数字图像处理

形态学及其应用

数学形态学概述

- 数学形态学
  - 数学形态学是法国和德国的科学家在研究岩石结构时建立的一门学科(1664)。
  - 形态学的用途是获取物体拓扑和结构信息，它通过物体和结构元素相互作用的某些运算，得到物体更本质的形态。
- 在图像处理中的应用主要是：
  - 利用形态学的基本运算，对图像进行观察和处理，从而达到改善图像质量的目的；
  - 描述和定义图像的各种几何参数和特征，如面积、周长、连通度、颗粒度、骨架和方向性等。

数学形态学概述

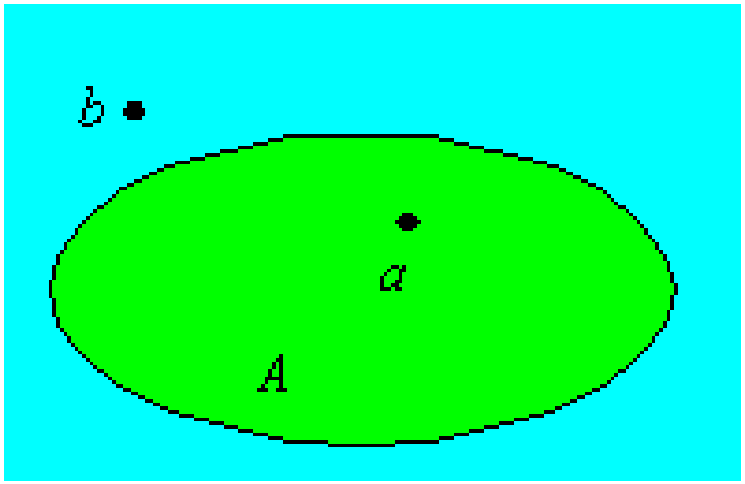
- 数学形态学的数学基础和所用语言是集合论，因此它具有完备的数学基础。这为形态学用于图像分析和处理、形态滤波器的特性分析和系统设计奠定了坚实的基础。
- 数学形态学的应用可以简化图像数据，保持它们基本的形状特性，并除去不相干的结构。
- 数学形态学方法利用一个称作结构元素的“探针”收集图像的信息。当探针在图像中不断移动时，便可考察图像各个部分之间的相互关系，从而了解图像的结构特征。

数学形态学概述

- 迄今为止， 还没有一种方法能像数学形态学那样既有坚实的理论基础，简洁、 朴素、 统一的基本思想，又有如此广泛的实用价值。有人称数学形态学在理论上是严谨的，在基本观念上却是简单和优美的。
- 数学形态学是一门建立在严格数学理论基础上的学科，其基本思想和方法对图像处理的理论和技术产生了重大影响。
- 数学形态学已经构成一种新的图像处理方法和理论，成为计算机数字图像处理的重要研究领域。

基本符号和定义

- 集合论概念
  - 在数字图像处理的数学形态学运算中，把一幅图像称为一个集合。
  - 对于一幅图像A，如果点a在A的区域以内， 那么就说a是A的元素，记为$a \in A$；否则，记作$a \notin A$.



基本符号和定义

- B包含于A
  - 设有两幅图像A和B。如果对于B中所有的元素a_i，都有$a_i \in A$，则称B包含于A，记作$B \subset A$.

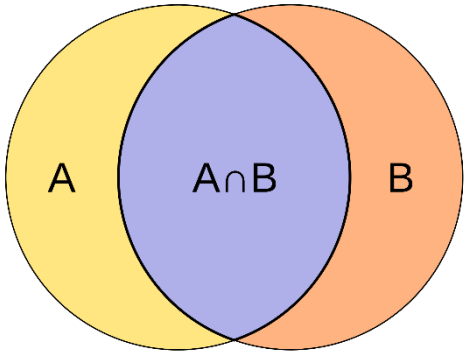
基本符号和定义

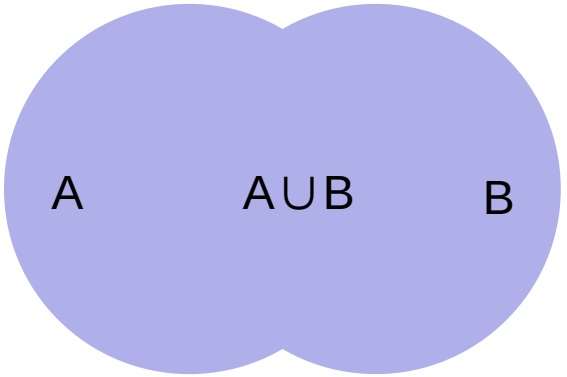
- 交集和并集
  - 两个图像集合A和B的公共点组成的集合称为两个集合的交集，记为$A \cap B$。即，

$$A \cap B = \{a | a \in A且a \in B\} 。$$

  - 两个图像集合A和B的所有元素组成的集合称为两个集合的并集，记为$A \cup B$。即，

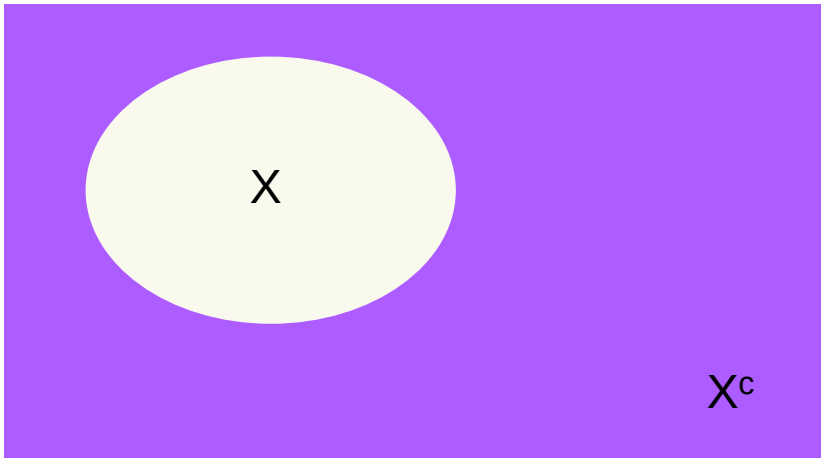
$$A \cup B = \{a | a \in A或a \in B\} 。$$





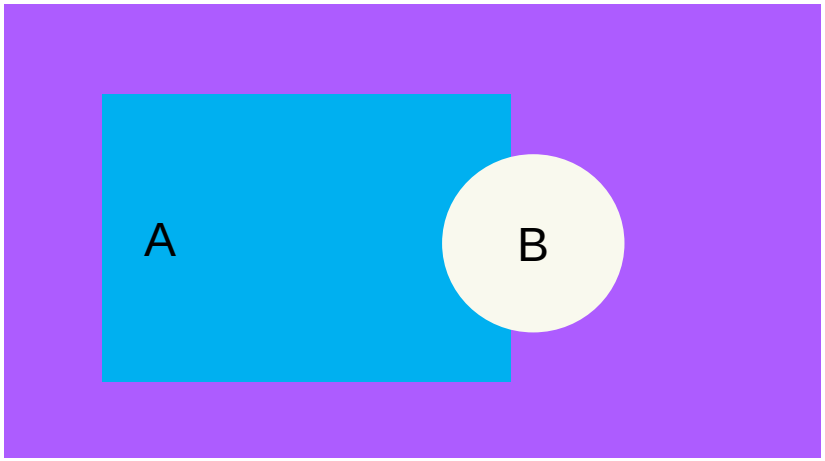
基本符号和定义

- 补集
  - 设有一幅图像X，所有X区域以外的点构成的集合称为X的补集，记作X^c。
  - 显然，如果$B \cap X = \emptyset$，则B在X的补集内。



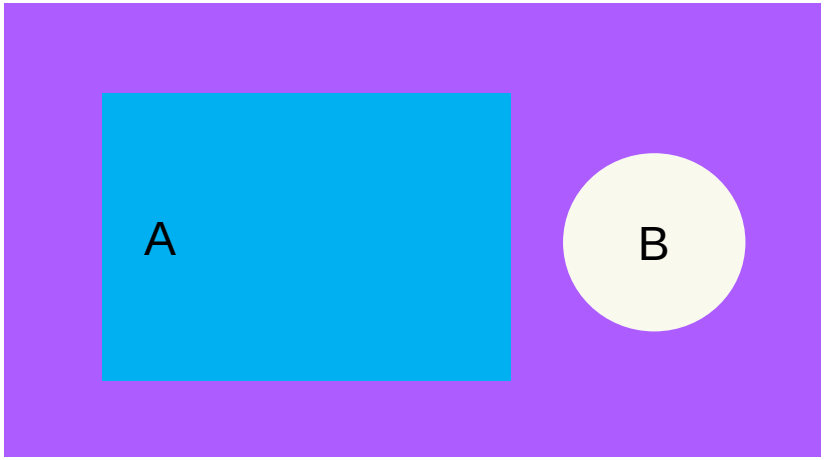
基本符号和定义

- 击中
  - 设有两幅图像B和A。若存在这样一个点，它既是B的元素，又是A的元素，$A \cap B \neq \emptyset$ 则称B击中A，记作$B \uparrow A$。



基本符号和定义

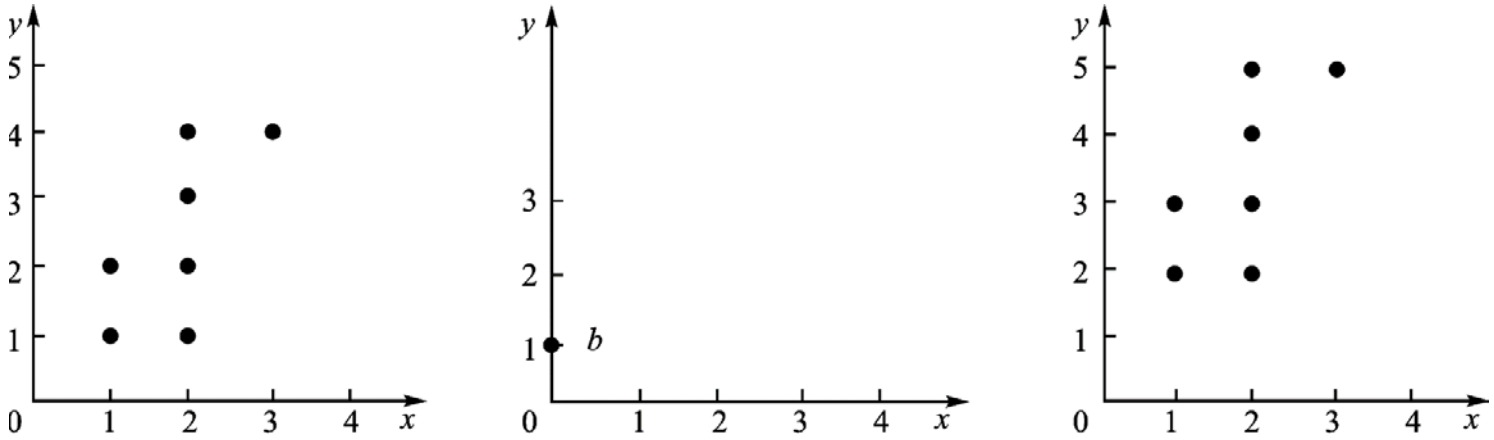
- 击不中
  - 设有两幅图像B和A。若不存在任何一个点，它既是B的元素，又是A的元素，即B和A的交集是空，则称B不击中A，记作$A \cap B \neq \emptyset$.



平移和对称集

- 平移

  设A是一幅数字图像，b是一个点，那么定义A被b平移后的结果为$A+b=\{a+b|a \in A\}$，即取出A中的每个点a的坐标值，将其与点b的坐标值相加，得到一个新的点的坐标值$a+b$，所有这些新点所构成的图像就是A被b平移的结果，记为$A+b$.

