第 9 章　形态学图像处理

形体与特征，面容与肢体，我与兄弟如此相像，以至于人们经常把我当做他，总是把我们的身份搞混。

——享利·桑布鲁克·利，《科克伊尼之颂：双胞胎》

引言

形态学(morphology)一词通常表示生物学的一个分支，该分支主要研究动植物的形态和结构。这里，我们使用同一词语表示数学形态学的内容，将数学形态学作为工具从图像中提取表达和描绘区域形状的有用图像分量，如边界、骨架和凸壳等。我们对预处理或后处理的形态学技术也感兴趣，比如形态学过滤、细化和修剪等。

在下面几节中，我们将建立并说明数学形态学中的几个重要概念。这里介绍的许多概念可在 n 维欧氏空间 E^n 中用公式表达，然而，我们的兴趣一开始是在二值图像上，这种图像的各个分量是 Z^2 的元素(见 2.4.2 节)。在 9.6 节，我们的讨论将扩展到灰度图像。

本章的材料开始从纯粹的输入和输出都是图像的图像处理，转变为输入是图像而输出是从这些图像中提取属性的处理，在这个意义上 1.1 节做了定义。像形态学及与其相关概念这样的工具数学基础的基石，它用来于从图像中提取“内涵”。本书的余下章节中将探讨和应用其他方法。

9.1　预备知识

数学形态学的语言是集合论。同样，形态学为大量的图像处理问题提供一种一致且有力的方法。数学形态学中的集合表示图像中的对象。例如，在二值图像中，所有白色像素的集合是该图像的一个完整的形态学描述。在二值图像中，问题中的集合是二维整数空间 Z^2 的元素(见 2.4.2 节)，在该空间中，集合的每个元素都是一个多元组(二维向量)，这些多元组的坐标是图像中一个白色(或黑色，取决于事先的约定)像素的坐标(x, y)。前一章讨论过的灰度数字图像可以表示为其分量在空间 Z^3 中的集合。在在这种情况下，集合中每个元素的两个分量提供一个像素的坐标，第三个分量则对应于其离散灰度值。更高维度空间中的集合可以包含其他的图像属性，譬如颜色和随时间变化的分量。

在继续阅读之前，请读者回顾一下 2.4.2 节和 2.6.4 节的内容。

除 2.6.4 节中基本的集合定义外，集合的反射和平移的概念在形态学中用得也很广泛。一个集合 B 的反射表示为 $\hat{B}$，定义如下：

集合反射操作类似于空间卷积中执行的翻转(旋转)操作(见 3.4.2 节)。

$$\hat{B}=\{w \mid w=-b,\ b\in B\} \tag{9.1-1}$$

如果 B 是描述图像中物体的像素的集合(二维点)，则 $\hat{B}$ 是 B 中(x, y)坐标被$(-x,-y)$替代的点的集合。图 9.1(a)和(b)显示了一个简单的集合及其反射[①]。

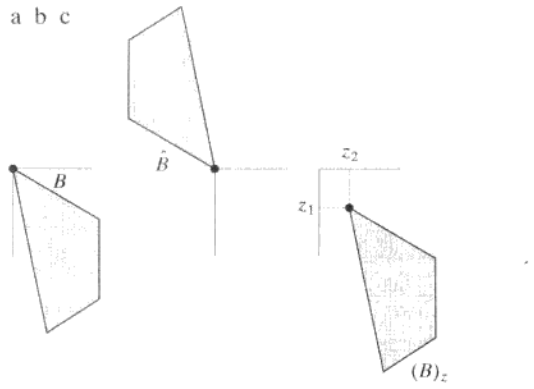


图 9.1　(a)一个集合；(b)该集合的反射；(c)距离为 z 的平移

集合 B 按照点 $z=(z_1, z_2)$ 表示为 $(B)_z$ 的平移定义如下：

$$(B)_z=\{c \mid c=b+z,\ b\in B\} \tag{9.1-2}$$

如果 B 是描述图像中物体的像素集合，则 $(B)_z$ 是 B 中(x, y)坐标被 $(x+z_1, y+z_2)$ 替代的点的集合。图 9.1(c)使用来自图 9.1(a)的集合 B 说明了这一概念。

在形态学中集合的反射和平移广泛用来表达基于结构元(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。图 9.2 的第一行显示了结构元的几个例子，其中每一个涂阴影的方块表示 SE 的一个成员。如果给定结构元中的一个位置是否是该 SE 集合的成员没有关系时，该位置用“×”来标记，表示一个“不关心”条件，如稍后在 9.5.4 节中定义的那样。除了元素是 SE 的成员的定义之外，还必须指定结构元的原点。图 9.2 中各种 SE 的原点由一个黑点指出(尽管将 SE 的中心放在其重心处是很普遍的，但通常原点的选择是依赖于问题的)。当 SE 对称且未显示原点时，则假定原点位于对称中心处。

当对图像操作时，我们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(图 9.2 中所示的非阴影部分)形成一个矩形阵列来实现的。图 9.2 第二行中的第一个和最后一个 SE 说明了该过程。该行中的其他 SE 已经是矩形形式。

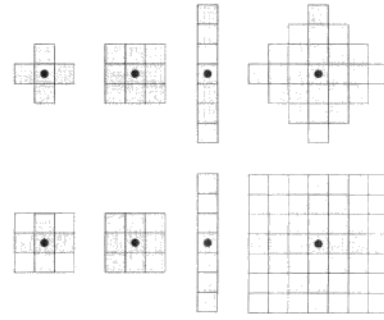


图 9.2　第一行：结构元的例子；第二行：转换为矩形阵列的结构元。点表示结构元的中心

为介绍在形态学中如何使用结构元，我们考虑图 9.3。图 9.3(a)和(b)显示了一个简单的集合和一个结构元。如前一段提到的那样，计算机实现要求用添加背景元素的方法把 A 也转换为一个矩形阵列。当结构元的原点位于原始集合的边界上时，背景边界要大到足以适应整个结构元(这类似于如

① 当对图像操作时，如图9.1 中的集合，我们使用阴影来表示正考虑集合的成员的点(像素)。在处理二值图像时，感兴趣的集合是对应于物体的像素。我们使用白色像素来显示这些集合，而其他像素则显示为黑色。术语前景和背景通常分别用于表示图像中为物体和非物体定义的像素集合。

650 ~ 651

3.4.2 节讨论的空间相关和卷积的填充操作)。在这种情况下，结构元的大小为 3×3，原点位于中心，所以包围整个集合的一个元素的边界是足够的，如图9.3(c)所示。就像在图9.2 中那样，必须使用最小可能数量的背景元素填充结构元，使它成为一个矩形阵列［见图 9.3(d)］。

在后面的说明中，我们会添加足够的背景点来形成矩形阵列，但在含义明确的情形下，为简化图形，我们假定已进行了填充操作。

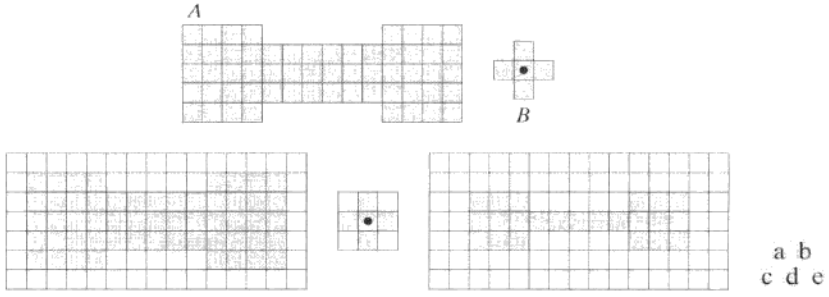


图 9.3 (a)一个集合(每个阴影方块都是集合的一个元素)；(b)结构元；(c)使用背景元素填充集合形成的一个矩形阵列，并且提供了一个背景边界；(d)矩形阵列形式的结构元；(e)由结构元处理过的集合

假定我们定义一个用结构元 B 在集合 A 上的操作如下：通过让 B 在 A 上运行，以便 B 的原点访问 A 的每一个元素，来创建一个新集合。在 B 的每个原点位置，如果 B 完全被 A 包含，则将该位置标记为新集合的一个成员(阴影所示)；否则，将该位置标记为非新集合的成员(非阴影所示)。图9.3(e)显示了这一操作的结果。我们看到，当 B 的原点位于 A 的边界元素上时，B 的一部分将不再包含在 A 中，从而排除了 B 处在中心位置的点作为新集合的成员的可能。最终结果是集合的边界被腐蚀掉了，如图 9.3(e)所示。当我们使用“结构元包含在集合中”这样的术语时，我们明确地指出 A 和 B 的元素完全重叠。换句话说，虽然我们用包含阴影和非阴影的元素阵列来说明 A 和 B，但在确定 B 是否包含于 A 中时，我们只考虑两个集合中的阴影元素。这些概念是下一节内容的基础，因此在继续学习之前全面理解图 9.3 中的概念是很重要的。

9.2 腐蚀和膨胀

我们通过研究腐蚀和膨胀这两个操作来开始形态学的讨论。这些操作是形态学处理的基础。事实上，本章中讨论的许多形态学算法都是以这两种原始操作为基础的。

9.2.1 腐蚀

作为 Z^2 中的集合 A 和 B，表示为 $A\ominus B$ 的 B 对 A 的腐蚀义为

$$A\ominus B=\{z\mid(B)_z\subseteq A\} \qquad (9.2\text{-}1)$$

表面上，该式指出 B 对 A 的腐蚀是一个用 z 平移的 B 包含在 A 中的所有的点 z 的集合。在下面的讨论中，假定集合 B 是一个结构元。式(9.2-1)是上节末尾讨论的图 9.3(e)中的例子的数学公式。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，故我们可以将腐蚀表达为如下的等价形式：

$$A\ominus B=\{z\mid(B)_z\cap A^c=\varnothing\} \qquad (9.2\text{-}2)$$

652 ~ 653

其中，如 2.6.4 节所定义的那样，A^c 是 A 的补集，$\varnothing$ 是空集。

图 9.4 显示了腐蚀的一个例子。A 和 B 的元素显示为阴影，背景显示为白色。图 9.4(c)中的实线边界是 B 的原点进一步移动的界限，若超出该界限，则会导致结构元不再完全包含于 A 中。这样，该边界内的点的轨迹(B 的原点位置)就构成了 B 对 A 的腐蚀。我们在图 9.4(c)中使用阴影显示了该腐蚀的结果。记住，腐蚀是简单地满足式(9.2-1)或式(9.2-2)的 z 值的一个集合。图 9.4(c)和(e)中虚线所示的集合 A 的边界仅供参考，它不是腐蚀操作的一部分。图 9.4(d)显示了一个拉长的结构元，图 9.4(e)显示了该结构元对集合 A 的腐蚀。注意，原来的集合已被腐蚀为一条直线。

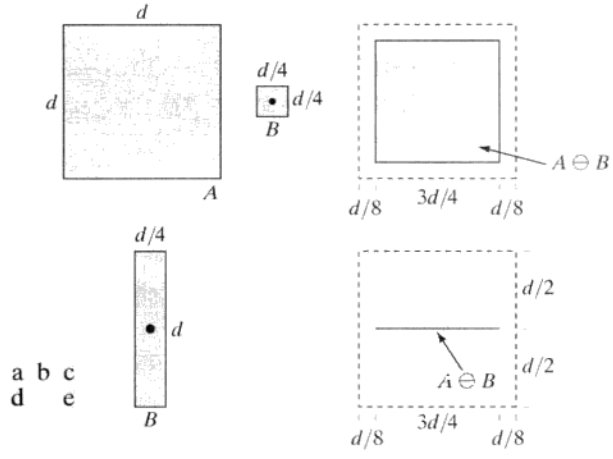


图 9.4　(a)集合 A；(b)方形结构元 B；(c)B 对 A 的腐蚀，如阴影部分所示；(d)拉长的结构元；(e)用拉长后的结构元 B 对 A 的腐蚀。(c)和(e)中的虚线边界是集合 A 的边界，仅供参考而显示

式(9.2-1)和式(9.2-2)不是腐蚀的唯一定义形式(见习题 9.9 和习题 9.10 中另外两种等价的定义形式)。然而，这些式子较之其他公式具有独特的优点，当把结构元 B 看成是一个空间模板时(见 3.4.1 节)，它们更直观。

例 9.1　使用腐蚀去除图像的某些部分。

假设我们希望去掉图 9.5(a)中连接中心区域到边界焊接点的线。使用一个大小为 11×11 且元素都是 1 的方形结构元腐蚀该图像，如图 9.5(b)所示，大多数为 1 的线条都被去除了。位于中心的两条垂直线被细化了，但没有被完全去除，原因是它们的宽度大于 11 个像素。把 SE 的大小改为 15×15，并再次腐蚀原图像，如图 9.5(c)所示，所有的连线都去除了[一种替代的方法是，也可以使用 11×11 的 SE 对图 9.5(b)的图像再进行腐蚀]。增大结构元的尺寸甚至会消除更大的部件。例如，使用大小为 45×45 的结构元，可去除边界的焊接点，如图 9.5(d)所示。

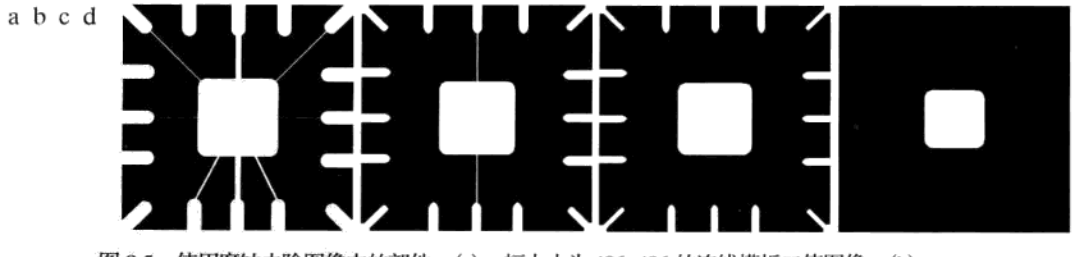


图 9.5　使用腐蚀去除图像中的部件：(a)一幅大小为 486×486 的连线模板二值图像；(b)~(d)分别使用大小为 11×11, 15×15 和 45×45 的结构元腐蚀的图像。SE 的元素都是 1

653 ~ 654

我们从这个例子看到，腐蚀缩小或细化了二值图像中的物体。事实上，我们可以将腐蚀看成是形态学滤波操作，这种操作将小于结构元的图像细节从图像中滤除(去除)了。在图 9.5 中，腐蚀执行了一个“线滤波”的功能。我们在 9.3 节和 9.6.3 节将会返回到形态学滤波这一概念。

9.2.2 膨胀

A 和 B 是 Z^2 中的集合，表示为 $A \oplus B$ 的 B 对 A 的膨胀定义为

$$A \oplus B = \left\{ z \mid (\hat{B})_z \cap A \neq \varnothing \right\} \tag{9.2-3}$$

这个公式是以 B 关于它的原点的映像，并且以 z 对映像进行平移为基础的(见图 9.1)。B 对 A 的膨胀是所有位移 z 的集合，这样，$\hat{B}$ 和 A 至少有一个元素是重叠的。根据这种解释，式(9.2-3)可以等价地写为

$$A \oplus B = \left\{ z \mid [(\hat{B})_z \cap A] \subseteq A \right\} \tag{9.2-4}$$

与以前一样，我们假定 B 是一个结构元，A 是被膨胀的集合(图像物体)。

式(9.2-3)和式(9.2-4)并不是当前所用的膨胀的唯一定义(见习题 9.11 和习题 9.12 中两种不同但等价的定义)。然而，当把结构元 B 视为一个卷积模板时，前述定义与其他定义形式相比有显著的优点，这种定义形式更直观。B 关于其原点翻转(旋转)，然后逐步移动以滑过整个集合(图像) A，这一基本过程类似于 3.4.2 节中介绍的空间卷积。然而，要记住，膨胀以集合操作为基础，因此，是一种非线性操作，而卷积是一种线性操作。

与腐蚀不同，腐蚀是一种收缩或细化操作，膨胀则会“增长”或“粗化”二值图像中的物体。这种特殊的方式和粗化的宽度由所用的结构元来控制。图 9.6(a)显示了与图 9.4 中所用集合相同的集合，图 9.6(b)显示了一个结构元(在这种情况下，$\hat{B} = B$，因为 SE 关于其原点对称)。作为参考，图 9.6(c)中的虚线显示了原始集合，实线显示了一个界限，$\hat{B}$ 的原点进一步移动 z，若超出了这一界限，会导致 $\hat{B}$ 和 A 的交集为空。因此，处在该边界上或该边界内的所有点就构成了 B 对 A 的膨胀。图 9.6(d)显示了一个设计用来实现垂直方向膨胀比水平方向膨胀更多的结构元，图 9.6(e)则显示了使用这个结构元进行膨胀的结果。

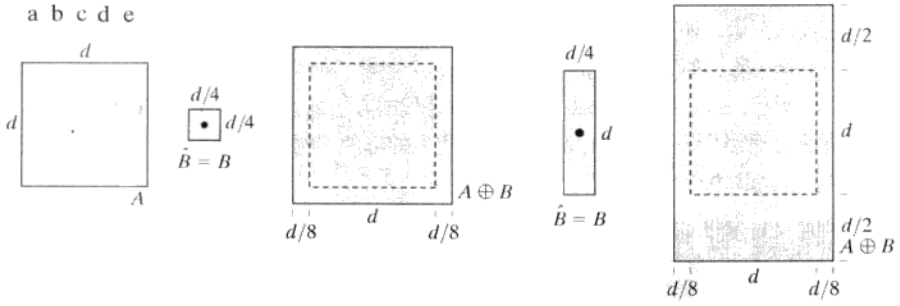


图 9.6 (a)集合 A；(b)方形结构元(黑点表示原点)；(c)B 对 A 的膨胀，显示为阴影；(d)拉长的结构元；(e)使用这个结构元对 A 膨胀。图(c)和图(e)中的点线的边是集合 A 的边界，仅作为参考而显示

例 9.2 膨胀的说明。

膨胀的最简应用之一是桥接裂缝。图9.7(a)显示了与我们研究过的图4.49相同的带有断裂的字符的图像，在图4.49中是用低通滤波进行连接的。已知断裂的最大长度为两个像素。图9.7(b)显示了能够修复这些断裂的结构元(注意，替代阴影，我们用 1 表示 SE 的元素，而用 0 表示背景；这是因为现在 SE 被当做

655 ~ 656

一幅子图像来处理，而不是一幅图形）。图 9.7(c) 显示了使用这个结构元对原图像进行膨胀后的结果。裂缝已被桥接。形态学方法较之我们在图 4.49 中用于桥接断裂的低通滤波方法的一个直接的优点，就是形态学方法可在一幅二值图像中直接得到结果。另一方面，低通滤波方法则从一幅二值图像开始，生成一幅灰度图像，它需要用一个阈值函数将灰度图像转换为二值图像。

