



第9章 数学形态学原理

(第一讲)

引入

形态学——通常代表生物学的一个分支，它是研究动植物的形态和结构的学科。

数学形态学 (Mathematical Morphology) ——应用于图像处理和模式识别领域的新的方法：把图像看成是点的集合，用集合论的各种观点来研究图的性质就是数学形态学。其理论基础是集合论，在数学形态学中用集合表示图像中的不同对象。

数学形态学图像处理——使用数学的方法，研究数字图像中的物体的形态和结构的图像处理方法。

数学形态学中集合如何代表图像中物体的形状？

如：在二进制图像中所有黑色像素点的集合就是对这幅图像的完整描述，当前集合指二维整形空间的成员，集合中的每个元素都是一个二维变量，用 (x, y) 表示。

灰度数字图像可以用三维集合来表示。集合中每个元素的前两个变量用来表示像素点的坐标，第三个变量代表离散的灰度值。


在更高维数的空间集合中可以包括其它的图像属性，如颜色和时间。

9.1 数学形态学的发展

“数学形态学”的历史可追溯到十九世纪的Eular. steiner. Crofton和本世纪的Minkowski。1964年法国学者J. Serra对铁矿石的岩相进行了定量分析以预测铁矿石的可轧性。同时G. Matheron研究了多孔介质的几何结构、渗透性及两者的关系，其研究成果直接导致“数学形态学”雏形的形成。


J. Serra和 G. Matheron在法国(巴黎矿业学院)共同建立了枫丹白露 (Fontainebleau) 数学形态学研究中心。此后，他们逐步建立并完善了“数学形态学”理论体系，研究了基于数学形态学的图像处理系统。






“数学形态学”建立在严格的数学理论基础上。
G. Matheron 于1973年出版的《Ensembles aleatoires geometrie integrale》一书严谨而详尽地论证了随机集论和积分几何，为数学形态学奠定了理论基础。

1982年，J. Serra出版的专著《Image Analysis and Mathematical Morphology》是数学形态学发展的里程碑，它表明数学形态学在理论上已趋于完备，在实际应用中不断深入。




此后，经过科学工作者的不断努力，J. Serra 主编的《Image Analysis and Mathematical Morphology》Volume2、Volume3相继出版，1986年，CVGIP（Computer Vision Graphics and Image Processing）发表了数学形态学专辑，从而使得数学形态学的研究呈现了新的景象。同时，枫丹白露研究中心的学者们又相继提出了基于数学形态学方法的纹理分析模型系列，从而使数学形态学的研究前景更加光明。




随着数学形态学逻辑基础的发展，其应用开始向边缘学科和工业技术方面发展。数学形态学的应用领域已不限于传统的微生物学和材料学领域，80年代初又出现了几种新的应用领域：

如工业控制、放射医学、运动场景分析等。数学形态学在我国的应用研究也很快，目前，已研制出一些以数学形态学为基础的实用图像处理系统，如：中国科学院生物物理研究所和计算机技术研究所负责，由软件研究所、电子研究所和自动化所参加研究的癌细胞自动识别系统等。



数学形态学综合了多学科知识，其**理论基础颇为艰深，但其基本观念却比较简单**。它体现了逻辑推理与数学演绎的严谨性，又要求具备与实践密切相关的实验技术与计算技术。它涉及微分几何、积分几何、测度论、泛函分析和随机过程等许多数学理论。总之，数学形态学是建立在**严格的数学理论基础**上而又密切联系实际的科学。



用于描述数学形态学的语言是**集合论**, 它可以提供一个统一而强大的工具来处理图像处理中所遇到的问题。利用数学形态学对物体几何结构的分析过程就是**主客体相互逼近**的过程。利用数学形态学的几个基本概念和运算, 将结构元灵活地组合、分解, 应用形态变换序列达到分析的目的。


- 数学形态学的研究问题的方法是试探!?
- 数学形态学图像处理的基本思想, 是利用一个称作结构元(素)的“探针”在图像中不断移动, 边移动边试探, 从而获得图像的信息, 并做出相应的处理.
- 什么是结构元(素)? 一个集合
- 试探什么? 图像上的点
- 如何试探? 由要处理的问题决定, 测试合格的点并入结果集, 否则舍弃该点。



利用数学形态学进行图像分析的基本步骤有：

- 1) 提出所要描述的物体几何结构模式，即提取物体的几何结构特征；
- 2) 根据该模式选择相应的结构元，结构元应该简单而对模式具有最强的表现力；
- 3) 用选定的结构元对图像进行击中与否（HMT）变换，便可得到比原始图像显著突出物体特征信息的图像。如果赋予相应的变量则可得到该结构模式的定量描述；
- 4) 经过形态变换后的图像突出了我们需要的信息，此时，就可以方便地提取信息；

数学形态学的核心运算是击中与否变换 (hit or miss transform ,HMT) ，在定义了HMT及其基本运算膨胀 (Dilation) 和腐蚀 (Erosion) 后，再从积分几何和体视学移植一些概念和理论，根据图像分析的各种要求，构造出统一的、相同的或变化很小的结构元素进行各种形态变换加以试探。在形态算法设计中，结构元的选择十分重要，其形状、尺寸的选择是能否有效地提取信息的关键。



一般情况，结构元的选择本着如下几个原则进行：

1) 结构元必须在几何上比原图像简单，且有界。当选择性质相同或相似的结构元时，以选择极限情况为益；

2) 结构元的凸性非常重要，对非凸子集，由于连接两点的线段大部分位于集合的外面，故而用非凸子集作为结构元将得不到什么信息。

总之，数学形态学的基本思想和基本研究方法具有一些特殊性，掌握和运用好这些特性是取得良好结果的关键。

9.2 数学形态学的基本概念和运算

在数学意义上，我们用形态学来处理一些图像，用以描述某些区域的形状如边界曲线、骨架结构和凸形外壳等。另外，我们也用形态学技术来进行预测和快速处理如形态过滤，形态细化，形态修饰等。而这些处理都是基于一些基本运算实现的。



9. 2. 1 数学形态学定量分析原则

9. 2. 2 数学形态学的基本定义及 基本算法

➤ 一些基本的定义

(1) 集合：具有某种性质的确定的有区别的事物的全体。如果某种事物不存在称为空集。集合常用大写字母 A, B, C, \dots 表示，空集用 ϕ 表示。

设 E 为一自由空间， $\wp(E)$ 是由集合空间 E 所构成的幂集，集合 $X, B \in \wp(E)$ ，则集合 X 和 B 之间的关系只能有以下包含、击中、相离三种形式：

- ①、集合 B 包含于 X （表示为 $B \subset X$ ）
- ②、集合 B 击中 X （表示为 $B \uparrow X$ ），即： $B \cap X \neq \emptyset$
- ③、集合 B 相离于 X （表示为 $B \subset X^c$ ），即：

$$B \cap X = \emptyset$$

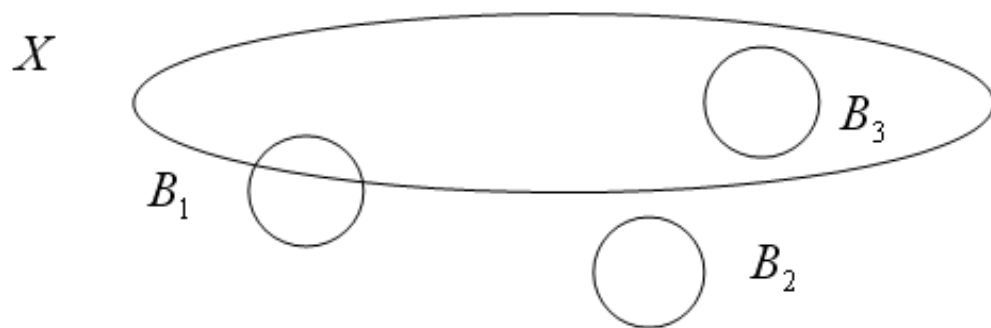


图 9-1 B_1 击中 X , B_2 相离于 X , B_3 包含于 X

(2) 元素：构成集合的每一个事物称之为元素，元素常用小写字母 a, b, c, \dots 表示，应注意的是任何事物都不是空集的元素。

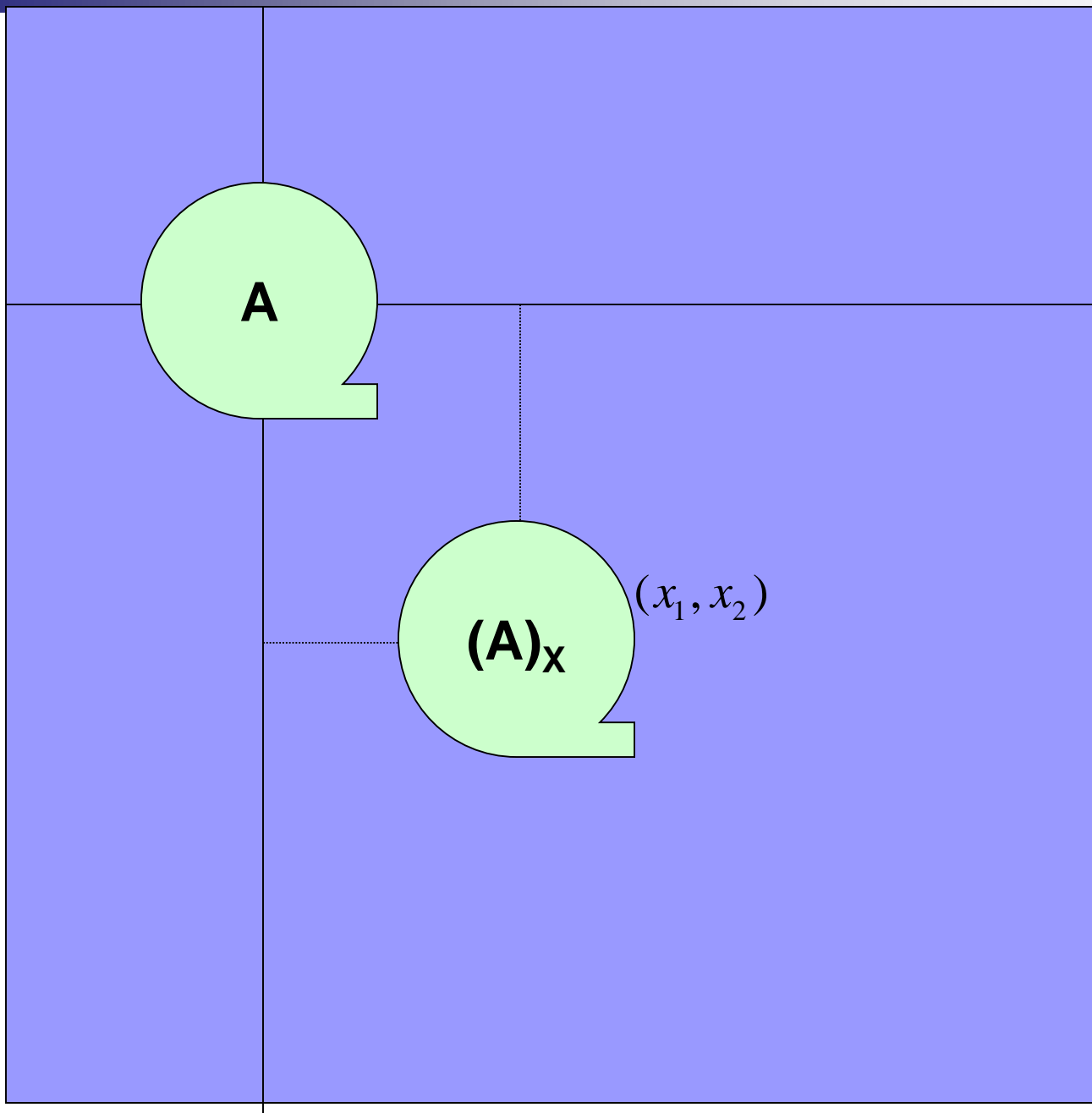
(3) 平移转换：

设A和B是两个二维集合，A和B中的元素分别是

$$a = (a_1, a_2), \quad b = (b_1, b_2)$$

定义 $x = (x_1, x_2)$ ，对集合的平移转换为：

$$(A)_x = \{c \mid c = a + x, \text{ for } a \in A\} \quad (9-8)$$



(4) 子集：当且仅当A集合的所有元素都属于B时，称A为B的子集。

(5) 补集：定义集合A的补集为：

$$A^c = \{x | x \notin A\} \quad (9-9)$$

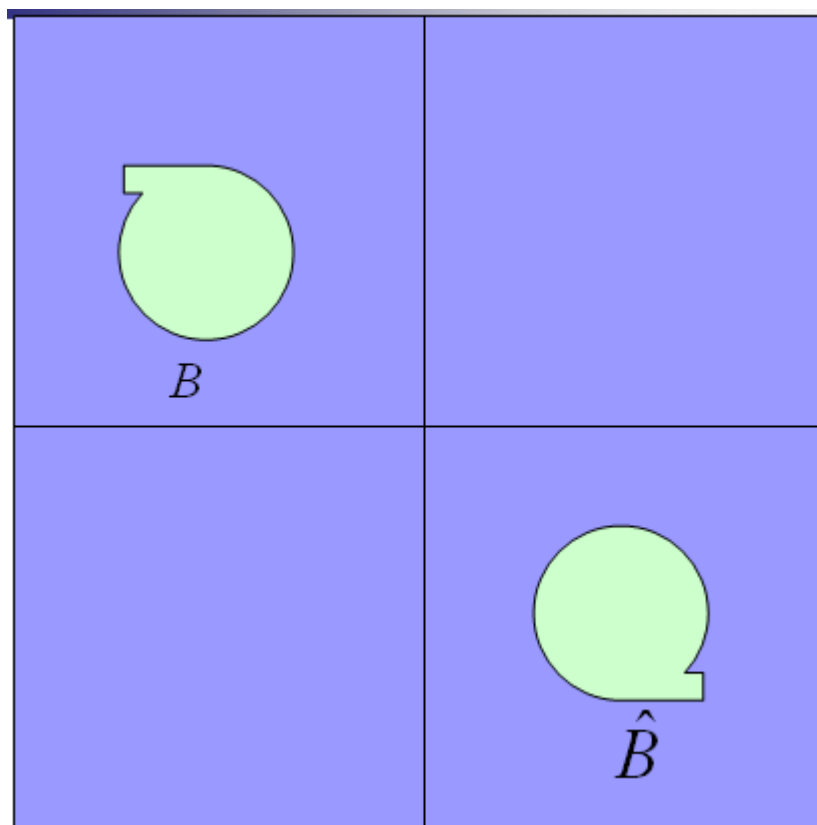
(6) 差集：定义集合A和B的差集为

$$A - B \quad (9-10)$$

$$A - B = \{x | x \in A, x \notin B\} = A \cap B^c \quad (9-11)$$

(7) 映像：定义集合B的映像为 \hat{B}

$$\hat{B} = \{x \mid x = -b, b \in B\} \quad (9-12)$$

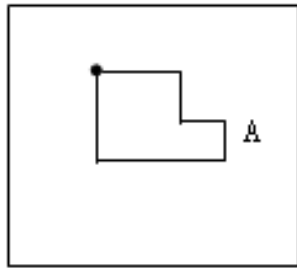


(8) 并集：由A和B的所有元素组成的集合称为A和B的并集。 $C = A \cup B$

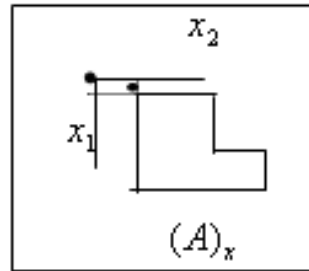
(9) 交集：由A和B的公共元素组成的集合称为A和B的交集。 $C = A \cap B$