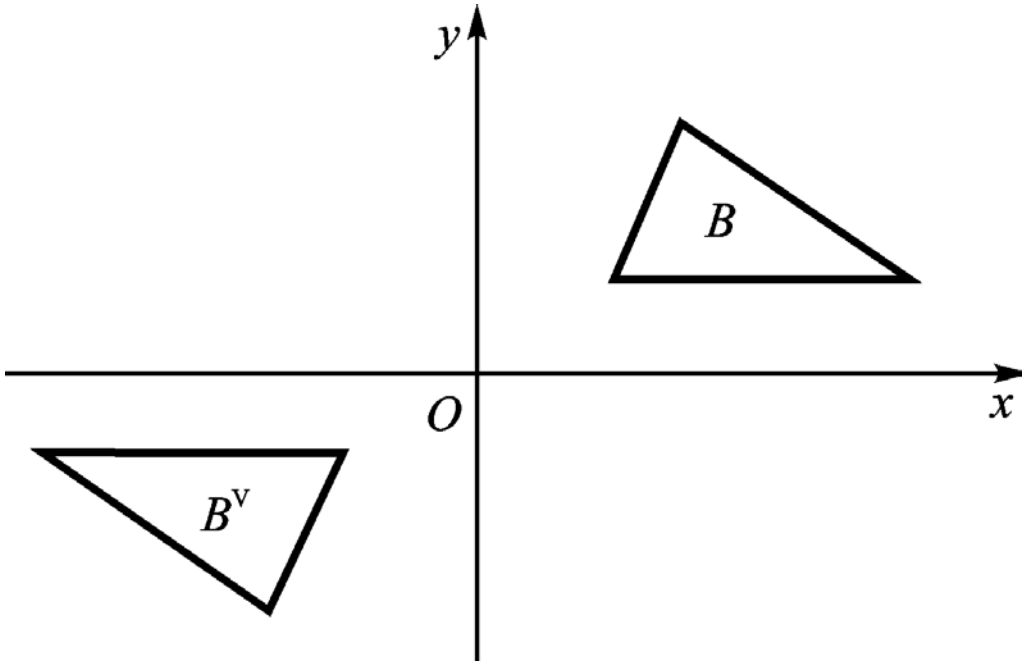


(a) 数字图像A　　(b) 结构元素b　　(c) A被b平移的结果

平移和对称集

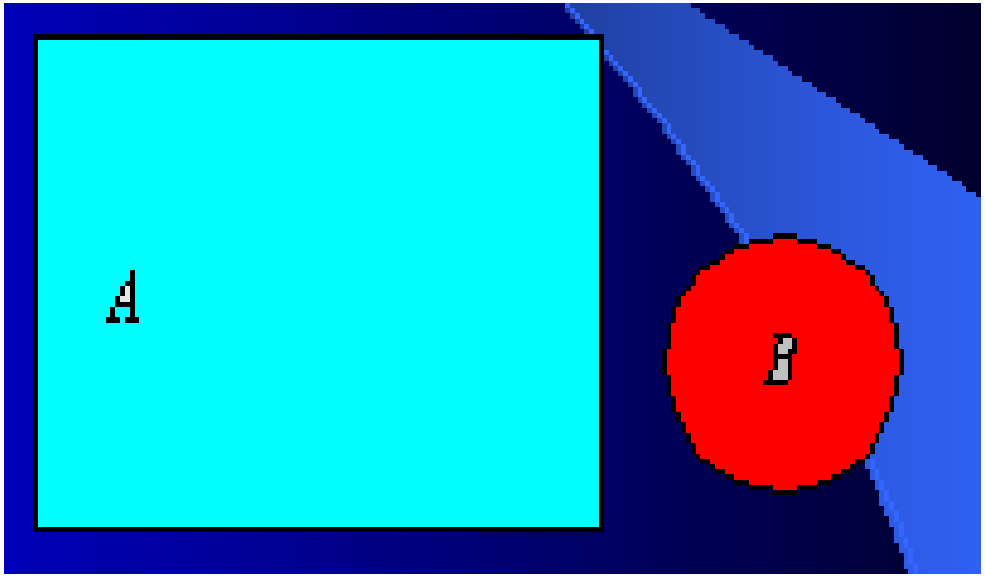
- 对称

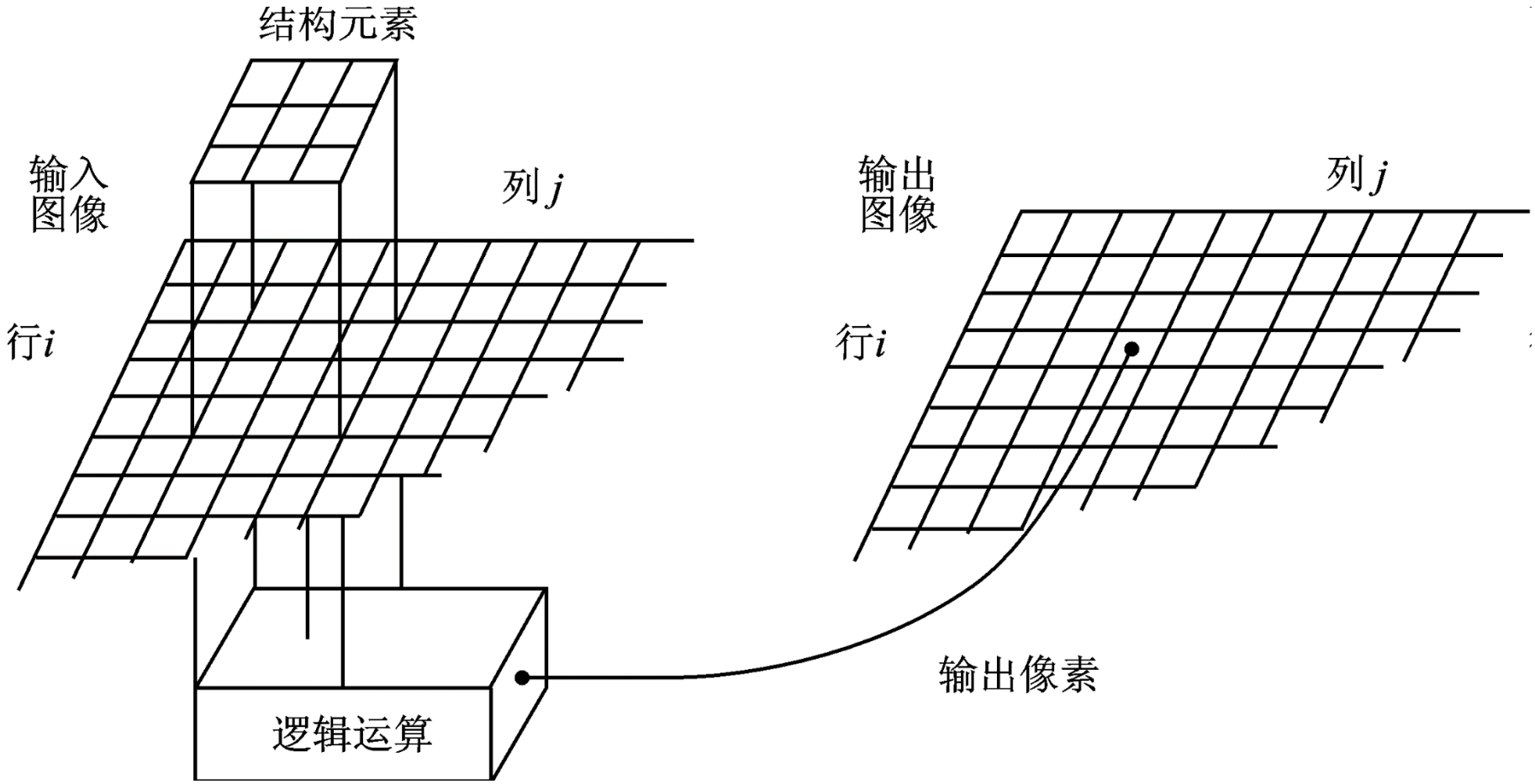
  设有一幅图像B，将B中所有元素的坐标取反，即令元素(x,y)变成$(-x,-y)$，所有这些点构成的新的集合称为B的对称集，记作B^v.



结构元素

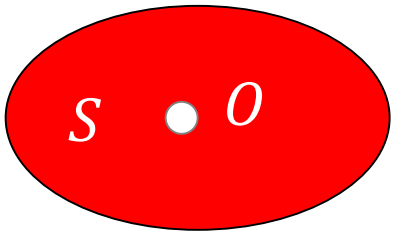
- 设有两幅图像B和A。若A是被处理的对象，而B是用来处理A的，则称B为结构元素，又被形象地称做刷子。结构元素通常都是一些比较小的图像。




结构元素
输入
图像
列 j
行 i
输出
图像
列 j
行 i
输出像素
逻辑运算

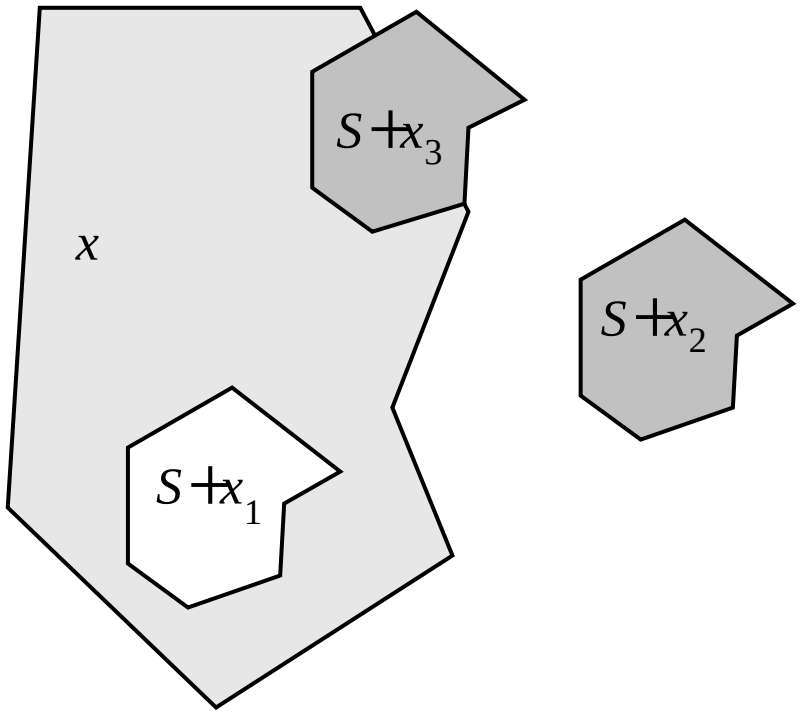
二值形态学

- 二值形态学中的运算对象是集合。设A为图像集合，S为结构元素，数学形态学运算是用S对A进行操作。
- 实际上结构元素本身也是一个图像集合。对每个结构元素可以指定一个原点，它是结构元素参与形态学运算的参考点。应注意，原点可以包含在结构元素中，也可以不包含在结构元素中，但运算的结果常不相同。



腐蚀

- 对一个给定的目标图像X和一个结构元素S，想象一下将S在图像上移动。在每一个当前位置x，$S+x$只有三种可能的状态：
  - (1) $S+x \subseteq X$；
  - (2) $S+x \subseteq X^C$；
  - (3) $S+x \cap X$与$S+x \cap X^C$均不为空。

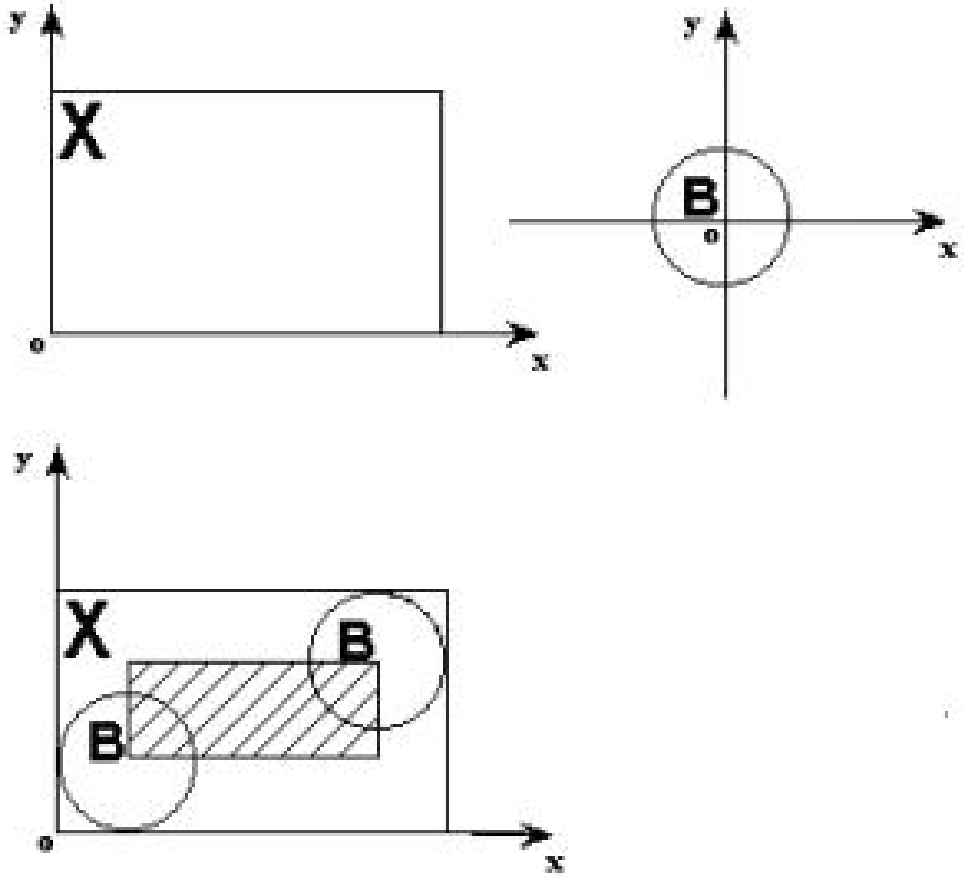


腐蚀

- 腐蚀是最基本的一种数学形态学运算。
- 腐蚀也可以用集合的方式定义，即

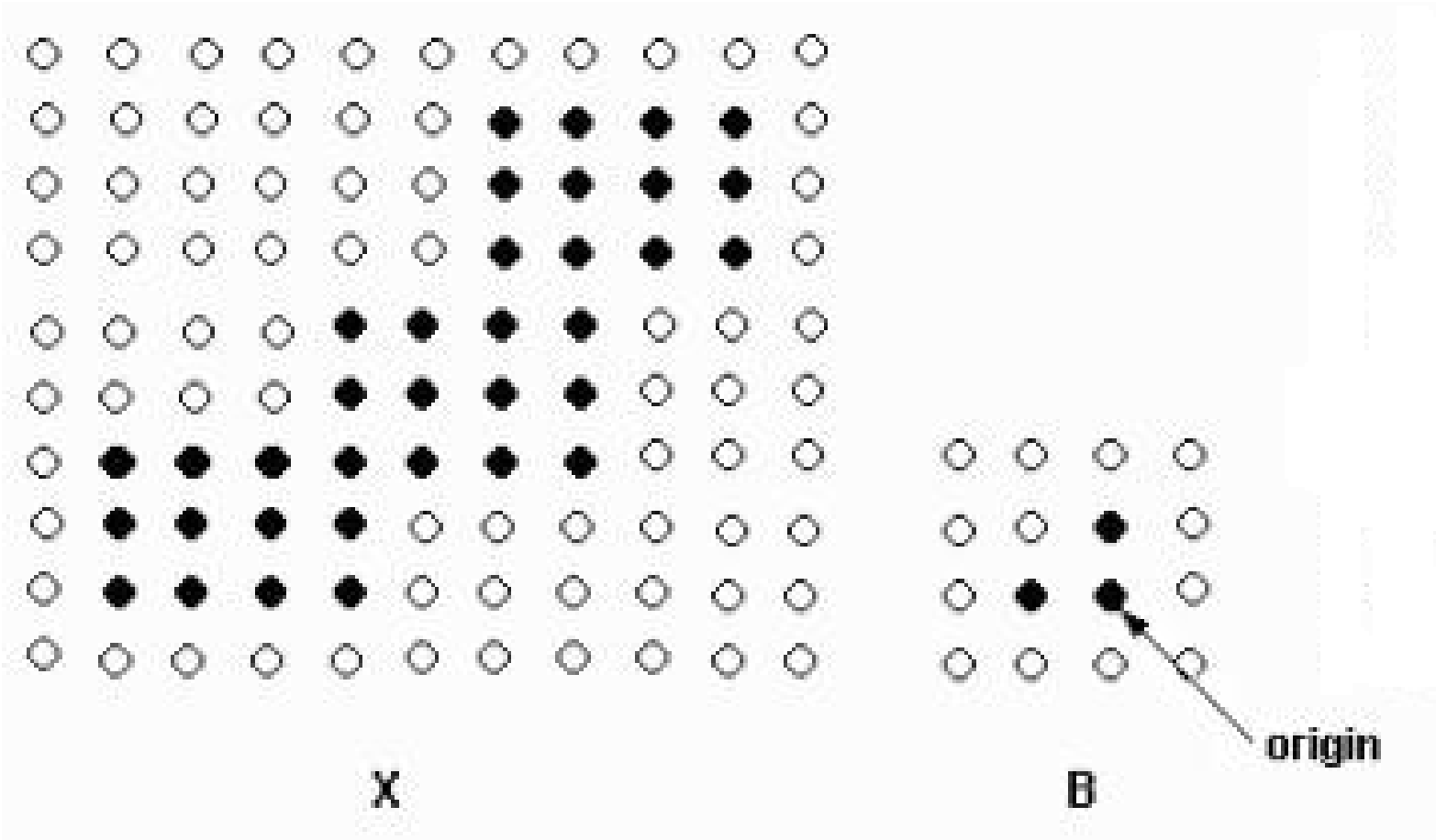
$$X \ominus S = \{x | S + x \subseteq X\}$$

