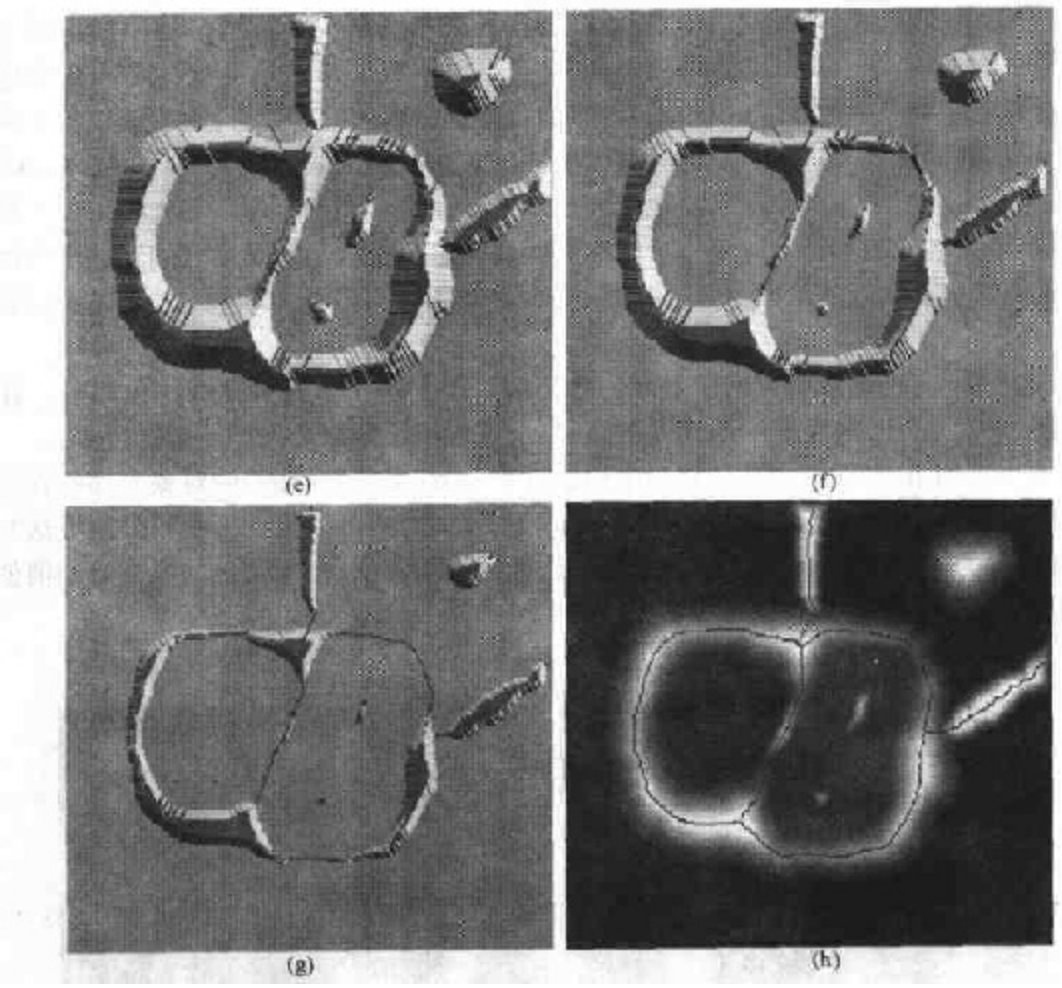

图 10.44(续) (e)进一步淹没的结果,(f)汇水盆地的水开始聚合(它们之间有一条短水坝),(g)长一些的水坝,(h)最后的分水线(分割)(由 CMM/Ecole des Mines de Paris 的 S. Beucher 博士提供)

10.5.2 水坝构造

在进行讨论之前,让我们考虑一下如何构造分水岭分割方法所需的水坝或分水线。水坝的构造是以二值图像为基础的,这种图像属于二维整数空间 Z^2(见 2.4.2 节)。构造水坝分离二元点集的最简单的方法是使用形态膨胀(见 9.2.1 节)。