图9-2解释了刚才几个定义，图中的黑点为集合的原点。图9-2(a)显示集合A；图9-2(b)表示A被平移，注意平移是在A的每个元素上加上 $x = (x_1, x_2)$ 。图9-2(c)表示集合B；图9-2(d)显示了B关于原点的反转。最后，图9-2(e)显示了集合A及其补，图9-2(f)显示了图9-2(e)的集合A与图9-2(f)中的集合B的差。

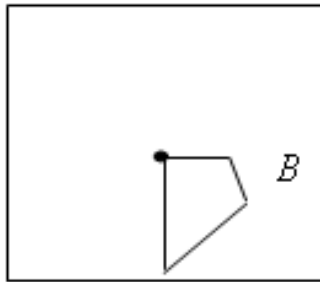
图9-2



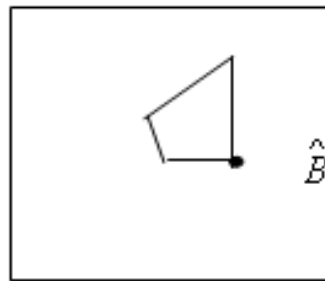
(a)



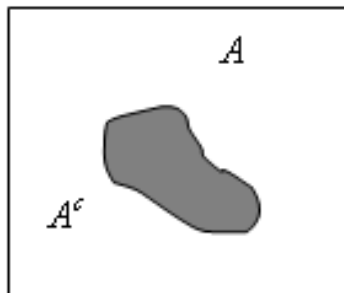
(b)



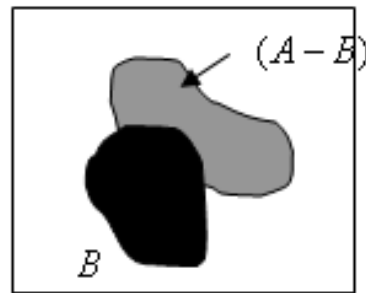
(c)



(d)



(e)



(f)

- (a) 集合A;
(b) 用 x 平移集合A后的结果;
(c) 集合B;
(d) B的反转;
(e) 集合A和它的补集;
(f) 两个集合的差集(如阴影所示)。

前四幅图的黑点表示了每个集合的起点。

补充：二值图像的逻辑运算

1. 主要逻辑运算

TABLE 9.1

The three basic
logical operations.

p	q	p AND q (also $p \cdot q$)	p OR q (also $p + q$)	NOT (p) (also \bar{p})
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

2. 二值图像的基本逻辑运算

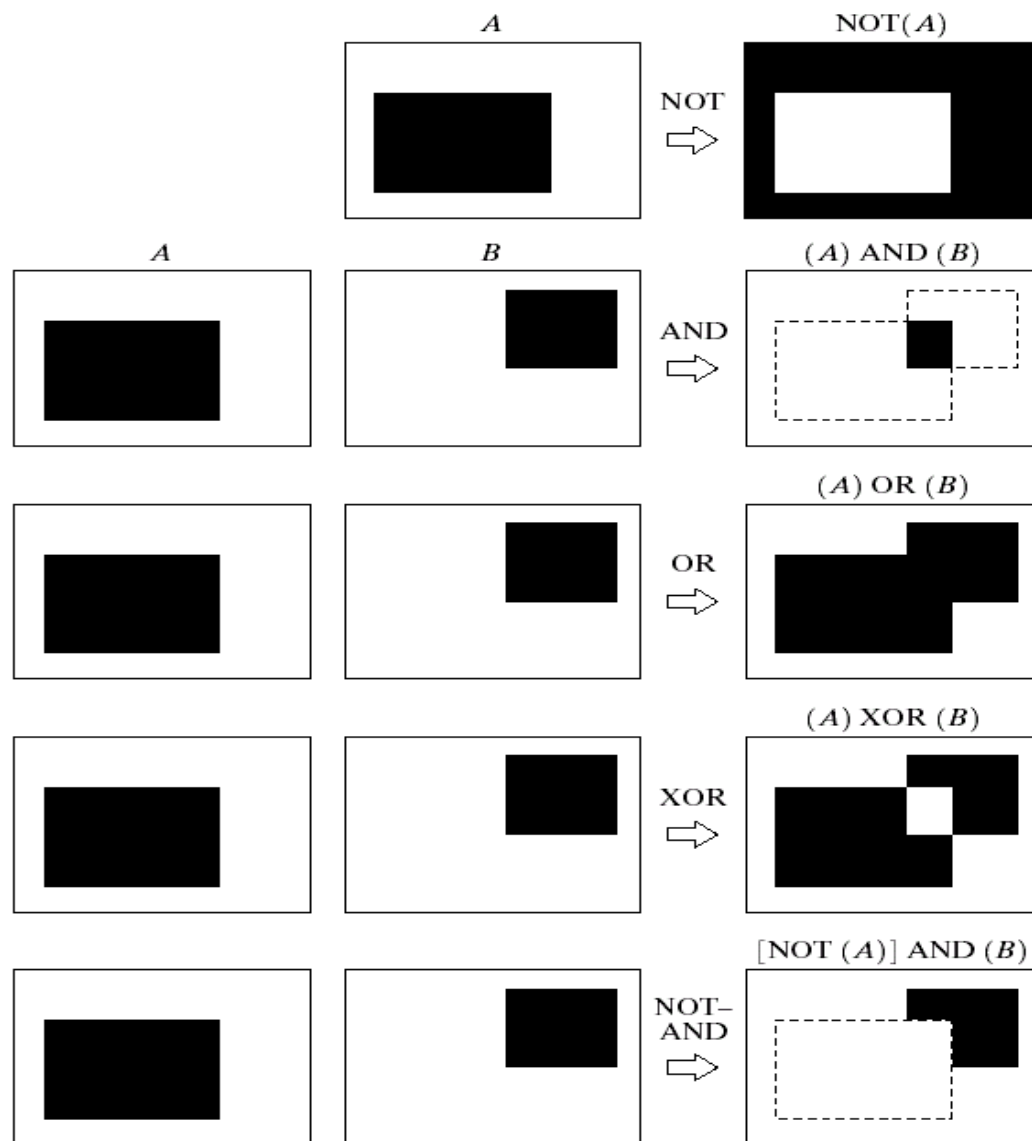


FIGURE 9.3 Some logic operations between binary images. Black represents binary 1s and white binary 0s in this example.

➤ 膨胀

A, B 为 Z^2 中的集合, \emptyset 为空集, A 被 B 的膨胀, 记为 $A \oplus B$, \oplus 为膨胀算子, 膨胀的定义为:

$$A \oplus B = \{ x \mid [(\hat{B})_x \cap A] \neq \emptyset \} \quad (9-12)$$

膨胀过程: B 首先做关于原点的映射, 然后平移 x 。 A 被 B 的膨胀是 \hat{B} 被所有 x 平移后与 A 至少有一个非零公共元素。

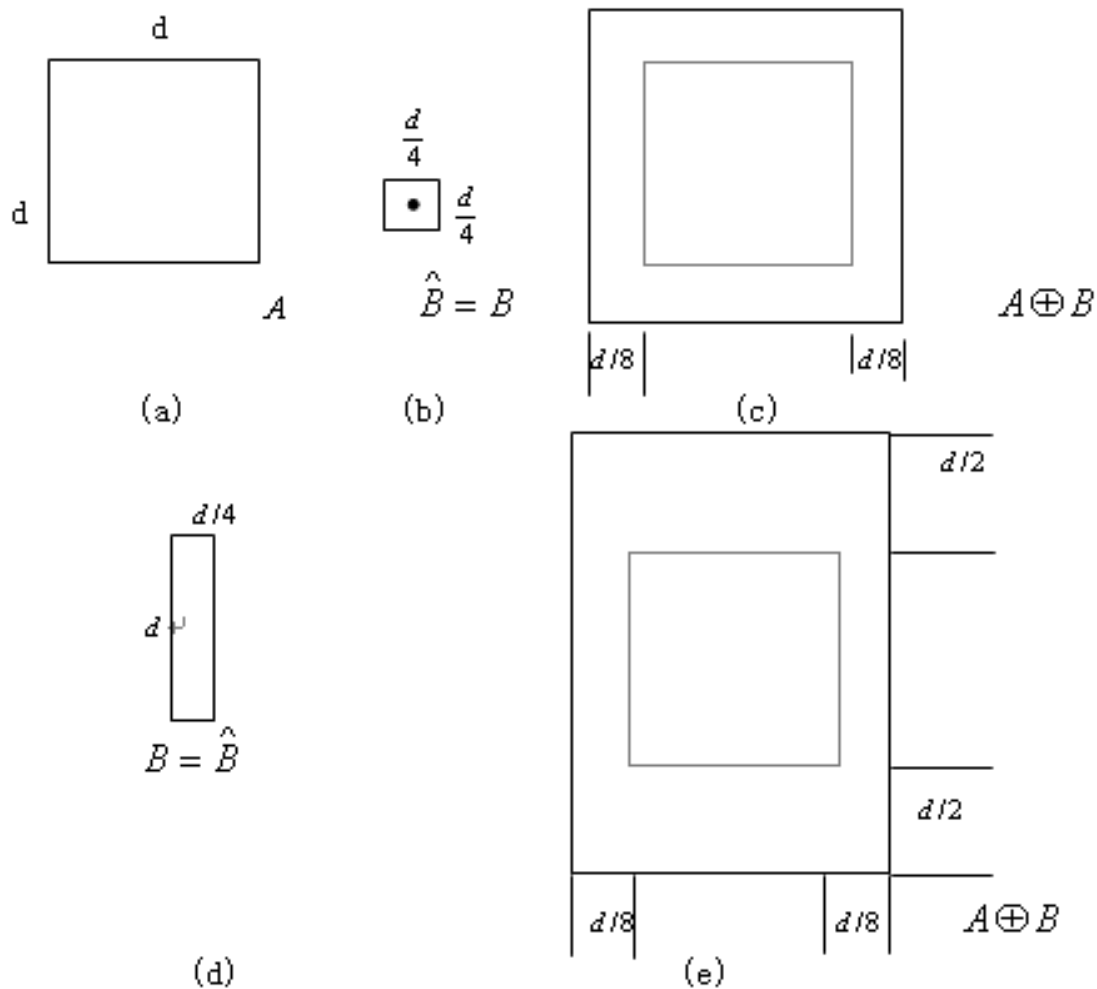
根据这个解释, 公式重写如下:

$$A \oplus B = \{ x \mid [(\hat{B})_x \cap A] \subseteq A \} \quad (9-13)$$

集合 B 在膨胀操作中通常被称为结构元素。

公式(9-12)不是现在形态学文献中膨胀的唯一定义。然而，前面这个定义有一个明显的优势，因为当结构元素 B 被看为卷积模板时有更加直观的概念。尽管膨胀是基于集合的运算，而卷积是基于算术运算，但是 B 关于原点的“映射”及而后连续的平移使它可以滑过集合(图像) A 的基本过程类似于卷积过程。

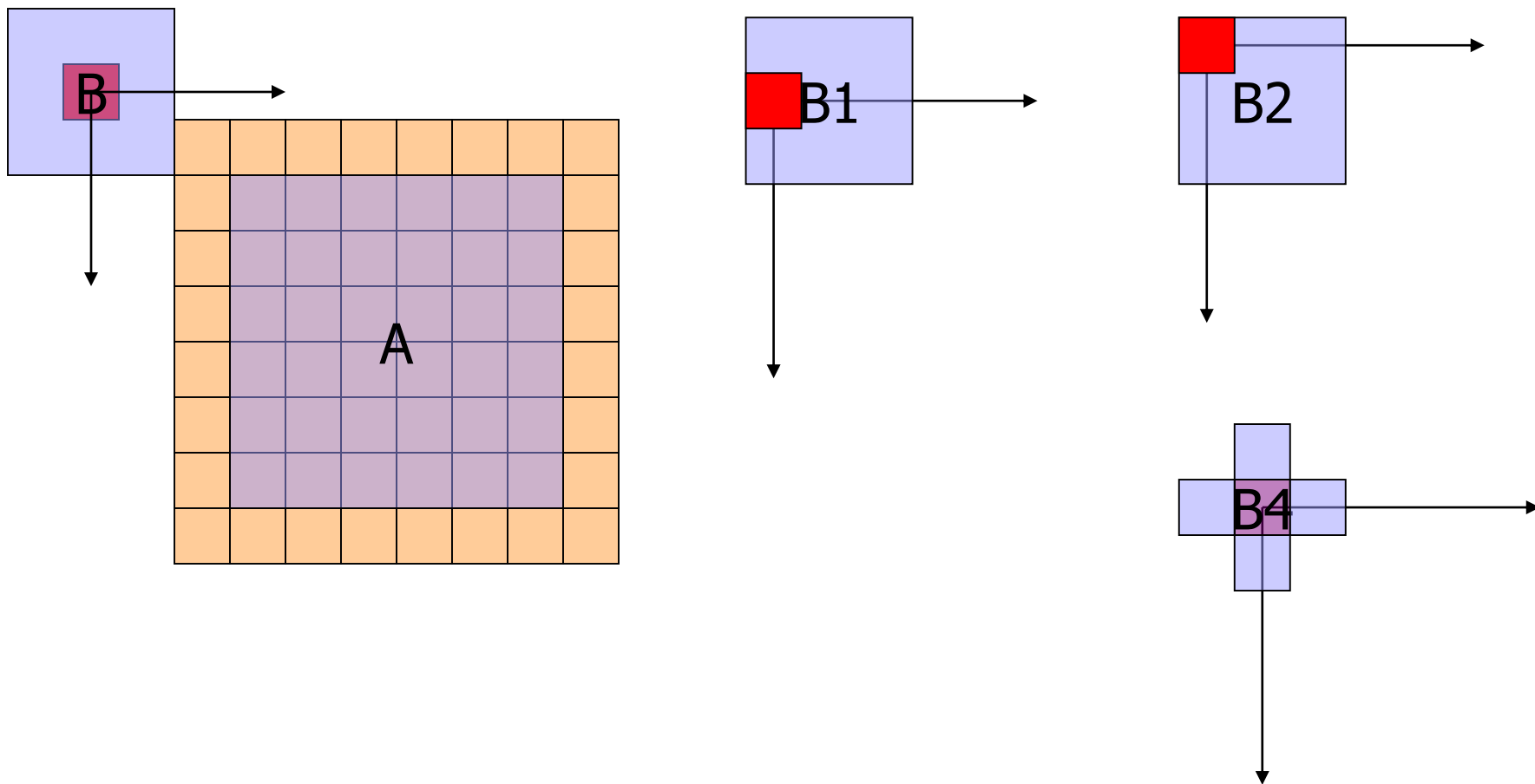
图9-3 (a) 表示一个简单的集合，图9-3 (b) 表示一个结构元素及其“映射”。在此图情况下，因为结构元素 B 关于原点对称，所以，结构元素 B 及其映射 \hat{B} 相同。图9-3 (c) 中的虚线表示作为参考的原始集合，实线示出若 \hat{B} 的原点平移至 x 点超过此界限，则 \hat{B} 与 A 的交集为空。

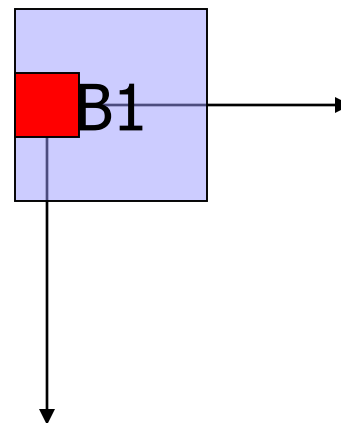
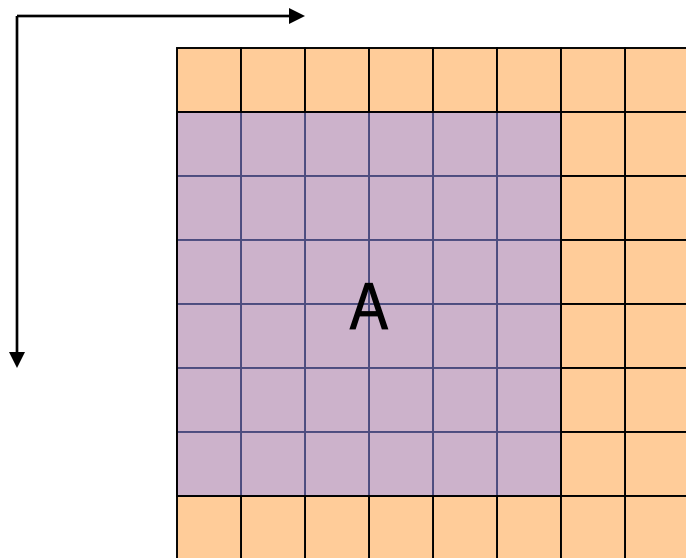


