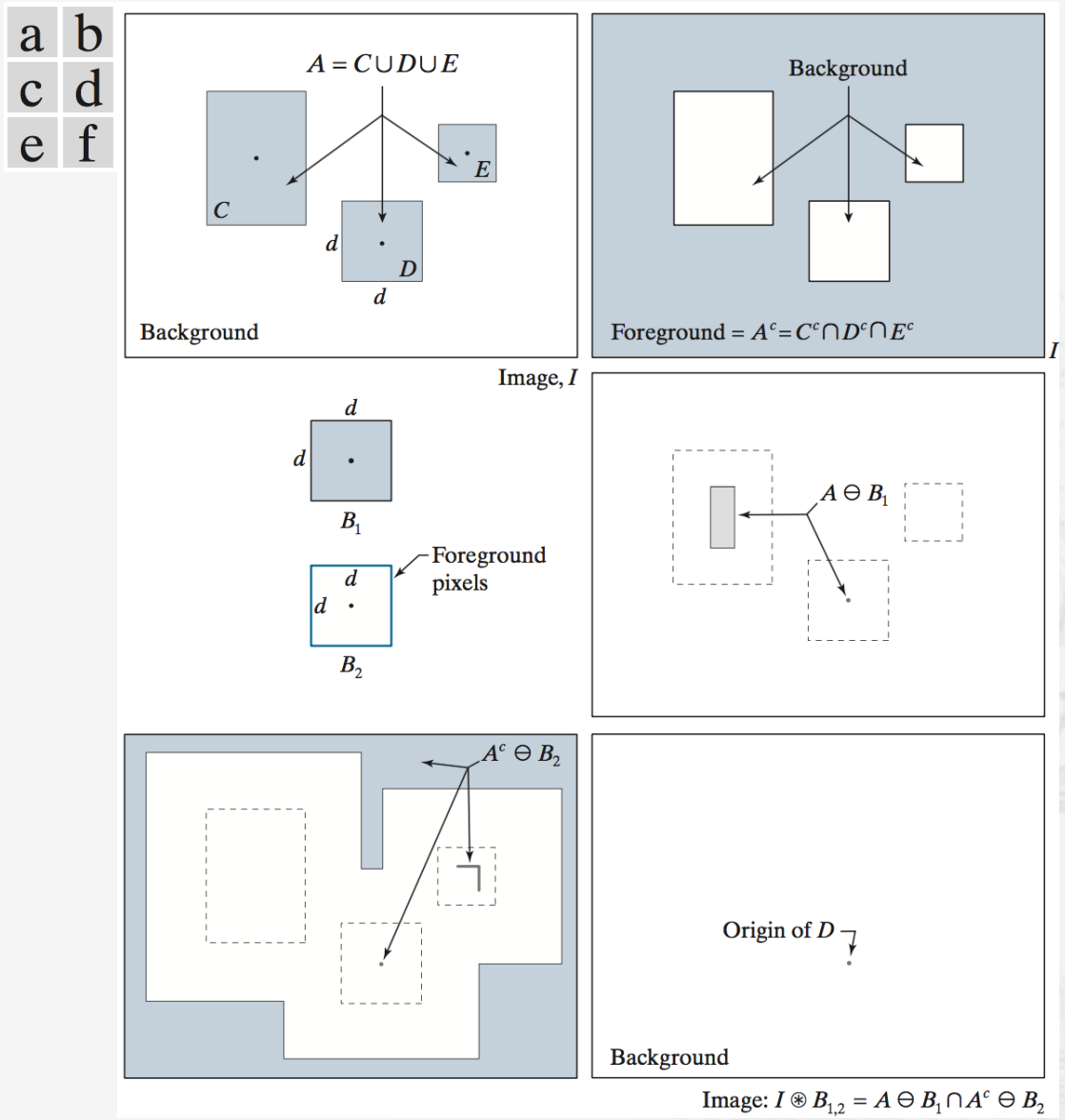
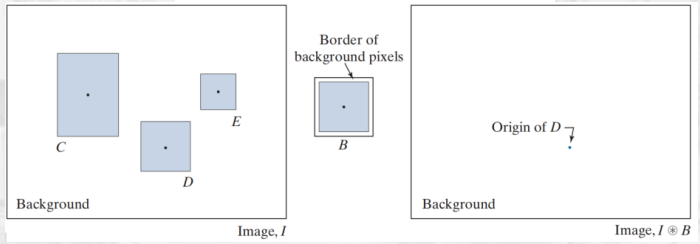



图7.4.1 (a)由前景（1值）组成的图像，是目标集合$A$和0值背景的并集；(b)前景定义为$A^C$的图像；(c)检测目标$D$的结构元；(d)$B_1$对$A$腐蚀；(e)$B_2$对$A^C$的腐蚀；(f) (d)与(e)的交集，该交集显示了$D$的原点位置。

B是对前景和背景都有要求的结构元

$$I \circledast B = \left\{ z \middle| (B)_z \subseteq I \right\}$$



# 7.4 击中-击不中变换



a b c
d e f
g h i

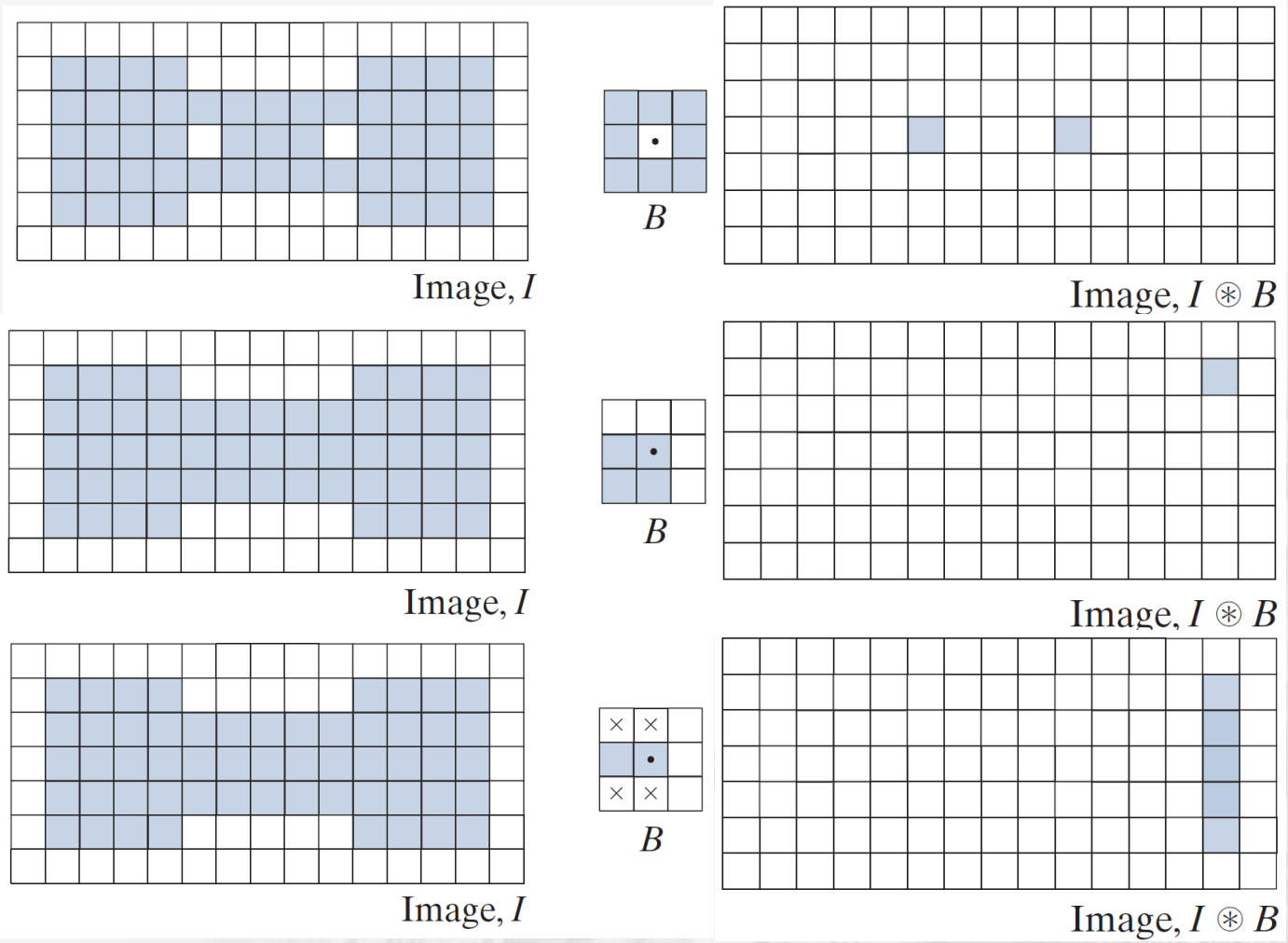



图7.4.2 使用单个结构元检测特定特征的3个例子。第一行：单像素孔检测；第二行：右上角的检测；第三行：多个特征检测。

# 7 形态学图像处理



7.1 预备知识

7.2 腐蚀与膨胀

7.3 开运算与闭运算

7.4 击中-击不中变换

**7.5 一些基本的形态学算法**

7.6 灰度级形态学

# 7.5 基本的形态学算法



7.5.1 边界抽取 (boundary extraction)