图 10.45 说明了如何使用形态膨胀构造水坝的基本点。图 10.45(a)显示了两个汇水盆地的部分区域在淹没步骤的第 $n-1$ 步时的图像。图 10.45(b)显示了淹没的下一步(第 n 步)的结果。水已经从一个盆地溢出到另一个盆地,所以,必须建造水坝阻止这种情况的发生。为了与紧接着要介绍的符号相一致,令 M_1 和 M_2 表示在两个区域极小值中包含的点的坐标集合。然后,将处于汇水盆地中的点的坐标集合与这两个在溢出的第 $n-1$ 个阶段的最小值联系起来,并用 $C_{n-1}(M_1)$和 $C_{n-1}(M_2)$表示。这就是图 10.45(a)中的两个黑色区域。

令这两个集合的联合用 $C[n-1]$表示。图 10.45(a)中有两个连通分量(见 2.5.2 节关于连通分量的部分),而图 10.45(b)中只有一个连通分量。这个连通分量包含着前面的两个分量,用虚线表示。两个连通分量变成一个连通分量的事实说明两个汇水盆地中的水在淹没的

第 n 步聚合了。用 q 表示此时的连通分量。注意，第 $n-1$ 步中的两个连通分量可以通过使用“与”操作（$q \cap C[n-1]$）从 q 中提取出来。我们也注意到，属于独立的汇水盆地的所有点构成了一个单一的连通分量。