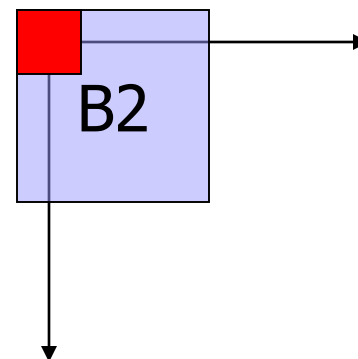
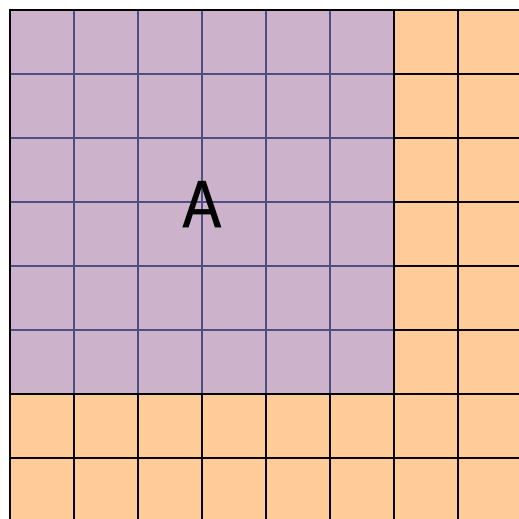
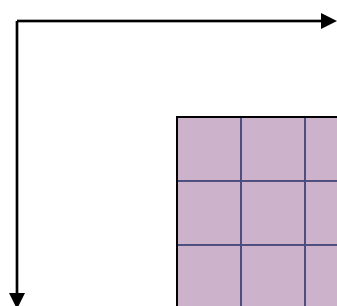
实线内的所有点构成了A被B的膨胀。图(d)表示预先设计的结构元素，其目的是为了得到一个垂直膨胀比水平膨胀大的结果。图(e)为用此构成元素膨胀后的结果。

图 9-3 膨胀操作的例子

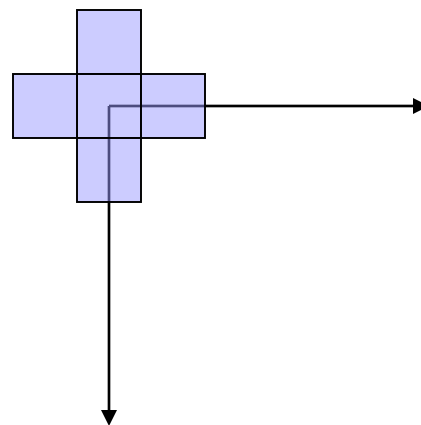
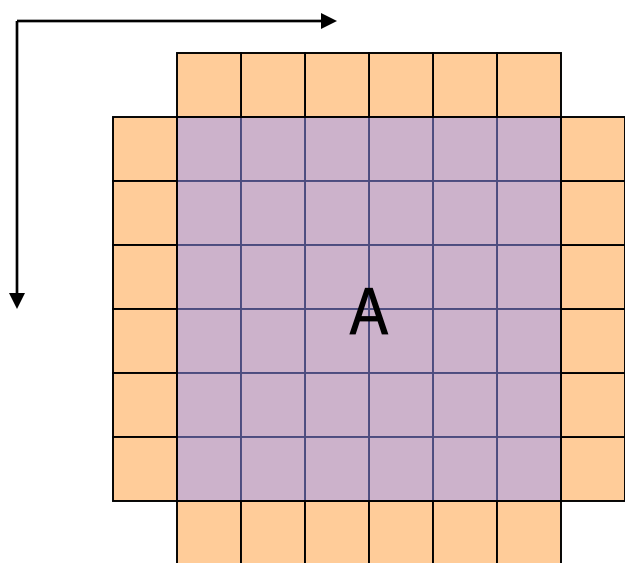
用下面的结构元素进行膨胀，结果是否相同？







相同形状位置不同的结构元素，膨胀后的结果形状相同，但位置不同



不同形状的结构元素，膨胀后的结果形状不同。

例题：将裂缝桥接起来的形态学膨胀的应用

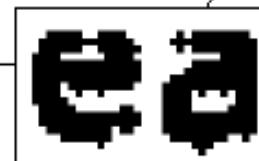
膨胀最简单的应用之一是将裂缝桥接起来。图中显示了带有间断的图像，这与低通滤波器的使用有关。已知间断的最大长度为两个像素。图(b)显示了能够修复这些间断的简单结构元素。图(c)显示了使用这个结构元素对原图进行膨胀后的结果。

形态学方法优于低通滤波方法的一个直接优点是这种方法在一幅二值图像中直接得到结果。

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.



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

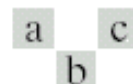


FIGURE 9.5

(a) Sample text of poor resolution with broken characters (magnified view).
 (b) Structuring element.
 (c) Dilation of (a) by (b). Broken segments were joined.

Matlab膨胀运算

`imdilate`功能：对图像实现膨胀操作。

用法：

- `SE= strel('square',6)%` 创建6*6的正方形
- `SE= strel('line',10,45)%` 创建直线长度10，角度45
- `IM2 = imdilate(IM,SE)` 膨胀灰度，二值，压缩二值图像IM，返回IM2。参数SE为由`strel`函数返回的结构元素或者结构元素对象组。

`IM2 = imdilate(IM,NH00D)` 膨胀图像IM，这里NH00D是定义结构元素邻域0和1的矩阵。

`IM2 = imdilate(IM,SE,PACKOPT)` 定义IM是否是一个压缩的二值图像。

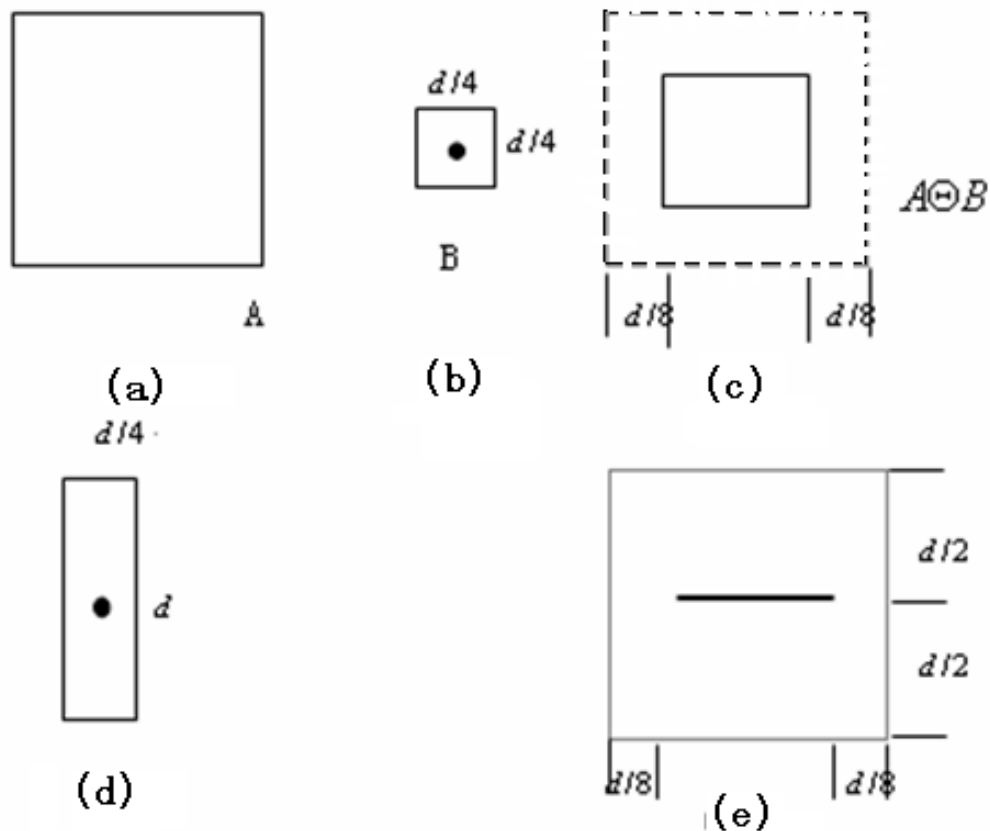
`IM2 = imdilate(...,PADOPT)` 指出输出图像的大小。

➤ 腐蚀

A, B 为 Z^2 中的集合, A 被 B 腐蚀, 记为 $A \ominus B$, 其定义为:

$$A \ominus B = \{x | (B)_x \subseteq A\} \quad (9-14)$$

也就是说 A 被 B 的腐蚀的结果为所有使 B 被 x 平移后包含于 A 的点 x 的集合。与膨胀一样, 公式 (9-14) 也可以用相关的概念加以理解。

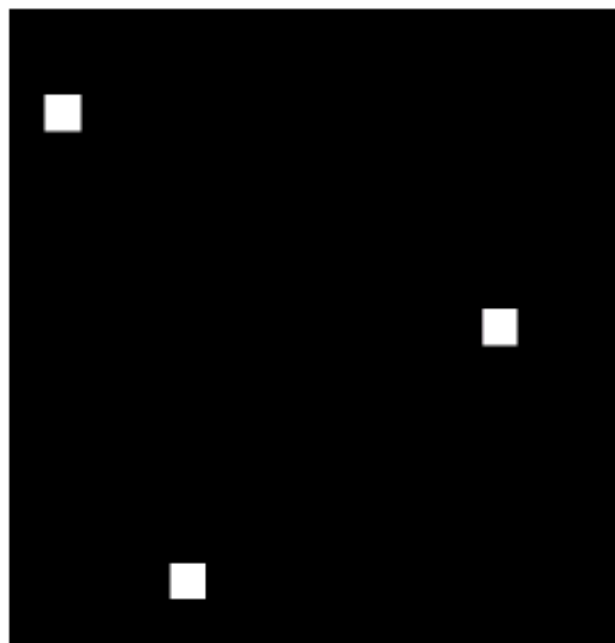
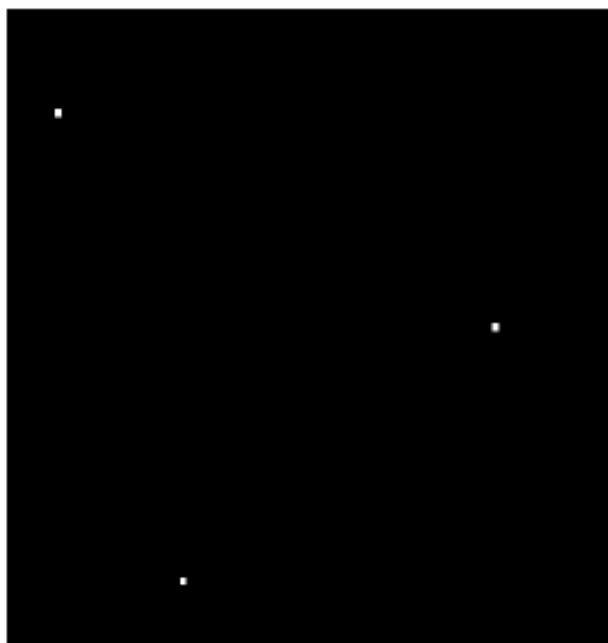
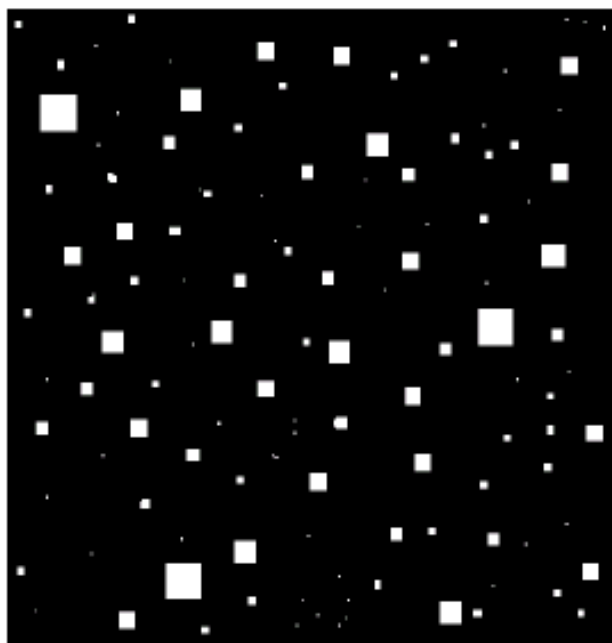


集合 A 在图(c)用虚线表示。实线表示若 B 的原点平移至 x 点超过此界限, 则 A 不能完全包含 B 。在这个实线边界内的点构成了 A 被 B 的腐蚀。图(d)画出了伸长的结构元素, 图(e)显示了 A 被此元素腐蚀的结果。原来的集合被腐蚀成一条线了。

图 9-4 腐蚀操作的例子

例题：使用形态学腐蚀除去图像的某些部分

腐蚀的一种简单用途是从二值图像中消除不相关的细节
图(a)显示的二值图像包含边长为1, 3, 5, 7, 9和15个像素的正方形。假设这里只留下最大的正方形而除去其他的正方形, 通过用比我们要保留的对象稍小的结构元素对图像进行腐蚀。我们选择 13×13 像素大小的结构元素。图(b)显示了腐蚀后的结果。图(c), 我们通过使用用来腐蚀的结构元素对这3个正方形进行膨胀恢复它们原来 15×15 像素的尺寸。



a b c

FIGURE 9.7 (a) Image of squares of size 1, 3, 5, 7, 9, and 15 pixels on the side. (b) Erosion of (a) with a square structuring element of 1's, 13 pixels on the side. (c) Dilation of (b) with the same structuring element.