7.5.2 区域填充 (region filling)

7.5.3 连接分量提取 (extraction of connected components)

7.5.4 细化 (thinning)

7.5.5 粗化 (thickening)

7.5.6 骨架 (skeletons)

# 7.5.1 边界抽取



## 边界抽取

$\beta(A)$表示为集合$A$的边界，可以用某一合适的结构元$B$对$A$先进行腐蚀，然后再把$A$减去腐蚀的结果，得出边界抽取的结果，即：

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

图例说明，当结构元大小为3×3时，边界厚度为1。

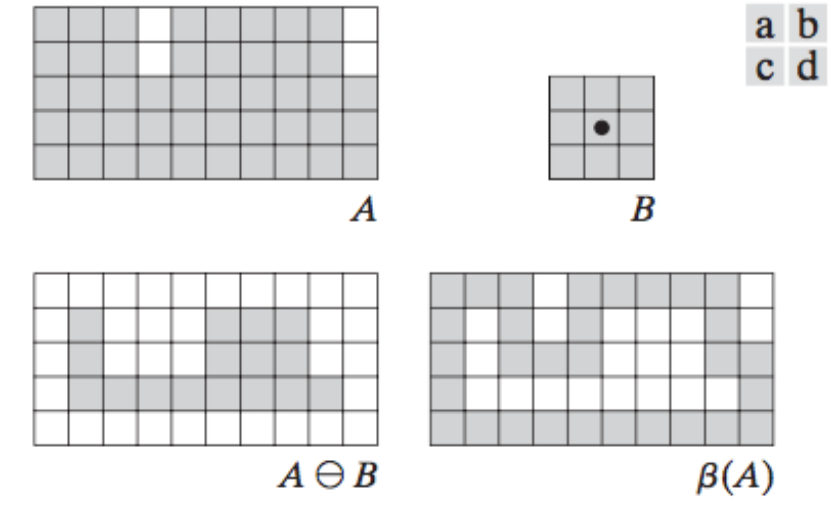
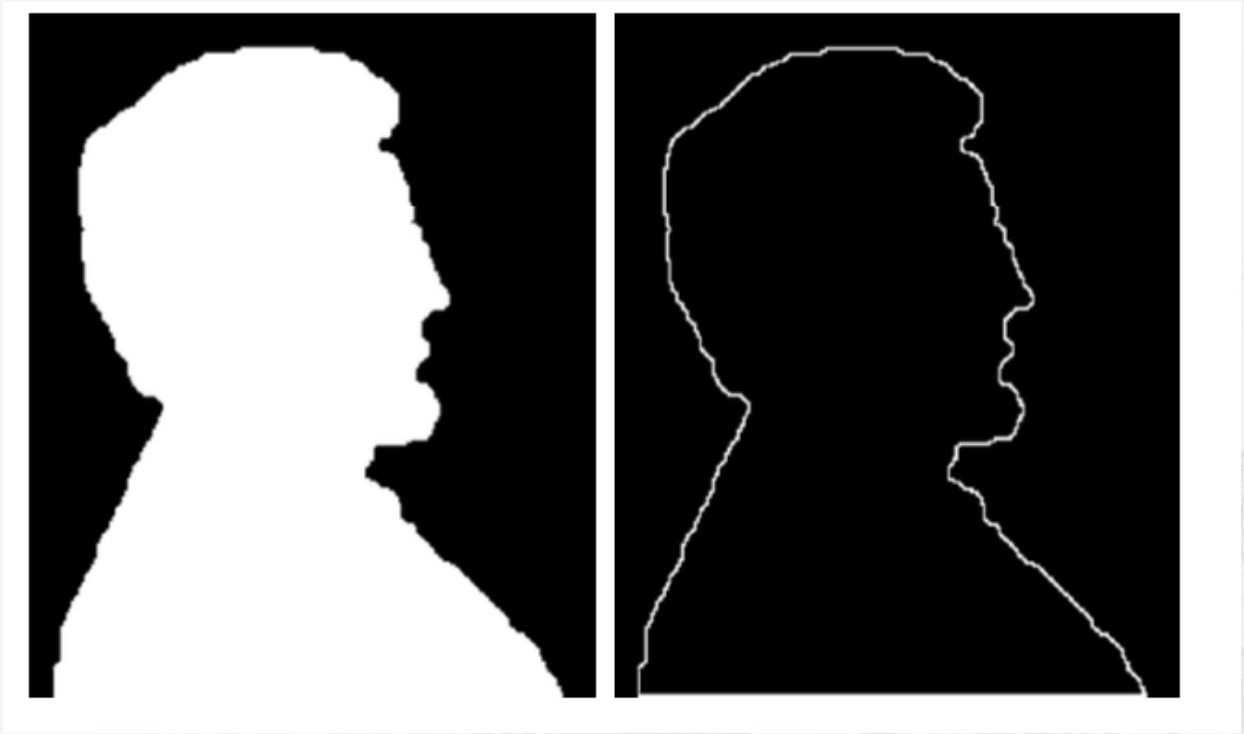



图7.5.1 (a)集合$A$ （b）结构元$B$；(c)$B$对$A$的腐蚀；(d)由A和其腐蚀结果的集合差给出的边界。

# 7.5.1 边界抽取



应用实例：人形上半身图像侧面轮廓提取



代码演示

# 7.5 基本的形态学算法



7.5.1 边界抽取 (boundary extraction)

7.5.2 区域填充 (region filling)

7.5.3 连接分量提取 (extraction of connected components)

7.5.4 细化 (thinning)

7.5.5 粗化 (thickening)

7.5.6 骨架 (skeletons)

# 7.5.2 区域填充

## 区域（孔洞）填充

基于膨胀、取补和交集的区域填充算法。当所需填充的区域是8连通的边界，先从界内初始区域$X_0$给定的一个点开始，用1去填充整个区域（设非边界元素为0），填充过程的公式如下：

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

式中：$B$为对称结构元，当$k$迭代到$X_k = X_{k-1}$时，算法终止，集合$X_k$和$A$的并集包含所有被填充的孔洞。上述过程每一步与$A^C$的交集把结果限制在我们感兴趣区域内（否则膨胀会一直进行，直至填满整个区域），所以上述过程也称条件膨胀。

# 7.5.2 区域填充



## 区域填充过程图示

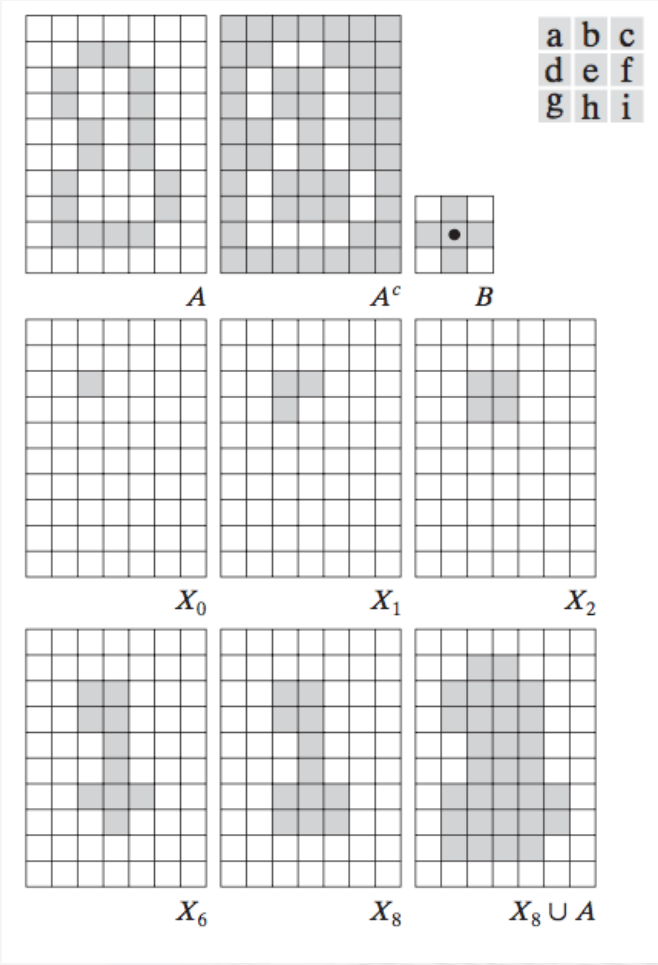



图7.5.2 孔洞填充：(a)集合$A$（显示为阴影）；(b) $A$ 的补集；(c)结构元$B$；(d)边界内的初始点；(e)~(h)填充过程；(i)最终结果[(a)和(h)的并集]。