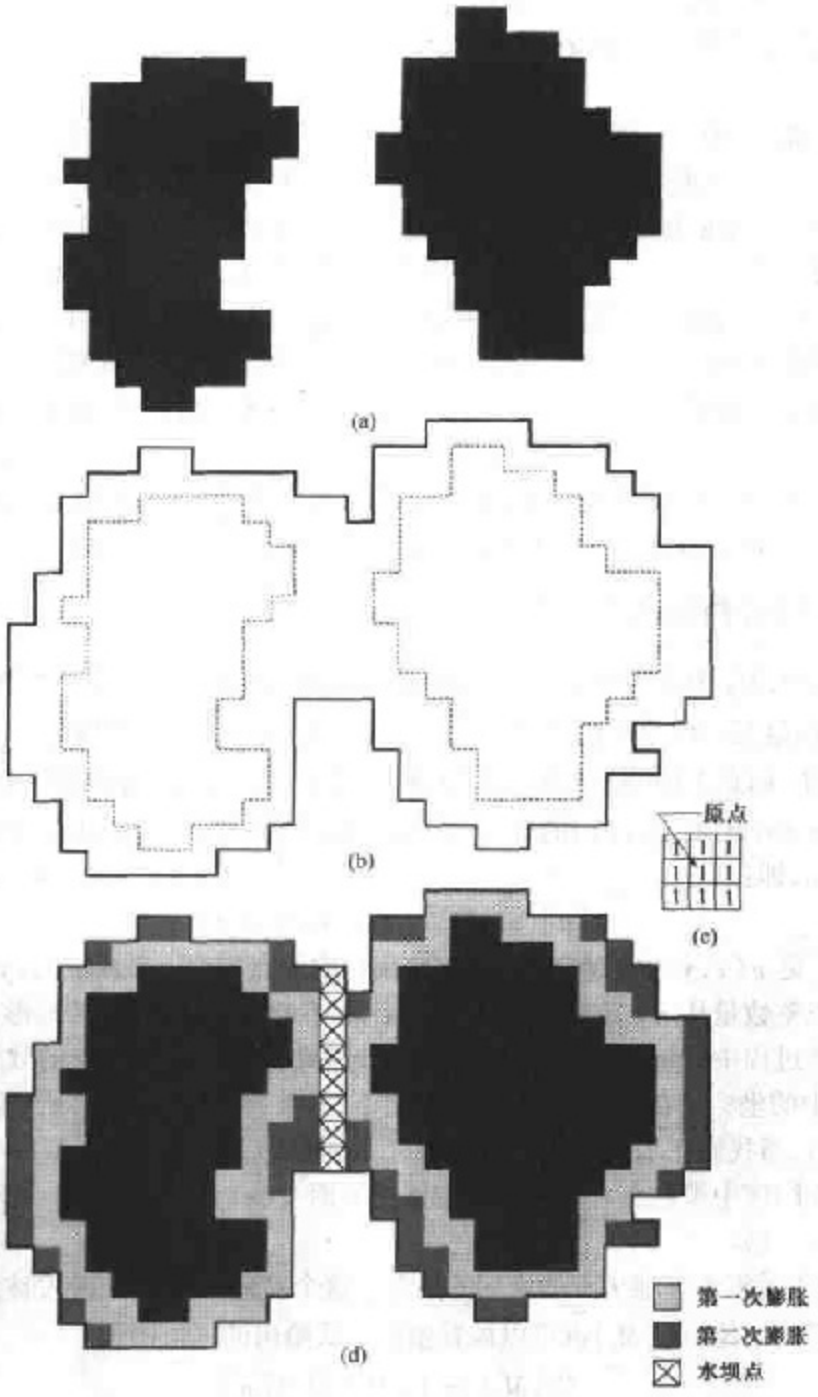

图 10.45　(a)在淹没的第 $n-1$ 个阶段淹没的汇水盆地的两个部分，(b)淹没的第 n 阶段，显示出两个盆地间的水已经溢出(为了显示清晰，水用白色表示而不是黑色)，(c)用于膨胀的结构元素，(d)扩展的结果和水坝的构造

假设图 10.45(a)中的每个连通分量通过使用图 10.45(c)中显示的结构元膨胀,在两个条件下:(1)膨胀受到 q 的约束(这意味着在膨胀的过程中结构化元素的中心只能定位于 q 中);并且(2)在引起集合聚合的那些点上不能执行膨胀(成为单一的连通分量)。图 10.45(d)显示首轮膨胀(浅灰色表示)使用了每个初始连通分量的边界。注意,在膨胀过程中每个点都满足条件(1)。条件(2)在膨胀处理中没有应用于任何的点;因此,每个区域的边界都进行了均匀的扩展。

在第二轮膨胀中(中等灰度表示),几个不满足条件(1)的点符合条件(2)时,得到图中显示的断开周界。很明显,只有满足上述两个条件的属于 q 中的点描绘了图 10.45(d)中交叉阴影线表示的一个像素宽度的连通路径。这条路径组成在淹没的第 n 个阶段我们希望得到的水坝。在这个淹没水平上,水坝的构造是由置所有刚好在这条路径上的点的值为比图像中灰度级的最大值还大的值完成的。所有水坝的高度通常设定为 1 加上图像中灰度级最大允许值。这样设定可以阻止在水位不断升高的情况下水越过部分水坝。应该特别注意到的是通过这一过程建立的水坝是连通分量,就是我们希望得到的分割边界。就是说,这种方法消除了分割线产生间断的问题。

尽管刚刚讨论的过程是用一个简单的例子说明的,但是处理更为复杂情况的方法是完全相同的,包括图 10.45(c)中显示的 3×3 对称结构元素的使用也是相同的。

10.5.3 分水岭分割算法

令 $M_1, M_2, \cdots, M_R$ 为表示图像 $g(x,y)$的局部最小值点的坐标的集合。如同在 10.5.1 节结尾说明的那样,这是一幅典型的梯度图像。令 $C(M_i)$为一个点的坐标的集合,这些点位于与局部最小值 M_i(回想无论哪一个汇水盆地内的点都组成一个连通分量)相联系的汇水盆地内。符号 min 和 max 代表 $g(x,y)$的最小值和最大值。最后,令 $T[n]$表示坐标(s,t)的集合,其中 $g(s,t)<n$,即:

$$T[n]=\{(s,t)\mid g(s,t)<n\} \tag{10.5.1}$$

在几何上,$T[n]$是 $g(x,y)$中的点的坐标集合,集合中的点均位于平面 $g(x,y)=n$ 的下方。

随着水位以整数量从 $n=\min+1$ 到 $n=\max+1$ 不断增加,图像中的地形会被水漫过。在水位漫过地形的过程中的每一阶段,算法都需要知道处在水位之下的点的数目。从概念上来说,假设 $T[n]$中的坐标处在 $g(x,y)=n$ 平面之下,并被"标记"为黑色,所有其他的坐标被标记为白色。然后,当我们在水位以任意增量 n 增加的时候,从上向下观察 xy 平面,会看到一幅二值图像。在图像中黑色点对应于函数中低于平面 $g(x,y)=n$ 的点。这种解释对于理解下面的讨论很有帮助。