Matlab腐蚀运算

imerode功能：对图像实现膨胀操作。

用法：

```
IM2 = imerode(IM, SE)
IM2 = imerode(IM, NHOOD)
IM2 = imerode(..., PACKOPT, M)
IM2 = imerode(..., SHAPE)
```

膨胀和腐蚀是关于集合补和反转的对偶。也就是，

$$(A \ominus B)^c = A^c \oplus \hat{B} \quad (9-15)$$

关于上式的正确性可证明于下：

从腐蚀的定义可知：

$$(A \ominus B)^c = \{x \mid (B)_x \subseteq A\}^c$$

如果集合 $(B)_x$ 包含于集合 A ，那莫 $(B)_x \cap A^c = \emptyset$ ，

在这种情况下，上式变为

$$(A \ominus B)^c = \{x \mid (B)_x \cap A^c = \phi\}^c$$


$$(A \ominus B)^c = \{x \mid (B)_x \cap A^c = \phi\}^c$$

但是满足 $(B)_x \cap A^c = \phi$ 的集合 x 的补集是使 $(B)_x \cap A^c \neq \phi$ 的 x 集合。这样

$$(A \ominus B)^c = \{x \mid (B)_x \cap A^c \neq \phi\} = A^c \oplus B$$

命题得证。

膨胀和腐蚀运算的一些性质对设计形态学算法进行图像处理和分析是非常有用的，下面列出几个较重要的性质：

①、交换性： $A \oplus B = B \oplus A$ (9-16)

②、结合性： $A \oplus (B \oplus C) = (A \oplus B) \oplus C$ (9-17)

③、递增性： $A \subseteq B \Rightarrow A \oplus C \subseteq B \oplus C$ (9-18)

$$A \subseteq B \Rightarrow A \ominus C \subseteq B \ominus C$$


④、分配性:

$$(A \cup B) \oplus C = (A \oplus C) \cup (B \oplus C) \quad (9-19)$$

$$A \oplus (B \cup C) = (A \oplus B) \cup (A \oplus C) \quad (9-20)$$

$$A \ominus (B \cup C) = (A \ominus B) \cap (A \ominus C) \quad (9-21)$$

$$(B \cap C) \ominus A = (B \ominus A) \cap (C \ominus A) \quad (9-22)$$



这些性质的重要性是显而易见的。如分配性，
如果用一个复杂的结构元素对图像作膨胀运算，则
可以把这个复杂结构元分解为几个简单的结构元素
的并集，然后用几个简单的结构元素对图像分别进
行膨胀运算，最后将结果再作并集运算，这样一来
就可以大大简化运算的复杂性。

➤ 开运算 (Opening) 和闭运算 (Closing)

膨胀扩大图像，腐蚀收缩图像。另外两个重要的形态运算是开运算和闭运算。

开运算一般能平滑图像的轮廓，削弱狭窄的部分，去掉细的突出。

闭运算也是平滑图像的轮廓，与开运算相反，它一般融合窄的缺口和细长的弯口，去掉小洞，填补轮廓上的缝隙。

开运算： matlab指令-imopen;

设 A 是原始图像， B 是结构元素图像，则集合 A 被结构元素 B 作开运算，记为 $A \circ B$ ，其定义为：

$$A \circ B = (A \ominus B) \oplus B \quad (9-23)$$

A 被 B 开运算就是 A 被 B 腐蚀后的结果再被 B 膨胀。

闭运算： matlab指令-imclose;

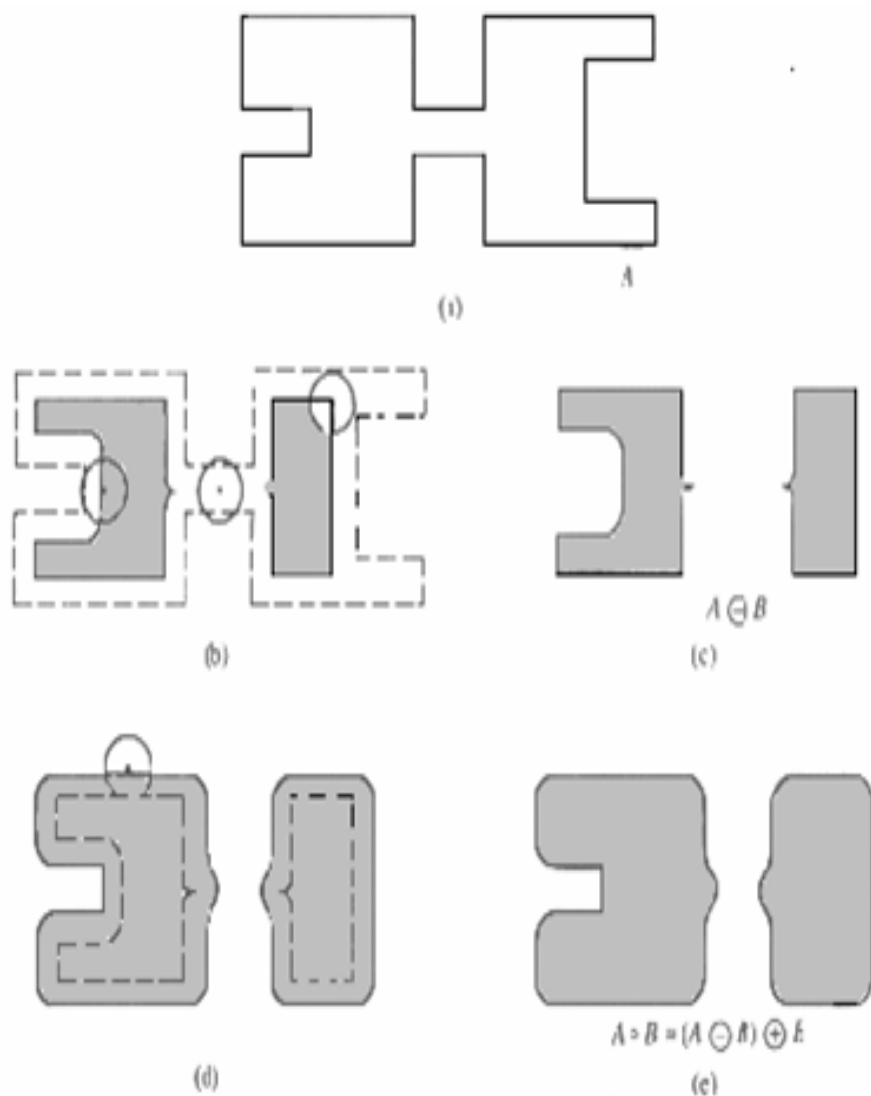
设 A 是原始图像， B 是结构元素图像，则集合 A 被结构元素 B 作闭运算，记为 $A \bullet B$ ，其定义为：

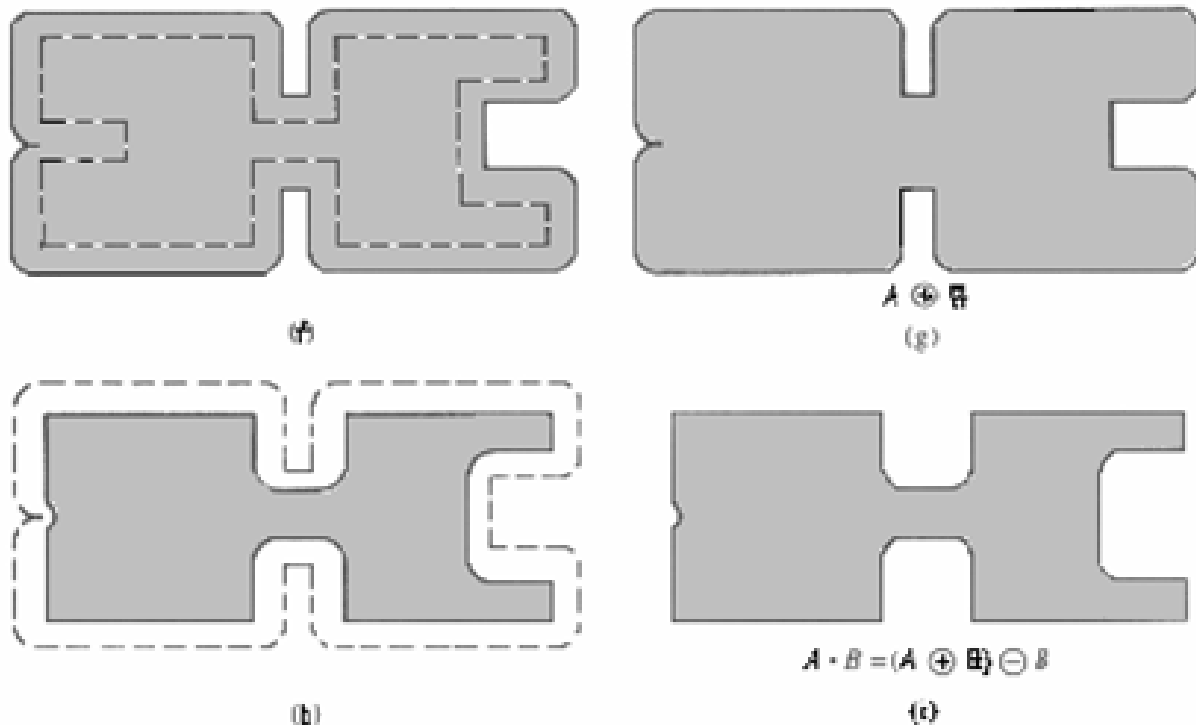
$$A \bullet B = (A \oplus B) \ominus B \quad (9-24)$$

A 被 B 开运算就是 A 被 B 膨胀后的结果再被 B 腐蚀。


图9—5图释了集合 A 被一个圆盘形结构元素作开运算和闭运算的情况。图9-5 (a) 是集合 A ， 9-5 (b) 示出了在腐蚀过程中圆盘结构元素的各个位置，当完成这一过程时，形成分开的两个图形示于图9-5 (c)。

A 的两个主要部分之间的桥梁被去掉了。
 “桥”的宽度小于结构元素的直径；也就是结构元素不能完全包含于集合 A 的这一部分被切除了。
 图9-5(d)画出了对腐蚀的结果进行膨胀的过程，而图9-5(e)示出了开运算的最后结果。





图(f)–(i)示出了用同样的结构元素对 A 作闭运算的结果。结果是去掉了 A 的左边对于 B 来说较小的弯。注意，用一个圆形的结构元素对集合 A 作开运算和闭运算均使 A 的一些部分平滑了。



开运算和闭运算有一个简单的几何解释。假设我们把圆盘形结构元素 B 看作一个（平面的）“滚动球”。 $A \circ B$ 的边界为 B 在 A 内滚动所能达到的最远处的 B 的边界所构成。这个解释能从图9-5(a)得到图9-5(e)。

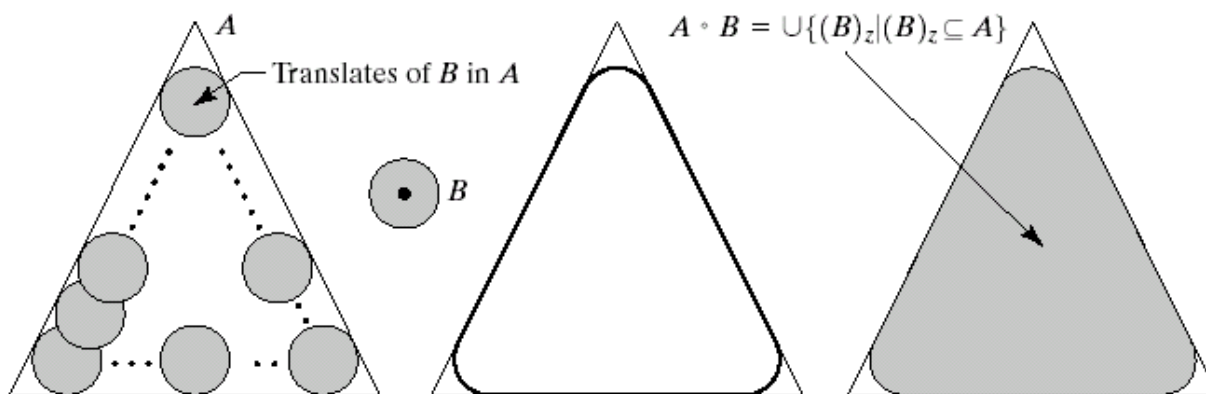
注意所有的朝外的突出角均被圆滑了，而朝内的则没有影响。突出的不能容下这球的部分被去掉。这种开运算的几何拟合性得出了集合论的一个定理：

A 被 B 的开运算就是 B 在 A 内 的 平 移
(保 证 $(B)_x \subseteq A$) 所得到的集合的并集。这样
开运算可以被描述为拟合过程，即：

$$A \circ B = \cup \{ (B)_x \mid (B)_x \subseteq A \} \quad (9-25)$$

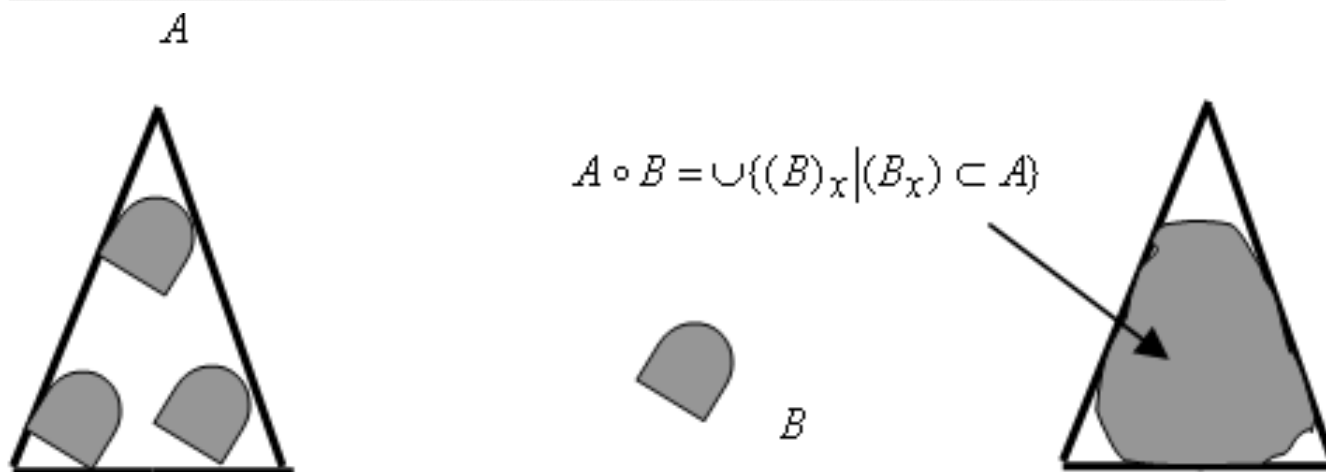
下面用图释这个概念，为了多样性这里我们
用了一个圆形和一个非圆形的结构元素。

■ 开操作的几何解释:



a b c d

FIGURE 9.8 (a) Structuring element B "rolling" along the inner boundary of A (the dot indicates the origin of B). (c) The heavy line is the outer boundary of the opening. (d) Complete opening (shaded).



闭运算也有类似的几何解释。再次用滚动球的例子，只不过我们在边界外边滚动该球（开运算和闭运算是对偶的，所以让小球在外面滚动是合理的）。

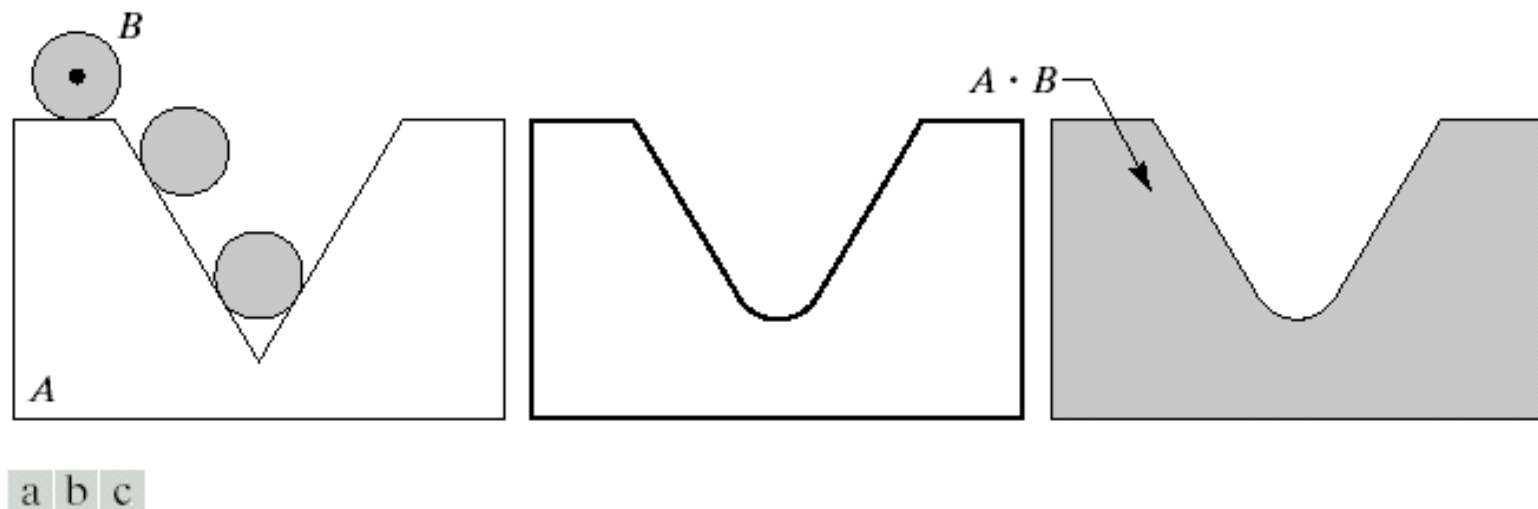


FIGURE 9.9 (a) Structuring element B “rolling” on the outer boundary of set A . (b) Heavy line is the outer boundary of the closing. (c) Complete closing (shaded).