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

# 7.5.2 区域填充



应用实例：消除球体二值扫描图像中心由于光放射造成的中心黑色区域。

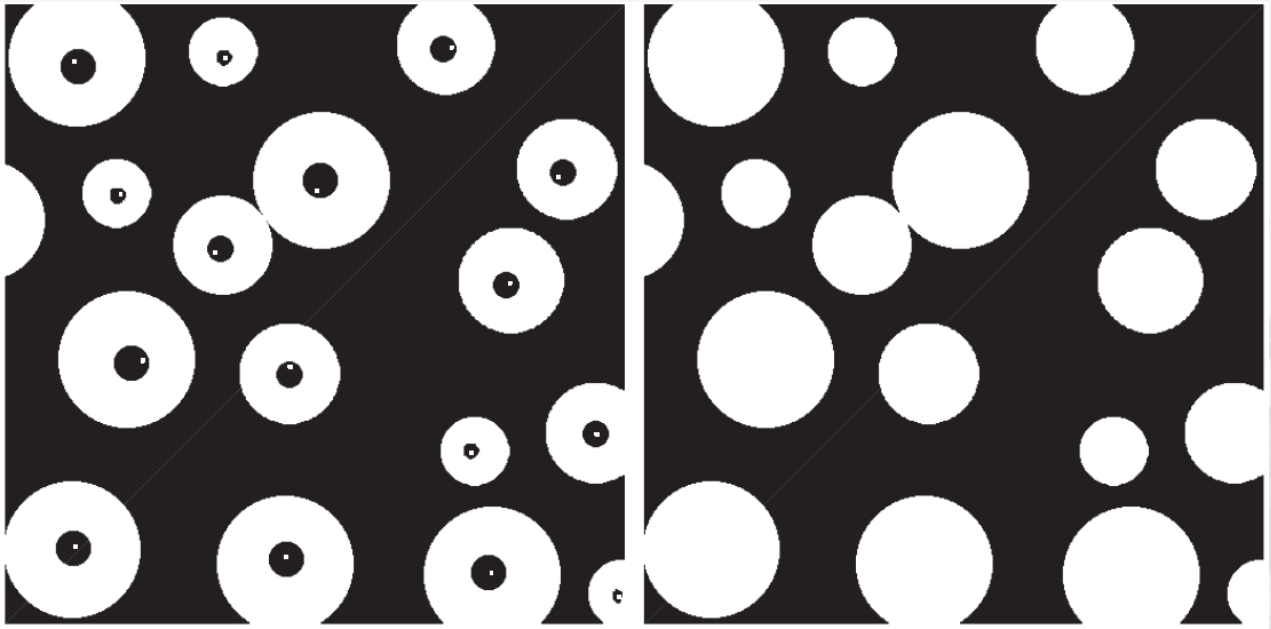


图7.5.3 (a)二值图像（区域内部的白点是孔洞填充算法的起始点）； (b)填充所有孔洞后的结果。

# 7.5 基本的形态学算法



7.5.1 边界抽取 (boundary extraction)

7.5.2 区域填充 (region filling)

7.5.3 连接分量提取 (extraction of connected components)

7.5.4 细化 (thinning)

7.5.5 粗化 (thickening)

7.5.6 骨架 (skeletons)

# 7.5.3 连接分量提取

## 连通分量的提取

从二值图像中提取连通分量是许多自动图像分析的核心。其公式如下：

$$X_k = (X_{k-1} \oplus B) \cap A$$

其中$B$是结构元，当$X_k = X_{k-1}$时，迭代结束。上述公式与形态学之区域填充公式只是用$A$代替了$A^c$。

# 7.5.3 连接分量提取



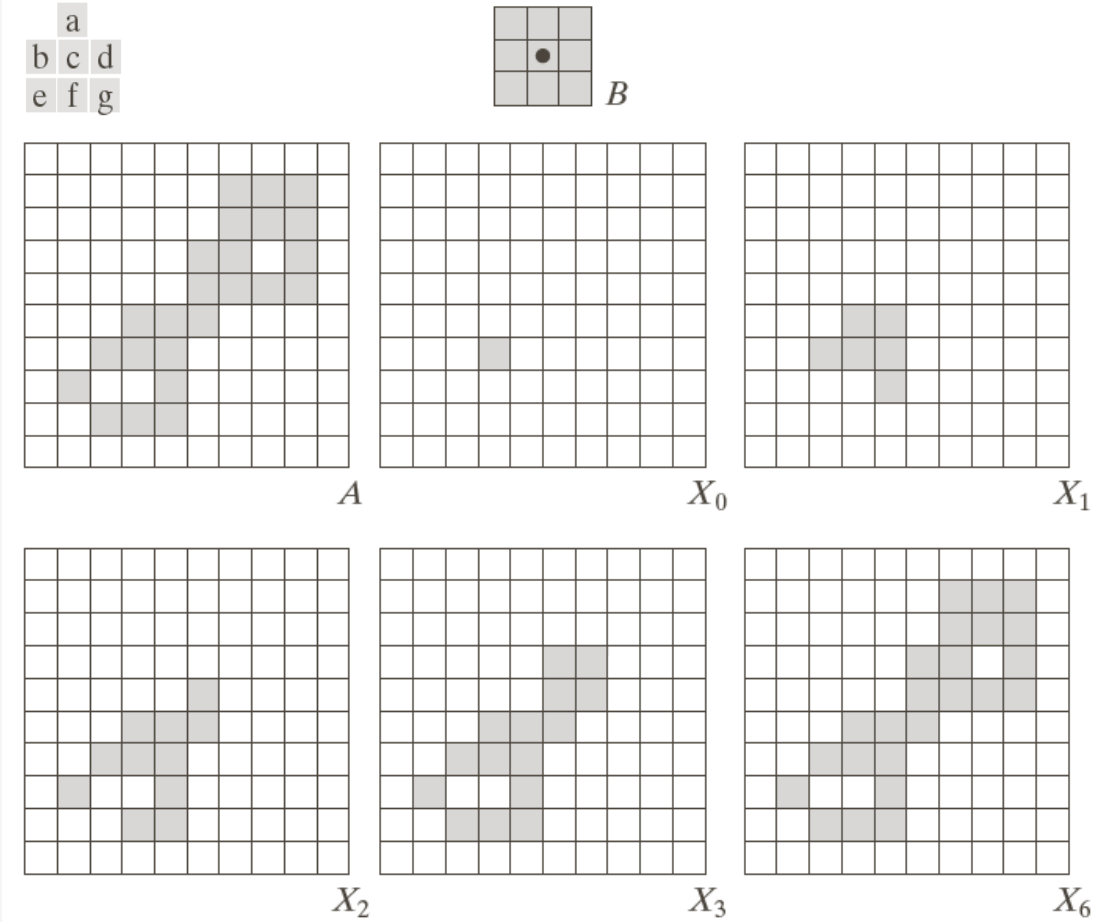
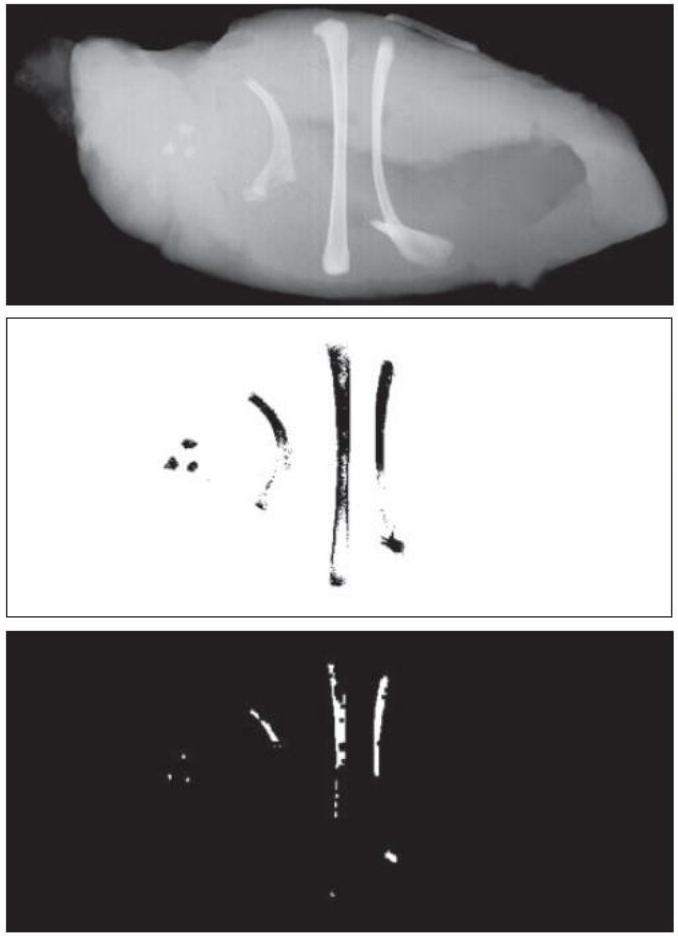