令 $C_n(M_i)$表示汇水盆地中点的坐标的集合。这个盆地与在第 n 阶段被淹没的最小值有关。参考前一段的讨论,$C_n(M_i)$也可以被看做由下式给出的二值图像:

$$C_n(M_i)=C(M_i)\cap T[n] \tag{10.5.2}$$

换句话说,如果$(x,y)\in C(M_i)$且$(x,y)\in T[n]$,则在位置(x,y)有 $C_n(M_i)=1$。否则 $C_n(M_i)=0$。对于这个结果几何上的解释是很简单的。我们只需在水溢出的第 n 个阶段使用"与(AND)"算子将 $T[n]$中的二值图像分离出来即可。$T[n]$是与局部最小值 M_i 相联系的集合。

接下来,我们令 $C[n]$表示在第 n 个阶段汇水盆地被水淹没的部分的合集:

$$C[n] = \bigcup_{i=1}^{R} C_n(M_i) \tag{10.5.3}$$

然后令 $C[\max + 1]$为所有汇水盆地的合集：

$$C[\max + 1] = \bigcup_{i=1}^{R} C(M_i) \tag{10.5.4}$$

可以看出(习题 10.29)处于 $C_n(M_i)$和 $T[n]$中的元素在算法执行期间是不会被替换的，而且这两个集合中的元素的数目与 n 保持同步增长。因此，$C[n-1]$是集合 $C[n]$的子集。根据式(10.5.2)和式(10.5.3)，$C[n]$是 $T[n]$的子集，所以，$C[n-1]$是 $T[n]$的子集。从这个结论我们得出重要的结果：$C[n-1]$中的每个连通分量都恰好是 $T[n]$的一个连通分量。

找寻分水线的算法开始时设定 $C[\min + 1] = T[\min + 1]$。然后算法进入递归调用，假设在第 n 步时，已经构造了 $C[n-1]$。根据 $C[n-1]$求得 $C[n]$的过程如下：令 Q 代表 $T[n]$中连通分量的集合。然后，对于每个连通分量 $q \in Q[n]$，有下列 3 种可能性：

(a) $q \cap C[n-1]$为空。

(b) $q \cap C[n-1]$包含 $C[n-1]$中的一个连通分量。

(c) $q \cap C[n-1]$包含 $C[n-1]$多于一个的连通分量。

根据 $C[n-1]$构造 $C[n]$取决于这 3 个条件。当遇到一个新的最小值时符合条件(a)，则将 q 并入 $C[n-1]$构成 $C[n]$。当 q 位于某些局部最小值构成的汇水盆地中时，符合条件(b)，此时将 q 合并入 $C[n-1]$构成 $C[n]$。当遇到全部或部分分离两个或更多汇水盆地的山脊线的时候，符合条件(c)。进一步的注水会导致不同盆地的水聚合在一起，从而使水位趋于一致。因此，必须在 q 内建立一座水坝(如果涉及多个盆地就要建立多座水坝)以阻止盆地内的水溢出。正如前一节中的解释，当用 3×3 个 1 的结构元素膨胀 $q \cap C[n-1]$并且需要将这种膨胀限制在 q 内时，一条一个像素宽度的水坝是能够构造出来的。

通过使用与 $g(x,y)$中存在的灰度级值相对应的 n 值，可以改善算法效率；根据 $g(x,y)$的直方图，可以确定这些值及其最小值和最大值。

例 10.18　分水岭分割算法的说明

分别考虑图 10.46(a)和(b)中显示的图像和它的梯度。应用刚才讨论的分水岭算法得到图 10.46(c)中显示的梯度图像的分水线(白色路径)。这些分割的边界叠加在原图上示于图 10.46(d)。如在本节中开始时所注意到的，分割边界具有那些被连接起来的路径的重要性质。

10.5.4　应用标记

直接以前一节中讨论的形式使用分水岭分割算法通常会由于噪声和其他诸如梯度的局部不规则性的影响造成过度分割。如图 10.47 所示，过度分割足以令应用算法得到的结果变得毫无用处。此时，过度分割意味着分割区域过多。一个较实际的解决方案是通过合并预处理步骤来限制允许存在的区域的数目，这些预处理步骤是为将附加知识应用于分割过程而设置的。