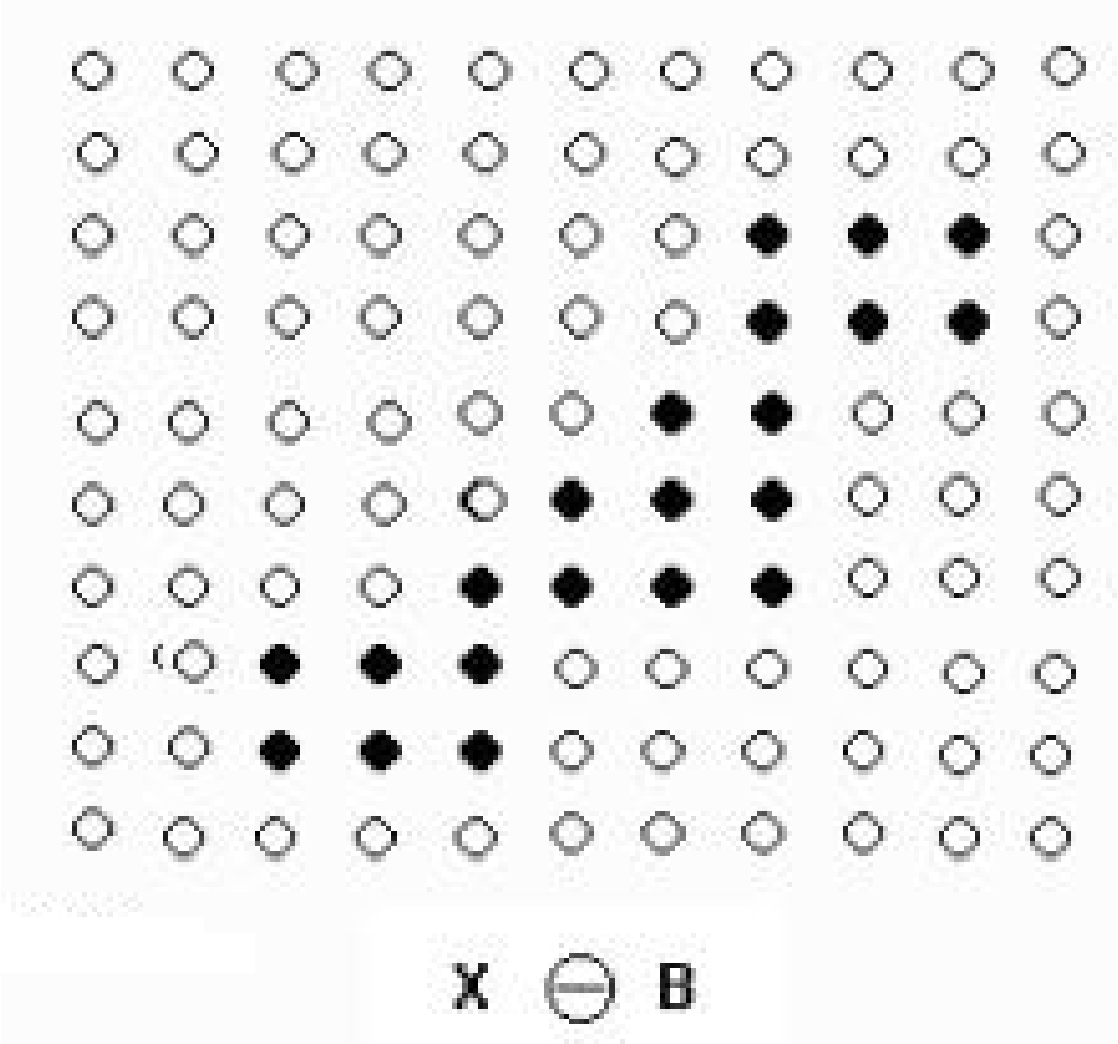
- X用S腐蚀的结果是所有使S平移x后仍在X中的x的集合。换句话说，用S来腐蚀X得到的集合是S完全包括在X中时S的原点位置的集合。

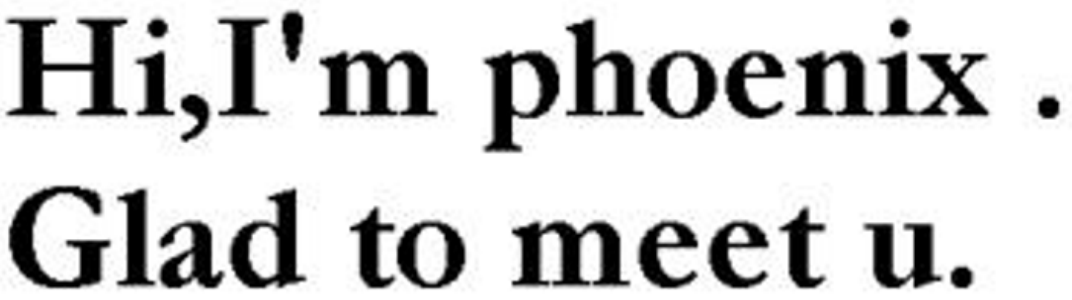


对于任意一个在阴影部分的点a，B_a 包含于X，所以X被B腐蚀的结果就是那个阴影部分。阴影部分在X的范围之内，且比X小，就像X被剥掉了一层似的，这就是叫腐蚀的原因

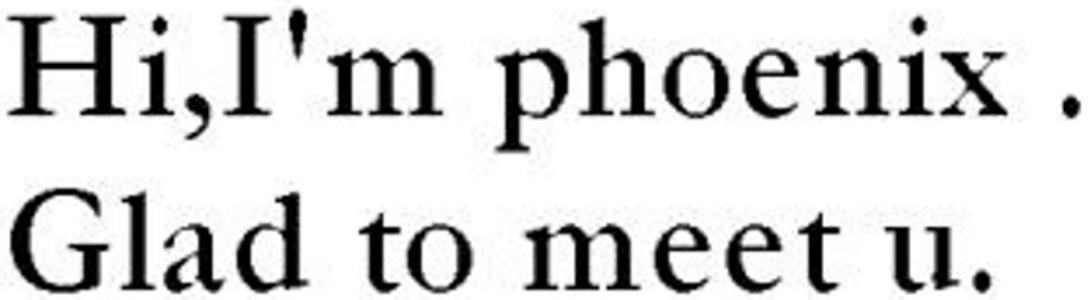
腐蚀在数学形态学运算中的作用是消除物体边界点。


origin
X
B


X
B



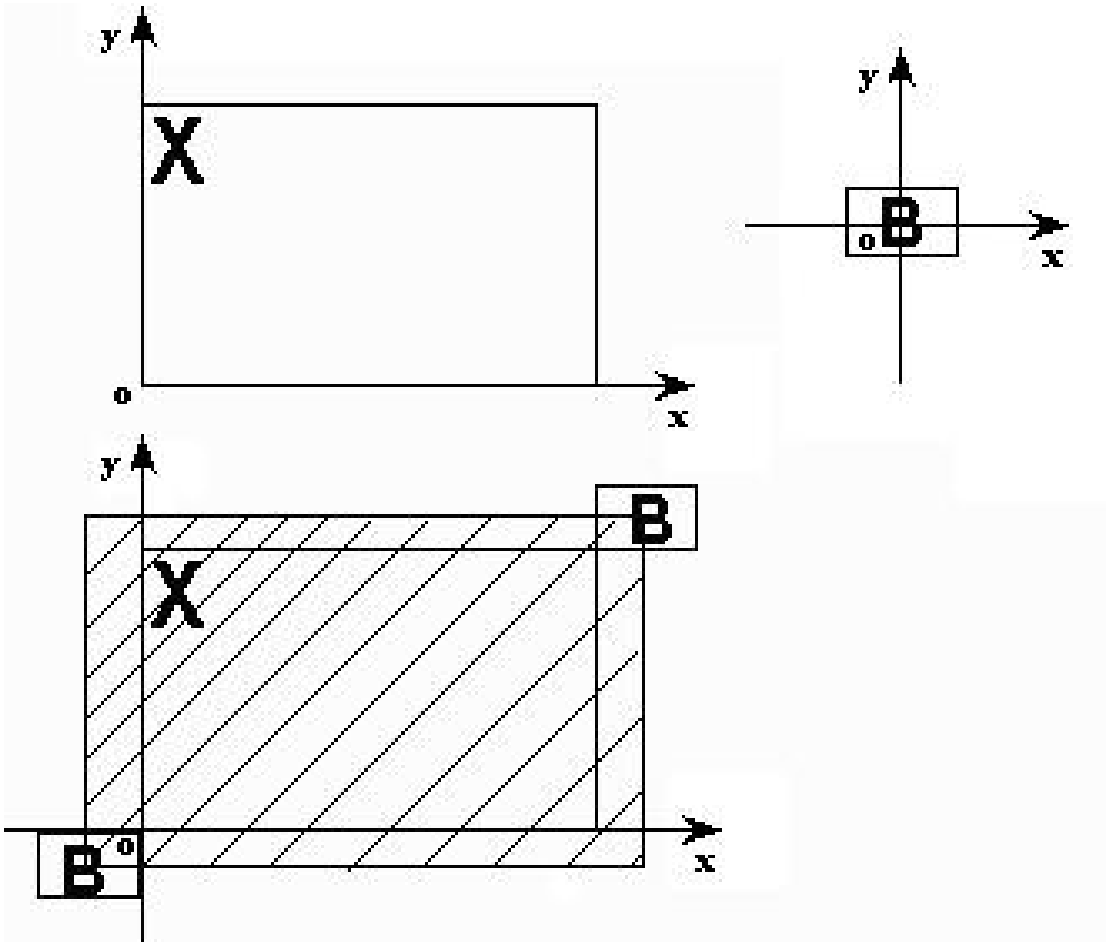
原图



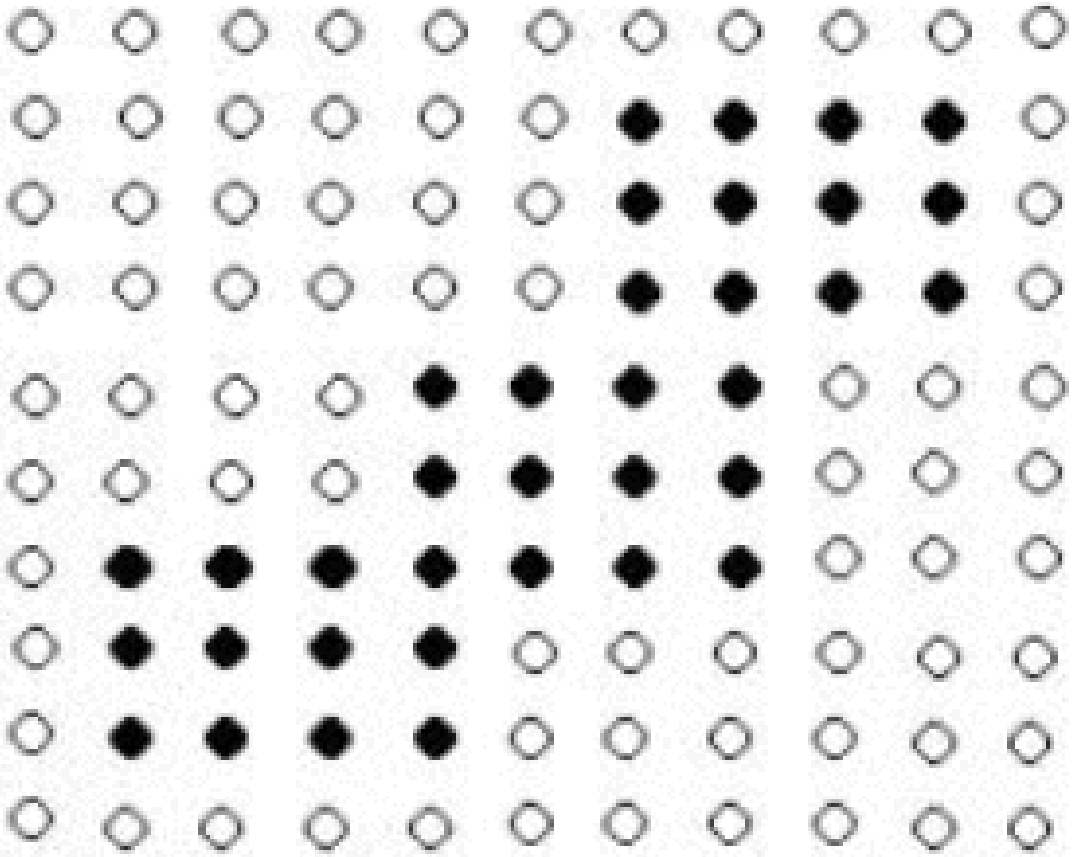
腐蚀之后的结果图

膨胀

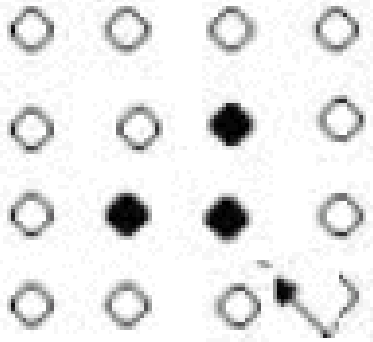
- 腐蚀与膨胀
  - 腐蚀可以看作是将图像X中每一与结构元素S全等的子集$S+x$收缩为点x。
  - 反之，也可以将X中的每一个点x扩大为$S+x$，这就是膨胀运算，记为$X \oplus S$。
  - 膨胀可以看做是腐蚀的对偶运算。
- 膨胀的定义是：把结构元素B平移a后得到B_a，若B_a击中X，我们记下这个a点。所有满足上述条件的a点组成的集合称做X被B膨胀的结果。
- 若用集合语言，膨胀的定义为$X \oplus S = \{x | S + x \cap X \neq \emptyset\}$



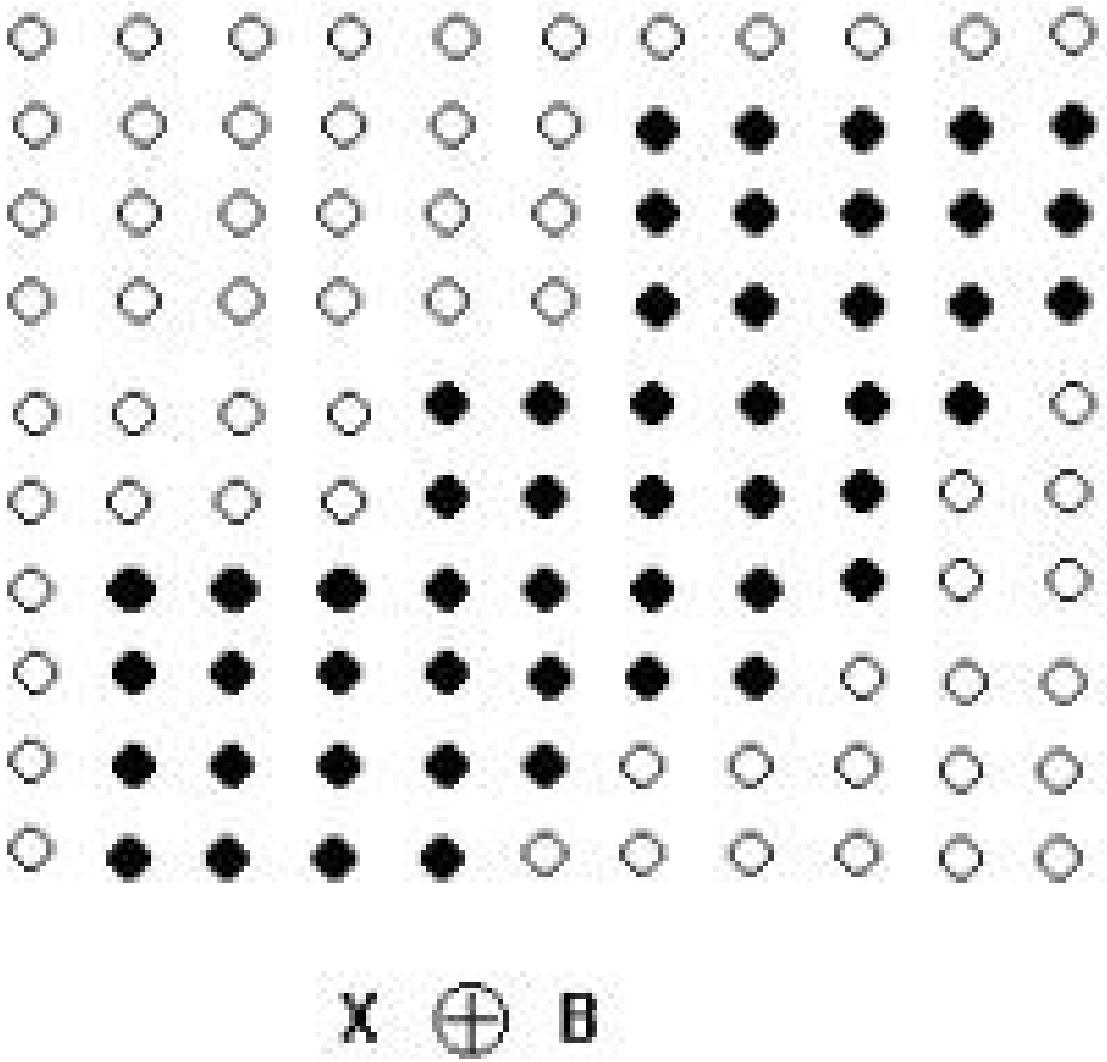
图中X是被处理的对象，B是结构元素，对于任意一个在阴影部分的点a，B_a击中X，所以X被B膨胀的结果就是那个阴影部分。阴影部分包括X的所有范围，就像X膨胀了一圈似的，这就是叫膨胀的原因。