图7.5.4 提取连通分量:(a)结构元；(b)包含有一个连通分量的集合的阵列；(c)在连通分量的区域中包含一个1的初始阵列；(d)-(g) 各个迭代步骤。

# 7.5.3 连接分量提取



提取连通分量的应用：检测包装食品中的外来物



a
b
c d

| 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 |

图7.5.5 (a)含有碎骨的鸡肉x射线图像；(b)经阈值处理后的图像；(c)使用元素为1、大小为5×5的结构元腐蚀后的图像；（d）图(c)的连通分量中的像素数。

# 7.5 基本的形态学算法



7.5.1 边界抽取 (boundary extraction)

7.5.2 区域填充 (region filling)

7.5.3 连接分量提取 (extraction of connected components)

7.5.4 细化 (thinning)

7.5.5 粗化 (thickening)

7.5.6 骨架 (skeletons)

# 7.5.4 细化

## 细化

细化是通过去掉物体$A$的边界来实现的，它可以根据击中-击不中变换来定义：

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

一种更有用的基于结构元序列的对称细化算法为：

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

通常的细化是多次细化，每次细化的对象是上次细化的结果。即对所有结果元素操作一遍后，如果没有收敛，再依次对各个结构元重复进行运算，直至没有变化为止。

# 7.5.4 细化



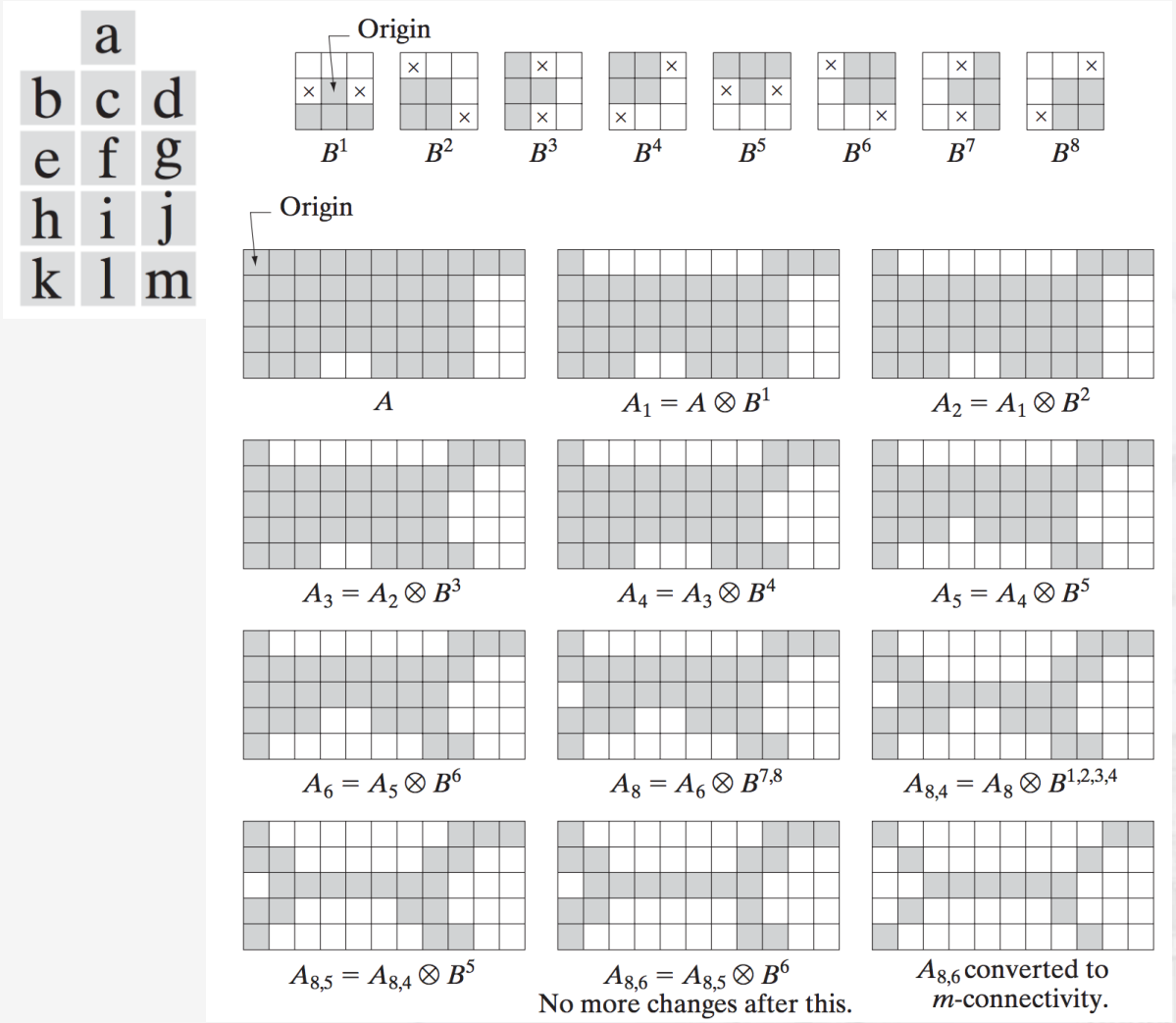



图7.5.6 (a)用于细化的经旋转后的结构元序列；(b)集合$A$；(c)使用第一个结构元细化得到结果；(d)~(g)使用接下来的7个结构元细化得到结果（使用第7个和第8个结构元细化得到的结果间没有变化）；(j)再次使用前4个结构元细化得到的结果；(l)收敛后的结果；(m)转换为m连通的结果。

# 7.5.4 细化



图像细化目的将原图像中线条宽度大于1个像素的线条细化成只有一个像素宽，能比较容易的分析图像，如提取图像的特征。

细化基本思想是“层层剥夺”，即从线条边缘开始一层一层向里剥夺，直到线条剩下一个像素的为止。

图像细化大大地压缩了原始图像的数据量，并保持其形状的基本拓扑结构不变，从而为文字识别中的特征抽取等应用奠定了基础。

代码演示

# 7.5 基本的形态学算法



7.5.1 边界抽取 (boundary extraction)

7.5.2 区域填充 (region filling)

7.5.3 连接分量提取 (extraction of connected components)

7.5.4 细化 (thinning)

7.5.5 粗化 (thickening)

7.5.6 骨架 (skeletons)