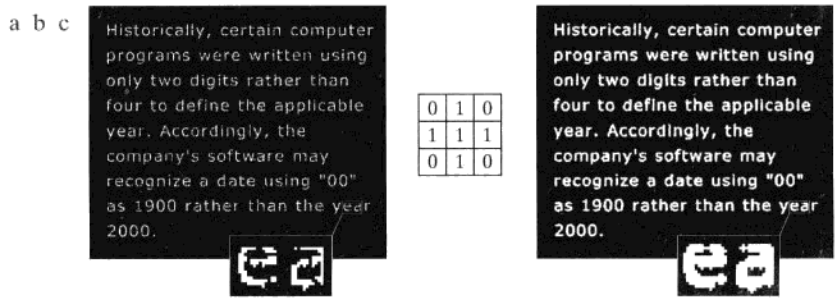


图 9.7　(a) 具有断裂字符的低分辨率样品文本（见放大的视图）；(b) 结构元；(c) 图 (b) 对 (a) 的膨胀。断裂线段被连起来了

9.2.3　对偶性

膨胀和腐蚀彼此关于集合求补运算和反射运算是对偶的，即

$$(A \ominus B)^c = A^c \oplus \hat{B} \tag{9.2-5}$$

和

$$(A \oplus B)^c = A^c \ominus \hat{B} \tag{9.2-6}$$

式 (9.2-5) 指出，B 对 A 的腐蚀是 $\hat{B}$ 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 $\hat{B} = B$，故对偶性特别有用。这样，我们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 A^c），并对该结果求补就可得到 B 对该幅图像的腐蚀。类似的说明适用于式 (9.2-6)。

为了说明建立形态学表达式有效性的一种典型方法，我们正式证明式 (9.2-5) 的有效性。由腐蚀的定义，我们有

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

如果集合 $(B)_z$ 包含在集合 A 中，则 $(B)_z \cap A^c = \varnothing$，在这种情况下，上式变为

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

但是，满足 $(B)_z \cap A^c = \varnothing$ 的 z 的集合的补集是满足 $(B)_z \cap A^c \neq \varnothing$ 的 z 的集合，因此，

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

其中，最后一步来自式 (9.2-3)。这就结束了该证明。采用类似的推理可以证明式 (9.2-6)（见习题 9.13）。

9.3　开操作与闭操作

如您所见到的那样，膨胀会扩大一幅图像的组成部分，而腐蚀则会缩小一幅图像中的组成部分。在本节中，我们讨论另外两个重要的形态学操作：开操作与闭操作。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，但与开操作相反，它通常会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。

656 ~ 657

结构元 B 对集合 A 的开操作，表示为 $A\circ B$，其定义如下：

$$A\circ B=(A\ominus B)\oplus B \tag{9.3-1}$$

因此，B 对 A 的开操作就是 B 对 A 的腐蚀，紧接着用 B 对结果进行膨胀。

类似地，用结构元 B 对集合 A 的闭操作，表示为 $A\bullet B$，定义如下：

$$A\bullet B=(A\oplus B)\ominus B \tag{9.3-2}$$

上式说明，B 对集合 A 的闭操作就是简单地用 B 对 A 膨胀，紧接着用 B 对结果进行腐蚀。

开操作有一个简单的几何解释(见图 9.8)。假设我们把结构元 B 视为一个(扁平的)“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 所能到达的 A 的边界的最远点。开操作的这种几何拟合特性导致了一个集合论公式，该公式表明 B 对 A 的开操作是通过拟合到 A 的 B 的所有平移的并集得到的。也就是说，开操作可以表示为一个拟合处理：

$$A\circ B=\bigcup\{(B)_z\mid(B)_z\subseteq A\} \tag{9.3-3}$$

其中 $\cup\{\cdot\}$ 表示大括号中所有集合的并集。

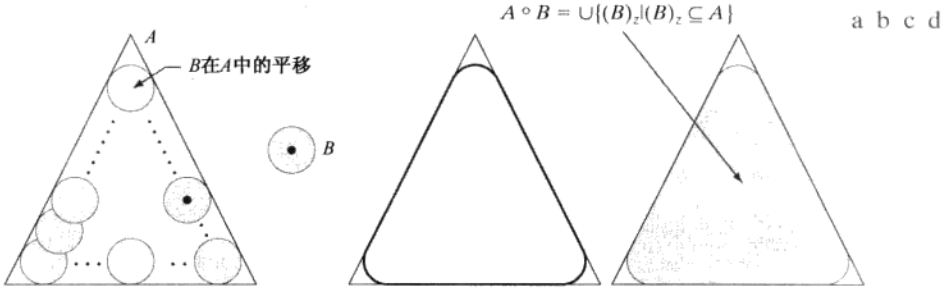


图 9.8 (a)结构元 B 沿集合 A 的内侧边界滚动(黑点表示 B 的原点)；(b)结构元；(c)粗线是开操作的外部边界；(d)完全的开操作(阴影部分)。为清楚起见，在图(a)中我们未对 A 加阴影

除了我们现在是在边界的外侧滚动 B(见图 9.9)之外，闭操作有类似的几何解释。如下面所讨论的那样，开操作和闭操作彼此对偶，所以闭操作在边界外侧滚动球体是意料之中的事情。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 $(B)_z\cap A\neq\varnothing$ 时，点 w 才是 $A\bullet B$ 的一个元素。图9.9 说明了闭操作这一基本的几何性质。

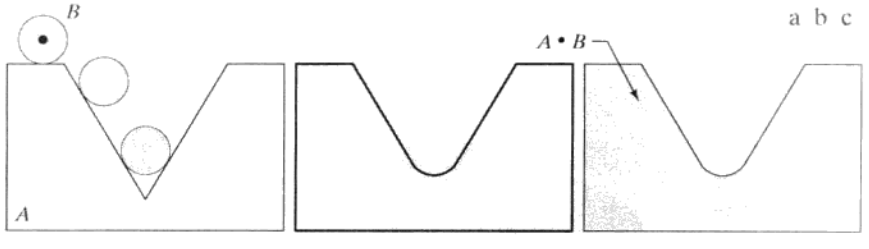


图 9.9 (a)结构元 B 沿集合 A 的外侧边界滚动；(b)粗线是闭操作的外部边界；(c)完全的闭操作(阴影部分)。为清楚起见，在图(a)中我们未对 A 加阴影

例 9.3 形态学开操作和闭操作的简单说明。

658 ~ 659

图 9.10 进一步说明了开操作和闭操作。图 9.10(a)显示了一个集合 A，图 9.10(b)显示了腐蚀处理期间一个圆盘形结构元的各个位置。当腐蚀完成后，得到图 9.10(c)所示的分离的图形。注意，两个主要部

分之间的桥接的消失。桥接部分的宽度与结构元的直径相比要细；也就是说，集合的这部分不能完全包含结构元，因此违反了式(9.2-1)的条件。该物体最右边的两个部分也是如此。圆盘无法拟合的突出部分已被消除。图 9.10(d)显示了对腐蚀后的集合进行膨胀的处理，图 9.10(e)显示了开操作的最终结果。注意，方向向外的角变圆了，而方向向内的角则未受影响。

类似地，图 9.10(f)到(i)显示了使用同一结构元对 A 进行闭操作的结果。我们注意到方向向内的角变圆了，而方向向外的角则保持不变。在 A 的边界上，最左边的突入部分的尺寸明显地减小了，因为在这个位置上圆盘无法拟合。还要注意使用圆盘形结构元对集合 A 进行开操作和闭操作所得到的物体的各个部分都平滑了。

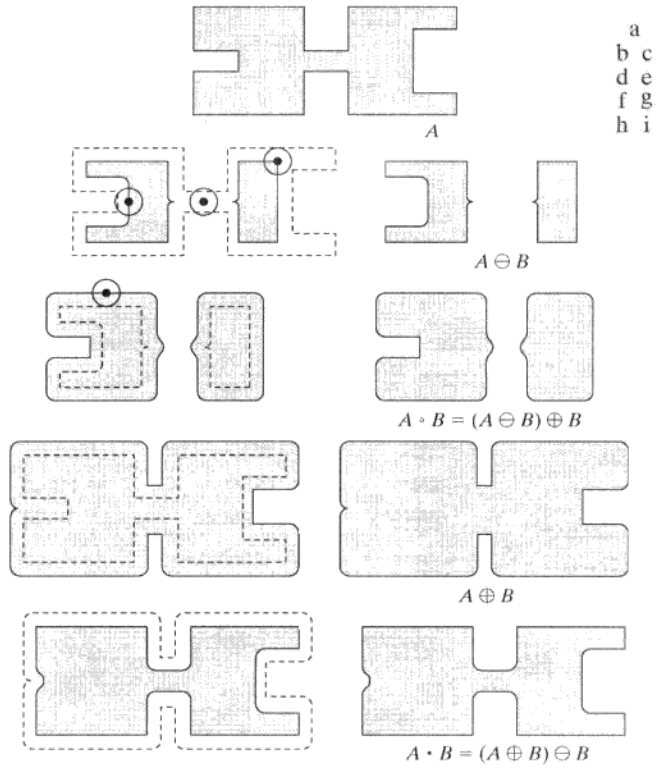


图 9.10　形态学开操作和闭操作。结构元是在图(b)中的各个位置显示的小圆。为清楚起见，SE 未加阴影。黑点是结构元的中心

如同膨胀和腐蚀的情形那样，开操作和闭操作彼此关于集合求补和反射也是对偶的，即

$$(A \bullet B)^c = (A^c \circ \hat{B}) \tag{9.3-4}$$

和

$$(A \circ B)^c = (A^c \bullet \hat{B}) \tag{9.3-5}$$

我们将该结果的证明作为练习留给读者(见习题 9.14)。

开操作满足下列性质：

(a) $A \circ B$ 是 A 的一个子集(子图像)。

(b) 如果 C 是 D 的一个子集，则 $C \circ B$ 是 $D \circ B$ 的一个子集。

(c) $(A \circ B) \circ B = A \circ B$。

659
~
660

类似地，闭操作满足下列性质：

(a) A 是 $A \bullet B$ 的一个子集(子图像)。

(b) 如果 C 是 D 的一个子集，则 $C \bullet B$ 是 $D \bullet B$ 的一个子集。

(c) $(A \bullet B) \bullet B = A \bullet B$ 。

注意，由两种情况下的条件(c)可知，算子应用一次后，一个集合的多次开操作或闭操作没有影响。

例 9.4 针对形态学滤波使用开操作和闭操作。

形态学操作可用于构造与第 3 章中讨论的空间滤波概念相类似的滤波器。图 9.11(a)中的二值图像显示了被噪声污染的指纹图像的一部分。这里，噪声本身表现为在黑色背景上的随机亮元素和指纹较亮部分上的暗元素。我们的目的是消除噪声及其对印刷的影响，同时使图像的失真尽可能小。由开操作后紧跟着闭操作组成的形态学滤波器可用来实现这一目的。

图 9.11 (a)噪声图像；(b)结构元；(c)腐蚀后的图像；(d)A 的开操作；(e)开操作的膨胀；(f)开操作的闭操作(原图由美国国家标准和技术研究所提供)

图 9.11(b)显示了所用的结构元。图 9.11 的余下部分显示了滤波操作一步步的顺序。图 9.11(c)显示了使用结构元对 A 进行腐蚀的结果。背景噪声在开操作的腐蚀阶段被完全消除，因为这种情况下的所有噪声分量都比结构元小。包含在指纹中的噪声元素(黑点)的尺寸实际上却增大了。原因是当物体被腐蚀时，这些元素在尺寸增大的内部边界。这种增大可通过对图 9.11(c)执行膨胀来抵消。图 9.11(d)显示了该结果，指纹中包含的噪声分量的尺寸被减小了，或完全被消除了。

刚才描述的两个操作构成了 B 对 A 的开操作。我们注意到，在图 9.11(d)中，开操作的实际效果实质上是从背景和指纹本身中消除所有的噪声。然而，在指纹纹路间却产生了新的断裂。为防止这种不希望的影响，我们在开操作上执行膨胀，如图 9.11(e)所示。大部分断裂被恢复了，但纹路却变粗了，这是可

660 ~ 661

由腐蚀来弥补的一种情形。图 9.11(f)所示的结果构成了图 9.11(d)的开操作的闭操作。最后的结果是噪声斑点清除得相当干净，但这种方法有缺点，即有些指纹纹路没有被完全修复，并还有间断。对这种情况也并非毫无指望，只是因为在保持连续性方面没插入任何的条件而已(我们将在例 9.8 中再次讨论这一问题，并在 11.1.7 节中论证处理这一问题的方法)。

9.4　击中或击不中变换

形态学击中或击不中变换是形状检测的一个基本工具。我们将借助于图 9.12 来说明这一概念。图 9.12 显示了一个由三种形状(子集)组成的集合 A，三种形状分别由 C, D 和 E 表示。图 9.12(a)到(c)中的阴影部分表示原始集合，而图 9.12(d)和(e)中的阴影部分显示形态学操作的结果。目的是找到三种形状之一的位置，譬如说 D 的位置。

令每种形状的原点位于其重心处。设 D 被一个小窗口 W 包围。关于 W 的 D 的局部背景定义为集合差$(W-D)$，如图 9.12(b)所示。图 9.12(c)显示了 A 的补集，后面将会用到它。图 9.12(d)显示了用 D 对 A 的腐蚀(虚线仅供参考)。回顾可知，D 对 A 的腐蚀是 D 的原点位置的集合，这样，D 完全包含在 A 中。换一种方法解释，$A \ominus D$ 在几何上可被看成是 D 的原点的所有位置的集合，在每个这样的位置，D 找到 A 中的一个匹配(击中)。记住，在图 9.12 中，A 仅由三个不相交的集合 C, D 和 E 组成。

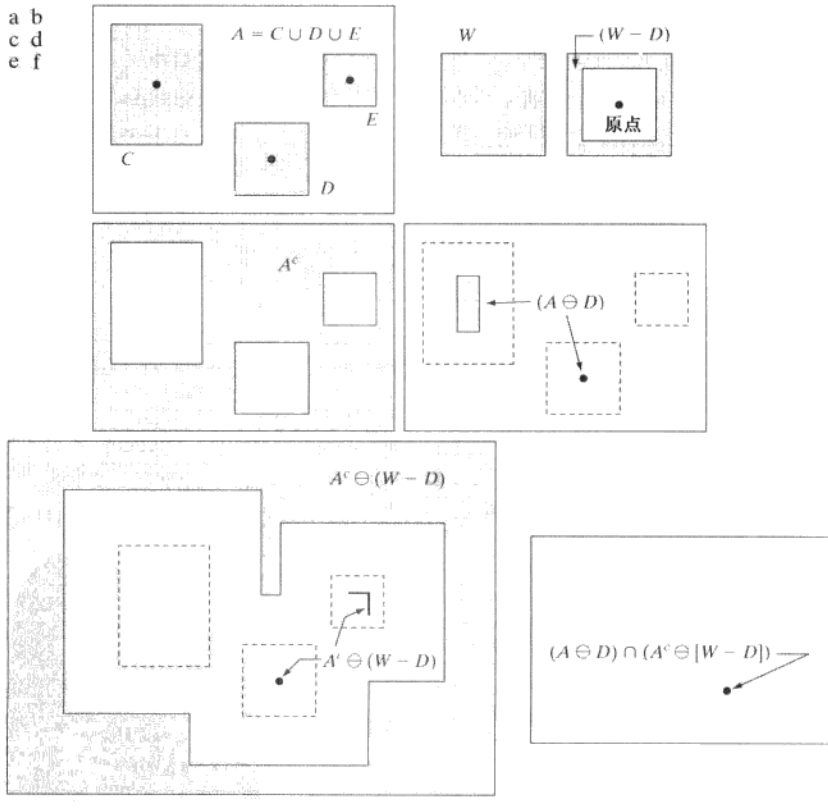


图 9.12　(a)集合 A；(b)窗口 W 和关于 W 的 D 的局部背景$(W-D)$；(c)A 的补集；(d)D 对 A 的腐蚀；(e)$(W-D)$对 A^c 的腐蚀；(f)(d)和(e)的交集，如所希望的那样，该交集显示了 D 的原点的位置。黑点是 C, D 和 E 的原点

661
~
663

图9.12(e)显示了局部背景集合$(W-D)$对A的补集的腐蚀。图9.12(e)中的外部阴影区域是腐蚀的部分。我们从图9.12(d)和(e)注意到，D精确地拟合A内部的位置集合是由D对A的腐蚀和$(W-D)$对A^c的腐蚀的交集，如图9.12(f)所示。这个交集正好是我们要寻找的位置。换句话说，如果B表示由D及其背景组成的集合，则B在A中的匹配（或匹配集合），表示为$A\circledast B$，是

$$A\circledast B=(A\ominus D)\cap[A^c\ominus(W-D)] \tag{9.4-1}$$

我们可以通过令$B=(B_1,B_2)$对这种表示法稍微做些推广，其中B_1是由与一个目标相联系的B的元素构成的集合，B_2是由与相应背景相联系的B的元素构成的集合。根据前面的讨论，$B_1=D,B_2=(W-D)$。用这种表示方法，式(9.4-1)变为

$$A\circledast B=(A\ominus B_1)\cap(A^c\ominus B_2) \tag{9.4-2}$$

因此，集合$A\circledast B$包含了所有的原点，同时，在这些原点处，B_1在A中找到了一个匹配(击中)，B_2在A^c中也找到了一个匹配。使用式(2.6-19)中给出的集合差的定义，及式(9.2-5)中给出的腐蚀和膨胀间的对偶关系，我们可将式(9.4-2)写为

$$A\circledast B=(A\ominus B_1)\cap(A\oplus\hat{B}_2) \tag{9.4-3}$$

然而，式(9.4-2)更为直观。我们将上述三个公式称为*形态学击中或击不中变换*。

使用与物体有关的结构元B_1和与背景有关的结构元B_2的原因基于一个假设的定义——仅当两个或多个物体形成相脱离的(断开的)集合时，这些物体才是可分的。通过要求每个物体至少被一个像素宽的背景所围绕，才可保证这一假设的成立。在某些应用中，我们可能对检测某个集合内由1和0组成的某些模式感兴趣，在这种情况下不需要背景。在这种场合，击中或击不中就简化为简单的腐蚀。正如前面指出的那样，腐蚀依然是匹配的集合，但检测单个物体时不需要额外的背景匹配。这种简化的模式检测方案将用于下节讨论的某些算法中。

9.5 一些基本的形态学算法

以前面的讨论为基础，我们现在可以考虑形态学的一些实际用途。在处理二值图像时，形态学的主要应用之一是提取用于表示和描述形状的图像成分。特别是我们要考虑提取边界、连通分量、凸壳和区域的骨架的形态学算法。我们还要探讨几种经常与这些算法相配合使用的预处理或后处理方法(对于区域填充、细化、粗化和修剪)。在介绍每一种形态学处理时，为了弄清楚每种处理的机理，我们将在这一小节中广泛地使用“迷你图像”。这些图像以图形的方式显示，用1表示阴影区域，而用0表示白色。