用于控制过度分割的方法是以标记的概念为基础的。一个标记是属于一幅图像的连通分量。我们有与重要对象相联系的内部标记，还有同背景相联系的外部标记。选择标记的典型过程包括两个主要步骤：(1)预处理；(2)定义一个所有标记必须满足的准则集合。为了对此进

行说明,再次考虑图 10.47(a)。导致图 10.47(b)中过度分割结果的一部分原因是大量隐含的最小值。由于这些区域的尺寸很小,所以这些最小值中有很多是不相关的细节。在前面的讨论中,已经不止一次地提到,将很小的细节对于图像的影响降至最低的有效方法是用一个平滑滤波器对图像进行过滤。在这种特殊情况下,这是一种合适的预处理方案。

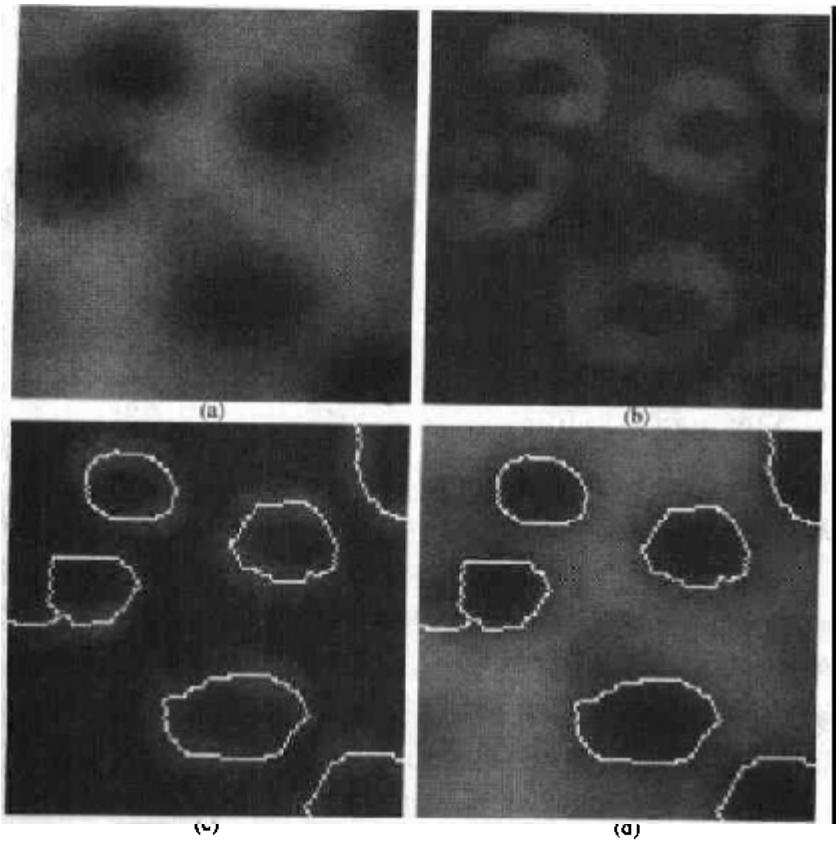

图 10.46 (a)带有斑点的图像,(b)梯度图像,(c)分水线,(d)叠加于原图中的分水线(由 CMM/Ecole des Mines de Paris 的 S. Beucher 博士提供)