# 7.5.5 粗化

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

粗化

粗化是细化的对偶。

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

与细化算法类似，粗化算法也可用下面的序列操作方式：

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

其中所使用的结构元序列与细化算法一样，并把所有的1和0互换。

# 7.5.5 粗化



## 粗化算法过程

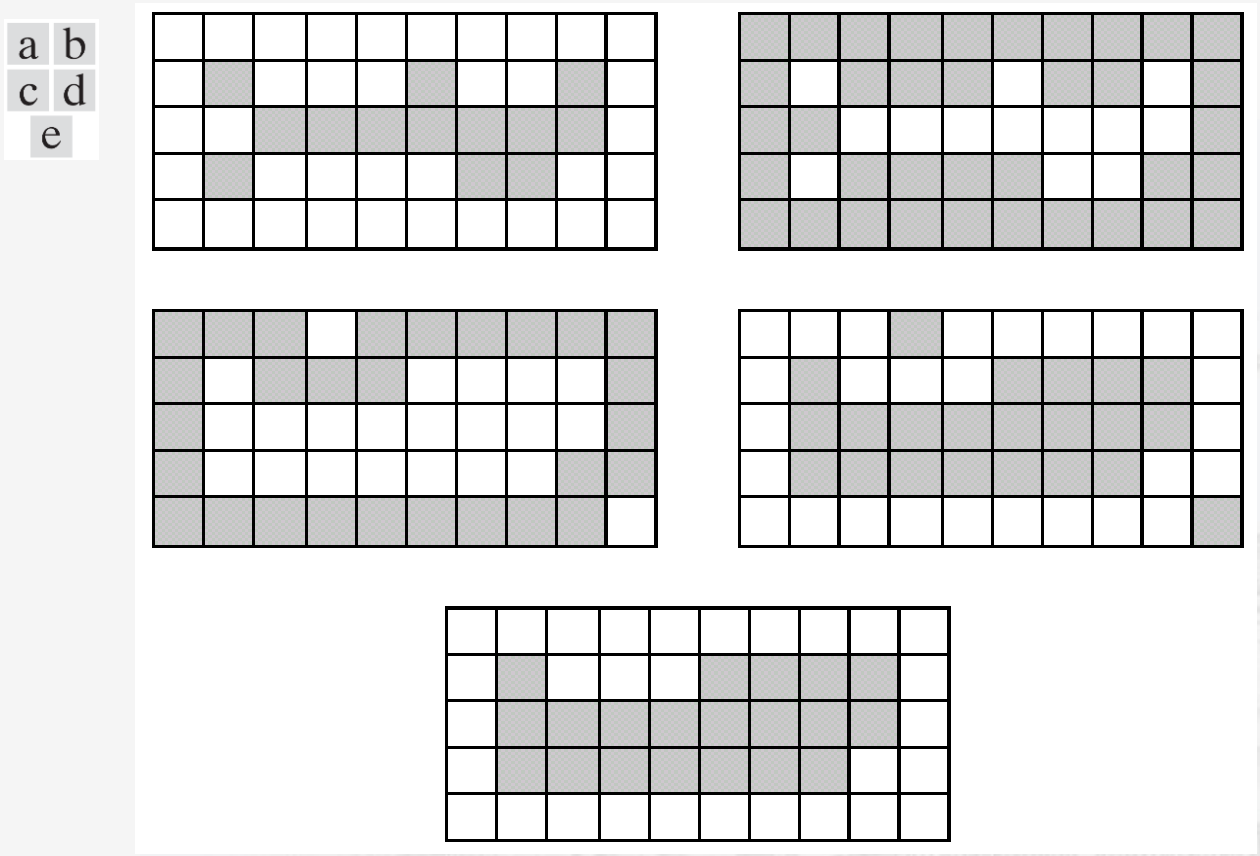



图7.5.7 (a)集合A；(b)A的补集；(c)对A的补集进行细化得到的结果；(d)对(c)求补得到的粗化后的集合；(e)没有断点的最终结果。

# 7.5 基本的形态学算法



7.5.1 边界抽取 (boundary extraction)

7.5.2 区域填充 (region filling)

7.5.3 连接分量提取 (extraction of connected components)

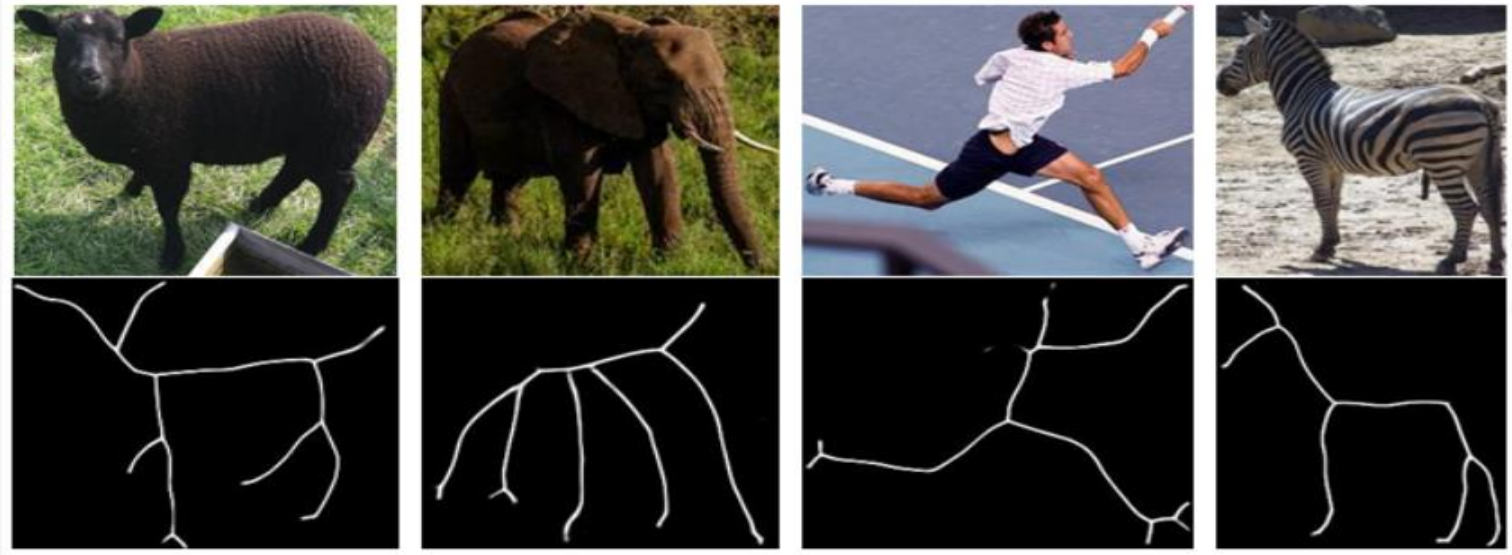
7.5.4 细化 (thinning)

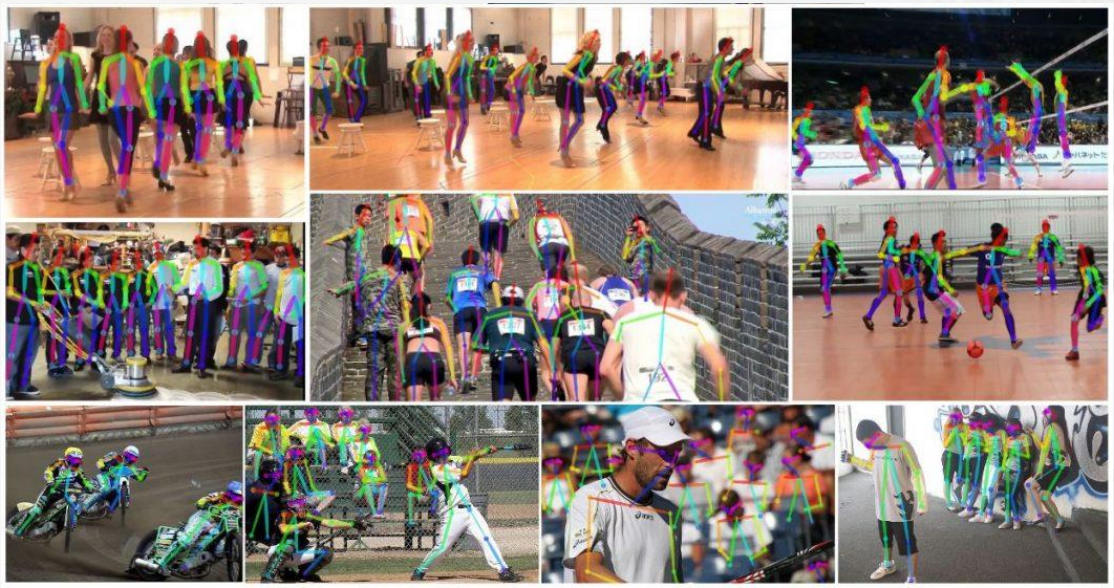
7.5.5 粗化 (thickening)

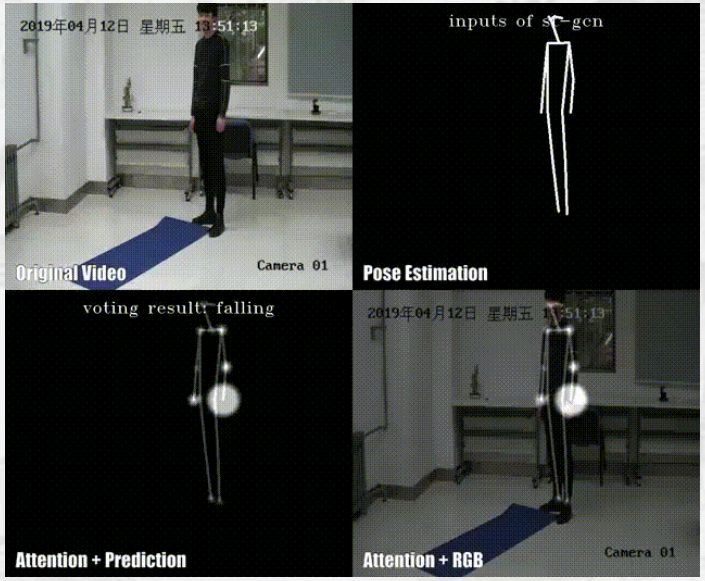
7.5.6 骨架 (skeletons)