9.5.1 边界提取

表示为$\beta(A)$的集合A的边界可以通过先用B对A腐蚀，而后执行A和腐蚀的结果之间的集合之差得到，即

$$\beta(A)=A-(A\ominus B) \tag{9.5-1}$$

其中B是一个适当的结构元。

图9.13说明了边界提取的机理。这幅图像显示了一个简单的二值物体、一个结构元B和使用式(9.5-1)得到的结果。尽管图9.13(b)中的结构元是最常用的，但它绝对不是唯一的。例如，使用由1组成的大小为5×5的结构元，将得到2~3个像素宽的边界。

663 ~ 664

从现在开始，我们将不再显式地给出边界填充。

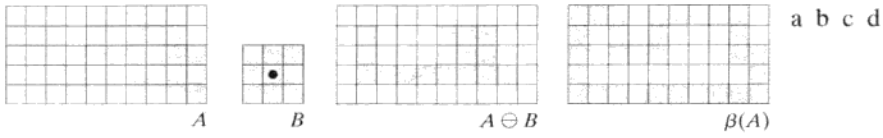


图 9.13　(a)集合 A；(b)结构元 B；(c)B 对 A 的腐蚀；(d)由 A 和其腐蚀间的集合差给出的边界

例 9.5　用形态学处理提取边界。

图 9.14 进一步说明了式(9.5-1)和由 1 组成的 3×3 结构元的用途。如本章中的所有二值图像那样，二进制值 1 显示为白色，二进制值 0 显示为黑色，因此该结构元的元素 1 也被当做白色来处理。由于所用结构元的尺寸，图 9.14(b)中的边界宽度为 1 个像素。

图 9.14　(a)一幅简单的二值图像，其中 1 表示白色；(b)使用式(9.5-1)和图 9.13(b)中的结构元得到的结果

9.5.2　孔洞填充

一个孔洞可被定义为由前景像素相连接的边界所包围的一个背景区域。在这一节中，我们将针对填充图像中的孔洞，开发一种基于集合膨胀、求补和交集的算法。令 A 表示一个集合，其元素是 8 连通的边界，每个边界包围一个背景区域(即一个孔洞)。当给定每个孔洞中的一个点后，目的就是用 1 填充所有的孔洞。

除了在每一个孔洞中对应于 X_0 中的位置给定的点之外，这一点我们已经置为 1 了，我们从形成一个由 0 组成的阵列 X_0 开始(该阵列与包含 A 的阵列的大小相同)。然后，如下过程将用 1 填充所有的孔洞：

$$X_k = (X_{k-1} \oplus B) \cap A^c \quad k = 1, 2, 3, \cdots \tag{9.5-2}$$

其中 B 是图 9.15(c)中的对称结构元。如果 $X_k = X_{k-1}$，则算法在迭代的第 k 步结束。然后，集合 X_k 包含所有被填充的孔洞。X_k 和 A 的并集包含所有填充的孔洞及这些孔洞的边界。

如果左边不加限制，那么式(9.5-2)中的膨胀将填充整个区域。然而，每一步中与 A^c 的交集操作将把结果限制到感兴趣区域内。这是我们如何制约形态学处理以满足希望特性的第一个例子。在当前应用中，它被适当地称为条件膨胀。图 9.15 的余下部分进一步说明了式(9.5-2)的机理。尽管该例子仅有一个孔洞，但在给定每个孔洞区域内的一个点的假设下，很清楚，这一概念可用于任何有限数量的孔洞。

例 9.6　形态学孔洞填充。

图 9.16(a)显示了一幅由内部带有黑色点的白色圆圈组成的图像。这样的图像可以通过将包含磨光的球体(如滚珠)的场景用阈值处理分为两个层次而得到。球体内部的黑点可能是反射的结果。我们的目的是通过孔洞填充来消除这些反射。图 9.16(a)显示了在一个球体中选择的一个点，图 9.16(b)显示了填充一

664 ~ 666

部分的结果。最后，图 9.16(c)显示了填充所有球体后的结果。因为黑点是背景点还是球体内部的点必须是已知的，所以完全自动化这一过程要求在算法中建立附加的“智能”。在 9.5.9 节中，我们将给出一个基于形态学重建的全自动方法(见习题 9.23)。

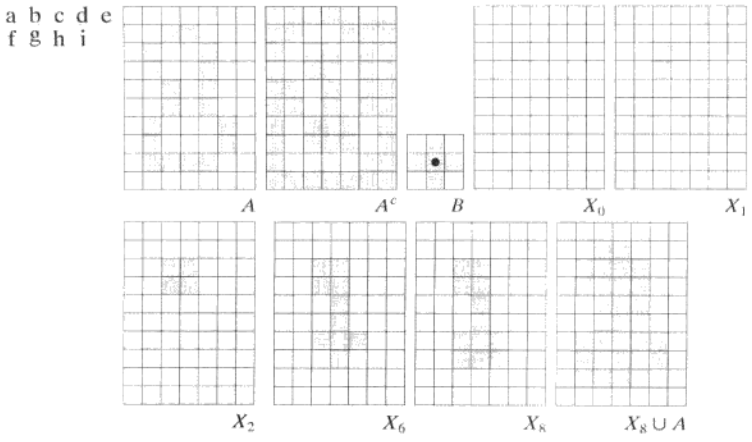


图 9.15 孔洞填充：(a)集合 A(显示为阴影)；(b)A 的补集；(c)结构元 B；(d)边界内的初始点；(e)~(h)式(9.5-2)的各个步骤；(i)最终结果［(a)和(h)的并集］

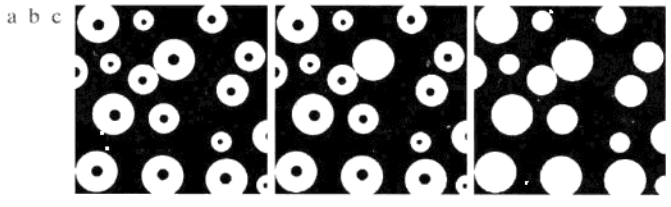


图 9.16 (a)二值图像(区域内部的白点是孔洞填充算法的起始点)；(b)填充该区域后的结果；(c)填充所有孔洞后的结果

9.5.3 连通分量的提取

连通性和连通分量的概念已在 2.5.2 节中介绍过。从二值图像中提取连通分量是许多自动图像分析应用中的核心。令 A 是包含一个或多个连通分量的集合，并形成一个阵列 X_0(该阵列的大小与包含 A 的阵列的大小相同)，除了在对应于 A 中每个连通分量中的一个点的已知的每一个位置处我们已置为 1(前景值)外，该阵列的所有其他元素均为 0(背景值)。如下迭代过程可完成这一目的：

$$X_k = (X_{k-1} \oplus B) \cap A \quad k = 1, 2, 3, \cdots \tag{9.5-3}$$

其中，B 是一个适当的结构元(如图9.17所示)。当 $X_k = X_{k-1}$ 时，迭代过程结束，X_k 包含输入图像中的所有的连通分量。注意式(9.5-3)与式(9.5-2)的相似性，唯一的差别是用 A 代替了 A^c。这并不奇怪，因为此处我们正在寻找前景点，而在 9.5.2 节的目的是寻找背景点。

图9.17说明了式(9.5-3)的机理，$k = 6$ 时即可收敛。注意，所用结构元的形状在像素间是基于 8 连通的。如果我们用图 9.15 中的 SE，它是基于 4 连通的，朝向图像底部的连通分量的最左侧元素将不会被检测到，因为对图的其余部分它是 8 连通的。如孔洞填充算法那样，假定在每一个连通分量内都已知一个点，式(9.5-3)对于任何在 A 中的有限数量的连通分量都是可用的。

关于事先不要求知道每个连通分量中的一个点的算法，请参阅习题 9.24。

667
~
668

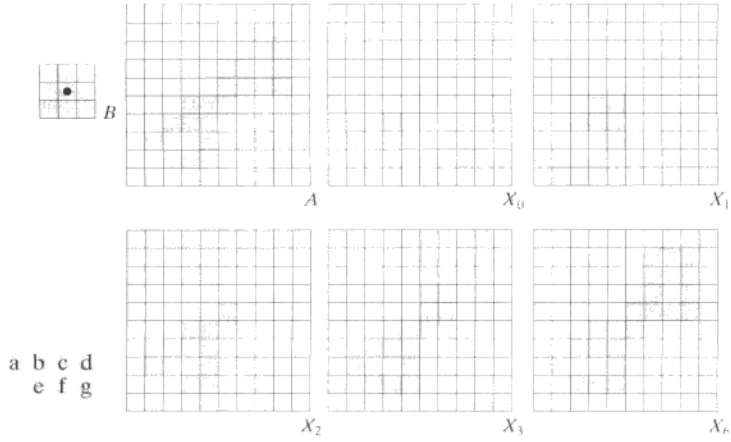


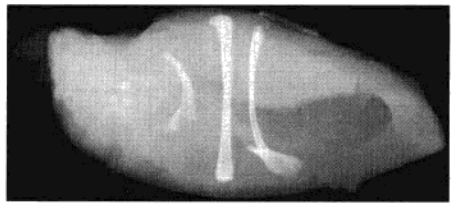
图 9.17　提取连通分量：(a)结构元；(b)包含有一个连通分量的集合的阵列；(c)在连通分量的区域中包含一个 1 的初始阵列；(d)~(g)式(9.5-3)的各个迭代步骤

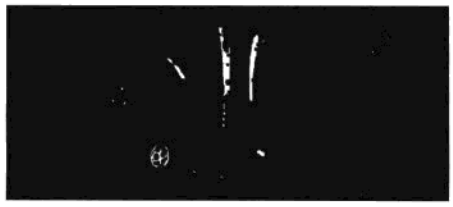
例 9.7　使用连通分量检测包装食品中的外来物。

连通分量经常用于自动检测。图 9.18(a)显示了一幅含有碎骨的鸡胸 X 射线图像。经过处理的食品在包装和/或运送之前能检测出这样的物体是很有意义的。在这种特殊情况下，骨骼的密度使得它们的正常灰度值与背景不同。这就使得使用单一阈值(阈值处理在 3.1 节介绍过了，10.3 节中将进行更详细的讨论)将骨骼从背景中提取出来成为一件简单的事情。结果是图 9.18(b)所示的二值图像。

这幅图中的最显著特征是保留下来的点聚集为物体(骨头)，而不是彼此孤立的毫无关系的点。我们可以确定，只有具有“有效”尺寸的物体才能对经阈值处理的图像进行腐蚀后而保留下来。在这个例子中，我们将“有效”尺寸的物体定义为：使用元素为 1、大小为 5×5 的结构元腐蚀图像后保留下来的任何物体。图 9.18(c)中显示了腐蚀的结果。下一步是分析保留下来的物体的尺寸。我们通过提取图像中的连通分量来标记(识别)这些物体。图 9.18(d)中的表格列出了提取的结果。总共有 15 个连通分量，其中 4 个尺寸较大。这足以确定包含在原图像中的重要的不希望的物体。如果需要，使用第 11 章讨论的技术进一步描述其特性(比如物体的形状)是可能的。

a b
c d





连通分量	连通分量中的像素数	连通分量	连通分量中的像素数
01	11	09	7
02	9	10	11
03	9	11	11
04	39	12	9
05	133	13	9
06	1	14	674
07	1	15	85
08	743		

图 9.18　(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(图像由 NTB Elektronische Geraete GmbH, Diepholz, Germany, www.ntbxray.com 提供)

668
~
669

9.5.4 凸壳

如果在集合 A 内连接任意两个点的直线段都在 A 的内部，则称集合 A 是凸形的。任意集合 S 的凸壳 H 是包含于 S 的最小凸集。集合差 $H-S$ 称为 S 的凸缺。正如 11.1.6 节和 11.3.2 节中详细讨论的那样，凸壳和凸缺对于物体描绘是很有用的。这里，我们介绍一种获得集合 A 的凸壳 $C(A)$的简单形态学算法。

令 B^i，$i=1,2,3,4$ 表示图 9.19(a)中的 4 个结构元。这个过程可通过执行下式实现：

$$X_k^i = (X_{k-1} \circledast B^i) \cup A \quad i=1,\ 2,\ 3,\ 4 \text{ 和 } k=1,\ 2,\ 3,\ \cdots \tag{9.5-4}$$

其中 $X_0^i = A$。当该过程收敛时(即当 $X_k^i = X_{k-1}^i$ 时)，我们令 $D^i = X_k^i$。则 A 的凸壳为

$$C(A) = \bigcup_{i=1}^{4} D^i \tag{9.5-5}$$

换句话说，该方法由反复使用 B^1 对 A 做击中或击不中变换组成；当不再发生进一步变化时，我们执行与 A 的并集运算，结果称为 D^1。这一过程使用 B^2 重复(应用于 A)，直到不发生进一步的变化，如此往复。得到的 4 个 D 的并集组成了 A 的凸壳。注意，我们使用不需要背景匹配的击中或击不中变换的简化的实现，如 9.4 节末尾所讨论的那样。

图 9.19 说明了式(9.5-4)和式(9.5-5)给出的过程。图 9.19(a)显示了用于提取凸壳的结构元。每个结构元的原点均位于其中心处。“×”项表示“不考虑”的条件。这意味着，如果结构元模板下 A 的 3×3 区域在该位置匹配模板的模式，则说结构元在 A 中找到了一个匹配。对于一个特殊的模板，当 A 中的这个 3×3 区域的中心为 0 时，而在阴影模板元素下的 3 个像素为 1 时，才会出现模式匹配。不必顾及该 3×3 区域内其他像素的值。此外，关于图 9.19(a)中的符号 B^i 是由 B^{i-1} 顺时针旋转 90°得到的。

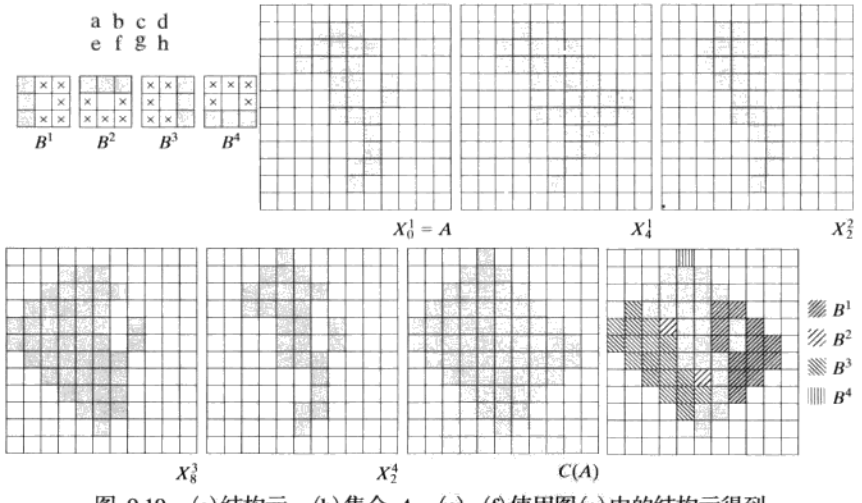


图 9.19 (a)结构元；(b)集合 A；(c)~(f)使用图(a)中的结构元得到的收敛结果；(g)凸壳；(h)显示了每个结构元的贡献的凸壳

图 9.19(b)显示了一个寻找凸壳的集合 A。从 $X_0^1 = A$ 开始，式(9.5-4)经过 4 次迭代后，得到图 9.19(c)中的集合。然后令 $X_0^2 = A$，并再次使用式(9.5-4)，得到图 9.19(d)中的集合(在这种情况下，仅经过两步即可收敛)。接下来的两个结果是以同样的方式得到的。最后，图9.19(c)，(d)，(e)和(f)中集合形成的并集得到图 9.19(g)显示的凸壳。每个结构元的贡献突出显示在图 9.19(h)所示的组合集合中。

669 ~ 670

上述过程的一个明显的缺点是凸壳可能超出确保凸性所需的最小尺寸。减少这种影响的一种简

单方法是限制生长，以便凸壳不会超过初始点集在水平和垂直方向上的尺寸。对图9.19中的例子施加这种限制产生图 9.20 所示的图像。更复杂的边界可用于进一步限制具有更多细节的图像的生长。例如，我们可以沿垂直、水平和对角线方向使用原始点集的最大尺寸。像这样的改进所付出的代价是额外增加了算法的复杂性和计算需求。

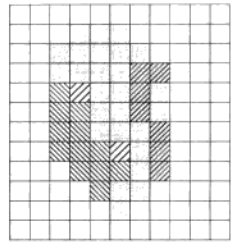


图 9.20　将凸壳生长算法限制到原始点集沿垂直方向和水平方向的最大尺寸的结果

9.5.5　细化

结构元 B 对集合 A 的细化可表示为 $A\otimes B$，它可以根据击中或击不中变换来定义：

$$A\otimes B = A-(A\circledast B) = A\cap(A\circledast B)^c \tag{9.5-6}$$

如前一节那样，我们仅对与结构元的模式匹配感兴趣，所以在击中或击不中变换中没有背景运算。针对对称地细化 A 的一种更有用的表达方式是以结构元序列为基础的：

$$\{B\} = \{B^1, B^2, B^3, \cdots, B^n\} \tag{9.5-7}$$

其中 B^i 是 B^{i-1} 旋转后的形式。使用这一概念，我们现在可以使用一个结构元序列将细化定义为

$$A\otimes\{B\} = ((\cdots((A\otimes B^1)\otimes B^2)\cdots)\otimes B^n) \tag{9.5-8}$$

这种处理是 A 被 B^1 细化一次，然后，得到的结果被 B^2 细化一次，如此进行下去，直到 A 被 B^n 细化一次。整个过程不断重复，直到得到的结果不再发生变化。每次单独的细化均使用式(9.5-6)来执行。

图 9.21(a)显示了一组通常用于细化的结构元，图 9.21(b)显示了将要使用刚才讨论的过程来细化的集合 A。图 9.21(c)显示了 A 被 B^1 细化一次后得到的结果。图 9.21(d)到(k)显示了用其他结构元细化多次的结果。使用 B^6 进行第二次细化后得到了收敛的结果。图 9.21(l)显示了细化的结果。最后，图 9.21(m)显示了转换为 m 连通的细化集合(见 2.5.2 节)，以消除多重路径。