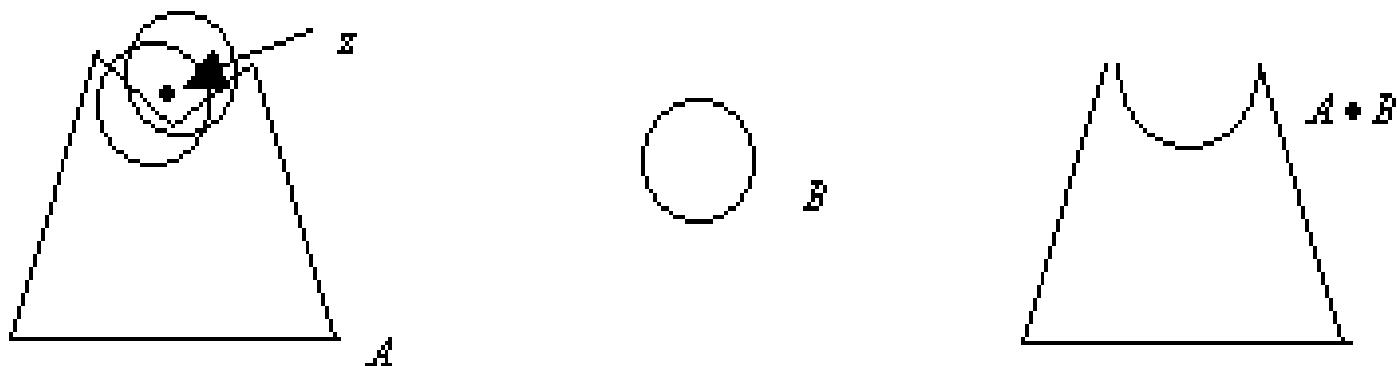


图 9—7 闭运算的几何解释

注意所有的朝内的突出角均被圆滑了，而朝外的则保持不变。集合 A 的最左边的凹入被大幅度减弱了。几何上，点 Z 为 $A \bullet B$ 的一个元素，当且仅当包含的 $(B)_x$ 与 A 的交集非空即， $(B)_x \cap A \neq \Phi$ 。图9-7解释了这一性质。

像膨胀和腐蚀一样，开运算和闭运算是关于集合补和反转的对偶。也就是

$$(A \bullet B)^c = (A^c \circ \hat{B}) \quad (9-26)$$

开运算有下列性质


- ①、 $A \circ B$ 是集合 A 的子集(子图)；
- ②、如果 C 是 D 的子集，则 $C \circ B$ 是 $D \circ B$ 的子集；
- ③、 $(A \circ B) \circ B = A \circ B$

同样，闭运算有下列性质：

①、 A 是集合 $A \bullet B$ 的子集(子图)；


②、如果 C 是 D 的子集，则 $C \bullet B$ 是 $D \bullet B$ 的子集；

③、 $(A \bullet B) \bullet B = A \bullet B$



这些性质有助于对用开运算和闭运算构成的形态滤波器时所得到的结果的理解。例如，用开运算构造一个滤波器。我们参考上面的性质：

(i) 结果是输入的子集；(ii) 单调性会被保持；
(iii) 多次同样的开运算对结果没有影响。最后一条性质有时称为幂等性。同样的解释适合于闭运算。



考虑图9-8 (a) 的简单的二值图像，它包含一个被噪声影响的矩形目标。这里噪声用暗元素(阴影)在亮的背景表示，而光使暗目标为空的。注意集合 A 包含目标和背景噪声，而目标中的噪声构成了背景显示的内部边界。目的是去除噪声及其对目标的影响，并对目标的影响越小越好。

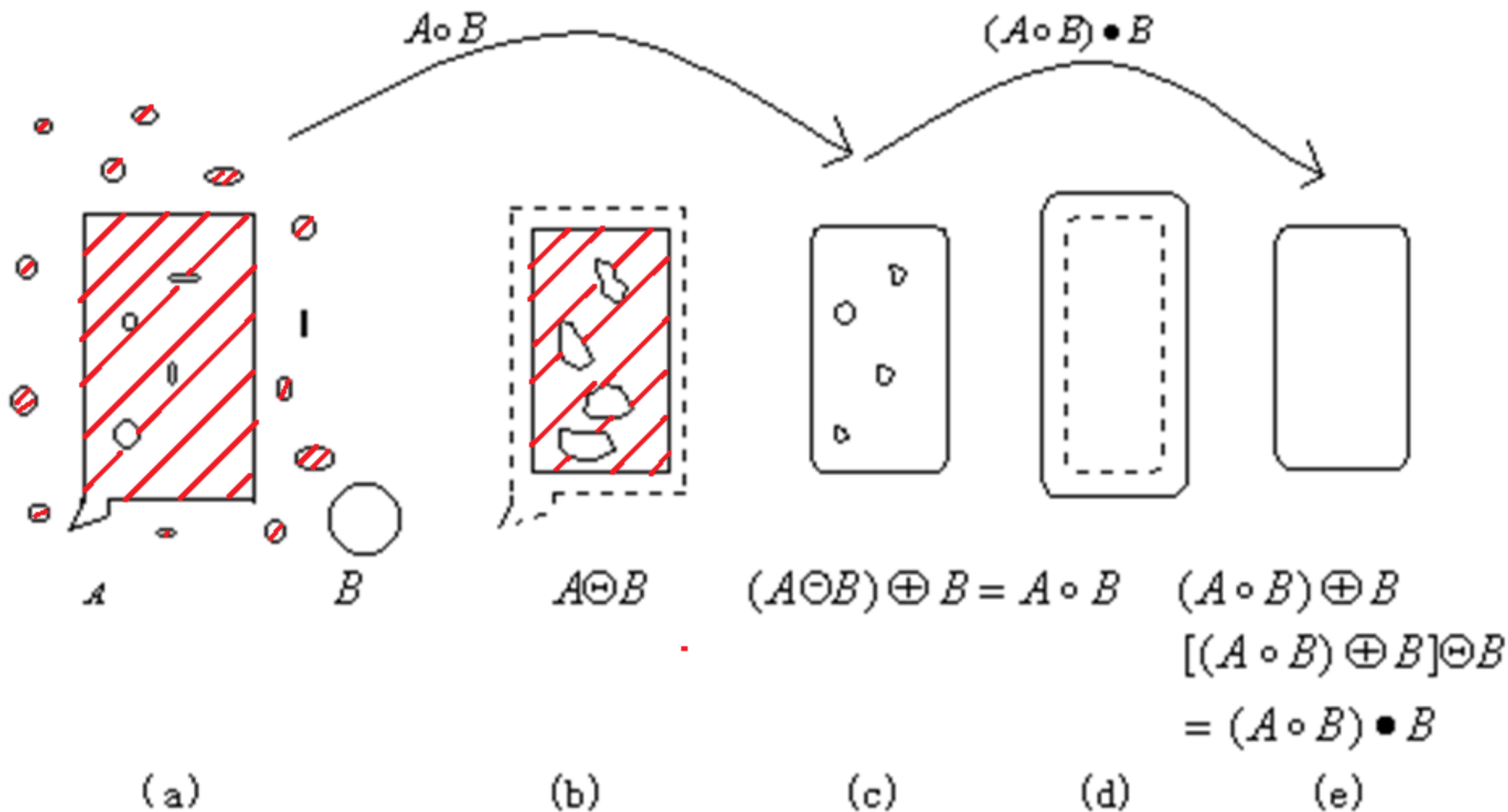
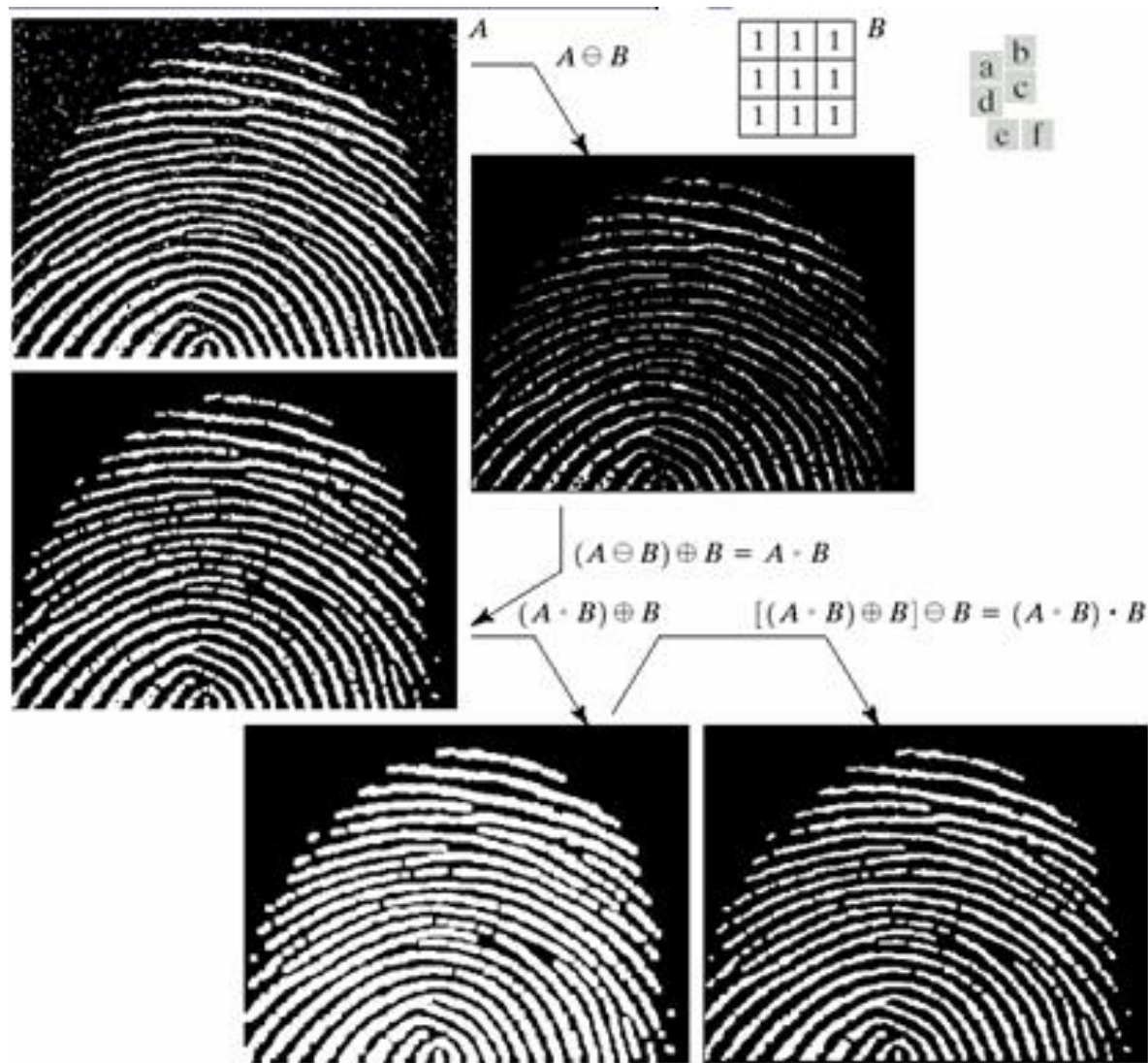


图 9-8 形态学滤波

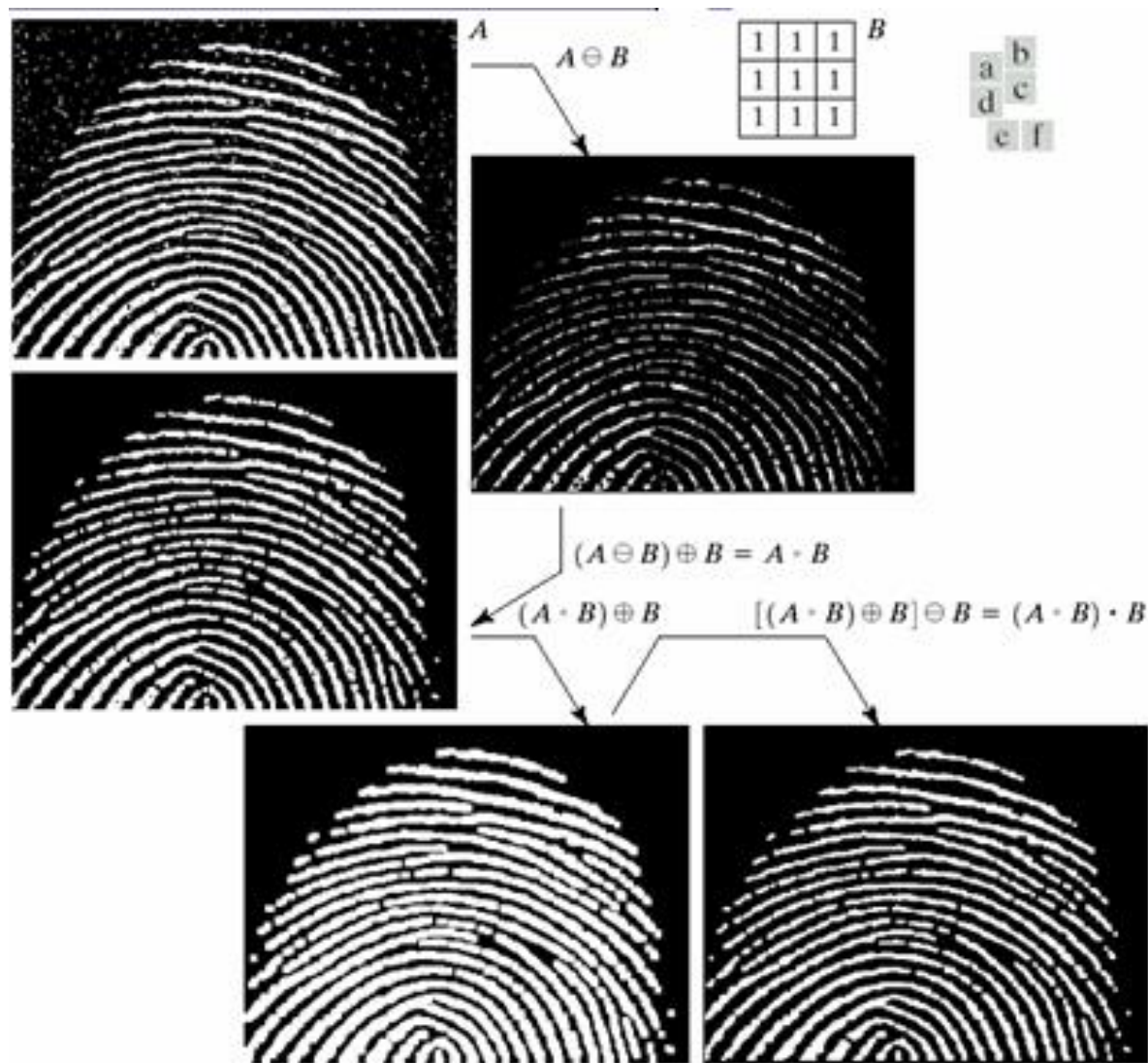
形态“滤波器” $(A \circ B) \bullet B$ 可以用来达到此目的。图9-8(c)显示了用一个比所有噪声成分都大的圆盘形结构元素对 A 进行开放运算的结果。注意这步运算考虑了背景噪声但对内部边界没有影响。