# 7.5.6 骨架








Original Video
Camera 01
Pose Estimation
voting result: falling
Attention + Prediction
Attention + RGB
Camera 01

# 7.5.6 骨架



## 骨架

即目标的中轴线，中轴线的点（像素点）定义为距离目标边界上两个点等距的那些像素。

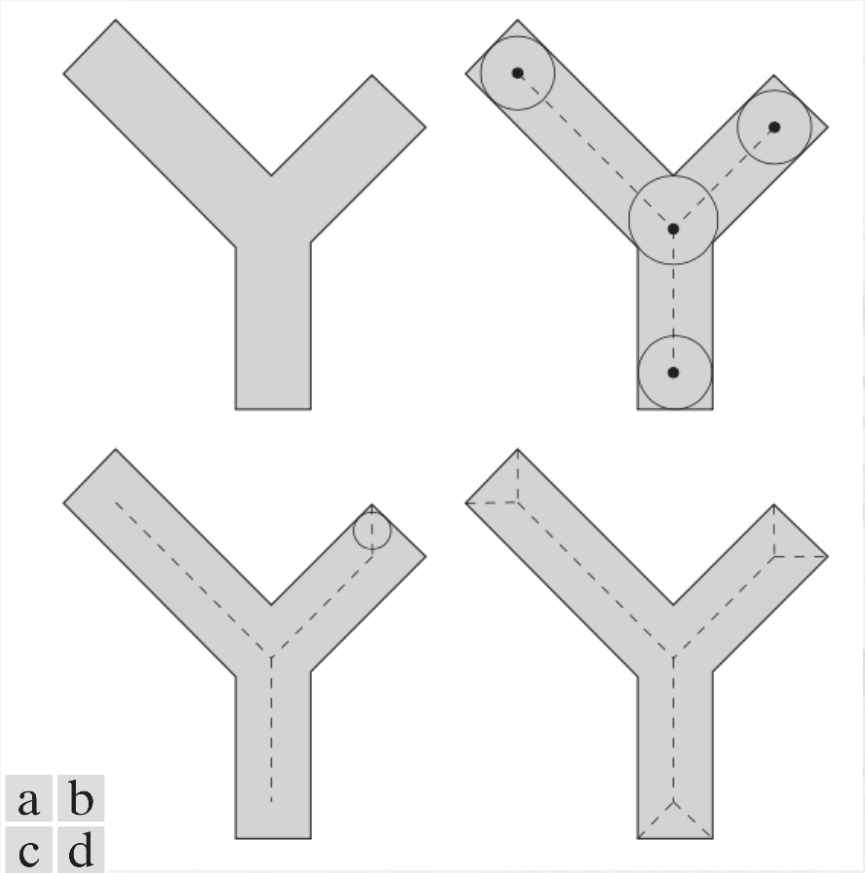



图7.5.8 (a)集合$A$；(b)中心在$A$的骨架上的最大圆盘的不同位置；(c)位于$A$的骨架的不同线段上的另一个最大圆盘；(d)完整的骨架。

# 7.5.6 骨架



集合$A$的骨架$S(A)$的概念上相当简单，由上图可以推出：

(a)如果$z$是$S(A)$的一个点，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（不必以$z$为中心）。圆盘$(D)_z$称为最大圆盘；

(b)若$(D)_z$是一个最大圆盘，它在两个或多个不同的位置与$A$的边界接触。

# 7.5.6 骨架



骨架可由腐蚀和开操作定义：$S(A) = \bigcup_{k=0}^{K} S_k(A)$

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

对$A$连续$k$次腐蚀：

$$(A \ominus kB) = ((\underbrace{\cdots(A \ominus B) \ominus B) \ominus \cdots) \ominus B}_{k\text{次}})$$

$K$是$A$被腐蚀为一个空集前最后一次迭代步骤：

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

由骨架重建A：$A = \bigcup_{k=0}^{K} (S_k(A) \oplus kB)$

# 7.5.6 骨架



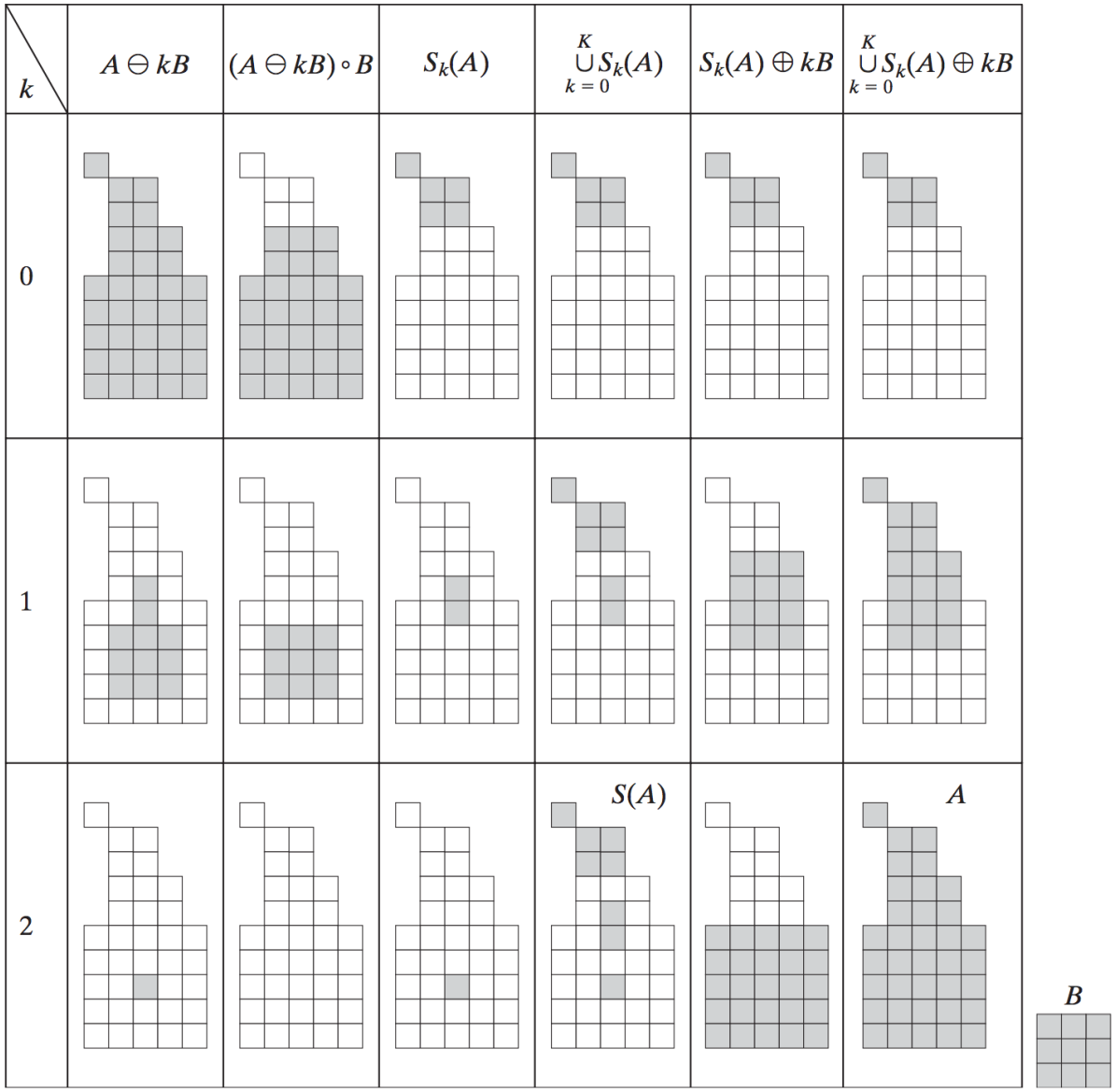



图7.5.9 骨架算法的执行过程。原始集合位于左上角，其形态学骨架位于第四列的底部。第六列底部为重建的集合。

# 7 形态学图像处理



7.1 预备知识

7.2 腐蚀与膨胀

7.3 开运算与闭运算

7.4 击中-击不中变换

7.5 一些基本的形态学算法

**7.6 灰度级形态学**

# 7.6 灰度级形态学



实际中遇到的图像更多是灰度图像，灰度形态学是二值形态学的自然扩展。灰度级图像由$f(x,y)$表示，$b(x,y)$ 表示为结构元。

灰度级形态学中的结构元所执行的基本功能与二值形态学中所对应的功能相同：它们作为一个“探测器"以明确的特性检验一幅给定的图像。灰度级形态学中的结构元属于两类：非平坦的和平坦的结构元。