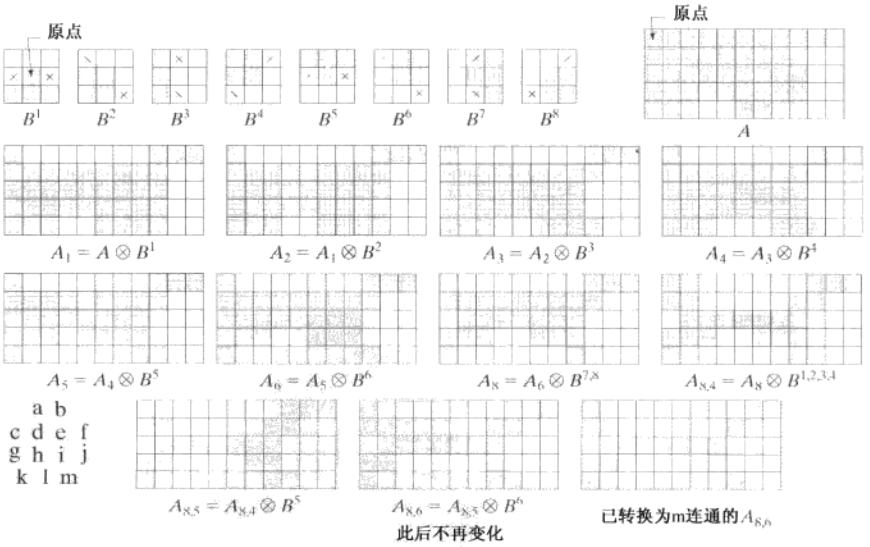


图 9.21　(a)用于细化的经旋转后的结构元序列；(b)集合 A；(c)使用第一个结构元细化得到的结果；(d)~(i)使用接下来的 7 个结构元细化得到的结果(使用第 7 个和第 8 个结构元细化得到的结果间没有变化)；(j)再次使用前 4 个结构元细化得到的结果；(l)收敛后的结果；(m)转换为 m 连通的结果

9.5.6 粗化

粗化是细化的形态学对偶，定义如下：

$$A\bullet B = A \cup (A \circledast B) \tag{9.5-9}$$

其中 B 是适合于粗化处理的结构元。与细化一样，粗化处理也可以定义为一个系列操作：

$$A \odot \{B\} = ((\cdots((A \odot B^1) \odot B^2)\cdots) \odot B^n) \tag{9.5-10}$$

用于粗化的结构元与图 9.21(a)所示的结构元具有相同的形式，但所有 1 和 0 要互换。然而，针对粗化的分离算法在实际中很少用到，取而代之的过程是先对问题中集合的背景进行细化，而后对结果求补集。换句话说，为粗化集合 A，我们先形成 $C = A^c$，而后细化 C，然后再求 C^c。图 9.22 说明了这一过程。

由于依赖于 A 的性质，这个过程可能产生某些断点，如图 9.22(d)所示。因此，通过这种方法的粗化处理通常会跟随一个后处理以消除断点。注意，根据图 9.22(c)可知，细化后的背景形成了粗化处理的一个边界。这一有用的特性在使用式(9.5-10)粗化的直接执行中并不存在，并且，它是使用背景细化来实现粗化的主要原因之一。

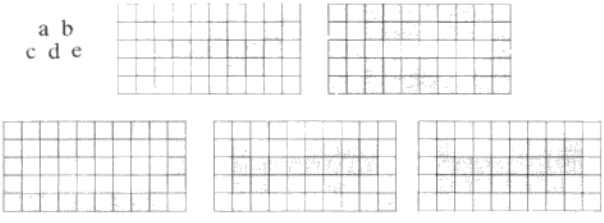


图 9.22 (a)集合 A；(b)A 的补集；(c)对 A 的补集进行细化得到的结果；(d)对(c)求补得到的粗化后的集合；(e)没有断点的最终结果

9.5.7 骨架

如图 9.23 所示，集合 A 的骨架 $S(A)$的概念直观上相当简单。由该图我们可以推出：

(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) 圆盘 $(D)_z$ 在两个或多个不同的位置与 A 的边界接触。

A 的骨架可以用腐蚀和开操作来表达，即骨架可以表示为(Serra[1982])

$$S(A) = \bigcup_{k=0}^{K} S_k(A) \tag{9.5-11}$$

其中，

$$S_k(A) = (A \ominus kB) - (A \ominus kB) \circ B \tag{9.5-12}$$

式中，B 是一个结构元，而 $(A \ominus kB)$ 表示对 A 的连续 k 次腐蚀：

$$(A \ominus kB) = ((\cdots(A \ominus B) \ominus B) \ominus \cdots) \ominus B) \tag{9.5-13}$$

672 ~ 674

K 是 A 被腐蚀为空集前的最后一次迭代步骤。换句话说，

$$K = \max\{k \mid (A \ominus kB) \neq \varnothing\} \tag{9.5-14}$$

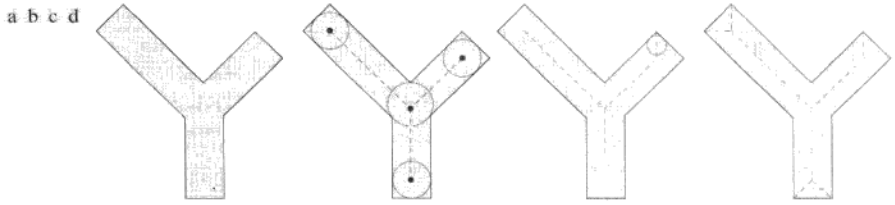


图 9.23　(a)集合 A；(b)中心在 A 的骨架上的最大圆盘的不同位置；(c)位于 A 的骨架的不同线段上的另一个最大圆盘；(d)完整的骨架

式(9.5-11)和式(9.5-12)给出的公式表明，$S(A)$可以作为骨架子集 $S_k(A)$ 的并集来得到。此外，还可证明使用下式可由这些子集来重建 A：

$$A=\bigcup_{k=0}^{K}(S_k(A)\oplus kB) \tag{9.5-15}$$

其中 $(S_k(A)\oplus kB)$ 表示对 $S_k(A)$ 的 k 次连续膨胀，即

$$(S_k(A)\oplus kB)=((\cdots((S_k(A)\oplus B)\oplus B)\oplus\cdots)\oplus B) \tag{9.5-16}$$

例 9.8　计算简单图形的骨架。

图 9.24 说明了刚才讨论的概念。第一列显示了原始集合(顶部)和使用结构元 B 的两次腐蚀。注意，对 A 再进行一次腐蚀将产生空集，因此，在这种情况下 $K=2$。第二列显示了使用 B 对第一列中的集合进行的开操作。结合图 9.8 中讨论的开操作的拟合特性，这些结果很容易解释。第三列只包含第一列和第二列间的集合差。

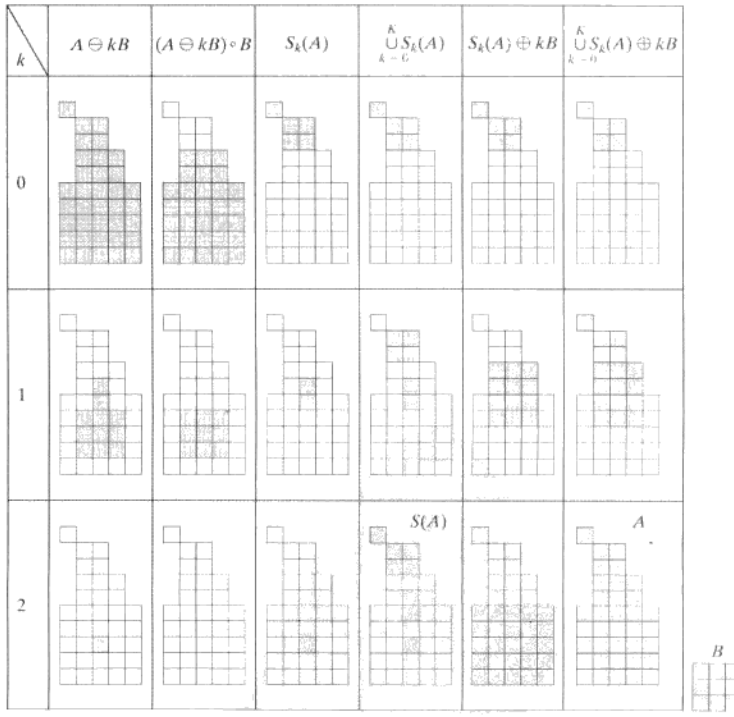


图 9.24　式(9.5-11)到式(9.5-15)的执行。原始集合位于左上角，其形态学骨架位于第四列的底部。第六列底部为重建的集合

674
~
675

第四列包含两部分骨架和最终结果(该列的底部)。最终得到的骨架不仅比需要的更粗，而且更重要的是它是不连通的。由于在前述的形态学骨架公式中没有任何条件保证连通性，所以该结果是意料之中的。形态学在给定集合的腐蚀和开操作中产生一个精致的公式。然而，如同通常情况下一样，骨架要求最大限度的细化、连通，且受到的腐蚀最小，就需要像 11.1.7 节中探讨的算法这一类有试探性的公式。

第五列显示了 $S_0(A), S_1(A) \oplus B$ 和 $(S_2(A) \oplus 2B) = (S_2(A) \oplus B) \oplus B$ 。最终，最后一列显示了根据式(9.5-15)重建的集合 A，该集合是第五列所示膨胀后的骨架子集的并集。

9.5.8 裁剪

裁剪方法本质上是对细化和骨架算法的补充，因为这些过程会保留某些寄生分量，因而需要用后处理来清除这些寄生分量。我们首先探讨裁剪问题，然后以前面小节中介绍的材料为基础探讨一种形态学上的解决方法。因此，我们抓住这个机会说明如何通过到现在为止讨论过的几种技术的联合使用解决这一问题。

在手写字符的自动识别中，通常使用的方法是分析每个字符的骨架形状。这些骨架经常带有许多“毛刺”(寄生分量)的特征。毛刺是在腐蚀过程中由组成字符的笔画的不均匀性造成的。为处理这一问题，我们将探讨一种形态学技术，先假定寄生成分的长度不超过指定的像素数。

图 9.25(a)显示了一个手写字符“a”的骨架。该字符最左侧的寄生成分就是我们要消除的部分。解决方案是以通过不断删除寄生分支的终点来抑制寄生分支为基础的。当然，这也会使得字符的其他分支变短(或删除)，但在缺乏其他结构信息的情况下，这个例子中的假设是任何具有三个或小于三个像素长度的分支都将被删除。使用一系列仅设计用来检测端点的结构元对输入集合 A 进行细化可以得到期望的结果。也就是说，令

我们可将端点定义为一个大小为 3×3 的区域的中心点，该区域满足图 9.25(b)或(c)中的任何排列。

$$X_1 = A \otimes \{B\} \tag{9.5-17}$$

式中，$\{B\}$表示图 9.25(b)和(c)中所示的结构元序列［见式(9.5-7)关于结构元序列］。这个结构元序列由两种不同的结构组成，每种结构对全部 8 个元素旋转了 90°。图 9.25(b)中的“×”表示一个“不必考虑”的条件，在这种意义上，该位置上的像素的值是 0 还是 1 无关紧要。在有关形态学的文献中报告的许多结果都是以运用类似图 9.25(b)中的单一结构元为基础的，只是沿着整个第一列有“不必考虑”的条件而已。这是不完善的。例如，该元素将位于图 9.25(a)中第 8 行、第 4 列的点识别为端点，因此将该点消除，从而破坏了笔画的连续性。

连续对 A 应用式(9.5-17)三次，得到图 9.25(d)中的集合 X_1。下一步是将字符“复原”成其原来的形状，但要去掉寄生分支。为做到这一点，首先需要求形成一个包含 X_1 中所有端点的集合 X_2［见图 9.25(e)］：

$$X_2 = \bigcup_{k=1}^{8} (X_1 \circledast B^k) \tag{9.5-18}$$

其中，B^k 是图 9.25(b)和(c)中所示的相同端点检测子。下一步是用集合 A 作为限定器，对端点进行三次膨胀：

$$X_3 = (X_2 \oplus H) \cap A \tag{9.5-19}$$

其中，H 是元素值为 1 的 3×3 结构元，并且在每一步之后都要与 A 求交集。如区域填充和提取连通分量的情况一样，这种有条件的膨胀可以防止在我们关注的区域外产生值为 1 的元素，正如图 9.25(f)所示结果证明的那样。最后，X_3 和 X_1 的并集就是我们想要的结果：

式(9.5-19)是膨胀形态学重建的基础，详见下一节中的解释。

675 ~ 677

$$X_4 = X_1 \cup X_3 \tag{9.5-20}$$

如图 9.25(g)所示。

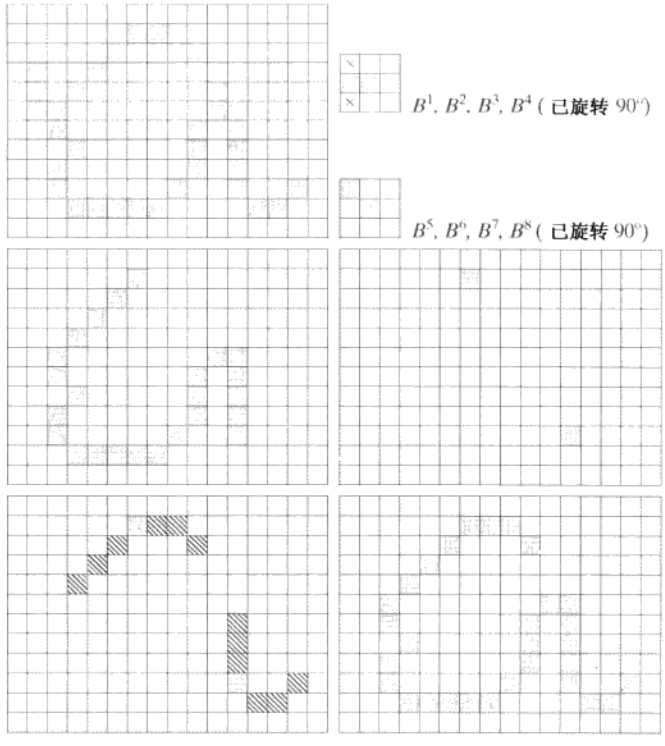


图 9.25　(a)原图像；(b)~(c)用于检测端点的结构元；(d)经过三轮细化后的结果；(e)图(d)的端点；(f)以图(a)为条件的端点膨胀；(g)修剪后的图像

在更复杂的情况下，使用式(9.5-19)有时会捡回某些寄生分支的“尖端”。当这些分支的端点离骨架很近时，这种情况就会发生。尽管使用式(9.5-17)可以消除它们，但在膨胀过程中会再次捡回这些点，因为它们是 A 中的有效点。除非整个寄生元素再次被捡回，(这些元素比有效笔画短的情况很少)检测并消除它们就很容易，因为它们都是不连续的区域。

此时的一种自然想法是，一定存在一种更为容易的方法来解决这个问题。例如，我们可以保留所有被删除点的轨迹，并将其中合适的点与那些应用式(9.5-17)后留下的端点进行再连接。这样的选择是可行的，但前面给出的公式的优点是使用简单的形态学结构即可解决整个问题。在实际情况下，当这一组工具可利用时，其优点是不必写新的算法。我们只需将必要的形态学函数结合到操作序列中。

9.5.9　形态学重建

迄今为止讨论的形态学概念只涉及一幅图像和一个结构元。在这一节中，我们讨论一种称为形态学重建的强有力的形态学变换，它涉及两幅图像和一个结构元。一幅图像是标记，它包含变换的起始点，另一幅图像是模板，它约束该变换。结构元用来定义连接性[①]。

① 在关于形态学重建的许多文献中，默认情形下结构元假设是各向同性的，且一般称为初级各向同性结构元。在本章的上下文中，这样的一个结构元的例子是其原点在中心处、元素为 1 且大小为 3×3 的阵列。

677 ~ 678

测地膨胀和腐蚀

形态学重建的核心是测地膨胀和测地腐蚀这两个概念。令 F 表示标记图像，G 表示模板图像。在本讨论中，假定两幅图像都是二值图像，且 $F \subseteq G$。令 $D_G^{(1)}(F)$ 表示大小为 1 的标记图像关于模板的测地膨胀定义为

$$D_G^{(1)}(F) = (F \oplus B) \cap G \tag{9.5-21}$$

其中，$\cap$ 表示集合的交［这里 $\cap$ 可解释为逻辑 AND(与)，因为对二值图像来说集合的交和逻辑 AND 操作是相同的］。F 关于 G 的大小为 n 的测地膨胀定义为

$$D_G^{(n)}(F) = D_G^{(1)}\left[D_G^{(n-1)}(F)\right] \tag{9.5-22}$$

其中，$D_G^{(0)}(F) = F$。在这个递推表达式中，式(9.5-21)中的集合求交在每一步中都执行[①]。注意，交集算子保证模板 G 将限制标记 F 的生长(膨胀)。图 9.26 显示了一个大小为 1 的测地膨胀的简单例子。图中的步骤是式(9.5-21)的直接执行。

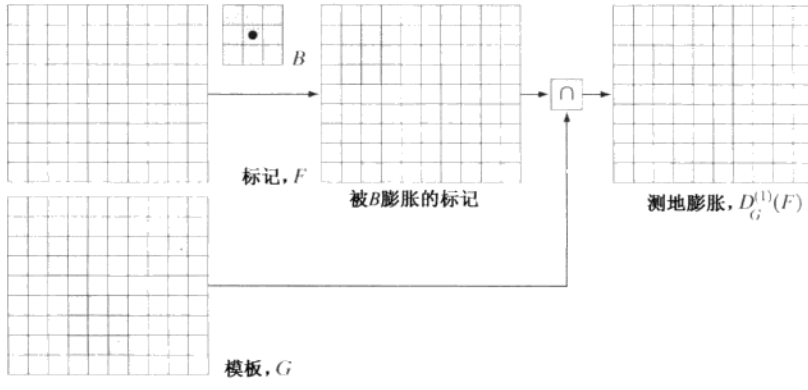


图 9.26 测地膨胀的说明

类似地，标记 F 关于模板 G 的大小为 1 的测地腐蚀定义为

$$E_G^{(1)}(F) = (F \ominus B) \cup G \tag{9.5-23}$$

其中，$\cup$ 表示集合的并(或者 OR 操作)。F 关于 G 的大小为 n 的测地腐蚀定义为

$$E_G^{(n)}(F) = E_G^{(1)}\left[E_G^{(n-1)}(F)\right] \tag{9.5-24}$$

其中，$E_G^{(0)}(F) = F$。式(9.5-23)中的并集操作在每一个迭代步骤中执行，并保证一幅图像的测地腐蚀仍然大于或等于其模板图像。如从式(9.5-21)和式(9.5-23)预期的那样，测地膨胀和测地腐蚀是关于集合的补集对偶的(见习题 9.29)。图 9.27 显示了一个大小为 1 的测地腐蚀的简单例子。图中的步骤是直接执行式(9.5-23)。

有限数量图像的测地膨胀和腐蚀经过有限数量的迭代步骤后总会收敛，因为标记图像的扩散或收缩受模板约束。

678
~
680

① 尽管使用递归公式来开发形态学重建方法更为直观(就像此处所做的那样)，但它们的实际实现通常基于计算上更有效率的算法(例如，见 Vincent[1993]和 Soille[2003])。本节中所有基于图像的例子均是使用这样的算法生成的。