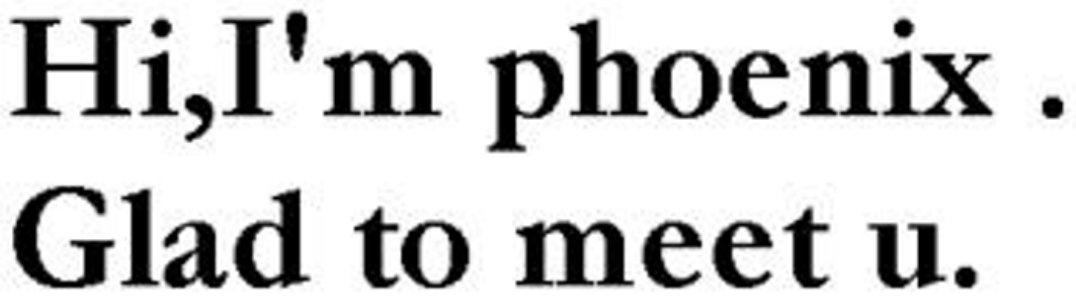


X



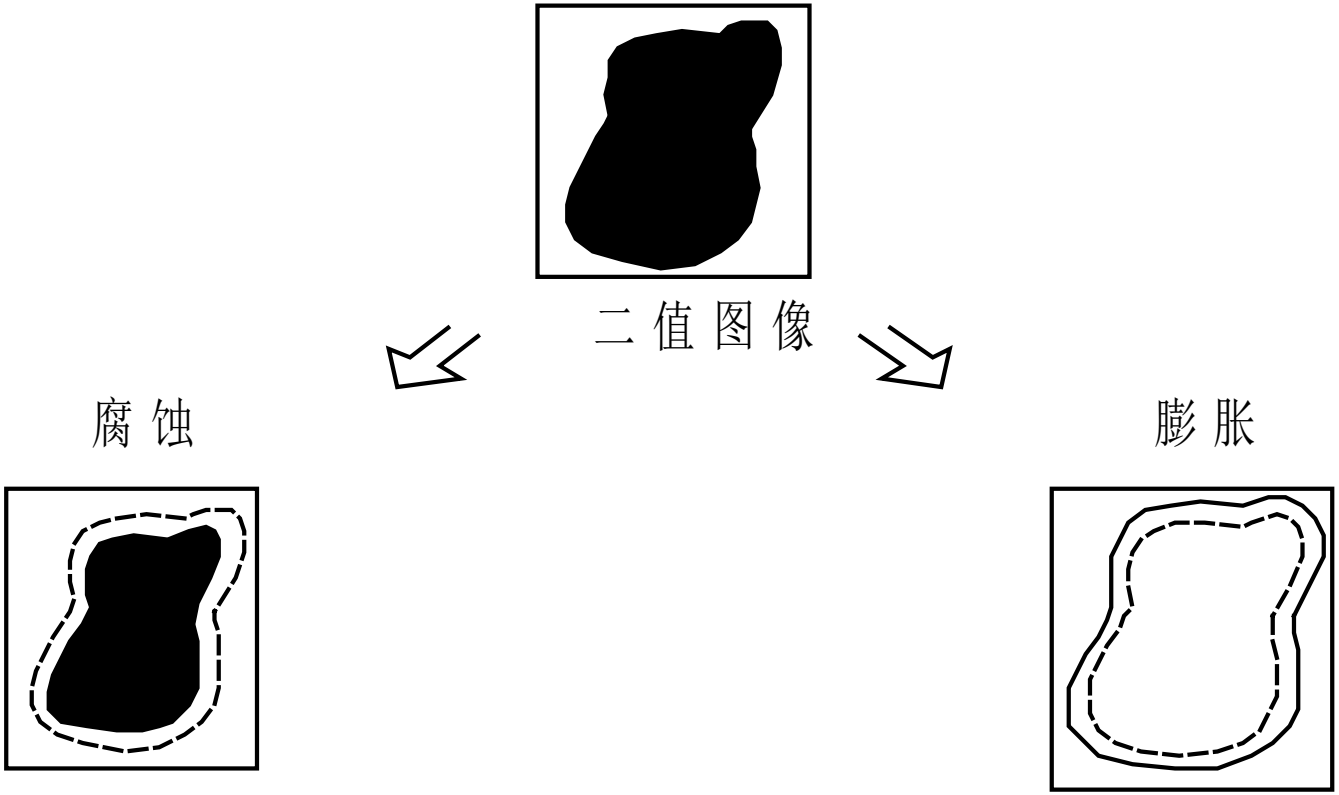
B


X
B

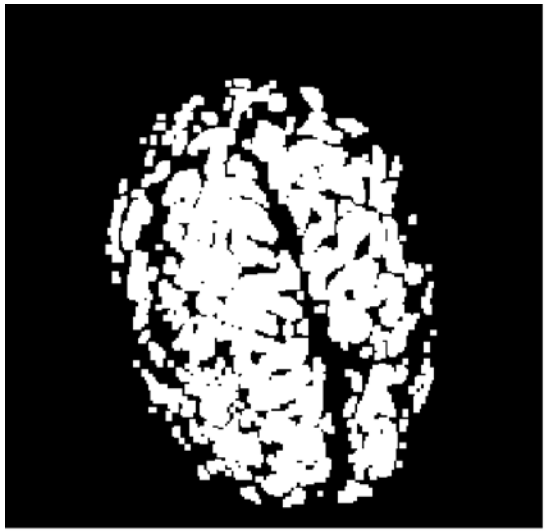


原图

膨胀之后的结果图



腐蚀与膨胀示意图



(a) 原始图像

(b) 腐蚀后图像1

(c) 膨胀后图像1

(d) 腐蚀后图像2

(e) 膨胀后图像2

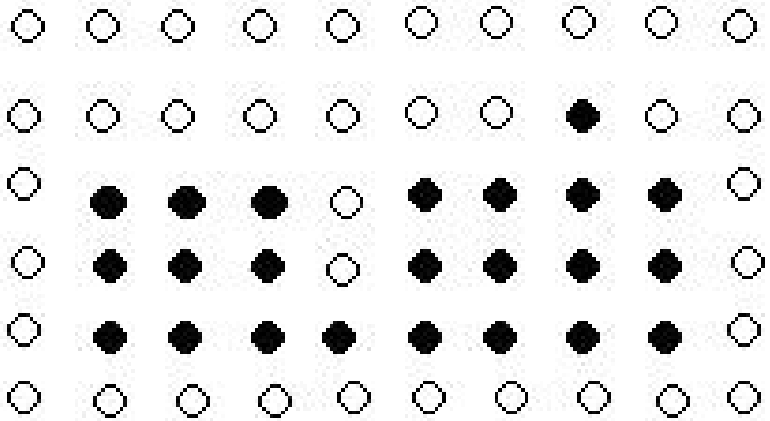
开操作与闭操作

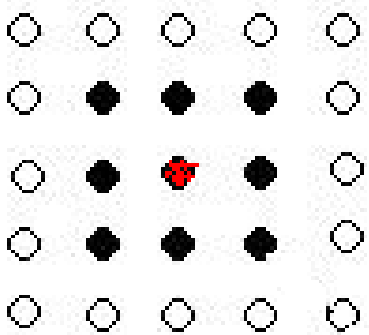
- 膨胀和腐蚀
  - 膨胀会扩大一幅图像的组成成分；
  - 腐蚀会缩小一幅图像的组成成分。
- 开操作和闭操作
  - 开操作一般会平滑物体的轮廓、断开较窄的狭颈并消除细的突出物；
  - 闭操作同样也会平滑轮廓的一部分，通常会弥合较窄的间断和细长的沟壑，消除小孔的空洞，填补轮廓线中的断裂。

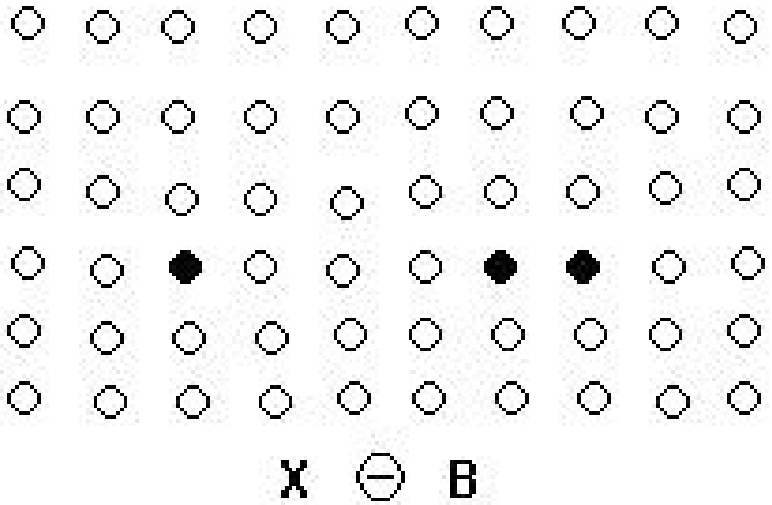
开操作与闭操作

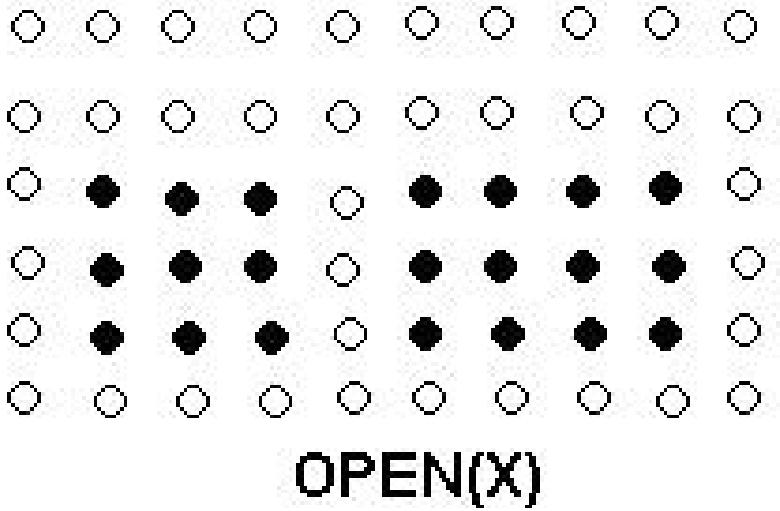
- 开操作
  - 先腐蚀后膨胀称为开
- 对图像X及结构元素S，用符号$X \circ S$表示S对图像X作开操作