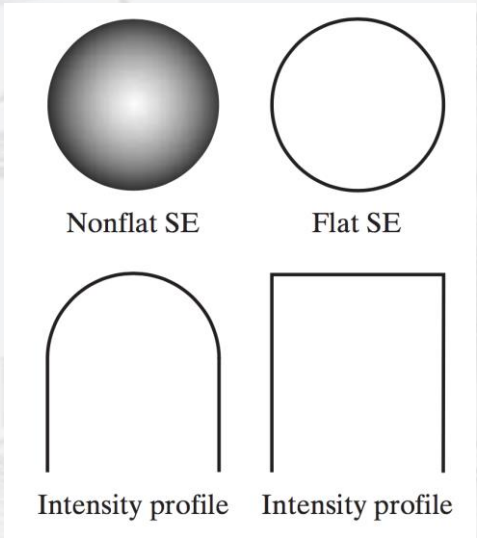



图7.6.1 非平坦和平坦结构元，以及对应的通过其中心的水平灰度剖面。本节中的所有例子均基于平坦结构元。

# 7.6 灰度级形态学



## 7.6.1 灰度腐蚀与膨胀

## 7.6.2 灰度开运算和闭运算

## 7.6.3 一些基本的灰度级形态学算法

# 7.6.1 灰度腐蚀与膨胀



平坦结构元$b$对一幅图像$f$在$(x,y)$处的腐蚀由下式给出：

$$[f \ominus b](x,y) = \min_{(s,t)\in b}\{f(x+s,\ y+t)\}$$

平坦结构元$b$对图像$f$的膨胀定义如下：

$$[f \oplus b](x,y) = \max_{(s,t)\in \hat{b}}\{f(x-s,y-t)\}$$

# 7.6.1 灰度腐蚀与膨胀



图7.6.2 (a)大小为448×425像素的灰度级x射线图像；(b)使用半径为2个像素的圆盘形平坦结构元对图像的腐蚀结果；(c)用相同结构元对图像的膨胀结果。

a b c

# 7.6.1 灰度腐蚀与膨胀



灰度腐蚀的效果：

- 腐蚀后的输出图像比输入图像更暗；
- 在比结构元还小的区域中的亮细节效果将减弱。

灰度膨胀的效果：

- 膨胀后的输出图像比输入图像更亮；
- 暗的细节被减少或消除。

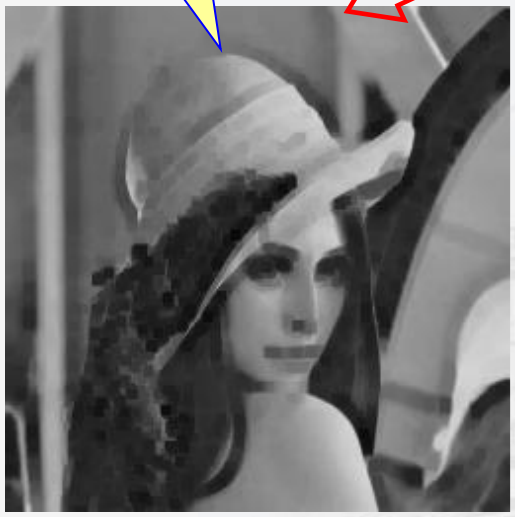
# 7.6.1 灰度腐蚀与膨胀



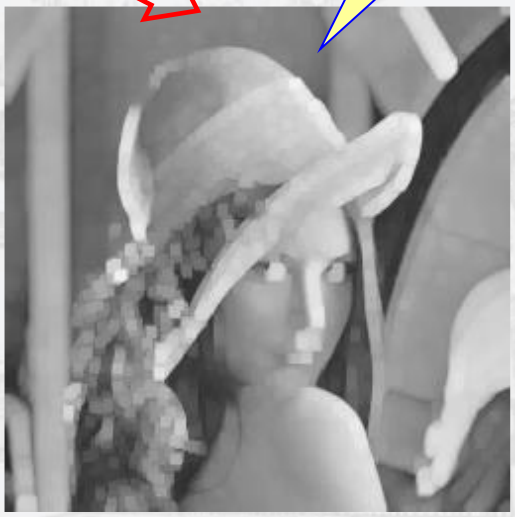
原始图像

腐蚀后，灰度值变小，图像变得更暗

膨胀后，灰度值变大，图像变得更亮



腐蚀后的图像



膨胀后的图像

# 7 形态学图像处理

7.6.1 灰度腐蚀与膨胀

**7.6.2 灰度开运算和闭运算**

7.6.3 一些基本的灰度级形态学算法

# 7.6.2 灰度开运算和闭运算

灰度级图像的开、闭操作的表达式与二值图像对应操作有相同形式，结构元$b$对图像$f$的开操作表示为$f \circ b$，即：

$$f \circ b = (f \ominus b) \oplus b$$

类似的，$b$对$f$的闭操作表示为 $f \bullet b$：

$$f \bullet b = (f \oplus b) \ominus b$$

开、闭运算也相对补集和结构元反射操作成对偶关系：

$$(f \bullet b)^c = f^c \circ \hat{b} \qquad (f \circ b)^c = f^c \bullet \hat{b}$$