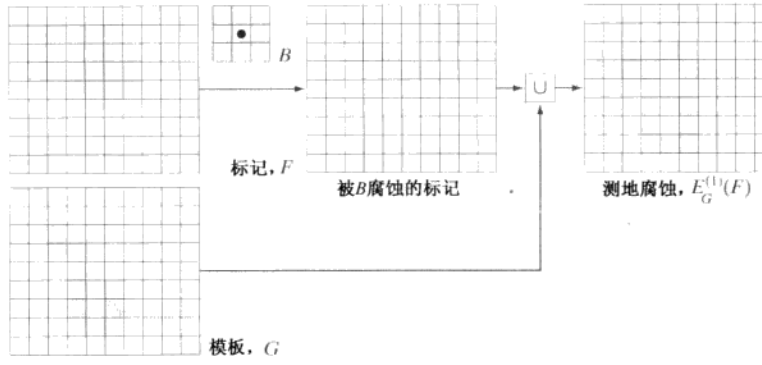


图 9.27　测地腐蚀的说明

用膨胀和腐蚀的形态学重建

基于前面的概念，来自标记图像 F 对模板图像 G 的膨胀形态学重建表示为 $R_G^D(F)$，它被定义为 F 关于 G 的测地膨胀，反复迭代直至达到稳定状态；即

$$R_G^D(F) = D_G^{(k)}(F) \tag{9.5-25}$$

迭代 k 次，直至 $D_G^{(k)}(F) = D_G^{(k+1)}(F)$。

图9.28 说明了使用膨胀的重建。图9.28(a)继续在图9.26中开始的处理；也就是说，得到 $D_G^{(1)}(F)$ 后，重建的下一步是膨胀该结果，然后，用模板 G 与其相“与”(AND)得到 $D_G^{(2)}(F)$，如图9.28(b)所示。$D_G^{(2)}(F)$ 的膨胀结果与模板 G 相“与”(AND)得到 $D_G^{(3)}(F)$，等等。重复这一过程，直至达到稳定。如果我们对该例子多执行一步，会发现 $D_G^{(5)}(F) = D_G^{(6)}(F)$，因此，采用膨胀的形态学重建的图像由 $D_G^D(F) = D_G^{(5)}(F)$ 给出，这正如式(9.5-25)指出的那样。注意，在这种情况下，重建的图像与模板相同，因为 F 包含了值为 1 的单个像素(这类似于一幅图像与一个冲激的卷积，该卷积简单地复制冲激位置处的图像，正如在 3.4.2 节解释的那样)。

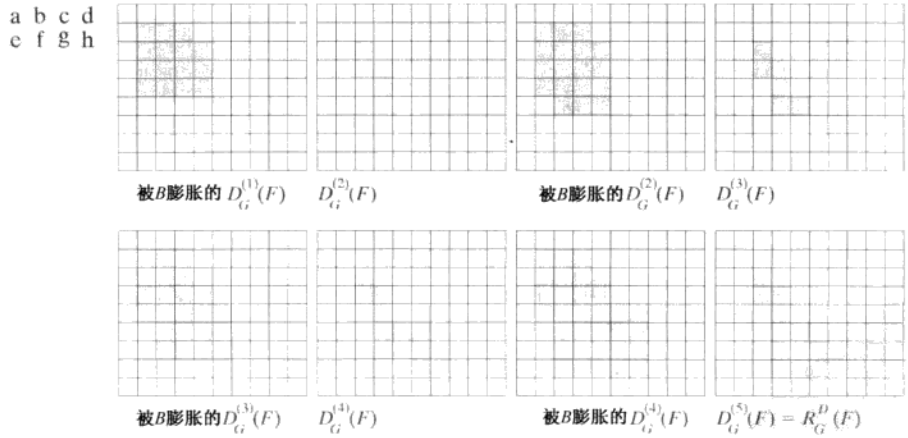


图 9.28　采用膨胀的形态学重建的说明。F, G, B 和 $D_G^{(1)}(F)$ 来自图 9.26

依照类似的方法，模板图像 G 对标记图像 F 的腐蚀的形态学重建表示为 $R_G^E(F)$，它被定义为 F 关于 G 的测地腐蚀，反复迭代直至达到稳定状态；即

$$R_G^E(F) = E_G^{(k)}(F) \tag{9.5-26}$$

迭代 k 次，直至 $E_G^{(k)}(F)=E_G^{(k+1)}(F)$。作为习题，请您针对腐蚀的形态学重建生成一幅类似于图 9.28 的图形。膨胀和腐蚀形态学重建是关于集合补集对偶的(见习题 9.30)。

应用实例

形态学重建有很宽的实际应用领域，每种应用都由标记图像和模板图像的选择、所用的结构元及前面讨论中定义的基本操作的组合来决定。下面的几个例子说明了这些概念的应用。

重建开操作：在形态学开操作中，腐蚀会删除小的物体，而后续的膨胀试图恢复遗留物体的形状。然而，这种恢复的准确性高度依赖于物体的形状和所用结构元的相似性。*重建开操作*可正确地恢复腐蚀后所保留物体的形状。一幅图像 F 的大小为 n 的重建开操作定义为来自 F 的大小为 n 的腐蚀的 F 的膨胀重建；即

$$O_R^{(n)}(F)=R_F^D[(F\ominus nB)] \tag{9.5-27}$$

其中，$(F\ominus nB)$ 表示 B 对 F 的 n 次腐蚀，如 9.5.7 节中解释的那样。注意，在该应用中，F 被用做模板。对于重建闭操作，可写出一个类似的表达式(见表 9.1)。

图 9.29 显示了重建开操作的一个例子。在这一说明中，我们的兴趣是从图 9.29(a)中提取长的、垂直笔画的字符。重建开操作至少要求一次腐蚀，因此我们首先执行这一步骤。图 9.29(b)显示了图 9.29(a)经腐蚀后的结果，该腐蚀采用长度与长字符的(51 个像素)平均高度成正比和宽度为 1 个像素的结构元。为了比较，我们用相同的结构元计算了开操作。图 9.29(c)显示了结果。最后，图 9.29(d)是用式(9.5-27)中给出的 F [即 $O_R^{(1)}(F)$] 的重建开操作(大小为 1)。该结果表明，包含长的垂直笔画的字符被准确地恢复了，而所有的其他字符则被去除了。

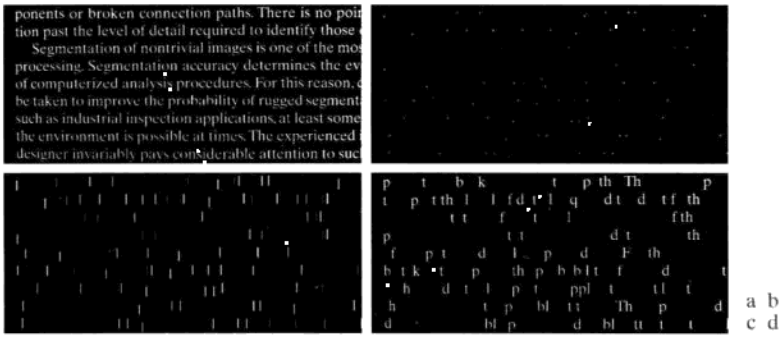


图 9.29 (a)大小为 918×2018 像素的文本图像。长的字符的近似平均高度是 51 个像素；(b)使用大小为 51×1 像素的结构元对图(a)的腐蚀；(c)使用相同的结构元对图(a)的开操作，用做参考；(d)重建开操作的结果

填充孔洞：在 9.5.2 节中，我们开发了一种填充孔洞的算法，该算法在图像中的每个孔洞都已知一个起始点。这里，我们开发一个基于形态学重建的全自动化过程。令 $I(x, y)$代表一幅二值图像，并假定我们形成了一幅标记图像 F，除了在该图像的边界位置为 $1-I$ 之外，在其他位置均为 0，即

$$F(x,y)=\begin{cases}1-I(x,y), & (x,y)\text{在}I\text{的边界上}\\ 0, & \text{其他}\end{cases} \tag{9.5-28}$$

则

$$H=\left[R_I^D(F)\right]^c \tag{9.5-29}$$

是一幅等于 I 且所有孔洞都被填充的二值图像。

680 ~ 682

让我们考虑式(9.5-29)中的各个分量，以了解该表达式事实上是如何填充图像中的所有孔洞的。图 9.30(a)显示了包含一个孔洞的简单图像 I，图 9.30(b)显示了它的补集。注意，因为 I 的补集将所有的前景像素(1 值)设置为背景(0 值)像素，并且反之亦然，该操作效果上建立了围绕孔洞的一面元素为 0 的“墙”。因为 I^c 被用做一个 AND(与)模板，这里我们所做的一切都是在该过程的迭代期间保护所有的前景像素(包括围绕孔洞的墙)。图 9.30(c)是根据式(9.5-28)形成的阵列 F，图 9.30(d)是使用所有元素都为 1 的 3×3 SE(结构元)膨胀后的 F。注意，标记 F 有一个元素为 1 的边缘(除 I 为 1 的位置之外)，故这些标记点 F 的膨胀在边缘处开始，并且向内处理。图 9.30(e)显示了使用 I^c 作为模板对 F 进行测地膨胀后的结果。如刚才指出的那样，我们看到，在这个结果中，对应于 I 的前景像素的所有位置都是 0，并且对于孔洞像素也是一样。另一次迭代将得到相同的结果，如式(9.5-29)要求的那样，求补后给出图 9.30(f)中所示的结果。操作 $H \cap I^c$ 生成一幅图像，该图像中对应于 I 中孔洞的位置像素值为 1，如图 9.30(g)所示。

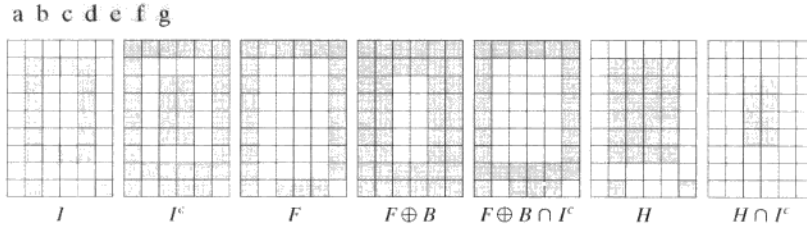


图 9.30　在简单图像上填充孔洞的说明

图 9.31 显示了一个更实际的例子。图 9.31(b)显示了图 9.31(a)中文本图像的补集，图 9.31(c)是由式(9.5-28)产生的标记图像 F。除了对应于原图像边界上的 1 的位置外，该图像有一个元素为 1 的边界。最后，图 9.31(d)显示了所有孔洞均被填充后的图像。

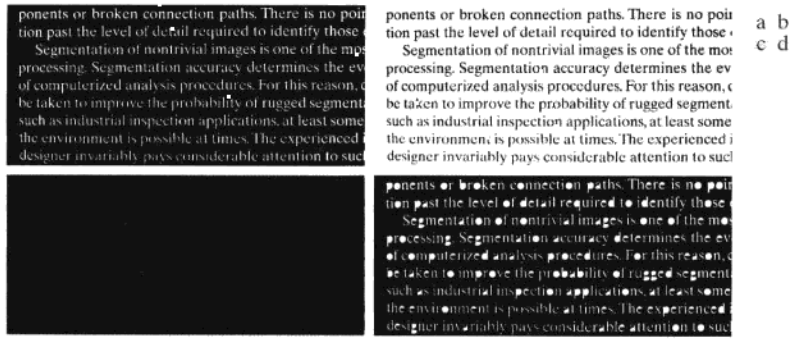


图 9.31　(a)大小为 918×2018 像素的文本图像；(b)作为模板图像使用的(a)的补集；(c)标记图像；(d)用式(9.5-29)填充孔洞的结果

边界清除：针对后续形状分析的图像物体提取是自动图像处理的基本任务。删除接触(即连接到)边界的物体的算法是一个很有用的工具，因为：(1)它可以用于屏蔽图像，以便对进一步处理只保留完整的物体，或者(2)它可用做在视野中存在部分物体的一个信号。作为本节介绍的概念的最后说明，我们现在提出一个基于形态学重建的边界清除过程。在这一应用中，我们用原图像作为模板，并使用下面的标记图像：

$$F(x, y) = \begin{cases} I(x, y), & (x, y) \text{位于 } I \text{ 的边界上} \\ 0, & \text{其他} \end{cases} \tag{9.5-30}$$

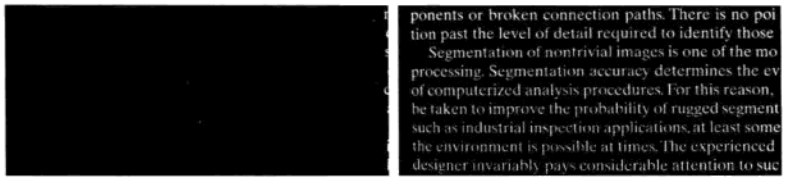
682 ~ 683

边界清除算法首先计算形态学重建 $R_I^D(F)$(简单地提取接触到边界的物体),然后计算差

$$X = I - R_I^D(F) \tag{9.5-31}$$

以得到一幅其中没有接触边界的物体的图像 X。

作为一个例子,再次考虑文本图像。图 9.32(a)显示了使用所有元素均为 1 的 3×3 的结构元得到的重建 $R_I^D(F)$(注意右侧接触边界的物体),图 9.32(b)显示了使用式(9.5-31)计算得到的图像 X。如果手边的任务是自动字符识别,则拥有一幅其中没有接触边界的字符的图像最有用,因为这样可以避免不得不识别部分字符的问题(充其量也只是一个困难的任务)。



a b

图 9.32 边界清除:(a)标记图像;(b)没有接触边界的物体的图像。原图像是图 9.29(a)

9.5.10 二值图像形态学操作小结

表 9.1 小结了前几节中提出的形态学结果。图 9.33 小结了迄今为止讨论过的用于各种形态学处理的结构元的基本类型。注意,表 9.1 第三列中的罗马数字是指图 9.33 中的结构元。

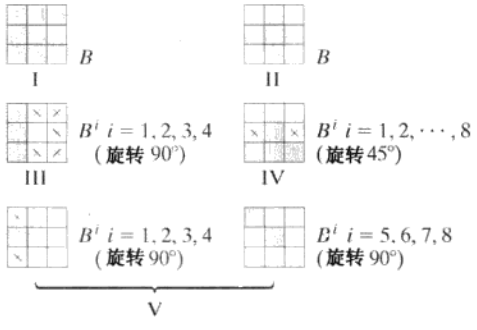


图 9.33 用于二值形态学的 5 个基本的结构元。每个结构元的原点均位于其中心处。×项表示“不考虑”的值

表 9.1 形态学操作及其性质小结

操 作	公 式	注 释 (罗马数字指图 9.33 中的结构元)
平移	$(B)_z = \{w \mid w = b + z,\ b \in B\}$	将 B 的原点平移到点 z
反射	$\hat{B}_z = \{w \mid w = -b,\ b \in B\}$	关于集合 B 的原点映射该集合的所有元素
求补集	$A^c = \{w \mid w \notin A\}$	不属于 A 的点的集合
求差集	$A - B = \{w \mid w \in A,\ w \notin B\}$ $= A \cap B^c$	属于 A 但不属于 B 的点的集合
膨胀	$A \oplus B = \{z \mid (\hat{B}_z) \cap A \neq \varnothing\}$	“扩展” A 的边界(I)
腐蚀	$A \ominus B = \{z \mid (B)_z \subseteq A\}$	“收缩” A 的边界(I)
开操作	$A \circ B = (A \ominus B) \oplus B$	平滑轮廓,切断狭窄区域,并消除小的孤岛和尖刺(I)
闭操作	$A \bullet B = (A \oplus B) \ominus B$	平滑轮廓,融合狭窄间断和细长沟堑,并消除小的孔洞(I)

683 ~ 685

（续表）

操　作	公　式	说　明 （罗马数字是指图 9.33 中的结构元）
击中或击不中变换	$A\circledast B=(A\ominus B_1)\cap(A^c\ominus B_2)$ $=(A\ominus B_1)-(A\oplus \hat{B}_2)$	点（坐标）的集合，在这样的点处，同时，B_1 在 A 中可找到一个匹配（击中），B_2 在 A^c 中可找到一个匹配
边界提取	$\beta(A)=A-(A\ominus B)$	在集合 A 的边界上的点的集合（I）
孔洞填充	$X_k=(X_{k-1}\oplus B)\cap A^c$; $k=1, 2, 3, \cdots$	填充 A 中的孔洞；X_0 是每个孔洞处元素为 1 而其他位置元素为 0 的阵列（II）
连通分量	$X_k=(X_{k-1}\oplus B)\cap A$; $k=1, 2, 3, \cdots$	寻找 A 中的连通分量；X_0 是每个连通分量中元素为 1 而其他位置元素为 0 的阵列（I）
凸壳	$X_k^i=(X_{k-1}^i\circledast B^i)\cup A$; $i=1, 2, 3, 4$; $k=1, 2, 3, \cdots$; $X_0^i=A$; $D^i=X_{\text{conv}}^i$	寻找集合 A 的凸壳 $C(A)$，其中 conv 表示 $X_k^i=X_{k-1}^i$（III）意义上的收敛，
细化	$A\otimes B=A-(A\circledast B)$ $=A\cap(A\circledast B)^c$ $A\otimes\{B\}=$ $((\cdots((A\otimes B^1)\otimes B^2)\cdots)\otimes B^n)$ $\{B\}=\{B^1, B^2, B^3, \cdots, B^n\}$	细化集合 A。前两个公式给出细化的基本定义。后两个公式表示使用一个结构元序列的细化。实际中通常使用这种方法（IV）
粗化	$A\odot B=A\cup(A\circledast B)$ $A\odot\{B\}=$ $((\cdots(A\odot B^1)\odot B^2\cdots)\odot B^n)$	粗化集合 A（见前面关于结构元序列的说明）。使用 IV，但把 0 和 1 颠倒
骨架	$S(A)=\bigcup_{k=0}^{K}S_k(A)$ $S_k(A)=\{(A\ominus kB)$ $-[(A\ominus kB)\circ B]\}$ A的重建： $A=\bigcup_{k=0}^{K}(S_k(A)\oplus kB)$	寻找集合 A 的骨架 $S(A)$。最后一个公式指出 A 可以由其骨架子集 $S_k(A)$ 重建。在所有 3 个公式中，K 是集合 A 被腐蚀为空集时的迭代次数。符号 $(A\ominus kB)$ 表示 B 对 A 连续腐蚀的第 k 次迭代（I）
裁剪	$X_1=A\otimes\{B\}$ $X_2=\bigcup_{k=1}^{8}(X_1\circledast B^k)$ $X_3=(X_2\oplus H)\cap A$ $X_4=X_1\cup X_3$	X_4 是裁剪集合 A 后的结果。必须指定使用第一个公式来得到 X_1 的次数。结构元 V 用于前两个公式。在第三个公式中，H 表示结构元 I
大小为 1 的测地膨胀	$D_G^{(1)}(F)=(F\oplus B)\cap G$	F 和 G 分别称为标记图像和模板图像
大小为 n 的测地膨胀	$D_G^{(n)}(F)=D_G^{(1)}\left[D_G^{(n-1)}(F)\right]$; $D_G^{(0)}(F)=F$	
大小为 1 的测地腐蚀	$E_G^{(1)}(F)=(F\ominus B)\cup G$	
大小为 n 的测地腐蚀	$E_G^{(n)}(F)=E_G^{(1)}\left[E_G^{(n-1)}(F)\right]$ $E_G^{(0)}(F)=F$	
膨胀形态学重建	$R_G^D(F)=D_G^{(k)}(F)$	k 满足 $D_G^{(k)}(F)=D_G^{(k+1)}(F)$
腐蚀形态学重建	$R_G^E(F)=E_G^{(k)}(F)$	k 满足 $E_G^{(k)}(F)=E_G^{(k+1)}(F)$
重建开操作	$O_R^{(n)}(F)=R_F^D[(F\ominus nB)]$	$(F\ominus nB)$ 表示 B 对 F 的 n 次腐蚀
重建闭操作	$C_R^{(n)}(F)=R_F^E[(F\oplus nB)]$	$(F\oplus nB)$ 表示 B 对 F 的 n 次膨胀
孔洞填充	$H=\left[R_{I^c}^D(F)\right]^c$	H 等于输入图像 I，但具有所有被填充的孔洞。标记图像 F 的定义见式（9.5-28）
边界清除	$X=I-R_I^D(F)$	X 等于输入图像 I，但带有所有被删除接触（连接到）边界的物体。标记图像 F 的定义见式（9.5-30）