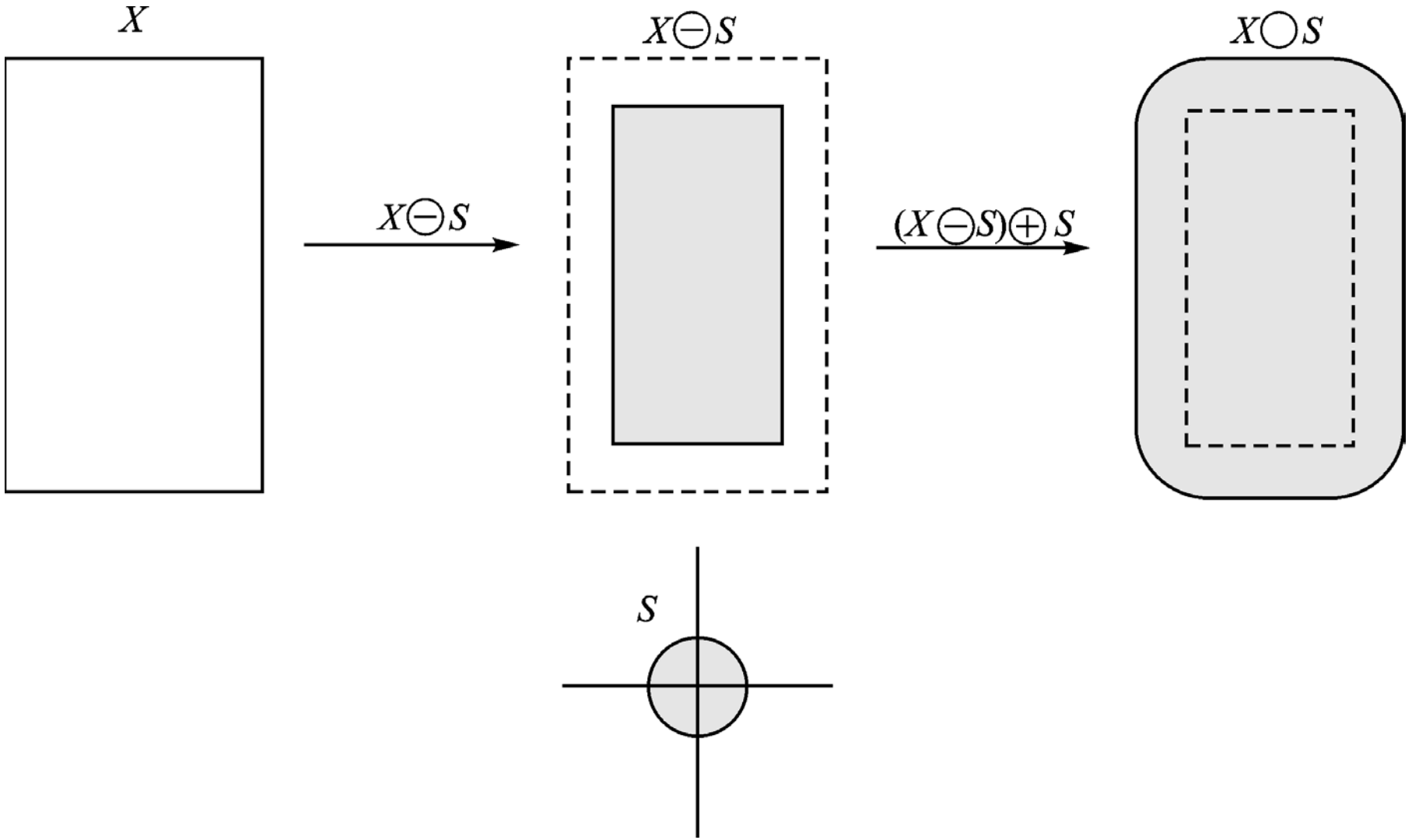
$$X \circ S = (X \ominus S) \oplus S$$

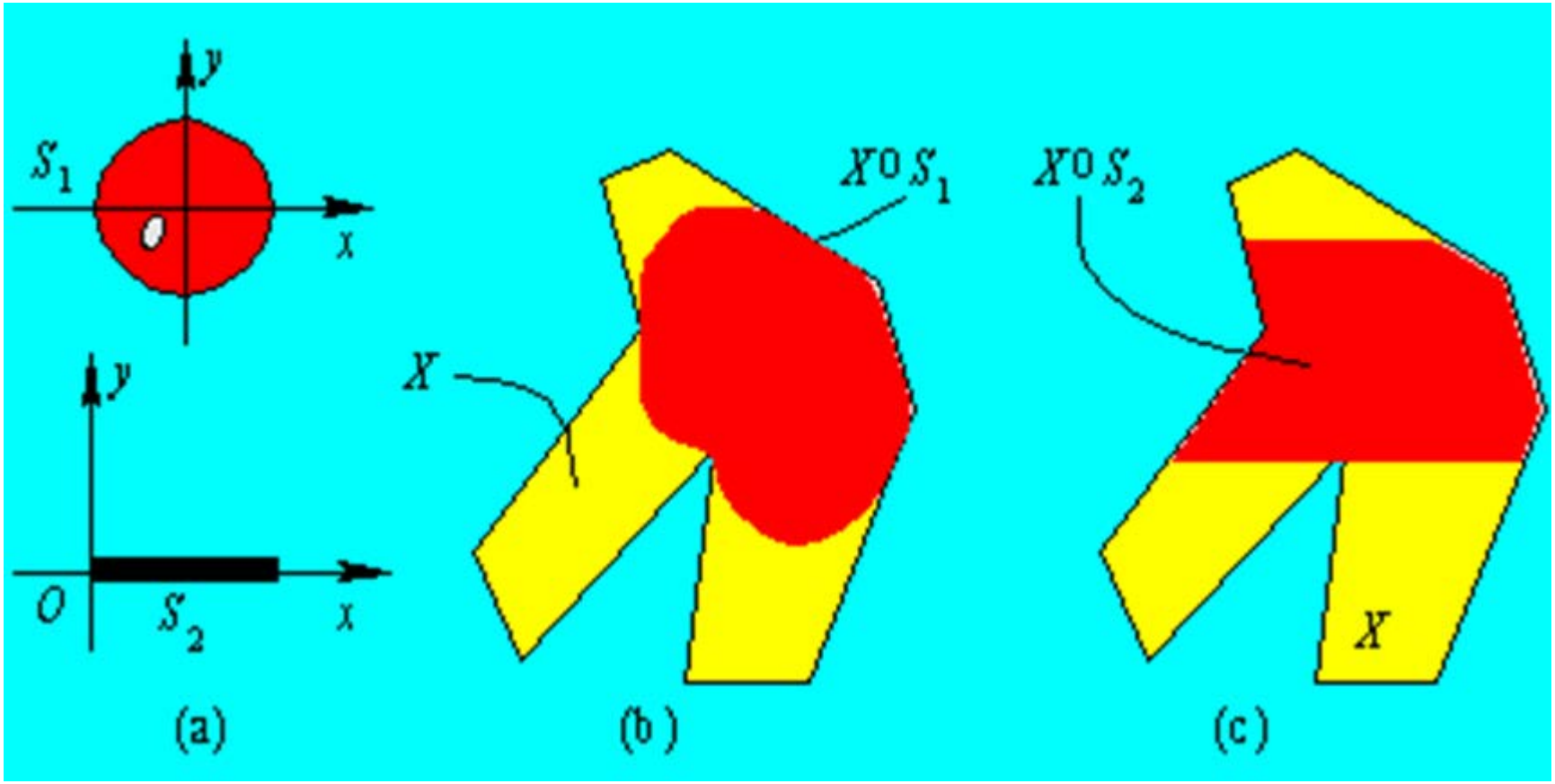

X


B


X ⊖ B


OPEN(X)

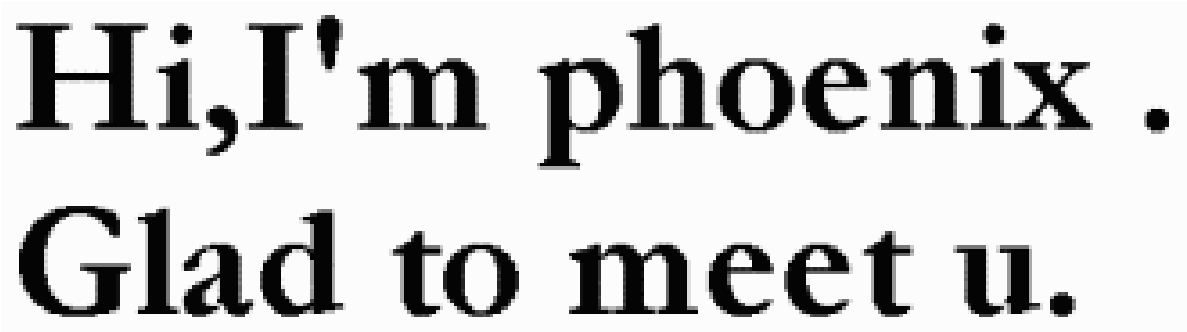

X
X⊖S
X⊖S
(X⊖S)⊕S
X◯S
S



开操作去掉了凸角

(a)结构元素S_1和S_2；(b) $X \circ S_1$；(c) $X \circ S_2$

原图



开操作之后的结果图

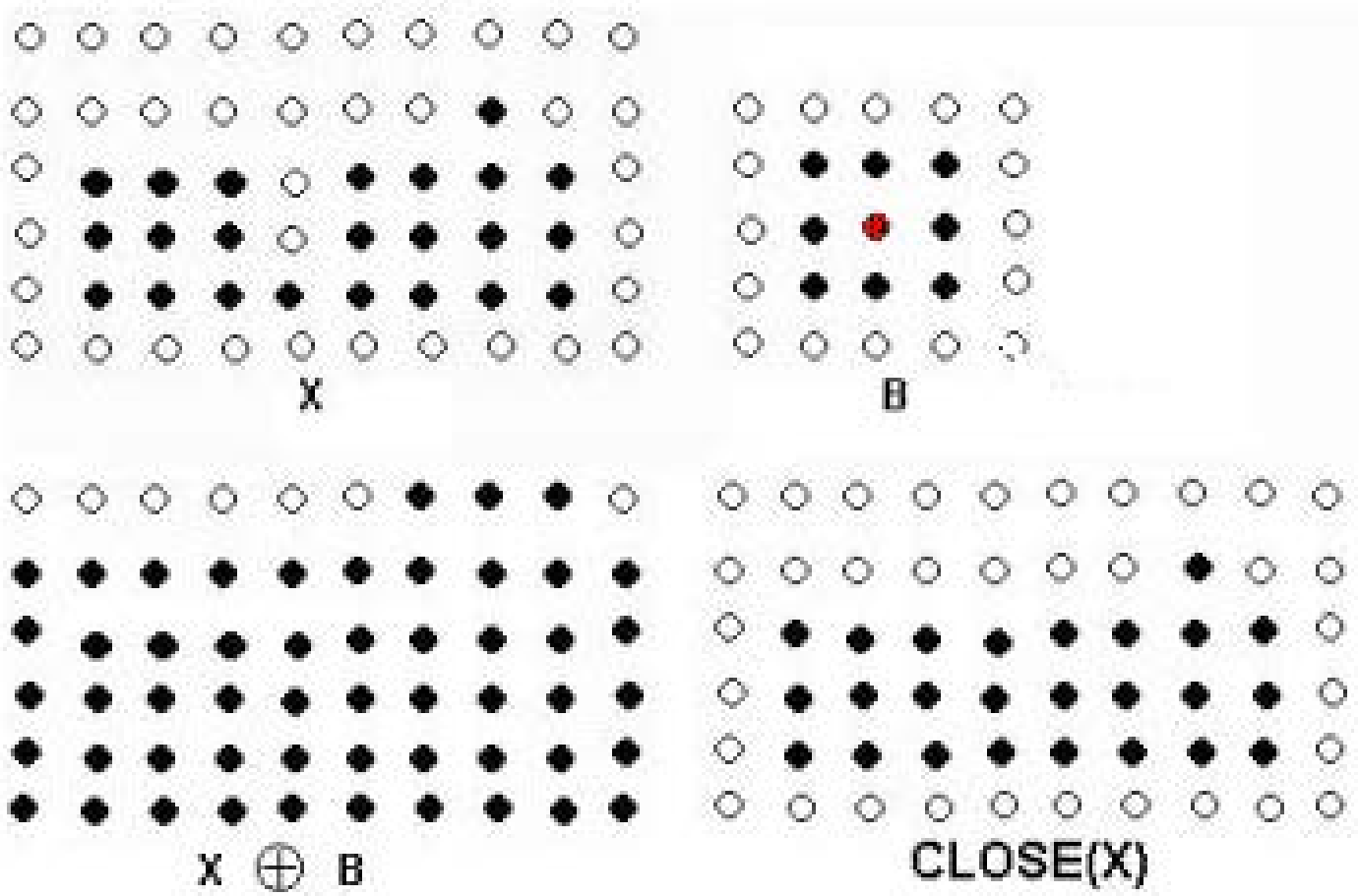
开操作的性质

- 我们可以得到关于开操作的几点结论：
  - 开操作能够除去孤立的小点，毛刺，而总的位置和形状不便；
  - 开操作是一个基于几何运算的滤波器；
  - 结构元素大小的不同将导致滤波效果的不同；
  - 不同的结构元素的选择导致了不同的分割，即提取出不同的特征。