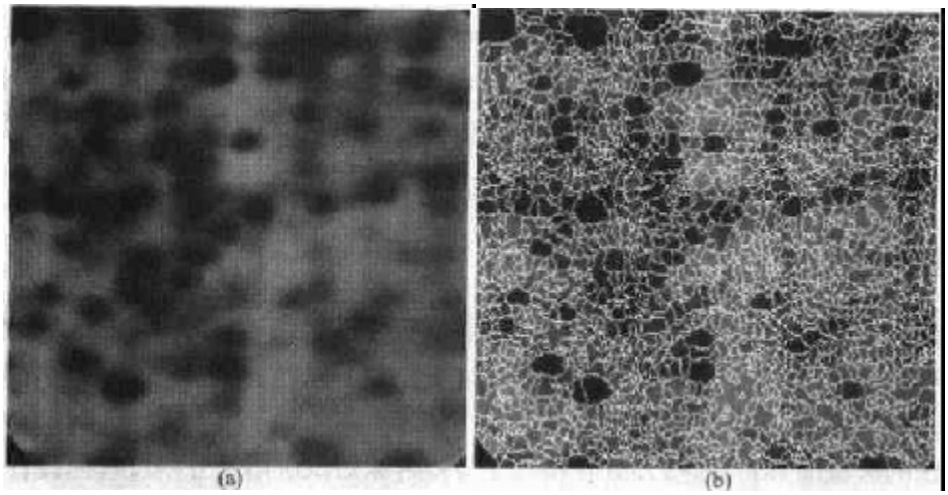

图 10.47 (a)电泳现象的图像,(b)对梯度图像使用分水岭分割算法得到的结果,过度分割现象很明显(由 CMM/Ecole des Mines de Paris 的 S. Beucher 博士提供)

假设在此时将内部标记定义为(1)被更高"海拔"点包围起来的区域;(2)区域中的点组成一个连通分量;并且(3)所有属于这个连通分量的点具有相同的灰度级值。在图像经过平滑处理之后,满足这些定义的内部标记以图 10.48(a)中浅灰色、斑点状区域表示。下一步,对平滑处理后的图像使用分水岭算法,并限制这些内部标记只能是允许的局部最小值。图 10.48(a)显示了得到的分水线。将这些分水线定义为外部标记。注意,沿着分水线的点是很好的背景候选点。因为它们经过相邻的标记之间的最高点。

图 10.48(a)中显示的外部标记有效地将图像分割成不同区域。每个区域包含一个惟一的内部标记和部分背景。问题是因此变为将每个这样的区域一分为二:单一的对象和它的背景。我们对这个简单的问题能够应用多种在本章前面讨论过的分割技术。另一种简单的方法是对每个单独的区域使用分水岭分割算法。就是说,我们只求得平滑后的图像的梯度[如图 10.46(b)],然后约束算法只对包含特定区域中标记的分水岭进行操作。使用这种方法得到的结果显示于图 10.48(b)中。相对于图 10.47(b)的改善之处是显而易见的。

标记的选择可以用基于灰度级值和连通性的简单的过程分类,如刚才所说明的那样,更复杂的描述涉及尺寸、形状、位置、相对距离、纹理内容等等(见第 11 章关于描绘子的部分)。关键是标记的使用给我们带来关于分割问题的某种带有先验性质的知识。应该提醒读者注意的是:人类经常以先验知识用日常的视觉辅助,进行各种分割和更高级的工作,最熟悉的例子就是在阅读上下文时所用到的方法。因此,分水岭分割方法提出了一种能有效使用这类知识的机制。这是这种方法的一个突出优点。

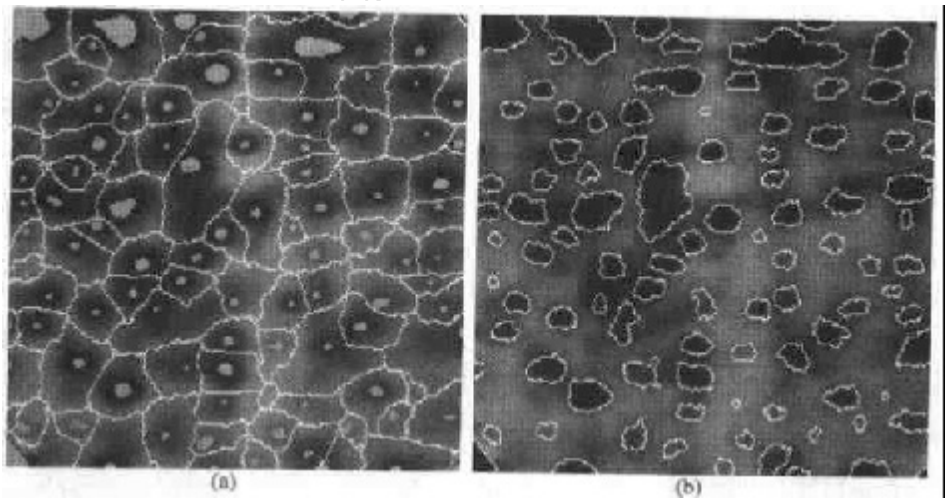

图 10.48　(a)显示有内部标记(浅灰色区域)和外部标记(分水线)的图像,(b)分割的结果。注意其相对于图10.47(b)的改进之处(由CMM/EcoledesMinesdeParis的S. Beucher博士提供)