685 ~ 686

9.6 灰度级形态学

在这一节中，我们将把膨胀、腐蚀、开操作和闭操作的基本操作扩展到灰度级图像。然后，我们将使用这些操作来探讨几个基本的灰度级形态学算法。

通过下面的讨论，我们将处理形如$f(x, y)$和$b(x, y)$的数字函数，其中$f(x, y)$是一幅灰度级图像，而$b(x, y)$是一个结构元。假设这些函数是 2.4.2 节中介绍过的离散函数。也就是说，如果 Z 表示实整数集合，则坐标(x, y)是来自笛卡儿积 Z^2 的整数，且f和b是对每个(x, y)坐标对赋以灰度值(来自实数集合 R 的一个实数)的函数。如果灰度级也是整数，则用 Z 代替 R。

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9.34 显示了每一类的一个例子。图9.34(a)以图像形式显示了一个半球状的灰度级结构元，图 9.34(c)是一个通过其中心的水平灰度剖面。图 9.34(b)显示了一个圆盘形的平坦结构元，图 9.34(d)是其对应的灰度剖面(该剖面的形状解释了“平坦”一词的由来)。为清楚起见，图 9.34 中的元素以连续量显示；它们的计算机实现是以数字近似为基础的(见图 9.2 中最右侧的圆盘形 SE“结构元”)。由于本节稍后讨论的一些困难，实际中灰度级结构元并不常用。最后，我们要提及的是，如在二值情况那样，必须清楚地确定结构元的原点。除非另外提及，本节中的所有例子都以高度为 1、对称的、平坦的结构元为基础，其原点位于中心处。灰度级形态学中结构元的反射的定义，如 9.1 节中的定义一样，且在下面的讨论中我们将它表示为$\hat{b}(x, y) = b(-x, -y)$。

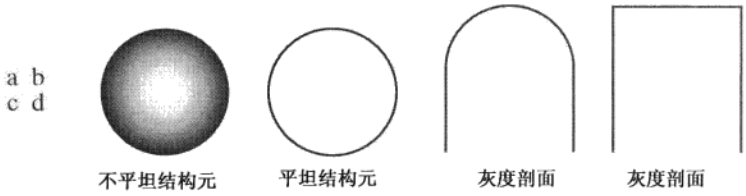


图 9.34 非平坦和平坦结构元，以及对应的通过其中心的水平灰度剖面。本节中的所有例子均基于平坦结构元

9.6.1 腐蚀和膨胀

当b的原点位于(x, y)处时，用一个平坦的结构元b在(x, y)处对图像f的腐蚀定义为图像f中与b重合区域的最小值。以公式的形式，结构元素b对一幅图像f在位置(x, y)处的腐蚀由下式给出：

$$[f \ominus b](x, y) = \min_{(s, t) \in b} \{f(x+s, y+t)\} \tag{9.6-1}$$

其中，用与 3.4.2 节讨论的相关过程类似的方式，x 和 y 是通过所有要求的值而增加的，以便 b 的原点能访问 f 中的每一个像素。也就是说，为寻求 b 对 f 的腐蚀，我们把结构元的原点放在图像每一个像素的位置。在任何位置的腐蚀由从包含在与 b 重合区域中的 f 的所有值中选取的最小值决定。例如，如果 b 是大小为 3×3 的方形结构元，则获得一点处的腐蚀要求寻找包含在由 b 定义的 3×3 区域中的 f 的 9 个值中其原点所在的那个点的最小值。

类似地，当$\hat{b}$的原点位于位置(x, y)处时，平坦结构元b在任何位置(x, y)处对图像f的膨胀，定义为图像f中与$\hat{b}$重合区域的最大值，即

687
~
688

$$[f \oplus b](x, y) = \max_{(s, t)\in b} \{f(x-s, y-t)\} \tag{9.6-2}$$

其中，我们用到了早些时候说明过的 $\hat{b} = b(-x,-y)$ 的事实。对该式的解释与前一段中的解释相同，但使用的是最大而不是最小操作，并且要记住结构元关于其原点反射，我们用 $(-s,-t)$ 考虑函数的自变量。这类似于空间卷积，就像 3.4.2 节中解释的那样。

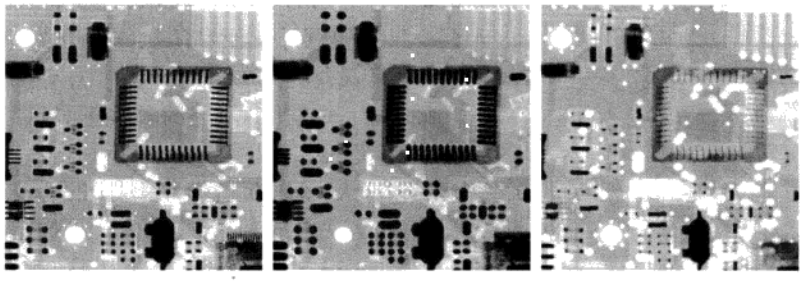
例 9.9　灰度级腐蚀和膨胀的说明。

因为使用一个平坦 SE（结构元）的灰度级腐蚀计算图像 f 中与 b 重合的 (x, y) 的每一个邻域的最小灰度值，因此通常我们希望被腐蚀的灰度级图像比原始图像暗，亮特征的尺寸（关于 SE 的尺寸）将被减小，而暗特征的尺寸将会增大。图 9.35(b) 显示了使用一个单位高度和半径为 2 个像素的圆盘形结构元素对图 9.35(a) 的腐蚀。刚才讨论的效果在腐蚀过的图像中清晰可见。例如，请注意小亮点亮度的降低程度，图 9.35(b) 中几乎已看不到这些小亮点，同时暗色特征则变浓了。腐蚀过的图像中的一般背景比原图像的背景要稍暗一些。类似地，图 9.35(c) 显示了使用相同 SE 膨胀后的结果。其效果与用腐蚀得到的效果相反，即亮特征变浓了，而暗特征降低了。请特别注意图 9.35(a) 中左侧、中间、右侧和底部的较细黑色连线在图 9.35(c) 中几乎已看不见了。黑点的尺寸已被减小，但与图 9.35(b) 中被腐蚀的小白点完全不同，在膨胀后的图像中，小白点却清晰可见。其原因是就结构元的尺寸相比，黑点的尺寸原来就比白点大。最后，请注意膨胀后的图像中的背景比图 9.35(a) 中的背景稍亮。

非平坦结构元具有随定义域而变化的灰度级。非平坦结构元 b_N 对图像 f 的腐蚀定义如下：

$$[f \ominus b_N](x, y) = \min_{(s, t)\in b_N} \{f(x+s, y+t) - b_N(s, t)\} \tag{9.6-3}$$

这里，我们实际上是从 f 中减去 b_N 的值来确定任意点处的腐蚀。与式 (9.6-1) 不同，这意味着使用非平坦结构元的腐蚀通常不受 f 值的限制，而这在解释结果时会存在问题。因为这一原因，实际中很少使用灰度级结构元；另外，与式 (9.6-1) 比较，为 b_N 选取有意义的元素和增加的计算负担也是潜在的困难。



a b c

图 9.35　(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（原图像由 Lixi 公司提供）

采用类似的方式，使用非平坦结构元的膨胀定义如下：

$$[f \oplus b_N](x, y) = \max_{(s, t)\in b_N} \{f(x-s, y-t) + b_N(s, t)\} \tag{9.6-4}$$

前一段给出的说明同样适用于使用非平坦结构元的膨胀。当 b_N 的所有元素都是常数时（即结构元是平坦的），式 (9.6-3) 和式 (9.6-4) 就分别简化为式 (9.6-1) 和式 (9.6-2)，其中一个标量常数等于该结构元的幅度。

如二值情况那样，腐蚀和膨胀是关于函数的补集和反射对偶的，即

$$(f \ominus b)^c(x, y) = (f^c \oplus \hat{b})(x, y)$$

688 ~ 689

其中，$f^c=-f(x,y)$ 且 $\hat{b}=b(-x,-y)$。相同的表达式对非平坦结构元也有效。为清楚起见，在下面的讨论中，我们将忽略所有函数的参量以简化表达式，在这种情况下，前一公式可写为

$$(f\ominus b)^c=(f^c\oplus\hat{b}) \tag{9.6-5}$$

类似地，

$$(f\oplus b)^c=(f^c\ominus\hat{b}) \tag{9.6-6}$$

自身的腐蚀和膨胀在灰度级图像处理中并不是特别有用。像在对应的二值情形那样，当组合用来推导高级算法时，这些操作会变得强有力，正如下节所示的材料那样。

尽管我们在本节剩余内容的例子中介绍的是平坦结构元，但讨论的概念同样适用于非平坦结构元。

9.6.2 开操作和闭操作

灰度级图像的开操作和闭操作的表达式与二值图像的对应操作具有相同的形式。结构元 b 对图像 f 的开操作表示为 $f\circ b$，即

$$f\circ b=(f\ominus b)\oplus b \tag{9.6-7}$$

像之前那样，开操作先只用 b 对 f 做腐蚀，随后用 b 对所得结果做膨胀。类似地，b 对 f 的闭操作表示为 $f\bullet b$，即

$$f\bullet b=(f\oplus b)\ominus b \tag{9.6-8}$$

灰度级图像的开操作和闭操作关于函数的补集和结构元的反射是对偶的：

$$(f\bullet b)^c=f^c\circ\hat{b} \tag{9.6-9}$$

和

$$(f\circ b)^c=f^c\bullet\hat{b} \tag{9.6-10}$$

因为 $f^c=-f(x,y)$，所以式(9.6-9)也可以写为 $-(f\bullet b)=(-f\circ\hat{b})$，对于式(9.6-10)也类似。

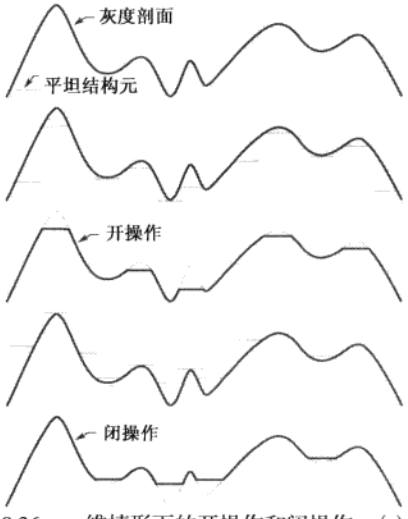


图 9.36 一维情形下的开操作和闭操作：(a)原始的一维信号；(b)从信号底部向上推动的平坦结构元素；(c)开操作；(d)沿信号顶部向下推动的平坦结构元素；(e)闭操作

图像的开操作和闭操作具有简单的几何解释。假设我们将一个图像函数 $f(x, y)$ 看成是一个三维表面；也就是说，它的灰度值可解释为 xy 平面上的高度值，如图 2.18(a)所示。然后，b 对 f 的开操作可从几何角度解释为，从 f 的下表面向上推动结构元。在 b 的每个原点位置，开操作是当从 f 的下表面向上推动结构元时，b 的任何部分所达到的最高值。这样，完全开操作就是由 b 的原点访问 f 的每一个坐标 (x, y) 所得到的所有值的集合。

图 9.36 以一维的形式说明了这个概念。假定图 9.36(a)中的曲线是沿图像中的一行的灰度剖面。图 9.36(b)显示了沿曲线底部向上推到不同位置的一个平坦结构元。图 9.36(c)的实线是完全开操作。因为结构元太大而不能完全拟合曲线上部峰值的内侧，故峰值的顶部被开操作剪切掉了，剪切的量与结构元能够到达峰值的

有时，开操作和闭操作可由在一条曲线的下侧和上侧滚动一个圆来说明。在三维情形下，圆可用球体来替代，得到的过程称为滚球算法。

689 ~ 691

距离成比例。通常，开操作用于去除较小的明亮细节，而保持整体灰度级和较大的明亮特征相对不变。

图 9.36(d) 是闭操作的图形说明。很明显，结构元从曲线的顶部向下推动，同时平移到所有位置。如图 9.36(e) 所示，当结构元从曲线的上侧滑动时，通过寻找结构元的任何部分所能到达的最低点，就可构建闭操作。

灰度级开操作满足如下性质：

(a) $f \circ b \hookleftarrow f$ 。

(b) 如果 $f_1 \hookleftarrow f_2$ ，则 $(f_1 \circ b) \hookleftarrow (f_2 \circ b)$ 。

(c) $(f \circ b) \circ b = (f \circ b)$ 。

符号 $e \hookleftarrow r$ 用来表示 e 的域是 r 的域的一个子集，且对于 e 的域中的任何 (x, y)，有 $e(x, y) \leqslant r(x, y)$。

类似地，闭操作满足如下性质：

(a) $f \hookleftarrow f \bullet b$ 。

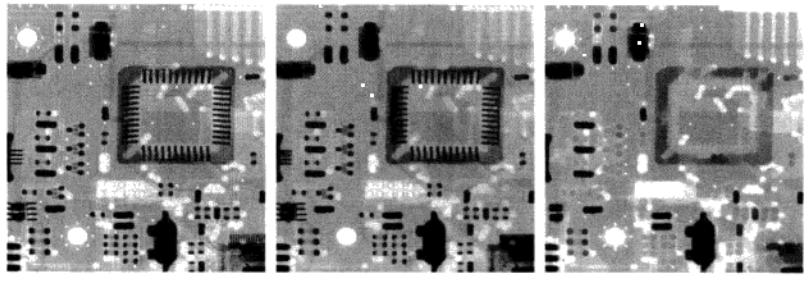
(b) 如果 $f_1 \hookleftarrow f_2$ ，则 $(f_1 \bullet b) \hookleftarrow (f_2 \bullet b)$ 。

(c) $(f \bullet b) \bullet b = (f \bullet b)$ 。

这些性质的用途类似于二值图像中的对应性质。

例 9.10　灰度级开操作和闭操作的说明。

图 9.37 将图 9.36 中说明的一维概念拓展到了二维。图 9.37(a) 与我们在例 9.9 使用的图像相同，图 9.37(b) 是使用单位高度和半径为 3 个像素的圆盘形结构元得到的开操作结果。如预料的那样，所有亮特征的灰度都降低了，降低的程度取决于这些特征相对于结构元的尺寸。该图与图 9.35(b) 相比，我们看到，与腐蚀的结果不同，开操作对图像的暗特征影响可忽略不计，也不影响背景。类似地，图 9.37(c) 显示了使用半径为 5 的圆盘形结构元得到的闭操作结果(小的圆黑点比小白点大，因此要达到可与开操作相比的结果，需要更大的圆盘形结构元)。在这幅图像中，亮的细节和背景相对来说未受影响，但削弱了暗特征，削弱的程度取决于这些特征相对于结构元的尺寸。



a b c

图 9.37　(a)一幅大小为 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得到的开操作结果；(c)使用半径为 5 个像素的结构元得到的闭操作结果

9.6.3　一些基本的灰度级形态学算法

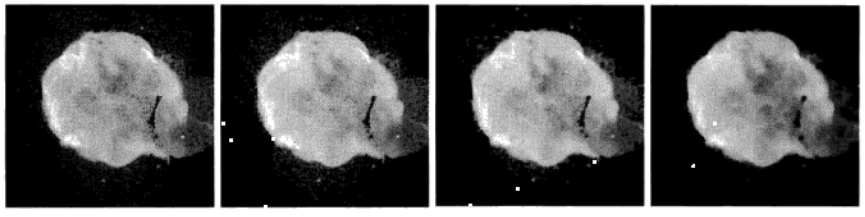
许多形态学技术是以迄今为止介绍的灰度级形态学概念为基础的。下面的讨论中我们将说明这些算法。

形态学平滑

因为开操作抑制比结构元小的亮细节，而闭操作抑制暗细节，所以它们常常以形态滤波的形式结合起来被用于图像平滑和噪声去除。考虑图 9.38(a)，它显示了一幅使用 X 射线波段拍摄的天鹅星

座环超新星图像(关于该图像的细节请参阅图 1.7)。为当前讨论的目的，假设中心的亮区域是我们感兴趣的目标，而较小的分量是噪声。我们的目的是去除噪声。图9.38(b)显示了用一个半径为 2 的平坦圆盘(结构元)对原始图像开操作，再用相同尺寸的 SE(结构元)进行闭操作的结果。图 9.38(c)和(d)分别显示了使用半径为 3 和 5 的结构元进行相同操作得到的结果。如期望的那样，该图像序列显示了消除作为结构元尺寸的函数的小分量的改进。在最后的结果中，我们看到，感兴趣的目标已被提取出来。图像底部的噪声分量未被完全去除，原因在于它们的密度。

图9.38 的结果是以对原始图像进行开操作，然后再闭操作为基础的。有时使用的一个过程是执行交替顺序滤波，在该过程中，对原始图像以开操作-闭操作顺序开始，但后续步骤对前一步骤的结果再执行开操作和闭操作。这种类型的滤波在自动图像分析中很有用，滤波过程中每一步的结果都针对一个指定的度量进行比较。通常，对于相同大小的结构元，与图9.38 中说明的方法相比，这种方法会产生更为模糊的结果。



a b c d

图 9.38 (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顺序的结果(原图像由 NASA 提供)

形态学梯度

膨胀和腐蚀可与图像相减结合起来得到一幅图像的形态学梯度，在这里，由 g 来定义：

$$g = (f \oplus b) - (f \ominus b) \tag{9.6-11}$$

膨胀粗化一幅图像中的区域，而腐蚀则细化它们。膨胀和腐蚀的差强调了区域间的边界。同质区域不受影响(只要 SE 相对较小)，因此相减操作趋于消除同质区域。最终结果是是边缘被增强而同质区域的贡献被抑制掉了的图像，从而产生“类似于微分”(梯度)的效果。