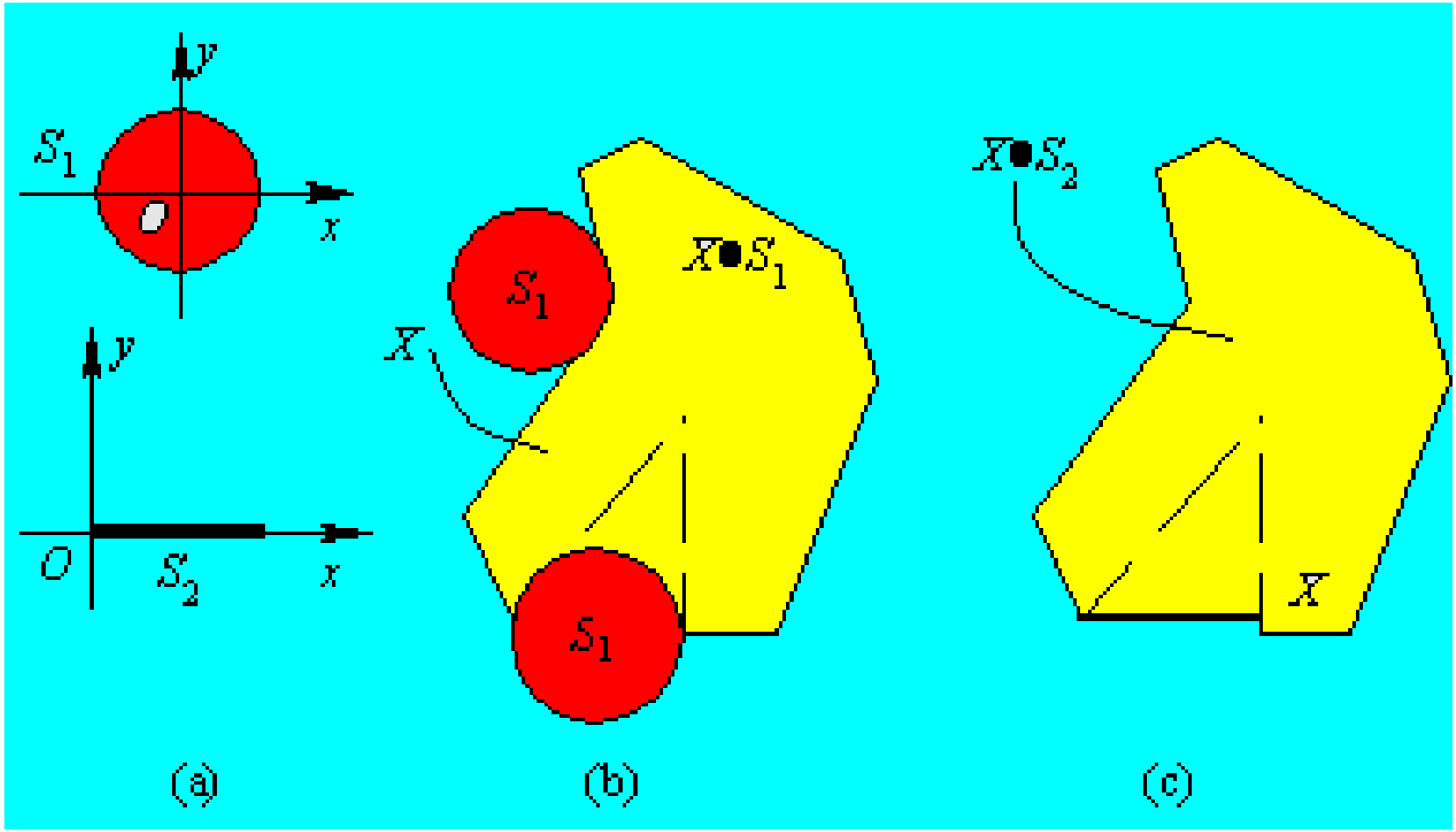
开操作与闭操作

- 闭操作
  - 先膨胀后腐蚀称为闭
- 对图像X及结构元素S，用符号$X \bullet S$表示S对图像X作闭操作

$$X \bullet S = (X \oplus S) \ominus S$$



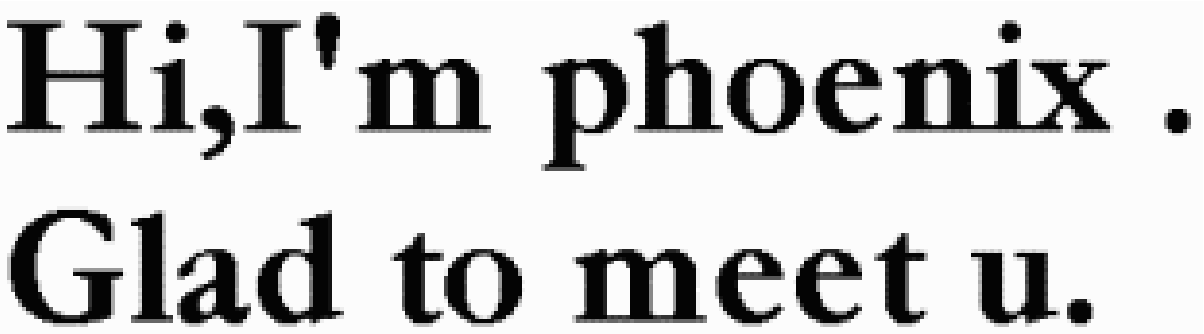
一般来说，闭操作能够填平小湖(即小孔)，弥合小裂缝，而总的位置和形状不变。



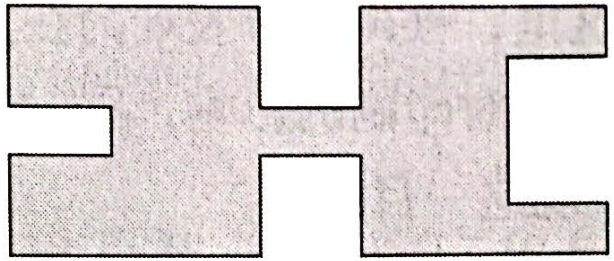
闭操作填充了凹角

(a) 结构元素S_1和S_2；(b) $X \bullet S_1$； (c) $X \bullet S_2$

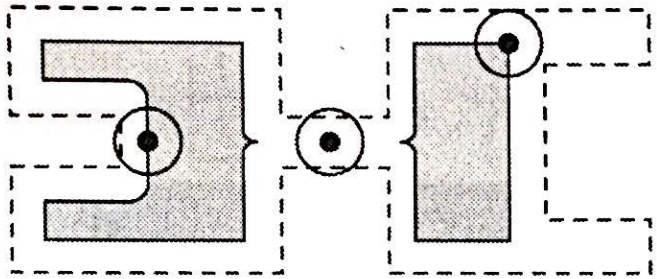
原图

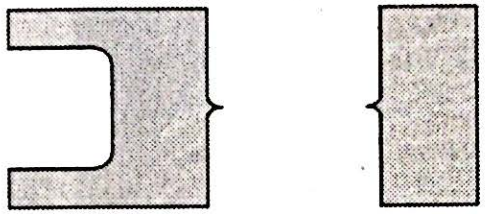


闭操作之后的结果图

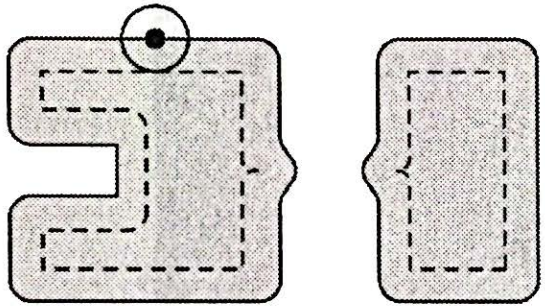


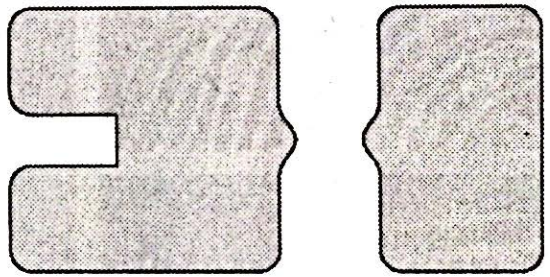
A



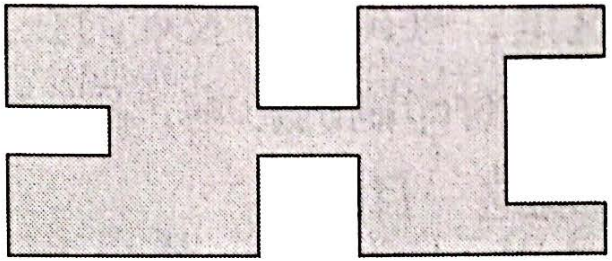


$A \ominus B$

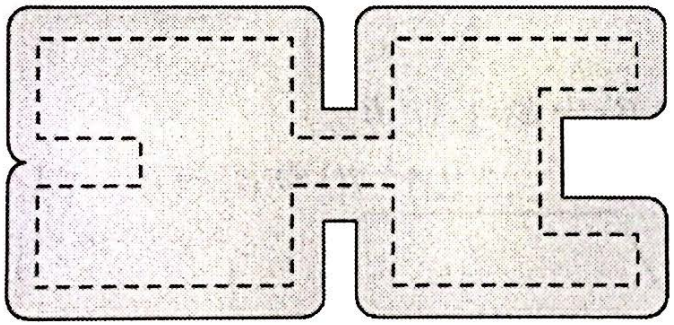


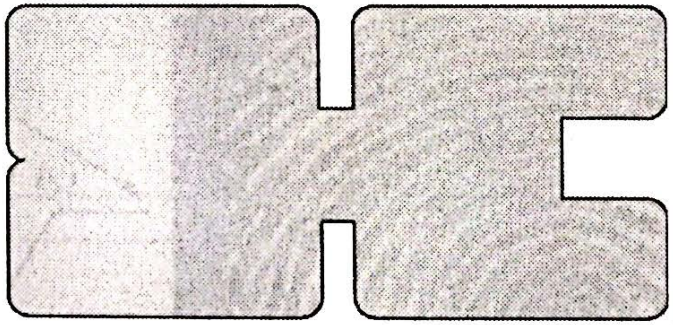


$A \circ B = (A \ominus B) \oplus B$

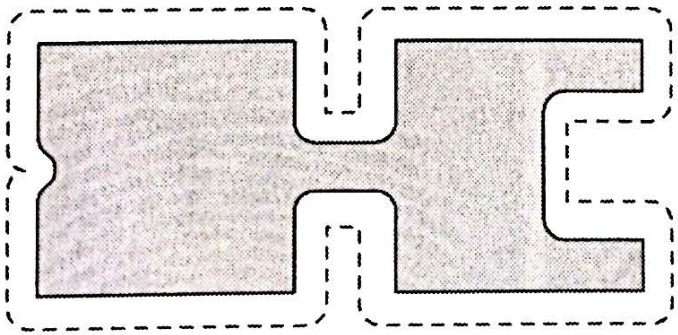


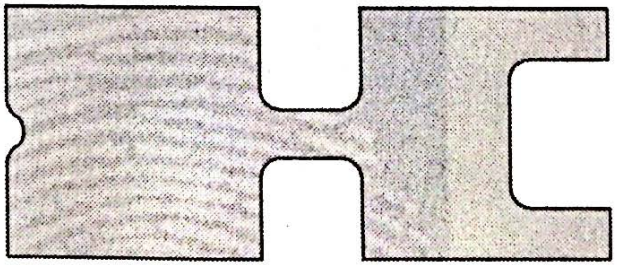
A





$A \oplus B$





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

闭操作的性质

- 我们可以得到关于闭操作的几点结论：
  - 闭操作能够填平小湖（即小孔），弥合小裂缝，而总的位置和形状不变；
  - 闭操作是通过填充图像的凹角来滤波图像的；
  - 结构元素大小的不同将导致滤波效果的不同；
  - 不同结构元素的选择导致了不同的分割。

开闭操作的代数性质

- 由于开、闭操作是在腐蚀和膨胀运算的基础上定义的， 根据腐蚀和膨胀运算的代数性质，我们不难得到下面的性质。
- 对偶性

$$(X^C \circ S)^C = X \bullet S\,, (X^C \bullet S)^C = X \circ S$$

- 扩展性（收缩性）

$$X \circ S \subseteq X \subseteq X \bullet S$$

  - 即开操作恒使原图像缩小，而闭操作恒使原图像扩大

开闭操作的代数性质

- 单调性
  - 如果$X \subseteq Y$，则

$$X \bullet S \subseteq Y \bullet S, X \circ S \subseteq Y \circ S$$

  - 如果$Y \subseteq Z$且$Z \bullet Y = Z$，那么

$$X \bullet Y \subseteq Y \bullet Z$$

根据这一性质可以知道，

结构元素的扩大只有在保证扩大后的结构元素对原结构元素闭操作不变的条件下方能保持单调性。

开闭操作的代数性质

- 平移不变性

$$(X+h)\bullet S=(X\bullet S)+h,\quad (X+h)\circ S=(X\circ S)+h$$

$$X\bullet(S+h)=X\bullet S,\quad X\circ(S+h)=X\circ S$$

- 等幂性

$$(X\bullet S)\bullet S\ =\ X\bullet S,\quad (X\circ S)\circ S\ =\ X\circ S$$

开、闭操作的等幂性意味着一次滤波就能把所有特定结构元素的噪声滤除干净，作重复的运算不会再有效果。这是一个与经典方法 (例如中值滤波、线性卷积)不同的性质。

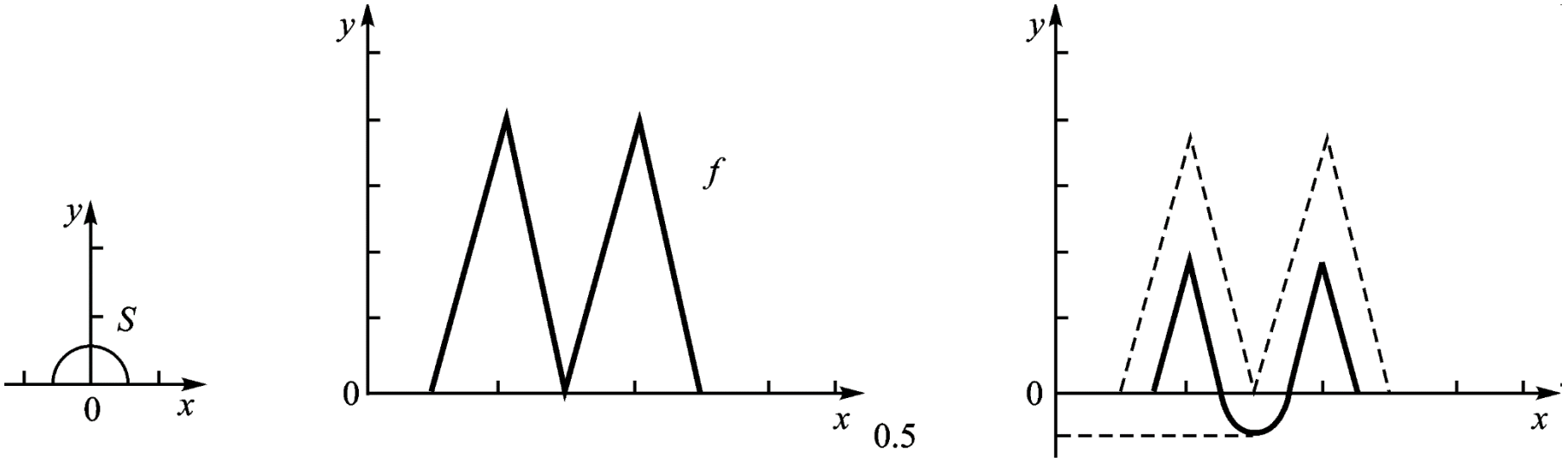
灰值形态学

- 灰度腐蚀
- 用结构元素对输入图像$f(x,y)$进行灰值腐蚀记为$f \ominus S$，其定义为

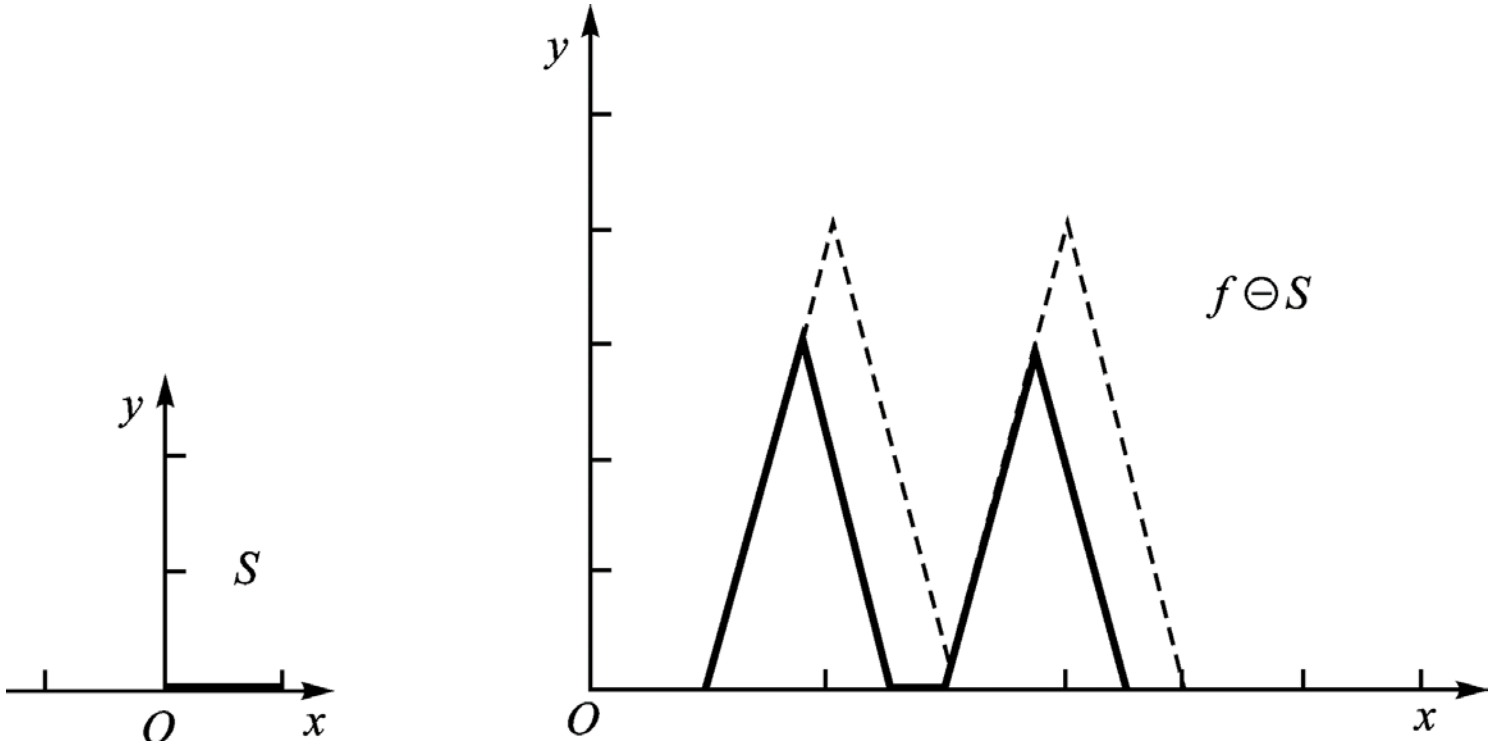
$$(f \Theta s)(t,m) = \min\{ f(t+x, m+y) - s(x,y) \,|\, t+x, m+y \in D_f, x+y \in D_s \}$$

式中，D_f和D_b分别是f和b的定义域。

这里限制$(t+x)$和$(m+y)$在f的定义域之内，类似于二值腐蚀定义中要求结构元素完全包括在被腐蚀集合中。



- 其效果相当于半圆形结构元素在被腐蚀函数的下面“滑动”时，其圆心画出的轨迹。
- 但是，这里存在一个限制条件，即结构元素必须在函数曲线的下面平移。
- 从图中不难看出，半圆形结构元素从函数的下面对函数产生滤波作用，这与圆盘从内部对二值图像滤波的情况是相似的。



采用了一个扁平结构元素对上图的函数作灰值腐蚀。

扁平结构元素是一种在其定义域上取常数的结构元素。注意这种结构元素产生的滤波效果。

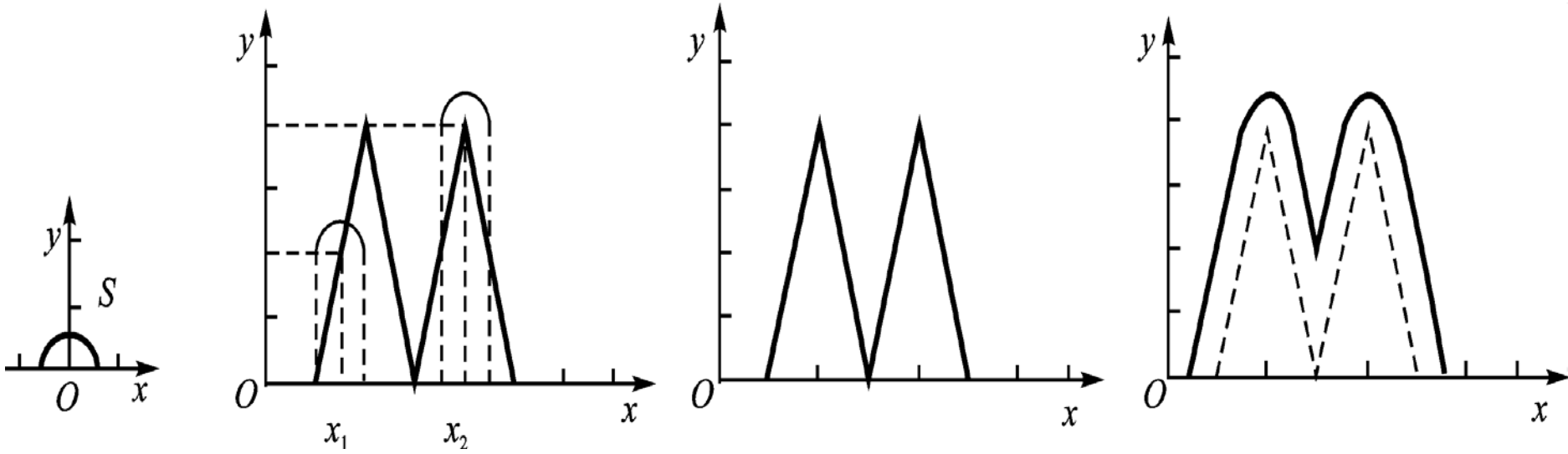
灰值形态学

- 灰值膨胀
- 用结构元素对$S(x,y)$输入图像$f(x,y)$进行灰值膨胀记为$f \oplus S$，其定义为

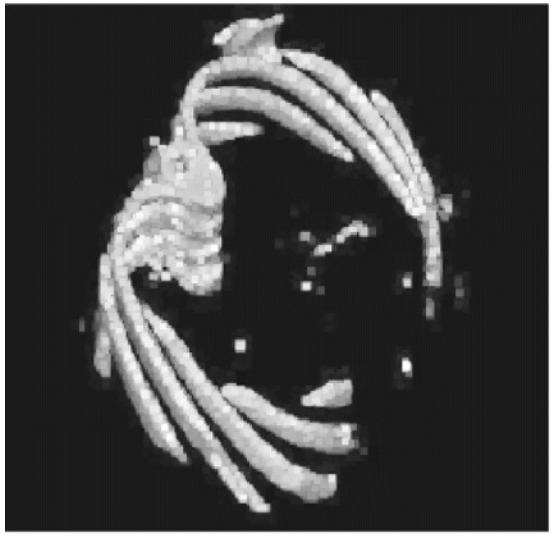
$$(f \oplus s)(t,m) = \max\{f(t-x,m-y)+s(x,y) \mid t-x, m-y \in D_f, x+y \in D_s\}$$