在这个理想的例子中所有背景噪声成分的物理大小均小于结构元素，背景噪声在开运算的腐蚀过程中被消除。而目标内的噪声成分的大小却变大了(图9-8(b))，原因目标中内部边界在腐蚀中会变大。最后内部的边界在闭运算后的膨胀运算中被消除了，如图9-8(d)所示。

实际例子：形态学滤波的开操作和闭操作



图(a)中的二值图像显示了受噪声污染的部分指纹图像。噪声表现为黑色背景上的亮元素和亮指纹部分的暗元素。由闭操作后紧跟着开操作形成的形态学滤波器可以消除噪声。图(b)显示了所使用的结构元素。



图(c)显示了使用结构元素对A腐蚀的结果。背景噪声在开操作的腐蚀过程中消除了。而包含于指纹中的噪声元素的尺寸却增加了。



A

$A \ominus B$

$$B = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$\begin{bmatrix} a & b \\ d & c \\ e & f \end{bmatrix}$



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

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

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



图(d)显示包含于指纹噪声分量的尺寸被减小。然而，指纹纹路间产生了新的间断。

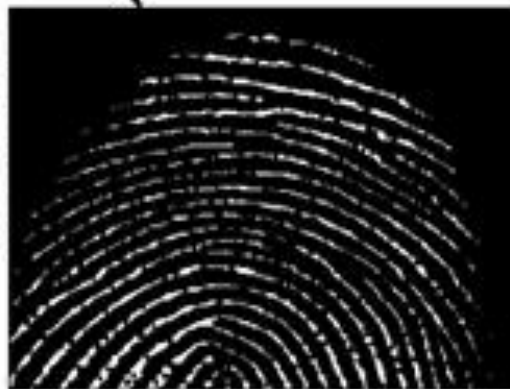


A

$A \ominus B$

$$B = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$\begin{matrix} & b \\ a & c \\ d & e \\ & f \end{matrix}$



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

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

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



我们在开操作的基础上进行膨胀，如图(e)所示。间断被恢复，但纹路变粗了，可以通过腐蚀弥补。



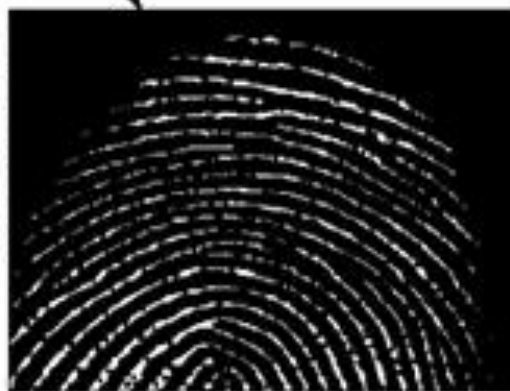
A

$A \ominus B$

1	1	1
1	1	1
1	1	1

B

a	b
d	c
e	f



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

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

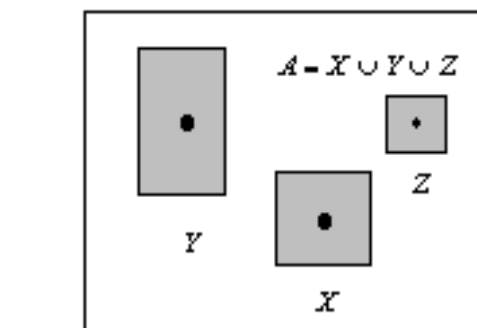
$$[(A \cdot B) \oplus B] \ominus B = (A \cdot B) \cdot B$$



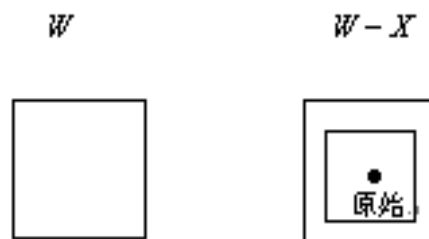
图(f)显示了对图(d)中开操作的闭操作。

➤ 击中 (Hit) 击不中(Miss)变换 (HMT)

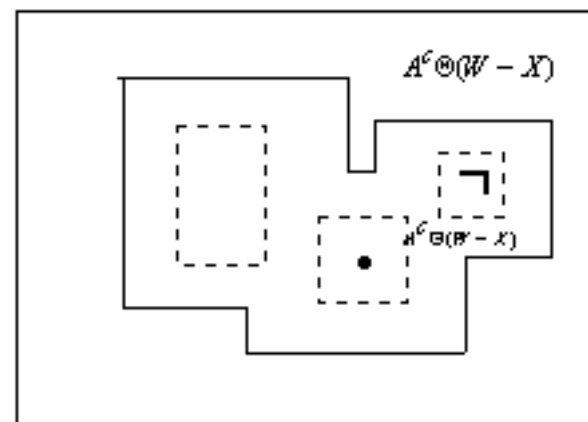
形态学中击中 (Hit) 击不中(Miss)变换是形状检测的基本工具。我们通过图9-9引入这个概念。图中集合 A 包含三个部分 (子集), 记为 X, Y, Z 。图9-9 (a)–(c) 中的图形为原始集合, 而图9-9 (d) 和 (e) 中的阴影为形态运算的结果。目标是找到一个图形 X 的位置。



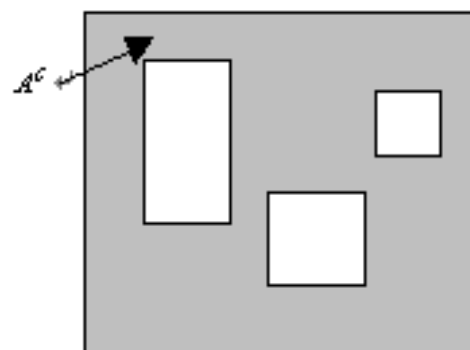
(a)



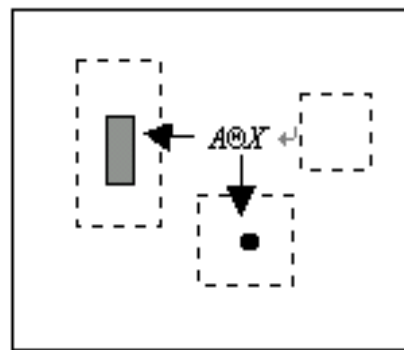
(b)



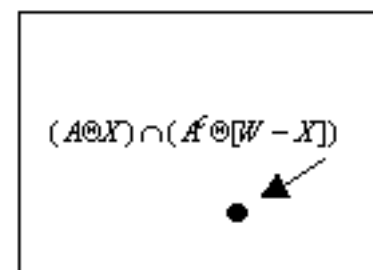
(e)



(c)



(d)



(f)

图 9-9 击中 (Hit) 击不中(Miss)变换图例

让每个图形的原点位于它的重心。如果用一个
小窗口 W 包含 X ， X 关于 W 的本地背景是图9-9(b)中
的集合差 $(W-X)$ 。图9-9(c)为集合 A 的补。图9-9(d)示
出 A 被 X 腐蚀的结果。 A 被 X 的腐蚀在 X 中只有 X 的原
点，这样 X 才能完全包含于 A 。图9-9(e)表示集合 A
的补被本地背景集合 $(W-X)$ 的腐蚀；外围阴影区域也
是腐蚀结果的一部分。

从图9-9 (d) 和 (e), 可以看出集合X在集合A中的位置是A被X的腐蚀和 A^c 被 $(W-X)$ 的腐蚀的交集, 如图9-9 (f) 所示。这个交集正是我们所要找的。换句话说, 如果B记为由X和其背景构成的集合, B在A中的匹配, 记为 $A \otimes B$, 则

$$A \otimes B = (A \ominus X) \cap [A^c \ominus (W - X)] \quad (9-27)$$

可以这样来概括这种表示法, 让 $B = (B_1, B_2)$,
其中 B_1 是由和目标相关的 B 的元素形成的集合,
而 B_2 是由和相应的背景相关的 B 的元素集合。
根据前面的讨论, $B_1 = X, B_2 = (W - X)$ 。用
这种表示法, 公式(9-27)变为


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

(9-28)


用集合差的定义及膨胀和腐蚀的对偶关系，也可以把公式(9-28)写为

$$A \otimes B = (A \ominus B_1) - (A \oplus \hat{B}_2) \quad (9-29)$$

这样集合 $A \otimes B$ 包括所有的点，同时， B_1 在 A 中找到了一个匹配“击中”， B_2 在 A^c 中找到了匹配“击中”。




数学形态学方法比其他空域或频域图像处理和
分析方法具有一些明显的优势。如：在图像
恢复处理中，基于数学形态学的形态滤波器可
借助于先验的几何特征信息利用形态学算子有
效地滤除噪声，又可以保留图像中的原有信息；



另外，数学形态学算法易于用并行处理方法有效的实现，而且硬件实现容易；基于数学形态学的边缘信息提取处理优于基于微分运算的边缘提取算法，它不象微分算法对噪声那样敏感，同时，提取的边缘也比较光滑；利用数学形态学方法提取的图像骨架也比较连续，断点少。

9.3 一些基本形态学算法

在前面讨论的背景知识基础之上，我们可以探讨形态学的一些实际应用。当处理二值图像时，形态学的主要应用是提取表示和描述图像形状的有用成分。特别是用形态学方法提取某一区域的边界线、连接成分、骨骼、凸壳的算法是十分有效的。

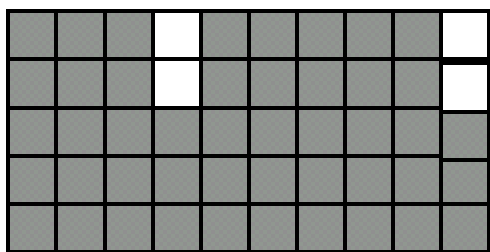


此外，区域填充、细化、加粗、裁剪等处理方法也经常与上述算法相结合在预处理和后处理中使用。这些算法的讨论大部分采用的是二值的图像，即只有黑和白两级灰度，1表示黑，0表示白。

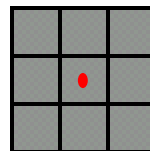
9.3.1 边缘提取算法

集合A的边界记为 $\beta(A)$ ，可以通过下述算法提取边缘：设B是一个合适的结构元素，首先令A被B腐蚀，然后求集合A和它的腐蚀的差。如下式所示：

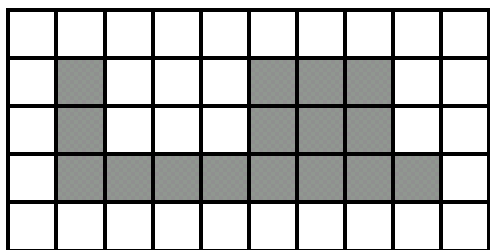
$$\beta(A) = A - (A \ominus B) \quad (9-30)$$



(a)

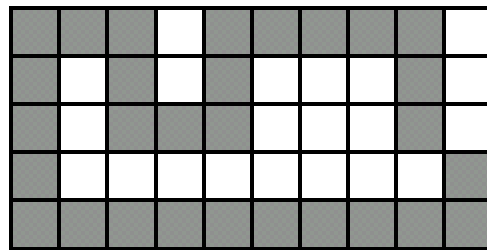


(b)



(c)

$A \oplus B$



(d)

$\beta(A)$

图9-10 边缘提取算法示意图

例题：使用形态学处理提取边界

图9.14 (a) 为一幅简单的二值图像，(b) 为使用图9.13 (b) 中的结构元素进行处理的结果。



a b

FIGURE 9.14

(a) A simple binary image, with 1's represented in white. (b) Result of using Eq. (9.5-1) with the structuring element in Fig. 9.13(b).

9.3.2 区域填充算法

下面讨论的是一种基于集合膨胀，取补和取交的区域填充的简单的算法。在图9-11中，A表示一个包含一个子集的集合，子集的元素为8字形的连接边界的区域。从边界内的一点P开始，目标是用1去填充整个区域。

假定所有的非边界元素均标为0，我们把一个值1赋给P开始这个过程。下述过程将把这个区域用1来填充：

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

其中， $X_0 = P$ ，B为对称结构元素。当 k 迭代到 $X_k = X_{k-1}$ 时，算法终止。集合 X_k 和 A 的并集包括填充的集合和边界。

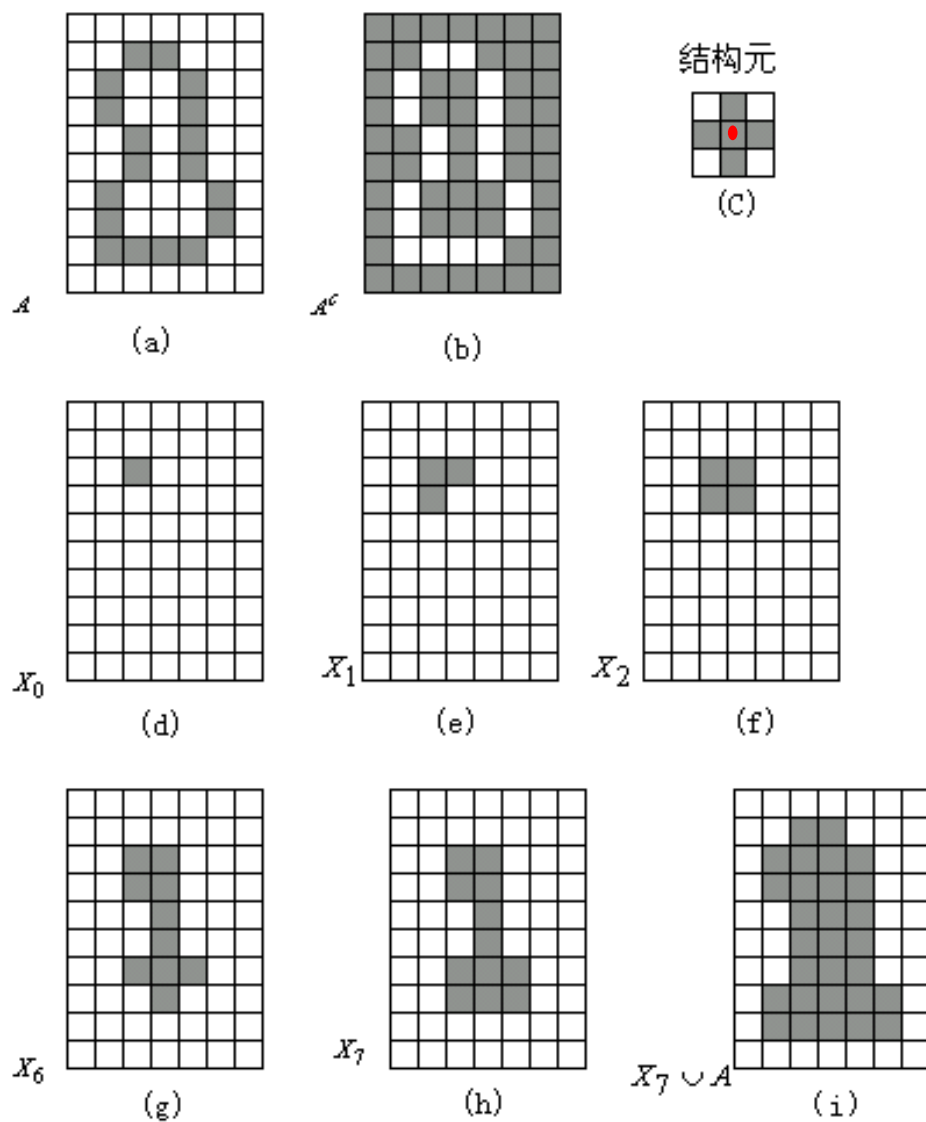


图 9-11 区域填充算法

✓ 通过区域填充消除白色圆圈内的黑点

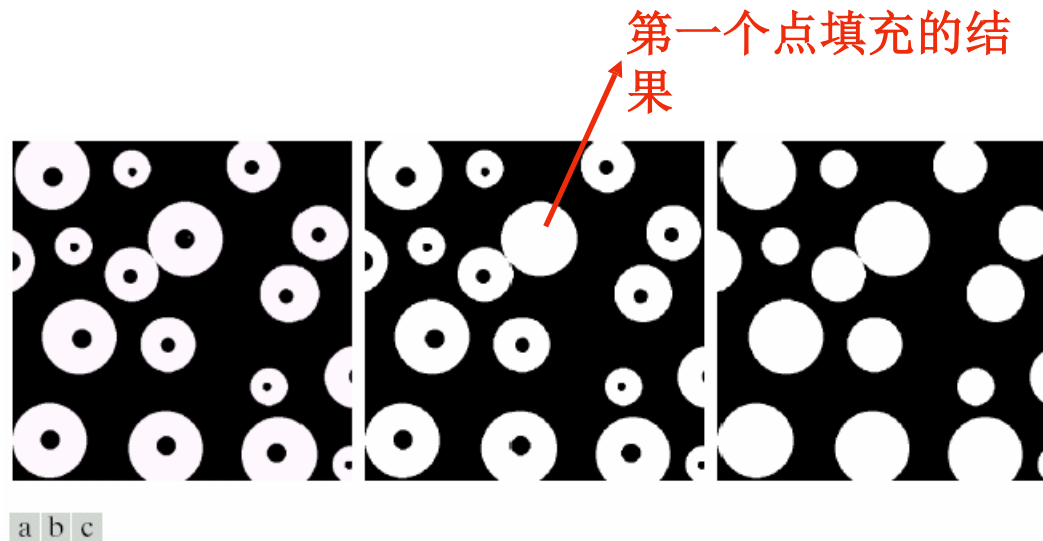


FIGURE 9.16 (a) Binary image (the white dot inside one of the regions is the starting point for the region-filling algorithm). (b) Result of filling that region (c) Result of filling all regions.