图 9.39 显示了一个例子。图 9.39(a)是一幅头部 CT 扫描图像，接下来的两幅图像是使用所有元素都为 1 的 3×3 结构元对该图像进行开操作和闭操作的结果。注意，刚刚提到的粗化和收缩。图 9.39(d)是使用式(9.6-11)得到的形态学梯度，其中，区域间的边界被清楚地描绘出来了，这与二维微分图像的预期相同。

a b c d

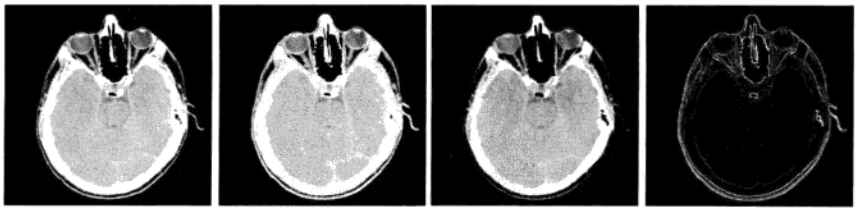


图 9.39 (a)512×512 的头部 CT 扫描图像；(b)膨胀的结果；(c)腐蚀的结果；(d)形态学梯度，计算图(b)和图(c)间的差(原图像由 Vanderbilt 大学的 David R. Pickens 博士提供)

692
~
694

顶帽变换和底帽变换

图像相减与开操作和闭操作相结合，会产生所谓的 Top-hat（顶帽）变换和 bottom-hat（底帽）变换。灰度级图像 f 的顶帽变换定义为 f 减去其开操作：

$$T_{\text{hat}}(f) = f - (f \circ b) \tag{9.6-12}$$

类似地，f 的底帽变换定义为 f 的闭操作减去 f：

$$B_{\text{hat}}(f) = (f \bullet b) - f \tag{9.6-13}$$

这些变换的主要应用之一是，用一个结构元通过开操作或闭操作从一幅图像中删除物体，而不是拟合被删除的物体。然后，差操作得到一幅仅保留已删除分量的图像。顶帽变换用于暗背景上的亮物体，而底帽变换则用于相反的情况。由于这一原因，当谈到这两个变换时，常常分别称为白顶帽变换和黑底帽变换。

顶帽变换的一个重要用途是校正不均匀光照的影响。正如我们将在下一章中看到的那样，合适（均匀）的光照在从背景中提取目标的处理中扮演核心的角色。称为分割的这一处理是自动图像分析中执行的第一步。一种常用的分割方法是对输入图像进行阈值处理。

为了说明，考虑图 9.40(a)，它显示了一幅大小为 600×600 的米粒的图像。该图像是在非均匀光照下得到的，如图像底部及最右侧的暗色区域就是明证。图9.40(b)显示了对该图像使用10.3.3 节讨论的 Otsu 最佳阈值处理方法得到的结果。非均匀光照的最终结果导致了暗区域的分割错误（一些米粒没有从背景中提取出来），且在图像的左上角，背景部分被错误地分类了。图 9.40(c) 显示了对该图像使用一个半径为 40 的圆盘形结构元进行开操作的结果。这个结构元足够大以致不会拟合任何物体。如结果那样，这些物体被消除了，仅留下一个近似的背景。阴影模式在该图像中很清楚。通过从原图像中减去该图像（即执行顶帽变换），背景应会变得更均匀。事实的确如此，如图 9.40(d) 所示。背景并不是非常均匀，但亮和暗之间不再存在极端的差别，这就足以得到正确的分割结果，其中所有的米粒均被检测出来，如图 9.40(e) 所示。以前，该图像是使用 Otsu 方法得到的。

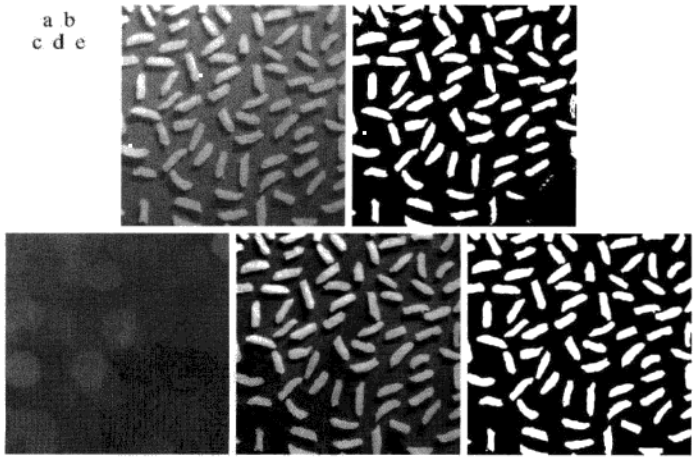


图 9.40　使用顶帽变换校正阴影：(a) 大小为 600×600 的原图像；(b) 阈值处理后的图像；(c) 用半径为 40 的圆盘形结构元进行开操作的图像；(d) 顶帽变换的图像（图像减去其开操作）；(e) 顶帽变换的图像经阈值处理的结果

粒度测定

在图像处理方面，粒度测定是属于判断图像中颗粒的尺寸分布的领域。实践中，颗粒很少能完

694 ~ 696

美地被分离，这就使得用识别单个颗粒来计数成为一个困难的任务。形态学可间接用于估计颗粒的尺寸分布，而不需要识别并测量图像中的每个颗粒。

这种方法原理上非常简单。对于比背景亮且具有规则形状的颗粒，该方法由使用逐渐增大的结构元对图像执行开操作组成。基本概念是，某个特殊尺寸的开操作应会对包含类似尺寸的颗粒的输入图像的区域产生最大的效果。对于每一次开操作，计算该开操作中像素值的和。该和有时称为表面区域，它会随着结构元的增大而减小，如我们早些时候注释的那样，因为开操作会降低亮特征的灰度。该过程会得到一个这样的数字的一维阵列，阵列中的每个元素等于对应于阵列中该位置的结构元素的大小的开操作中的像素之和。为了强调连续开操作间的变化，我们计算一维阵列的相邻元素的差。为了形象化该结果，我们画出了该差的图形。曲线中的峰值表明了图像中颗粒的主要大小分布。

作为一个例子，考虑图 9.41(a)，它是两种不同大小的木钉的图像。对木钉中的木颗粒可能需要引入不同的开操作，因此，平滑是一种合理的预处理步骤。图 9.41(b)显示了使用早期讨论过的形态学滤波器平滑后的图像，该滤波器用半径为 5 的圆盘形结构元。图 9.41(c)到(f)显示了使用半径为 10, 20, 25 和 30 的圆盘形结构元对图像进行开操作的例子。注意，在图 9.41(d)中，由于较小木钉，灰度的贡献几乎已被消除。图 9.41(e)中，大木钉的贡献已被显著地降低了，图 9.41(f)更厉害。[观察图 9.41(e)，靠近图像右上方的大木钉比其暗得多，因为它的尺寸较小。如果我们试图检测有缺陷的木钉，那么这将是有用的信息]。

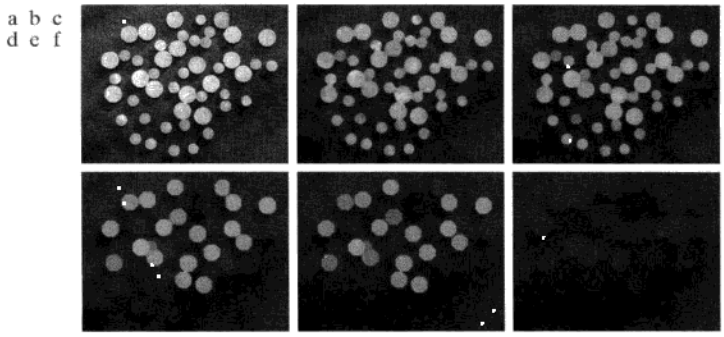


图 9.41 (a)大小为 531×675 的木钉图像；(b)平滑后的图像；(c)~(f)分别用半径为 10, 20, 25 和 30 像素的圆盘形结构元对图像进行开操作后的结果(原图像由 MathWorks 公司的 Steve Eddins 博士提供)

图 9.42 显示了该差值阵列的曲线。如前边提到的那样，我们期望半径附近的差值较大(曲线中的峰值)，在该处结构元足够大，以包围近似相同直径的一组颗粒。图 9.42 中的结果有两个明显的峰值，这清楚地表明图像中存在两种主要的物体尺寸。

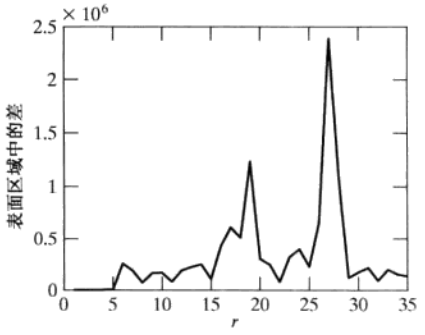


图 9.42 表面区域中的差，它是圆盘形结构元半径 r 的函数。两个峰值表明图像中存在两个主要的颗粒尺寸

纹理分割

图 9.43(a)显示了一幅在亮背景上添加了暗斑点的噪声图像。该图像有两个纹理区域：右边大斑点组成的区域和左边小斑点组成的区域。目的是以纹理内容为基础找到两个区域的边界(我们将在 11.3.3 节讨论纹理)。如早期解释的那样，将一幅图像分为区域的处理称为分割，分割是第 10 章的话题。

696 ~ 697

感兴趣的物体比背景暗，并且我们知道，如果使用一个比小斑点大的结构元对图像做闭操作，这些斑点将被删除。在图 9.43(b)中，使用半径为 30 个像素的圆盘形结构元对输入图像做闭操作的结果表明情况的确如此(斑点的半径近似为 25 像素)。因此，我们得到了一幅亮背景上带有暗的大斑点的图像。如果我们使用尺寸大于这些斑点间的间隔的结构元对该图像做开操作，则最终结果应该是这样一幅图像，在该图像中，斑点间的亮的间隔已被删除，留下了暗的斑点，并且这些斑点之间现在已成为相同的暗间隔。图 9.43(c)显示了使用半径为 60 的圆盘形结构元得到的结果。

图 9.43　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)中两个区域间的边界叠加到原图像上后的结果。边界是使用形态学梯度操作得到的

使用一个其元素全为 1 且大小为 3×3 的结构元对该图像执行形态学梯度操作将给出这两个区域间的边界。图 9.43(d)显示了将形态学梯度操作结果叠加到原图像后所得到的边界。该边界右侧的所有像素表示属于由大斑点表征的纹理区域，而该边界左侧的所有像素属于由小斑点表征的纹理区域。您将发现，若使用图 9.36 中对开操作和闭操作的图形模拟，那么我们可更直观地理解这一例子。

9.6.4　灰度级形态学重建

灰度级形态学重建基本上按照与9.5.9 节针对二值图像所介绍的相同的方法来定义。令 f 和 g 分别代表标记图像和模板图像。我们假设 f 和 g 是大小相同的灰度级图像，且 $f \leqslant g$。f 关于 g 的大小为 1 的测地膨胀定义为

很容易理解这些表达式是 (x, y) 的函数。为简化表示，我们省略了 (x, y)。

$$D_g^{(1)}(f) = (f \oplus b) \wedge g \tag{9.6-14}$$

式中，$\wedge$ 代表点方式的最小算子。该式指出，大小为 1 的测地膨胀是由先计算 b 对 f 的膨胀，然后选择在每一个 (x, y) 点处该结果和 g 间的最小者。如果 b 是一个平坦结构元，则膨胀由式(9.6-2)给出，否则膨胀由式(9.6-4)给出。f 关于 g 的大小为 n 的测地膨胀定义为

$$D_g^{(n)}(f) = D_g^{(1)}\left[D_g^{(n-1)}(f)\right] \tag{9.6-15}$$

并有 $D_g^{(0)}(f) = f$。

类似地，f 关于 g 的大小为 1 的测地腐蚀定义为

$$E_g^{(1)}(f) = (f \ominus b) \vee g \tag{9.6-16}$$

其中，$\vee$ 表示点方式的最大算子。f 关于 g 的大小为 n 的测地腐蚀定义为

$$E_g^{(n)}(f) = E_g^{(1)}\left[E_g^{(n-1)}(f)\right] \tag{9.6-17}$$

并有 $E_g^{(0)}(f) = f$。

697 ~ 699

灰度级标记图像 f 对灰度级模板图像 g 的膨胀形态学重建定义为 f 关于 g 的测地膨胀反复迭代直至达到稳定；即

本节中表达式间的对偶关系列表见习题 9.33。

$$R_g^D(f) = D_g^{(k)}(f) \tag{9.6-18}$$

并有 k 应使 $D_g^{(k)}(f) = D_g^{(k+1)}(f)$。$f$ 对 g 的腐蚀的形态学重建类似地定义为

$$R_g^E(f) = E_g^{(k)}(f) \tag{9.6-19}$$

并有 k 应使 $E_g^{(k)}(f) = E_g^{(k+1)}(f)$。

如在二值情况那样，灰度级图像重建的开操作首先腐蚀输入图像，并用它作为标记图像。一幅图像 f 的大小为 n 的重建开操作定义为先对 f 进行大小为 n 的腐蚀，再由 f 的膨胀重建；即

$$O_R^{(n)}(f) = R_f^D[(f \ominus nb)] \tag{9.6-20}$$

其中，$(f \ominus nb)$ 表示 b 对 f 的 n 次腐蚀，如 9.5.7 节解释的那样。回忆式(9.5-27)针对二值图像的讨论，重建开操作的目的是保护腐蚀后留下的图像分量的形状。

类似地，图像 f 的大小为 n 的重建闭操作定义为先对 f 进行大小为 n 的膨胀，再由 f 的腐蚀重建；即

$$C_R^{(n)}(f) = R_f^E[(f \oplus nb)] \tag{9.6-21}$$

其中，$(f \oplus nb)$ 表示 b 对 f 的 n 次腐蚀。因为对偶性，图像的重建闭操作可以用图像的求补得到，先得到重建开操作，然后再求结果的补。最后，如下例所示，称为重建顶帽的一种有用技术是从一幅图像中减去其重建开操作。

例 9.11 使用形态学重建展平复杂的背景。

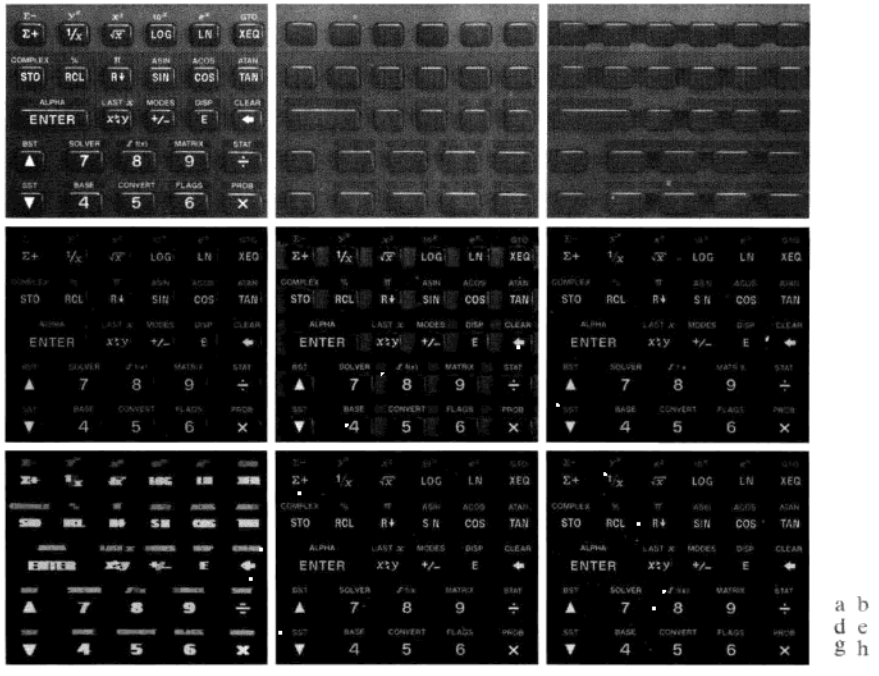
在这个例子中，我们用规格化图 9.44(a)中不规则图像背景的一些步骤来说明灰度级重建的用途，即仅留下恒定灰度背景上的正文。这一问题的解决是形态学概念的能力的很好的说明。我们从抑制各个键顶部的水平反射开始。在图像中反射比任何单个字符都宽，因此，我们应该能在腐蚀操作中用一条长的水平线通过执行重建的开操作抑制它们。该操作将产生包含键及其反射的背景。从原图像减去这个背景(即执行一个重建顶帽操作)将从原图像中消除水平反射和变化的背景。

图 9.44(b)显示了在腐蚀操作中用一条大小为 1×71 像素的水平线对原图像执行重建开操作的结果。我们已经能够仅用一个开操作就除去字符，但是，得到的背景将不会如图 9.44(c)所示那样均匀（例如，比较两幅图像中各个键之间的区域）。图 9.44(d)显示了从图 9.44(a)中减去图 9.44(b)的结果。如期望的那样，水平反射和背景的变化被抑制了。为便于比较，图 9.44(e)显示了执行刚才的顶帽变换的结果(如本节之前讨论的那样，即从图像中减去“标准的”开操作)。从图 9.44(c)中背景的特征可以预料，图 9.44(e)中的背景不像图 9.44(d)中的那样均匀。

下一步是从键的边缘删除垂直反射，这在图 9.44(d)中非常明显。通过使用一个其宽度近似等于这些反射的线状结构元来执行一个重建开操作，我们可删除垂直反射(在这种情况下，结构元的宽度约为 11 个像素)。图 9.44(f)显示了对图 9.44(d)执行该操作的结果。垂直反射被抑制了，有用字符的垂直笔划也被细化了(如 SIN 中的 I)，因此，我们不得不寻找一种恢复该字符的方法。由于被抑制的字符非常靠近其他字符，因此，如果我们水平地膨胀遗留下来的字符，则膨胀后的字符将与先前被抑制字符所占的区域重叠。使用大小为 1×21 的线状结构元膨胀图 9.44(f)后得到的图 9.44(g)的结果显示事实的确如此。

699 ~ 701

此时所有余下的问题是复原被抑制的字符。考虑图 9.44(g)中膨胀后的图像和图 9.44(d)中重建顶帽图像间点方式的最小值形成的一幅图像。图 9.44(h)显示了最小值图像(虽然该结果看起来接近我们的目标，但 SIN 中仍遗漏了 I)。在灰度级重建中，通过将这幅图像作为标记图像，并将膨胀后的图像作为模板图像 [见式(9.6-18)]，我们得到如图 9.44(i)所示的最终结果。这幅图像表明所有字符都已完全地从原图像的不规则背景，包括键的背景中提取出来了。图 9.44(i)中的背景都是均匀的。



a b c
d e f
g h i

图 9.44　(a)大小为 1134×1360 像素的原图像；(b)用腐蚀中的一条 71 像素长的水平线对图(a)的重建开操作；(c)用相同的线对图(a)的开操作；(d)重建顶帽操作；(e)顶帽操作；(f)用 11 个像素长的水平线对图(d)的重建开操作；(g)使用 21 个像素长的水平线对图 f 的膨胀操作；(h)图(d)和图(g)的最小操作；(i)最后的重建结果(原图像由 MathWork 公司的 Steve Eddins 博士提供)

小结

本章介绍的形态学概念和技术构成了从一幅图像中提取感兴趣特征的一组强有力的工具。形态学图像处理最吸引人的一个方面就是来自形态学技术已有发展的广泛的集合论基础。实现方面的一个显著优点是膨胀和腐蚀都是基本操作，它们是各类形态学算法的基础。正如下章所示，形态学可以作为广泛应用的图像分割程序的基础。如第 11 章中讨论的那样，形态学技术在图像描述中也起着重要的作用。

参考文献

Serra[1982]的书籍是关于形态学图像处理的基础性参考资料，也可以参阅 Serra[1988]，Giardina 和 Dougherty[1988]及Haralick 和 Shapiro[1992]的著述。与我们的讨论相关的其他早期参考文献包括 Blum[1967],

Lantuéjoul[1980], Maragos[1987]和 Haralick 等人[1987]等的著述。关于二值形态学和灰度级形态学的综述见 Basart 和 gonzalez[1992]及 Basart 等人[1992]的著述。这组参考文献为 9.1 节到 9.4 节涵盖的内容提供了丰富的背景。9.5 节和 9.6 节中材料的较好综述见 Soille[2003]的论著。

实现形态学算法,如 9.5 节和 9.6 节中给出的算法的重要刊物见 Jones 和 Svalbe[1994], Park 和 Chin [1995], Sussner 和 Ritter[1997], Anelli 等人[1998]及 Shaked 和 Bruckstein[1998]的论文。关于实现灰度级形态学算法的实践细节见 Vincent[1993]的论文,也可以参阅 Gonzalez, Woods 和 Eddins[2004]的书籍。

关于形态学图像处理理论和应用的其他读物见 Goutsias 和 Bloomberg[2000]的图书和 *Pattern Recongnition* [2000]的一期特刊。也可以参阅 Rosenfeld[2000]编辑的参考文献。Marchand-Maillet 和 Sharaiha[2000]关于二值图像处理的书籍及 Ritter 和 Wilson[2001]关于图像代数学的书籍也是重要的文献。关于形态学技术在图像处理方面的应用的当前工作请参阅 Kim[2005]及 Evans 和 Liu[2006]的论文。

标有星号的习题的详细解答,可在本书的网站上找到。该网站中还包含有基于本章内容而建议的项目。

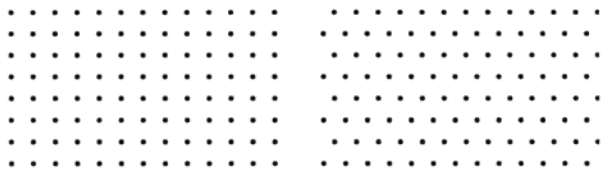
习题

9.1 本书中的数字图像嵌入了方形排列的网格,且像素允许是 4, 8 或 m 连通的。然而,其他网格排列也是可能的。特别地,有时也会使用 6 连通的六边形网格排列(见下面的图形)。

(a) 如何将图像从方形网格转换到六边形网格?