式中，D_f和D_S分别是f和S的定义域。

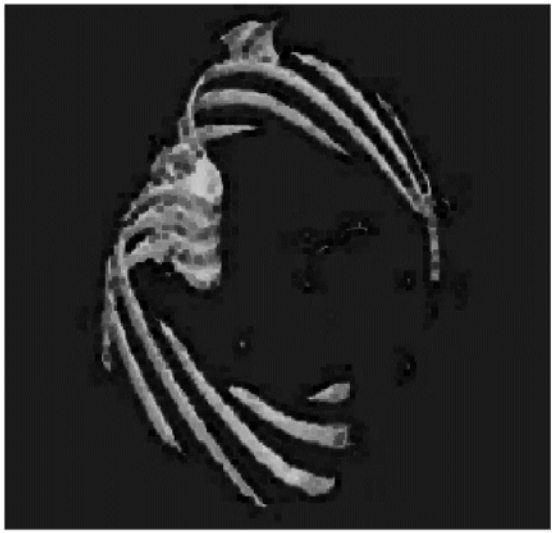
这里限制$(t-x)$和$(m-y)$在f的定义域之内，类似于二值膨胀定义中中要求两个运算集合至少有一个（非零）元素相交。。



灰度膨胀可以通过将结构元素的原点平移到与信号重合，然后，对信号上的每一点求结构元素的最大值得到。

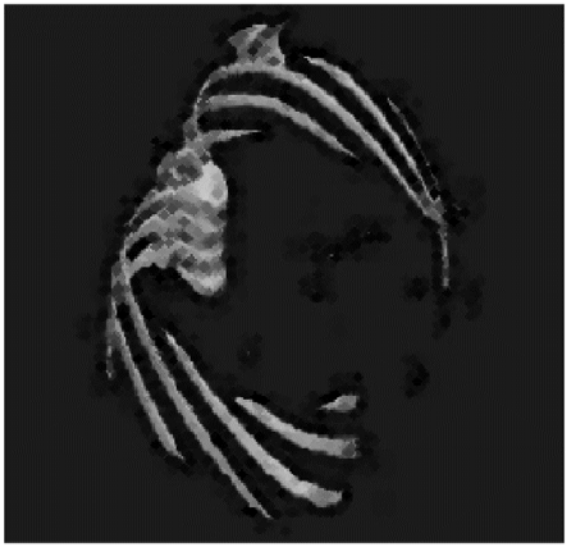


(a) 原始图像

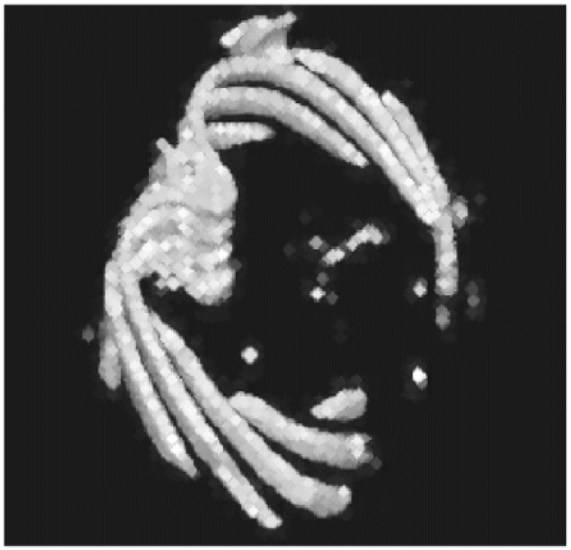


(b) 腐蚀后图像1

(c) 膨胀后图像1



(d) 腐蚀后图像2

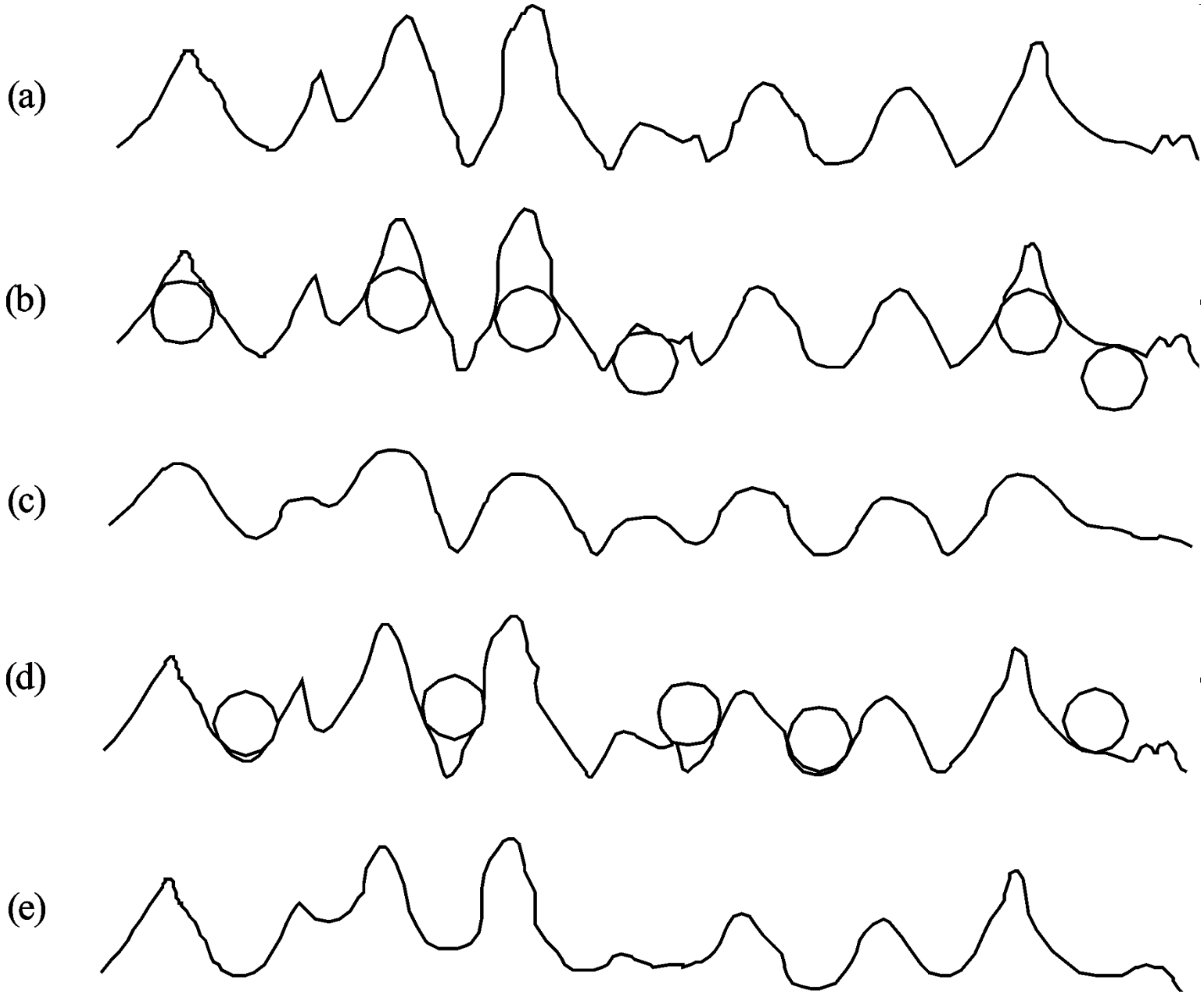


(e) 膨胀后图像2

开操作与闭操作

- 灰度开操作
  - 用结构元素S(灰值图像)对灰值图像f做开操作记为$f \circ S$，其定义为 $f \circ S = (f \ominus S) \oplus S$

- 灰度闭操作
  - 用结构元素S(灰值图像)对灰值图像f做开操作记为$f \bullet S$，其定义为$f \bullet S = (f \oplus S) \ominus S$


(a)
(b)
(c)
(d)
(e)

开操作与闭操作

- 开操作
  - 可看作将b贴着f的下沿从一端滚到另一端。对所有比b的直径小的山峰其高度和尖锐度都减弱了。
  - 开操作消除与结构元素相比尺寸较小的亮细节，而保持图像整体灰度值和大的亮区域基本不受影响。
  - 腐蚀去除了小的亮细节并同时减弱了图像亮度，膨胀增加了图像亮度，但又不重新引入前面去除的细节。

开操作与闭操作

- 闭操作
  - 可看作将b贴着f的上沿从一端滚到另一端，所有比b的直径小的山谷得到了“填充”。
  - 闭操作操作消除与结构元素相比尺寸较小的暗细节，而保持图像整体灰度值和大的暗区域基本不受影响。
  - 膨胀去除了小的暗细节并同时增强了图像亮度， 腐蚀减弱了图像亮度但又不重新引入前面去除的细节。

(a) 原始图像

(b) 开运算后图像1

(c) 闭运算后图像1

(d)开运算后图像2

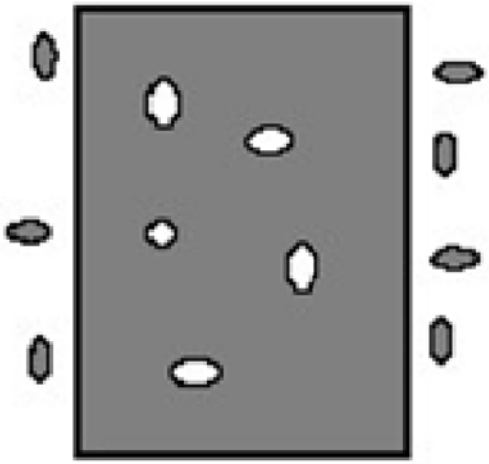
(e) 闭运算图像2

形态学的应用

- 前面已经介绍了二值形态学和灰值形态学的基本运算—腐蚀，膨胀，开和闭操作及其一些性质，通过对它们的组合可以得到一系列二值形态学和灰值形态学的实用算法。
- 灰值形态学的主要算法有灰值形态学梯度，形态学平滑，纹理分割等。
- 本节主要介绍形态学滤波，骨架抽取等重要算法。注意到，在实际应用形态学方法时，通常需要对输入图像做预处理，以便适合于使用这些算法。同时对输出图像可能还要做一些处理，以便产生满意的结果。

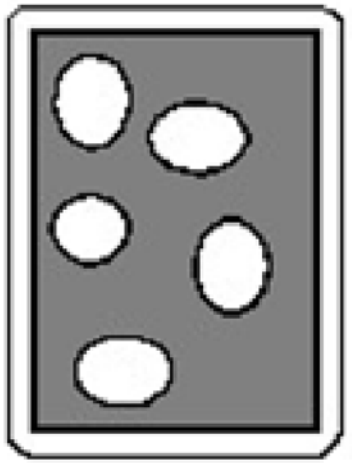
形态学滤波

- 由于开、闭操作所处理的信息分别与图像的凸、凹处相关。 因此，它们本身都是单边算子。
- 可以利用开、闭操作去除图像的噪声、恢复图像，也可交替使用开、闭操作以达到双边滤波目的。
- 一般，可以将开、闭操作结合起来构成形态学噪声滤波器，例如$(X \circ S)\bullet S$或$(X\bullet S) \circ S$等。

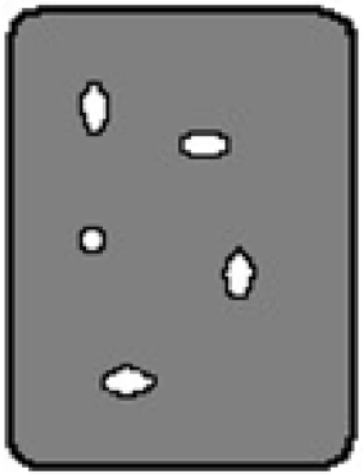


(a) 含噪声的图像

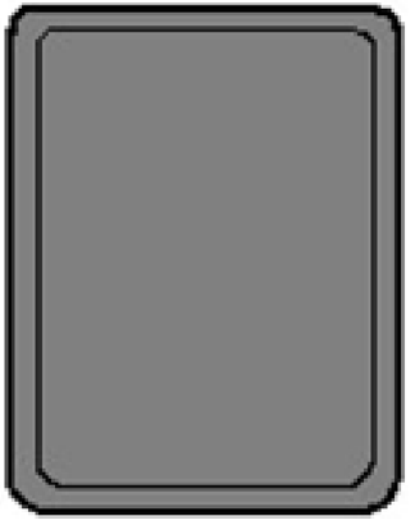
(b) 结构元素



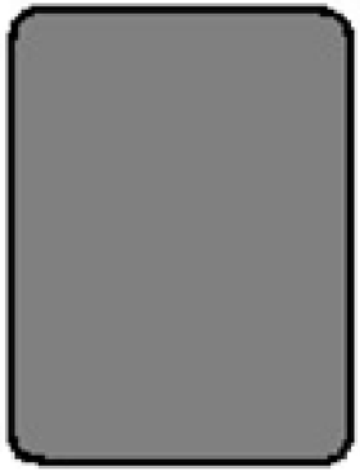
(c) 腐蚀后的图像



(d) 开运算后的图像



(e) 开运算基础上的膨胀图像

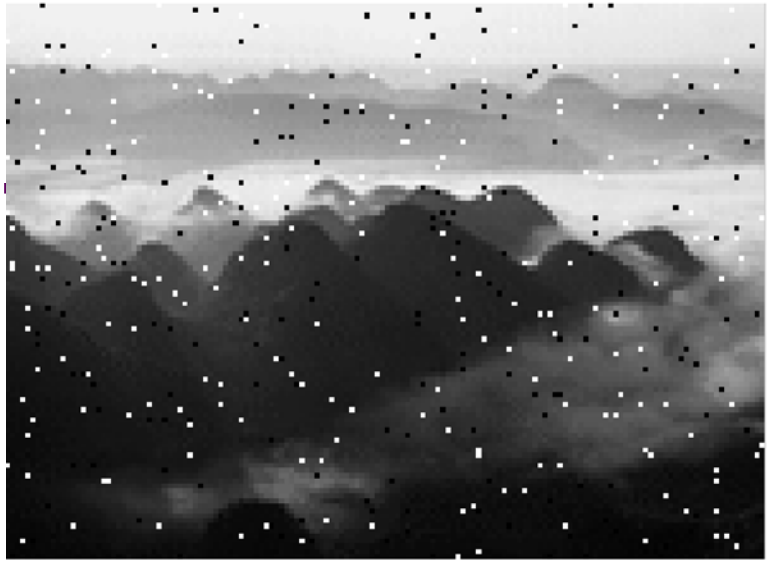


(f) 先开运算后闭运算的图像

上述整个过程是先做开操作再做闭操作，可以写为

$$\{[(X \ominus S) \oplus S] \oplus S\} \ominus S = (X \circ S) \bullet S$$

- 可看出目标区域内外的噪声都消除掉了，而目标本身除原来的4个直角变为圆角外没有太大的变化。
- 在利用开、 闭操作滤除图像的噪声时，选择圆形结构元素会得到较好的结果。
- 为了能使从噪声污染的图像X中恢复原始图像X0的结果达到最优，在确定结构元素的半径时，可以采用优化方法。