# 7.6.2 灰度开运算和闭运算



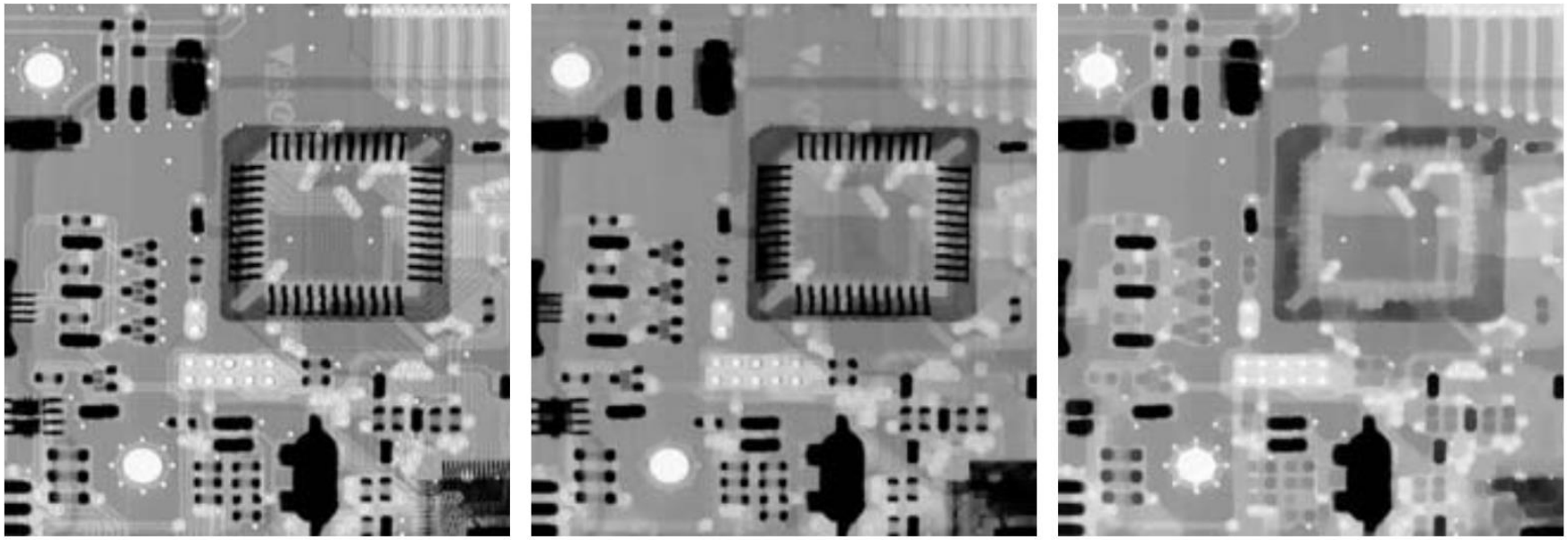
开、闭运算的效果

开运算通常用于去除小的（相对于结构元而言）亮细节，而保留总体的灰度特征不变。

闭运算通常用于去除小的（相对于结构元而言）暗细节，而保留总体的灰度特征不变。

# 7.6.2 灰度开运算和闭运算





a b c

图7.6.3 (a)大小为448×425像素的灰度级x射线图像；(b)使用半径为3个像素的圆盘形结构元对图像的开操作结果；(c)用5个像素对图像的闭操作结果。

7.6.1 灰度腐蚀与膨胀

7.6.2 灰度开运算和闭运算

**7.6.3 一些基本的灰度级形态学算法**

# 7.6.3 基本的灰度级形态学算法



一些基本的灰度级形态学算法：

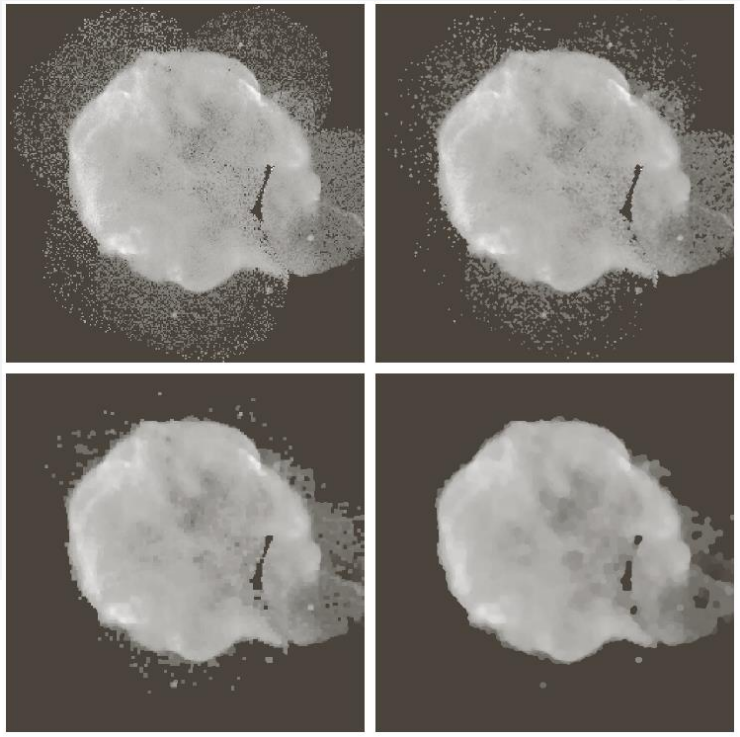
1. 形态学平滑
2. 纹理分割

# 7.6.3 基本的灰度级形态学算法



## 1. 形态学平滑

因为开操作抑制比结构元小的亮细节，闭操作抑制暗细节，所以常使用开运算后接闭运算操作，用于消除亮的和暗的伪迹或噪声。



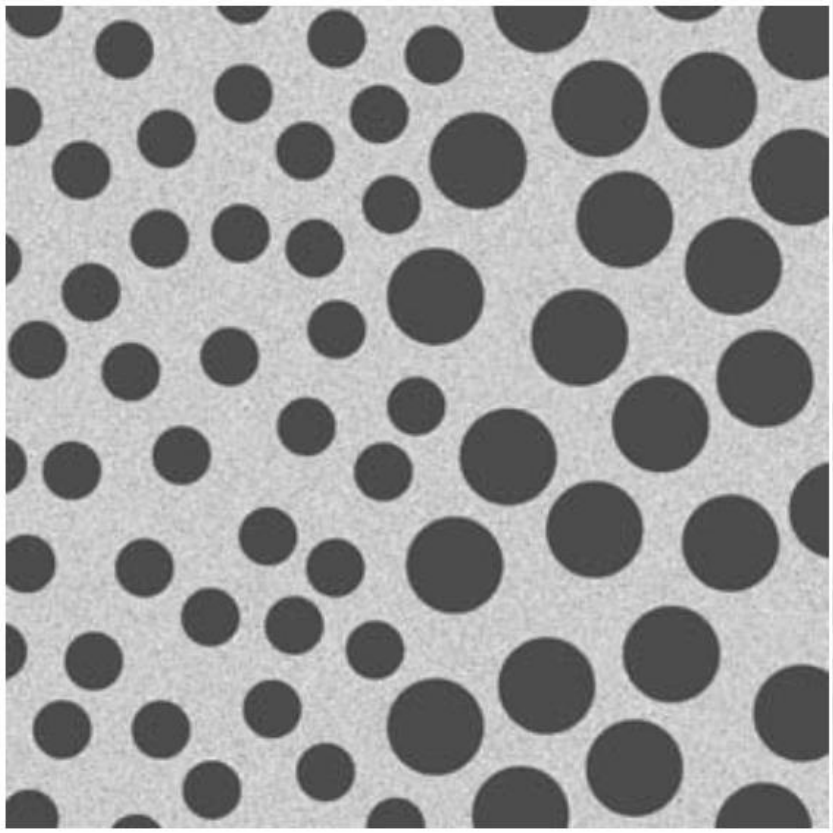
a b
c d

图7.6.4 (a)由NASA的哈勃望远镜在x射线波段拍摄的天鹅星座环超新星的566×566图像；(b)-(d)分别使用半径为1，3和5的圆盘形结构元对原图像执行开操作和闭操作的结果。

# 7.6.3 基本的灰度级形态学算法

## 2. 纹理分割

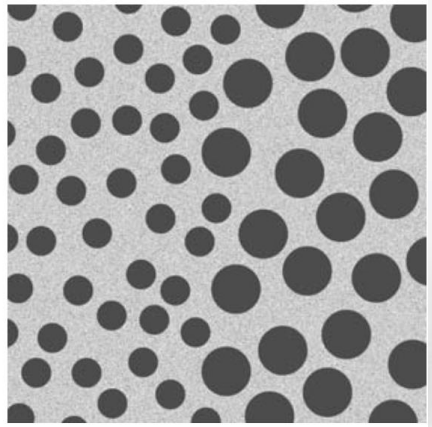
先看处理任务：分开下图中由不同大小圆球组成的两个纹理区域。



# 7.6.3 基本的灰度级形态学算法



步骤：



1. 用比小斑点稍大的结构元（此时左边的球相当于暗细节）对图像进行闭操作来消除小斑点，左边只留下亮背景，右边不变。

2. 用大于大斑点之间间距的结构元对上述结果做开运算，此时图像右边的背景区域（相当于亮细节）被消除，致使图像右边全成了黑色。这样就得到左边为白色，右边为黑色的二值图像。

# 7.6.3 基本的灰度级形态学算法



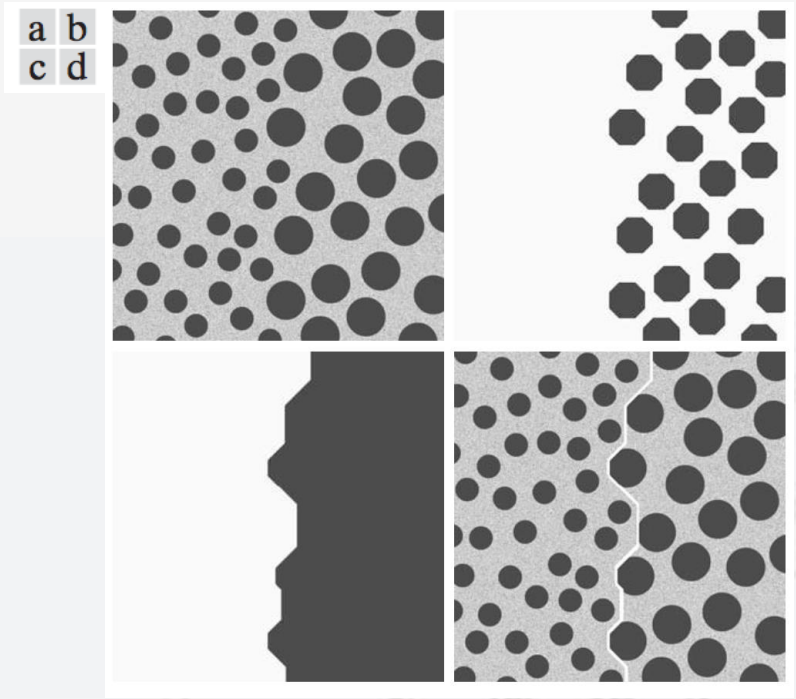



图7.6.5 纹理分割：(a)由两种斑点组成的大小为600×600的图像；(b)对图(a)执行闭操作后删除了小斑点的图像；(c)对图(b)执行开操作后删除了大斑点间的亮间隔的图像；(d)将图(c)中两个区域的边界叠加到原图像上的结果。


西安交通大學
XI'AN JIAOTONG UNIVERSITY


# 本章结束