● 连通分量的提取

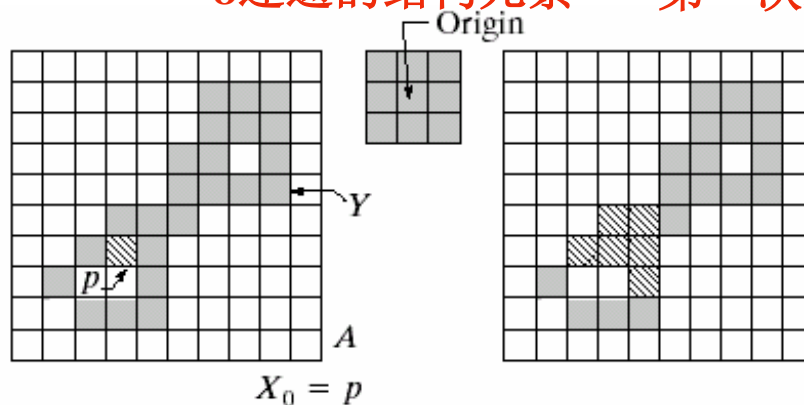
✓ 令Y表示一个包含于集合A中的连通分量，并假设Y 中的一个点p是已知的。用下列迭代式生成Y的所有 元素：

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

$x_0 = p$, 如果 $X_k = X_{k-1}$, 算法收敛, 令
 $Y = X_k$

8连通的结构元素

第一次迭代的结果



第二次迭代的结果

最终结果

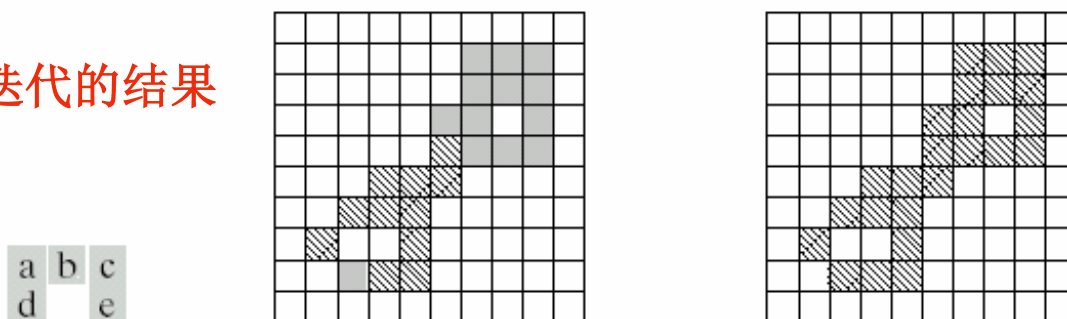
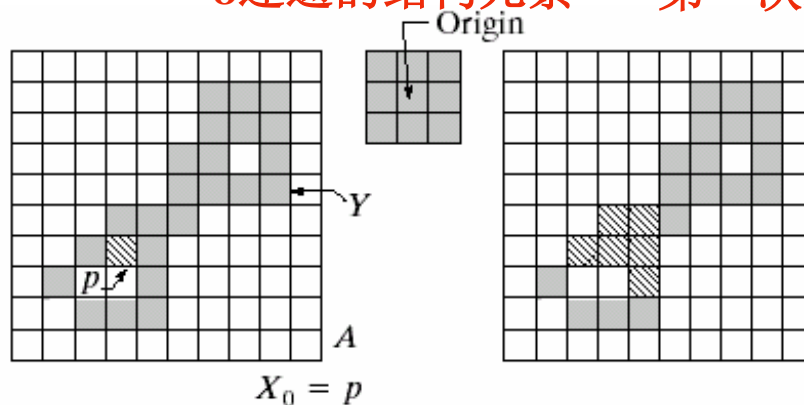


FIGURE 9.17 (a) Set A showing initial point p (all shaded points are valued 1, but are shown different from p to indicate that they have not yet been found by the algorithm). (b) Structuring element. (c) Result of first iterative step. (d) Result of second step. (e) Final result.

8连通的结构元素

第一次迭代的结果



第二次迭代的结果

最终结果

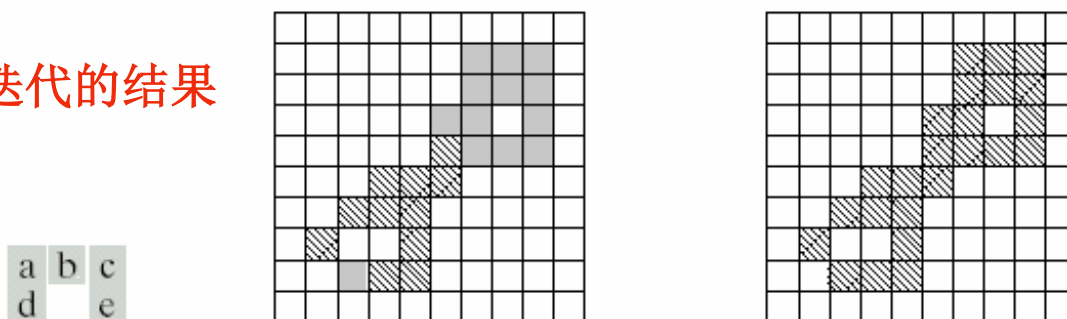
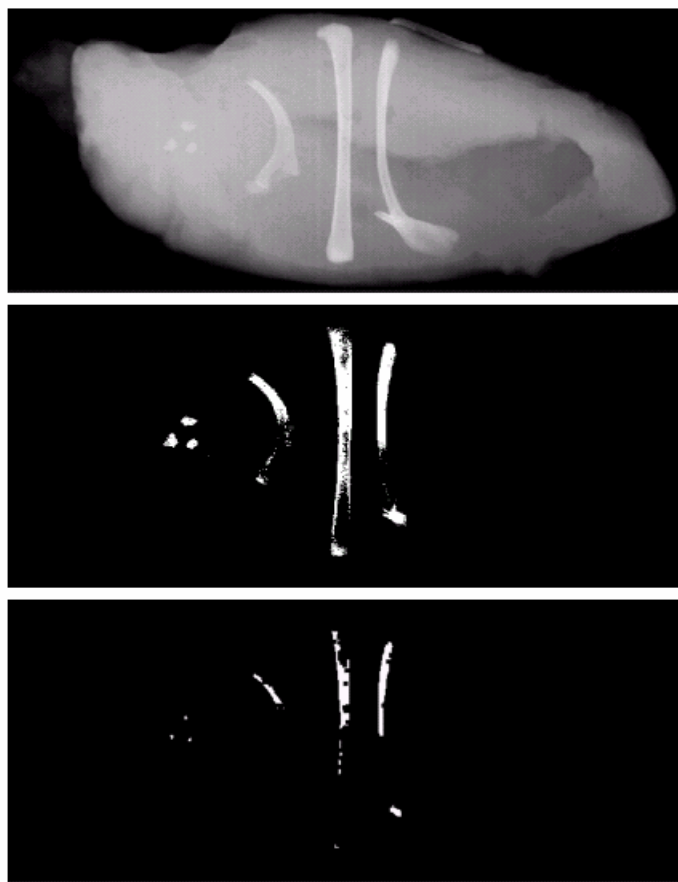


FIGURE 9.17 (a) Set A showing initial point p (all shaded points are valued 1, but are shown different from p to indicate that they have not yet been found by the algorithm). (b) Structuring element. (c) Result of first iterative step. (d) Result of second step. (e) Final result.

a
b
c d

FIGURE 9.18

(a) X-ray image of chicken filet with bone fragments.
(b) Thresholded image.
(c) Image eroded with a 5×5 structuring element of 1's.
(d) Number of pixels in the connected components of (c). (Image courtesy of NTB Elektronische Geraete GmbH, Diepholz, Germany, www.ntbxray.com.)



含有碎骨的鸡胸X光图像

使用阈值将骨头从背景中 提取出来

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

消除细节，对阈值处理后的图像进行腐蚀，保留大尺寸物体

提取连通分量，识别大尺寸对象，其中4个具有最大尺寸

9.3.4 凸壳算法

集合的凸壳是一个有用的图像描述工具。在此，我们提出一种获得集合A凸壳C(A)的简单形态学算法。设 B^i , $i= 1, 2, 3, 4$, 代表四个结构元素。这个处理过程由下述公式实现：

$$X_k^i = (X \otimes B^i) \cup A \quad i = 1, 2, 3, 4 \quad k = 1, 2, 3 \dots$$

其中 $X_0^i = A$ 。现令 $D^i = X_{conv}^i$, 下标
“conv” 表示当时收敛。

那么, A的凸壳为

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

然后令 $X_0^2 = A$ 再次利用公式(9-33)得到的结果示于图9-13(d) (注意只用两步就收敛了)。下两个结果用同样的方法得到。最后，把图9-13(c), (d), (e)和(f)中的集合求并的结果就为所求凸壳。每个结构元素对结果的贡献在图9-13(h)的合成集合中用不同加亮表示。

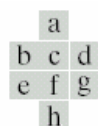
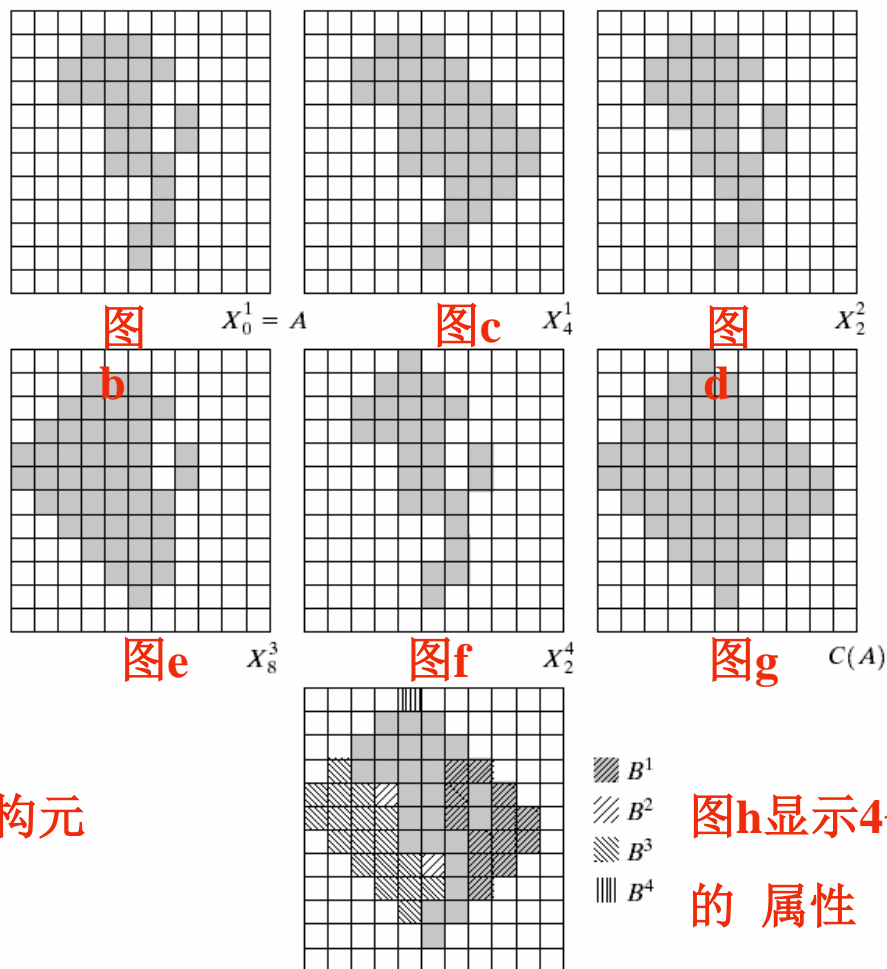
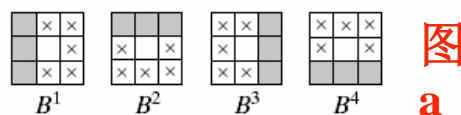


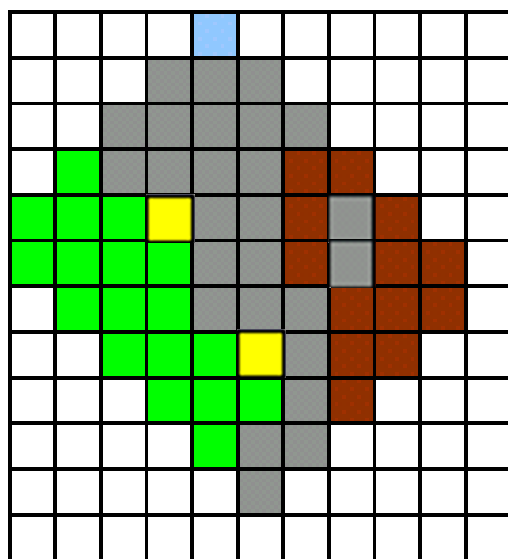
FIGURE 9.19
 (a) Structuring elements. (b) Set A . (c)–(f) Results of convergence with the structuring elements shown in (a). (g) Convex hull. (h) Convex hull showing the contribution of each structuring element.