(a) 原始图像

(b) 开—闭运算结果

(c) 闭—开结果运算

(d) 交替顺序滤波后的图像

骨架抽取

- 1. 细化
  - 细化就是从原来的图中去掉一些点，但仍要保持原来的形状。实际上，是保持原图的骨架。

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

即$X \otimes B$为X与$X \ominus B$的差集。更一般地，利用结构对序列B^1，B^2，…，B^N迭代地产生输出序列

$$X^1 = X \otimes B^1, X^2 = X^1 \otimes B^2, \cdots, X^N = X^{N-1} \otimes B^N$$

随着迭代的进行，得到的集合也不断细化

骨架抽取

- 2. 细化的重要性
  - 利用细化技术得到区域的细化结构是常用的方法。寻找二值图像的细化结构是图像处理的一个基本问题。在图像识别或数据压缩时，经常要用到这样的细化结构。

骨架抽取

- 3. 骨架
  - 所谓骨架，可以理解为图像的中轴.
  - 例如 ：一个长方形的骨架是它的长方向上的中轴线；圆的骨架是它的圆心； 直线的骨架是它自身； 孤立点的骨架也是自身 。
    - 例1：基于烈火模拟： 设想在$t=0$时刻，将目标边界各处同时点燃，火的前沿以匀速向目标内部蔓延，当前沿相交时火焰熄灭，火焰熄灭点的集合就构成了骨架。

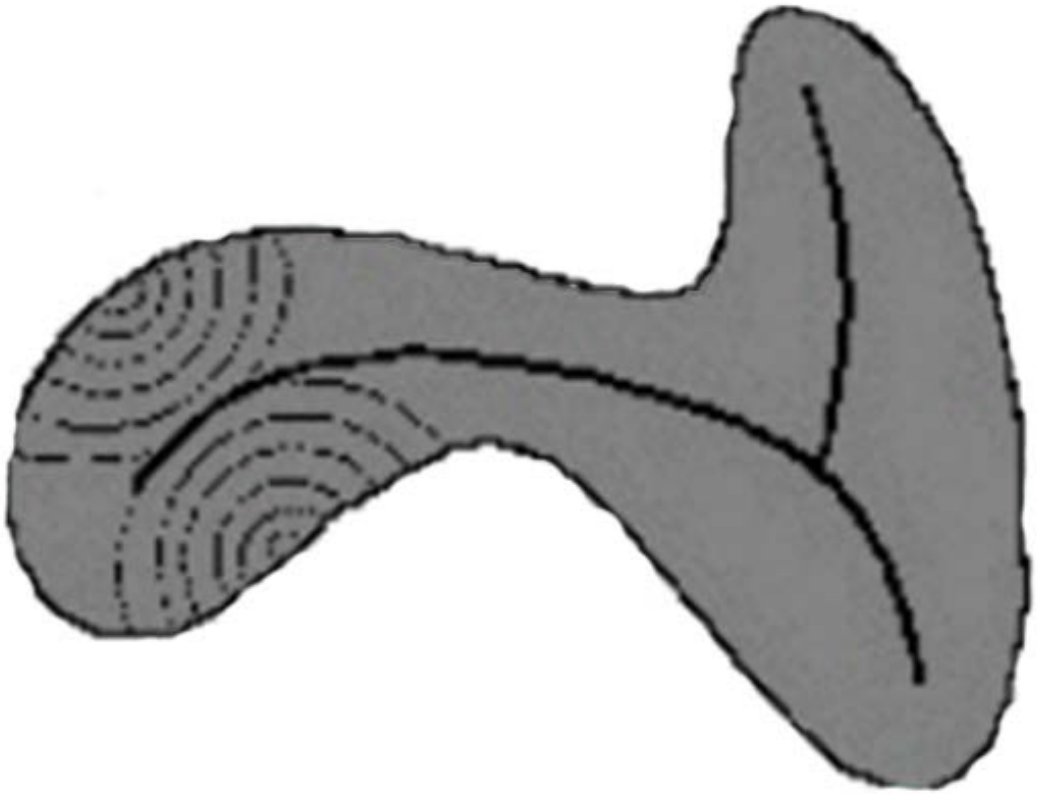


图 骨架的定义

骨架抽取

- 3. 骨架
  - 所谓骨架，可以理解为图像的中轴.
  - 例如 ：一个长方形的骨架是它的长方向上的中轴线；圆的骨架是它的圆心； 直线的骨架是它自身； 孤立点的骨架也是自身 。
    - 例2：基于最大圆盘，目标X的骨架由X内所有最大内切圆盘的圆心组成，最大圆盘不是其他任何完全属于X的圆盘的子集，并且至少有两点与目标边界轮廓相切。

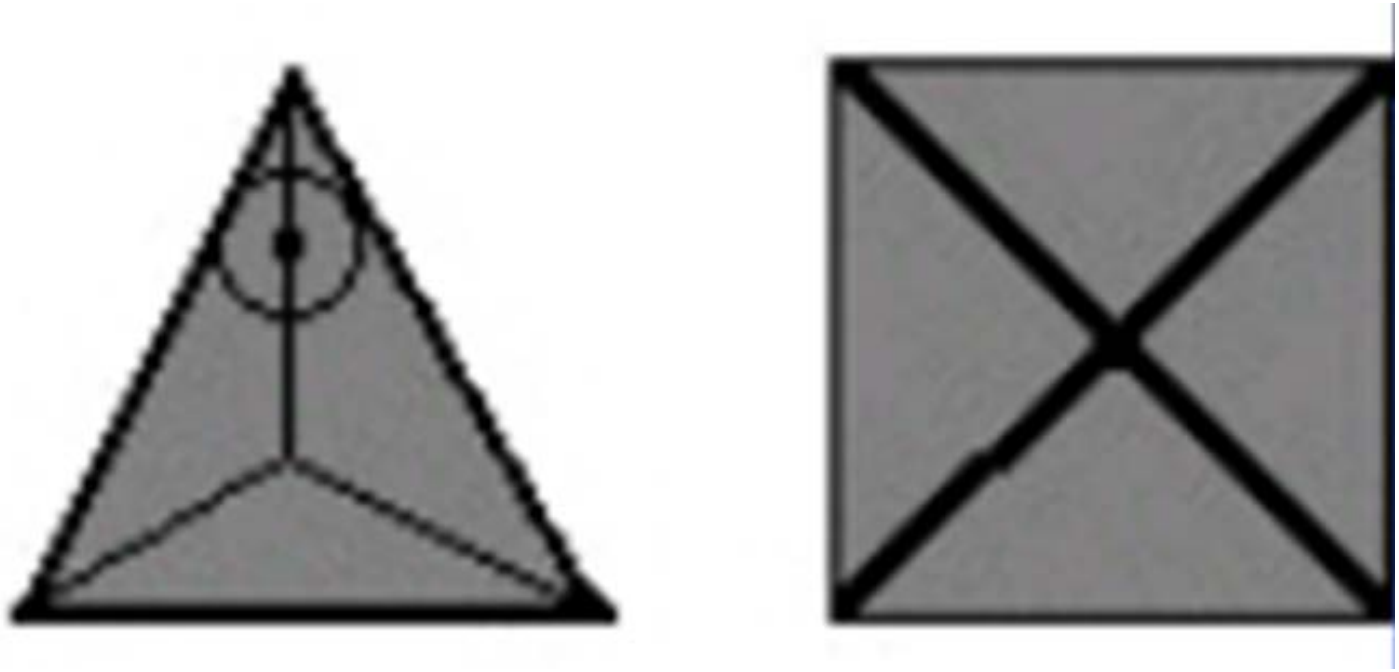


图 基于最大圆盘的骨架定义

骨架抽取

- 定义骨架子集$S_k(X)$为图像X内所有最大圆盘kB的圆心x构成的集合，对于$k=0,1,2,\ldots$
- 从骨架的定义可知，骨架是所有骨架子集的并， 即

$$S(X)=\cup\{S_k(X)\,|k=0,1,2,\ldots\}$$

- 可以证明骨架子集为

$$S_k(X)=(X\ominus kB)-[(X-kB)\circ B]$$

式中，B为结构元素，$(X\ominus kB)$代表连续k次用B对X腐蚀

$$S(X)=\cup\{(X\ominus kB)-[(X\ominus kB)\circ B]|\,k=0,1,2,\ldots\}$$

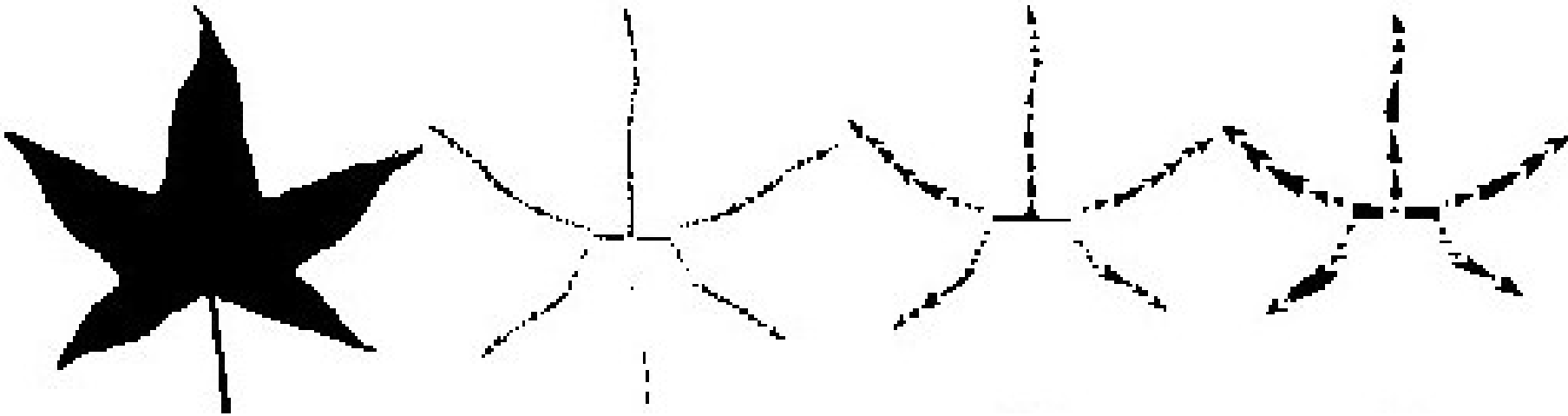


图 骨架抽取示例

(a) 一幅二值图像； (b) 用3×3的结构元素S得到的骨架；

(c) 用5×5的结构元素得到的骨架； (d) 用7×7的结构元素得到的骨架

细化算法举例

- 细化算法有很多，在这里介绍的是一种简单而且效果很好的算法，用它就能够实现从文本抽取骨架的功能。
- 判断一个点是否能去掉的判据：

  八个相邻点（八连通）的情况

  - (1) 内部点不能删除；
  - (2) 孤立点不能删除；
  - (3) 直线端点不能删除；
  - (4) 如果P是边界点，去掉P后，如果连通分量不增加，则P可以删除。

细化算法举例

- 假设一个像素点，定义该点为p1，则它的八邻域点p2->p9位置如下图所示，该算法考虑p1点邻域的实际情况，以便决定是否删除p1点。假设我们处理的为二值图像，背景为黑色，值为0，要细化的前景物体像素值为1。

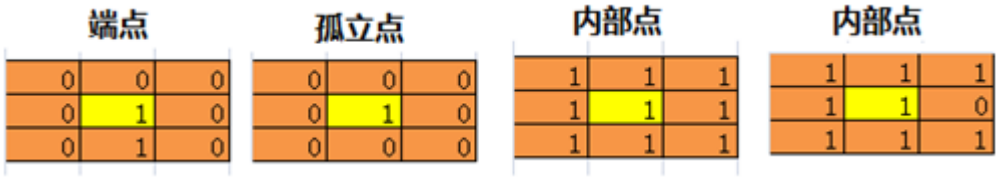
p9	p2	p3
p8	p1	p4
p7	p6	p5

细化算法举例

- 算法的描述如下
  - 1. 对图像进行一次扫描，对于不为0的点，如果满足以下四个条件，则在图像中删除该点(就是设置该像素为0)，这里p2，…，p9是对应位置的像素灰度值(其为1或者0)。
  - A. 2<= p2+p3+p4+p5+p6+p7+p8+p9<=6
  - B. p2->p9的排列顺序中，01模式的数量为1
  - C. p2*p4*p6 = 0
  - D. p4*p6*p8 = 0

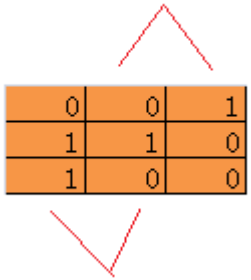
细化算法举例

- A. 2<= p2+p3+p4+p5+p6+p7+p8+p9<=6
  - 大于等于2会保证p1点不是端点或孤立点，因为删除端点和孤立点是不合理的，小于等于6保证p1点是一个边界点，而不是一个内部点。等于0时候，周围没有等于1的像素，所以p1为孤立点，等于1的时候，周围只有1个灰度等于1的像素，所以是端点（注：端点是周围有且只能有1个值为1的像素)。

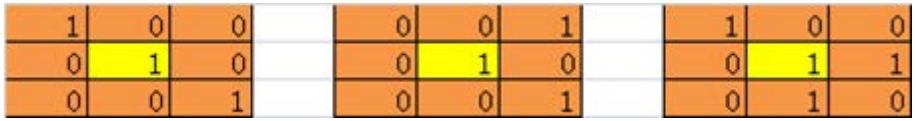


细化算法举例

- B. p2->p9的排列顺序中，01模式的数量为1。比如下面的图中，有p2p3 => 01, p6p7=>01,所以该像素01模式的数量为2。

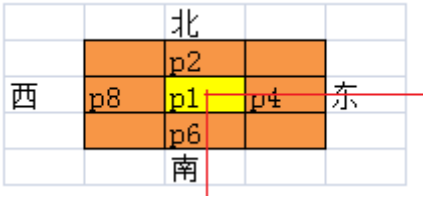


- 之所以要01模式数量为1，是要保证删除当前像素点后的连通性。比如下面的图中，01模式数量大于1，如果删除当前点p1，则连通性不能保证。



细化算法举例

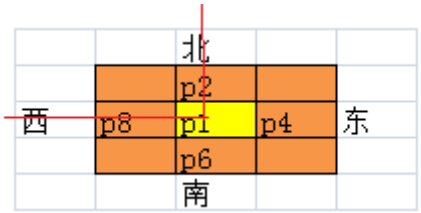
- C. P2*p4*p6 = 0
- D. p4*p6*p8 = 0



- 在第一次子迭代中，只是移去东南的边界点，而不考虑西北的边界点，注意p4, p6出现了2次，就是说它们有一个为0，则C，D就满足。

细化算法举例

- 2. 对图像再进行一次扫描，如果不为0的点的八邻域满足以下4个条件，则在图像中删除该点(就是设置该像素为0)
  - A. 2<= p2+p3+p4+p5+p6+p7+p8+p9<=6
  - B. p2->p9的排列顺序中，01模式的数量(这里假设二值图非零值为1)为1。
  - C. p2*p4*p8 = 0
  - D. p2*p6*p8 = 0



第二次会移去西北的边界点，注意p2, p8出现了2次，就是说它们有一个为0，则C，D就满足。

细化算法举例

- 执行完上面2个步骤后，就完成了一次细化算法。多次迭代执行上述过程，得到最终的图像骨架。