(b) 讨论由方形网格和六边形网格表示的物体的形状旋转不变性。

(c) 在六边形网格中,是否像在 8 连通情况下那样(见 2.5.2 节),可能存在有歧义的对角线配置?



9.2 ★(a) 针对把 4 连通的二值边界转换为 m 连通的边界(见 2.5.2 节)给出一个形态学算法。可以假设边界是全连通的,且边界的宽度为 1 个像素。

(b) 您的算法的操作需要用每个结构元进行多于一次的迭代过程吗? 试解释原因。

(c) 您的算法的性能是否与应用结构元的顺序无关? 如果您的回答是无关,请证明之;否则请举出一个例子,说明您的过程与应用结构元的顺序有关。

9.3 只要 B 的原点包含于 B 中,则结构元 B 对集合 A 的腐蚀就是 A 的一个子集。给出腐蚀 $A \ominus B$ 位于 A 之外或部分位于 A 之外的一个例子。

9.4 下列 4 种说法是正确的。请给出能证明这些说法正确的论据。(a)通常是正确的;(b)到(d)仅对数字集合才是正确的。为了证明(b)至(d)的正确性,请画出一个离散的方形网格(如习题 9.1 所示),并针对每种情况,使用由该网格上的点构成的集合给出一个例子。(提示:在仍能确定这些说法的正确性时,尽可能保持每种情况下的点数最少。)

★(a) 用一个凸结构元对一个凸集的腐蚀仍是一个凸集。

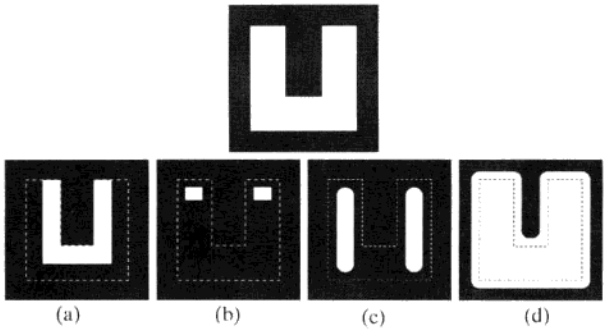
★(b) 用一个凸结构元对一个凸集的膨胀并不总是凸的。

(c) 一个数字凸集内的点并不总是连通的。

(d) 有可能存在一个这样的点集,在该集合中,连接每个点对的线位于该集合内,但该集合不是凸的。

702 ~ 703

★9.5 参考所显示的图像,给出生成图(a)至图(d)所示结果的结构元和形态学操作。请清晰地说明每个结构元的原点。图中的虚线表示原始集合的边界,仅供参考显示。注意图(d)中的所有角都是圆角。



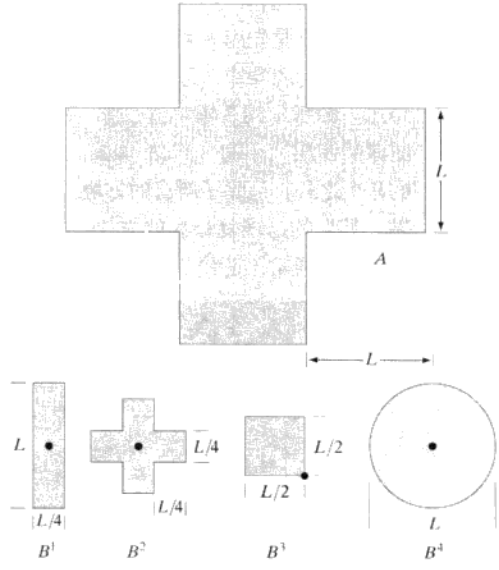
9.6 令 A 表示右图中阴影所示的集合。参考所示的结构元(黑点表示原点)，画出下列形态学操作的结果：

(a) $(A \ominus B^2) \oplus B^4$

(b) $(A \ominus B^1) \oplus B^3$

(c) $(A \ominus B^3) \oplus B^1$

(d) $(A \oplus B^3) \ominus B^2$



★9.7 (a) 反复膨胀一幅图像的极限效果是什么？假设不使用只有一个点的结构元。

(b) 为使(a)的答案成立，您能从其开始的最小图像是什么？

9.8 (a) 反复腐蚀一幅图像的极限效果是什么？假设不使用只有一个点的结构元。

(b) 为使(a)的答案成立，您能从其开始的最小图像是什么？

★9.9 腐蚀的另一种可替换定义是

$$A \ominus B = \{w \in Z^2 \mid w + b \in A, \text{ 对于每个 } b \in B\}$$

证明该定义等价于式(9.2-2)中的定义。

9.10 (a) 证明习题 9.9 中给出的腐蚀定义，仍等价于如下腐蚀的另一个定义：

$$A \ominus B = \bigcap_{b \in B} (A)_{-b}$$

(若用 b 代替 $-b$，则该表达式称为两个集合的闵可夫斯基相减。)

(b) 证明(a)中的表达式也等价于式(9.2-2)的定义。

★9.11 膨胀的一个可替换的定义是

$$A \oplus B = \{w \in Z^2 \mid w = a + b, \text{ 对于一些 } a \in A \text{ 和 } b \in B\}$$

证明该定义等价于式(9.2-4)中的定义。

9.12 (a) 证明习题 9.11 中给出的膨胀定义，等价于如下关于膨胀的另一个定义：

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

(该表达式也称为两个集合的闵可夫斯基相加。)

(b) 证明(a)中的表达式也等价于式(9.2-4)中的定义。

9.13 证明式(9.2-6)和式(9.2-5)中对偶表达式的正确性。

★9.14 证明对偶表达式 $(A \bullet B)^c = (A^c \circ \hat{B})$ 和 $(A \circ B)^c = (A^c \bullet \hat{B})$ 的正确性。

703 ~ 705

9.15 证明下列表达式的正确性：

★(a) $A \circ B$ 是 A 的一个子集(子图像)。

(b) 若 C 是 D 的一个子集，则 $C \circ B$ 是 $D \circ B$ 的一个子集。

(c) $(A \circ B) \circ B = A \circ B$。

9.16 证明下列表达式的正确性［假设 B 的原点包含在 B 中，且习题 9.14 和习题 9.15 为真］：

(a) A 是 $A \bullet B$ 的一个子集(子图像)。

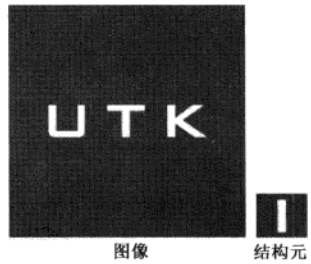
(b) 若 C 是 D 的一个子集，则 $C \bullet B$ 是 $D \cdot B$ 的一个子集。

(c) $(A \bullet B) \bullet B = A \bullet B$。

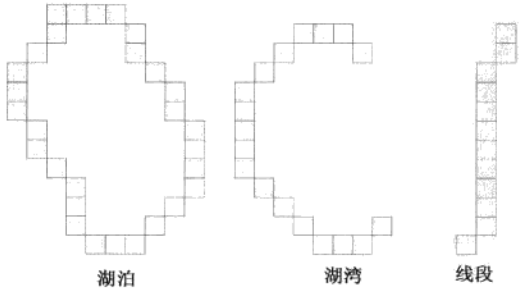
9.17 参考下图所示的图像和结构元。画出经如下操作后集合 C, D, E 和 F 是什么：$C = A \ominus B$; $D = C \oplus B$; $E = D \ominus B$; $F = E \oplus B$。初始集合 A 由显示为白色的所有图像分量组成，但结构元 B 除外。注意，这些操作仅是先用 B 对 A 进行简单的开操作，接着用 B 对开操作的结果进行闭操作。您可以假设 B 刚好大到包围每个噪声分量。

★9.18 考虑下图所示的 3 幅二值图像。左侧的图像由边长为 1, 3, 5, 7, 9 和 15 像素的方块组成。中间的图像是使用大小为 13×13 像素、元素为 1 的方形结构元对左侧图像进行腐蚀生成的，除了最大的几个之外，消除了所有的方块。最后，右侧的图像是使用相同的结构元对中间图像膨胀后的结果，其目的是恢复最大的方块。您知道，先腐蚀再膨胀实际上是对图像的开操作，并且您还知道开操作通常不能将物体恢复为原始形式。请解释这种情形下为何能完全重建较大的方块。

9.19 画出对右侧图像应用击中或击不中变换得到的结果和结构元。请清楚地指出您为结构元选取的原点和边界。



★9.20 用于区分图像中细化后物体的三个特征(湖泊、湖湾和线段)如下图所示。开发一种区分这些形状的形态学/逻辑算法。算法的输入是这三种形状之一。输出必须是输入的特征。可以假设这些特征的宽度都为 1 像素，且每个特征都是完全连通的，然而，它们出现时的方向可以是任意的。



705 ~ 707

9.21 在下列每种情况下，讨论您期望得到的结果是什么：

(a) 9.5.2 节中孔洞填充算法的起始点是物体边界上的一点。

(b) 该孔洞填充算法中的起始点位于物体边界之外。

(c) 使用 9.5.4 节中给出的算法，计算习题 9.6 中图形的凸壳外表特征是什么。请画出计算出的凸壳。假设 L 为 4 个像素。

9.22★(a) 若使用图 9.15(c)中给出的结构元来替代图 9.13(b)中给出的结构元，试讨论对提取边界的影响。

(b) 在式(9.5-2)给出的孔洞填充算法中，若使用元素值为 1、大小为 2×2 的结构元代替图 9.15(c)所示的结构元，请问会有什么影响?

9.23★(a) 针对图 9.16 中的自动化例子，请提出一种方法(使用 9.1 节到 9.5 节的任何技术)。可以假设球体互不接触，并且不接触图像的边界。

(b) 重复(a)，但允许球体以任意方式接触，包括接触边界。

★**9.24** 9.5.3 节中给出的提取连通分量的算法，要求在每个连通分量中有一个点是已知的，以便提取所有的点。假如您已有一幅包含任意数量(未知)连通分量的二值图像。请提出一种完全自动化地提取所有连通分量的过程。假设属于连通分量的点标记为 1，背景点标记为 0。

9.25 给出一个基于膨胀重建的、能提取一幅二值图像中的所有孔洞的表达式。

9.26 参考 9.5.9 节中的孔洞填充算法：

(a) 如果 f 的所有边界点都是 1，解释会发生什么情况。

(b) 如果(a)中的结果是您所希望的结果，请解释原因。如果不是，请解释如何改进该算法，以便算法能像您所期望的那样工作。

★**9.27** 如果结构元是值为 1 的单个点，解释二值图像的腐蚀和膨胀将会发生什么情况。对于您的回答，请给出原因。

9.28 如式(9.5-27)和 9.6.4 节中解释的那样，重建开操作可保持腐蚀后保留的图像分量的形状。请问重建闭操作的作用是什么?

★**9.29** 证明测地腐蚀和测地膨胀(见 9.5.9 节)关于补集对偶。也就是说，证明 $E_G^{(n)}(F)=[D_{G^c}^{(1)}[D_{G^c}^{(n-1)}(F^c)]]^c$，反之 $D_G^{(n)}(F)=[E_G^{(1)}[E_G^{(n-1)}(F^c)]]^c$。假定结构元关于其原点对称。

9.30 证明膨胀重建和腐蚀重建(见 9.5.9 节)关于补集对偶。也就是说，证明 $R_G^D(F)=[R_{G^c}^E(F^c)]^c$，反之亦然，$R_G^E(F)=[R_{G^c}^D(F^c)]^c$。假定结构元关于其原点对称。

★**9.31** 提出一个论据证明：

(a) $[(F\ominus nB)]^c=(F^c\oplus n\hat{B})$，其中 $(F\ominus nB)]$ 表示 B 对 F 的 n 次腐蚀。

(b) $[(F\oplus nB)]^c=(F^c\ominus n\hat{B})$。

9.32 证明二值重建闭操作与重建开操作关于补集对偶，即 $O_R^{(n)}(F)=[C_R^{(n)}(F^c)]^c$ 并且，类似地，$C_R^{(n)}(F)=[O_R^{(n)}(F^c)]^c$。假设结构元关于其原点对称。

9.33 证明下列灰度级形态学表达式的正确性。您可以假设 b 是一个平坦结构元。回忆一下，$f^c(x,y)=-f(x,y)$ 且 $\hat{b}(x,y)=b(-x,-y)$。

★(a) 腐蚀和膨胀的对偶性：$(f\ominus b)^c=f^c\oplus\hat{b}$ 和 $(f\oplus b)^c=f^c\ominus\hat{b}$。

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

★(c) $D_g^{(n)}(f)=[E_{g^c}^{(1)}[E_{g^c}^{(n-1)}(f^c)]]^c$ 和 $E_g^{(n)}(f)=[D_{g^c}^{(1)}[D_{g^c}^{(n-1)}(f^c)]]^c$。假设一个对称结构元。

(d) $R_g^D(f)=[R_{g^c}^E(f^c)]^c$ 和 $R_g^E(f)=[R_{g^c}^D(f^c)]^c$。

(e) $(f\ominus nb)^c=(f^c\oplus n\hat{b})$，其中，$(f\ominus nb)$ 表示 b 对 f 的 n 次腐蚀。还有，$[(f\oplus nb)]^c=(f^c\ominus n\hat{b})$。

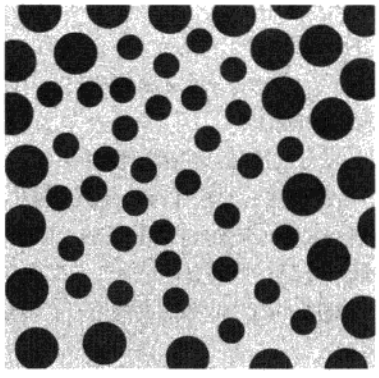
(f) $O_R^{(n)}(f)=[C_R^{(n)}(f^c)]^c$ 和 $C_R^{(n)}(f)=[O_R^{(n)}(f^c)]^c$。假设结构元关于其原点对称。

707 ~ 708

9.34 在图 9.43 中，建立两个不同纹理区域间的边界并不困难。考虑下图，它显示了一个由大圆区域包围的小圆区域。

(a) 用于生成图 9.43(d)的方法也可用于该图像吗？解释您的理由，包括为使该方法处理而需要做出的任何假设。

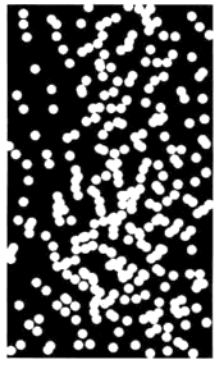
(b) 如果您的回答是肯定的，请画出边界的外表特征。



9.35 一幅灰度级图像 $f(x, y)$被非重叠的噪声尖刺所污染，它们可以用半径为 $R_{\min} \leqslant r \leqslant R_{\max}$，幅度为 $A_{\min} \leqslant a \leqslant A_{\max}$ 的较小的圆柱形干扰来建模。

★(a) 请开发一种清理该图像的形态学滤波方法。

(b) 重复(a)，但现在假设最多存在 4 个重叠的噪声尖刺。



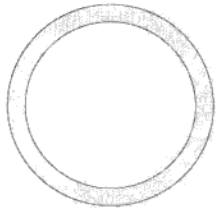
9.36 显微应用中一个预处理步骤是从两组或更多组重叠的类似颗粒(见右图)中分离出单个独立的一种圆颗粒。假设所有颗粒的大小相同，提出一种产生 3 幅图像的形态学算法，这 3 幅图像分别仅由如下物体组成：

★(a) 仅与图像边界融合在一起的颗粒。

(b) 仅彼此重叠的颗粒。

(c) 没有重叠的颗粒。

9.37 一家高技术制造工厂获得了一份制造形如下图的高精度垫圈的政府合同。合同要求使用一个图像系统来检测所有垫圈的形状。在合同文本中，形状检测是指检测垫圈内边缘和外边缘的偏差。您可以假定：(1) 一幅可接受的垫圈的“金”图像是可用的(关于该习题而言的一幅完美图像)；(2) 该系统中最终使用的成像和定位系统的精度可高到足以允许您忽略数字化和定位引起的误差。假设您被聘为该系统视觉检测部分的顾问。请基于形态学/逻辑操作，提出一种检测方案。请以方框图的形式给出您的答案。



708 ~ 709