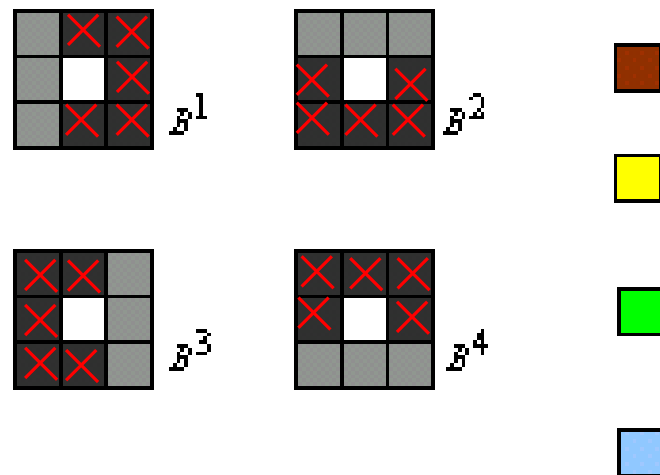
图c-f是用图a中的结构元素 得到的收敛结果

图h显示4个结构元素的 属性

图9-13 凸壳算法示例



(h)



(a)

图9-13 凸壳算法示例

9.3.5 细化

集合A被结构元素的细化用 $A \otimes B$ 表示, 根据击中 (hit) (或击不中miss)变换定义:

$$\begin{aligned} A \otimes B &= A - (A \otimes B) \\ &= A \cap (A \otimes B)^c \end{aligned}$$

对称细化A的一个更有用的表达是基于结构元素序列:

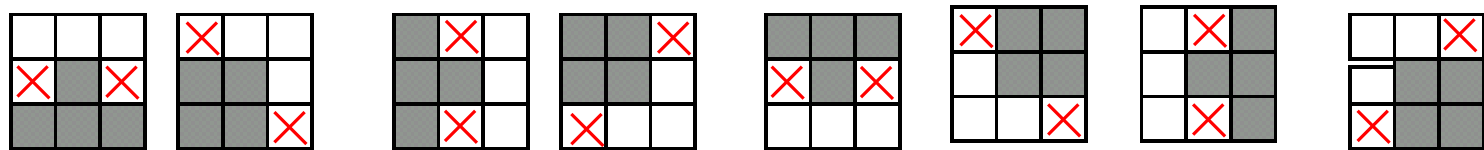
$$\{B\} = \{B^1, B^2, B^3, \dots, B^n\}$$

其中 B^i 是 B^{i-1} 的旋转。

根据这个概念，我们现定义被一个结构元素序列的细化为

$$A \otimes \{B\} = (((\dots((A \otimes B^1) \otimes B^2) \dots) \otimes B^n) \quad (9-37)$$

换句话说，这个过程是用 B^1 细化A，然后用 B^2 细化前一步细化的结果等等，直到A被 B^n 细化。整个过程重复进行到没有进一步的变化发生为止。



B^1

B^2

B^3

B^4

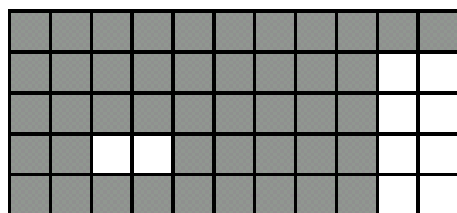
B^5

B^6

B^7

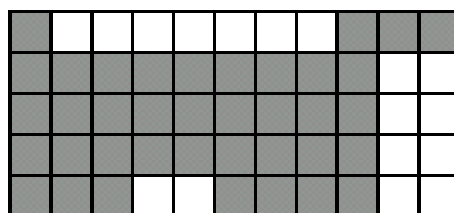
B^8

(a)



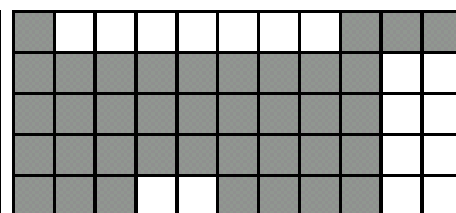
(b)

A



(c)

$A \otimes B^1$



(d)

$A \otimes B^2$

图 9-14 细化处理

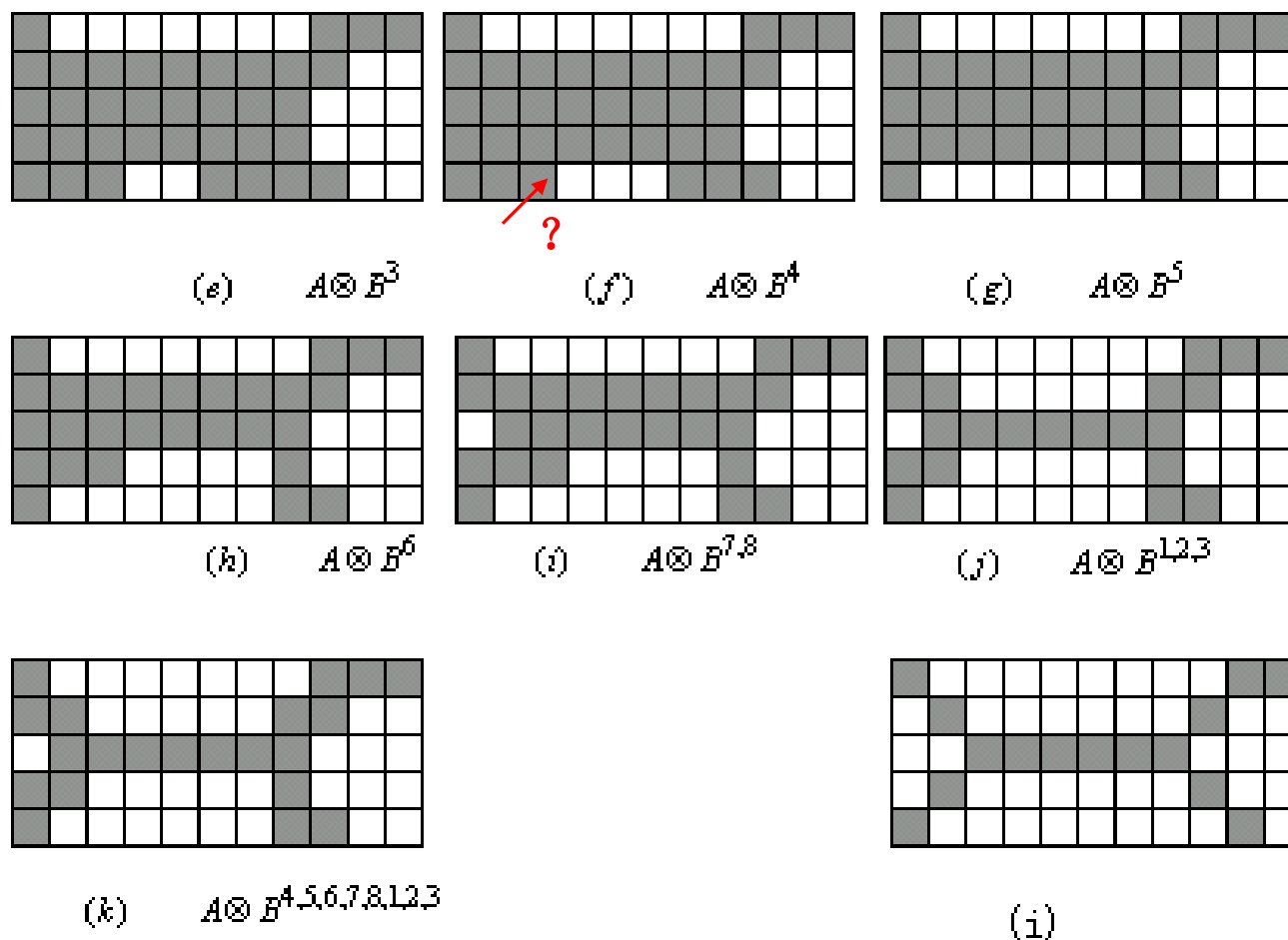


图 9-14 细化处理


9.3.6 粗化运算

粗化是细化的形态学上对偶，记为 $A \odot B$ ，定义为

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

其中 B 是适合粗化的结构元素。象细化一样，粗化可以定义为一个序列运算：

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



用来粗化的结构元素同细化的结构元素具有相同的形式。只是所有的0和1交换位置。然而，在实际中，粗化的算法很少使用。相反的，通常的过程是细化集合的背景，然后求细化结果的补而达到粗化的结果。

C^c

● 粗化

- ✓ 粗化可以通过细化算法求补集实现： 为了对集合A进行粗化，先令 $C=A^c$ ，然后对C进行细化，最后形成 C^c

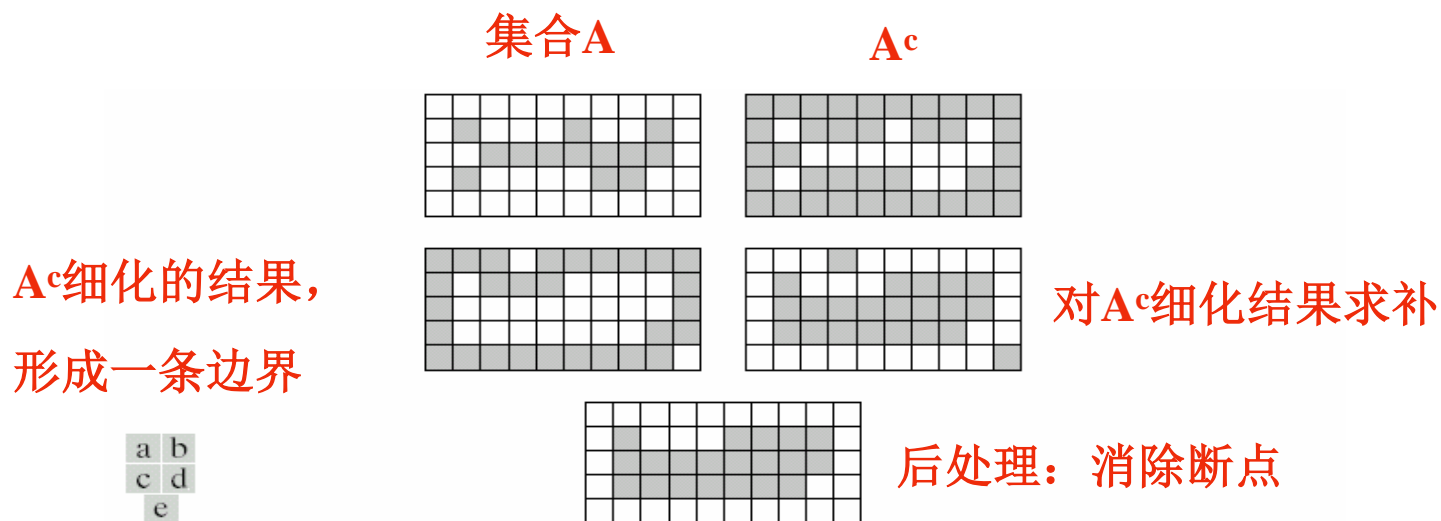


FIGURE 9.22 (a) Set A. (b) Complement of A. (c) Result of thinning the complement of A. (d) Thickened set obtained by complementing (c). (e) Final result, with no disconnected points.

表9—1总结了前边讨论的数学形态学算法及其结果，图9.18示出了所使用的基本结构元素。

操作	等式	评述
平移	$(A)_x = \{c c = a + x, a \in A\}$	把 A 的原点平移到 x 点
映射	$\hat{A} = \{x x = -b, b \in A\}$	相对于原点映射关于集合 A 的所有元素
补集	$A^c = \{x x \notin A\}$	所有不属于 A 的点集
差集	$A - B = \{x x \in A, x \notin B\} = A \cap B$	属于 A 但不属于 B 的点集
膨胀	$A \oplus B = \{x (\hat{B})_x \cap A \neq \emptyset\}$	“扩展” A 的边界。
腐蚀	$A \ominus B = \{x (B)_x \subseteq A\}$	“收缩” A 的边界。
开	$A \circ B = (A \ominus B) \oplus B$	平滑轮廓, 切除狭区, 去除小的孤岛及突刺。
闭	$A \bullet B = (A \oplus B) \ominus B$	平滑轮廓, 连接小狭区, 纵向细化沟渠并且去除小洞。

表9-1 形态学结论和特性的总结

击中 和 击中不中变换	$A \otimes B = (A \ominus B_1) \cap (A^c \ominus B_2)$ $= (A \ominus B_1) - (A \oplus \hat{B}_2)$	在一个点集上, B_1 在 A 中, B_2 在 A^c 中同时找到了匹配点
边缘提取	$\beta(A) = A - (A \ominus B)$	提取集合 A 的边界上的点集。
区域填充	$X_k = (X_{k-1} \oplus B) \cap A^c;$ $X_0 = p \quad k=1, 2, 3, \dots$	给定 A 的一个区域中的一点 p , 填充这个区域。
连接的部分	$X_k = (X_{k-1} \oplus B) \cap A;$ $X_0 = p \quad k=1, 2, 3, \dots$	给定 A 中一个连接的部分 Y 中的一点 p , 提取 Y
凸壳	$X_k^i = (X_{k-1}^i \oplus B^i) \cap A; \quad i=1, 2, 3, 4$ $k=1, 2, 3, \dots \quad X_0^i = A$ $D^i = X_{conv}^i \quad C(A) = \bigcup_{k=1}^4 D^i$	<p>找到集合 A 的凸壳 $C(A)$, *</p> <p>其中“conv”在 $X_k^i = X_{k-1}^i$ 的意义上表示收敛。</p>

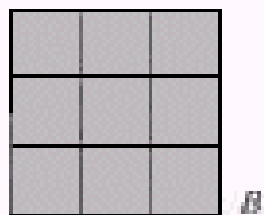
表9-1 形态学结论和特性的总结（续）

细化	$A \otimes B = A - (A \oplus B)^c$ $= A \cap (A \oplus B)^c$ $A \otimes \{B\} = ((\dots((A \otimes B^1) \otimes B^2)\dots) \otimes B^n$ $\{B\} = \{B^1, B^2, B^3, \dots, B^n\}$	<p>细化集合 A。前两个等式给出了细化的基本定义。后两个等式表示用一系列结构元素来进行的细化。</p>
粗化	$A \odot B = A \cup (A \oplus B)^c$ $A \odot \{B\} = ((\dots((A \odot B^1) \odot B^2)\dots) \odot B^n)$	<p>粗化集合 A。(参阅以前关于结构元素序列的部分)。把细化中的 0 和 1 调换来使用即可粗化。</p>

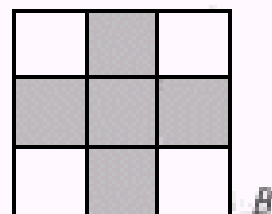
表9-1 形态学结论和特性的总结（续）

骨骼	$S(A) = \bigcup_{k=0}^K S_k(A)$ $S_k(A) = \bigcup_{k=0}^K \{(A \ominus kB) - [(A \ominus kB) \circ B]\}$ $A = \bigcup_{k=0}^K (S_k(A) \oplus kB)$	<p>找到集合 A 的骨骼 $S(A)$。最后一个公式表明 A 可由它的骨骼子集 $S_k(A)$ 重构。在所有等式中, K 是迭代步骤的次数, 超过 K 次集合 A 被腐蚀为空集。符号 $(A \ominus kB)$ 表示连续腐蚀 k 次的迭代。</p>
裁剪	$X_1 = A \otimes \{B\}$ $X_2 = \bigcup_{k=1}^K (X_1 \otimes B^k)$ $X_3 = (X_2 \oplus H) \cap A$ $X_4 = X_1 \cup X_3$	<p>X_4 是裁剪集合 A 后的结果。用于获得 X_1 的第一等式的使用次数必须指定。等式分别使用了前边介绍过的结构元素。</p>

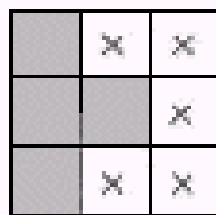
表9-1 形态学结论和特性的总结（续）



I

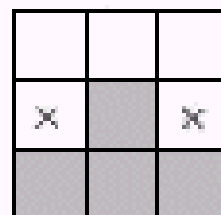


II



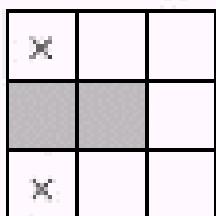
III

$B^i, i = 1, 2, 3, 4$
(rotate 90°)

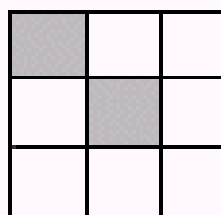


IV

$B^i, i = 1, 2, \dots, 8$
(rotate 45°)



$B^i, i = 1, 2, 3, 4$
(rotate 90°)



$B^i, i = 5, 6, 7, 8$
(rotate 90°)

图9-18 基本形态学结构元素