10.6　分割中运动的应用

运动是人类和动物使用的用于将重要的对象从不相关的背景细节中提取出来的强有力的提示。在成像应用领域,运动是由于感觉系统和视觉场景之间的相对位移而形成的,比如机器人应用、自主导航和动态视觉分析等等方面都是如此。在下面的各节中,我们将考虑运动在空间和频域分割中的运用。

10.6.1 空间技术

基本方法

分别检测两帧图像 $f(x,y,t_i)$和 $f(x,y,t_j)$之间在时间 t_i 和 t_j 时的变化的最简单的方法是将两幅图像逐个像素进行对比。这样做的过程是形成一幅差值图像。假设我们有一幅仅包含不动部分的参考图像。将这幅图像和随后生成的同一场景但包含一个运动对象的图像相比较,去掉两幅图像中的固定元素,余下的与非固定部分相对应的非零项就是两幅图像的差值。

两幅图像在时间 t_i 和 t_j 时的差值图像可以定义为:

$$d_{ij}(x,y)=\begin{cases}1 & |f(x,y,t_i)-f(x,y,t_j)|>T\\ 0 & \text{其他}\end{cases} \tag{10.6.1}$$

这里 T 是一个特定的门限。注意,如果在两幅图像间灰度差在其坐标上有相当的不同时,$d_{ij}(x,y)$具有值 1,差异程度取决于事先确定的门限 T。假设所有的图像具有同样的尺寸大小。最后,注意到式(10.6.1)中的坐标(x,y)的值跨越了这些图像的尺寸,所以差值图像 $d_{ij}(x,y)$与上述这些序列中的图像大小相同。

在动态图像处理过程中,$d_{ij}(x,y)$中所有值为 1 的像素被认为是对象运动的结果。这种方法只有在两幅图像都做了空间配准且在 T 值设定的范围内亮度相对较为稳定的情况下才是合适的。实际上,在 $d_{ij}(x,y)$中的 1 值项常常是由于噪声造成的。很典型的,在差异图像中,这些点是孤立的,去除它们的简单方法是在 $d_{ij}(x,y)$中构造 1 的 4 连通或 8 连通区域,并且忽略所有输入项少于预定数目的区域。尽管这种方法会忽略掉小的和/或慢速运动的对象,但这种方法改进了在差值图像中保留运动结果的几率。

差异积累

当试图从一图像序列中将运动的部分分离出来时,源自噪声的孤立项并不是一个无足轻重的问题。尽管通过门限处理的连通性分析可以减少这类项的数目,但这类过滤操作也会去掉前一节中提到的小的或慢速运动的对象。处理这一问题的方法是考虑几帧图像中同一像素的变化,这样要将"记忆"引入这一处理过程。其所包含的思想是忽略在序列帧上零星出现的变化并将其归于随机噪声的影响。

考虑一个图像帧序列 $f(x,y,t_1),f(x,y,t_2),\cdots,f(x,y,t_n)$,并令 $f(x,y,t_1)$为基准图像。一幅累积差异图像(ADI)是由将基准图像和图像序列中后续图像进行对比得到的。当基准图像和序列中图像之间在某个像素位置上出现一次差异就令一个计数器计数一次,这种计数器在积累差异图像的每个像素的位置上都有一个。因此,当第 k 帧图像与基准图像相比较时,积累差异图像中一个给定像素的输入项给出在此位置上对应的像素与基准图像中同一位置的像素间灰度级变化的次数。差异是设定好的,例如通过使用式(10.6.1)进行。

通常考虑三种类型的积累差异图像是有用的:绝对 ADI、正 ADI 和负 ADI。假设运动对象的灰度值大于背景的灰度值,这三种 ADI 可以被定义如下。令 $R(x,y)$表示基准图像,且为了简化符号,令 k 表示 t_k,则有 $f(x,y,k)=f(x,y,t_k)$。假设 $R(x,y)=f(x,y,1)$,然后记住作为计数的ADI值,并对所有 $k>1$,对(x,y)的所有相关的值定义如下:

$$A_k(x,y) = \begin{cases} A_{k-1}(x,y) + 1 & |R(x,y) - f(x,y,k)| > T \\ A_{k-1}(x,y) & \text{其他} \end{cases} \tag{10.6.2}$$

$$P_k(x,y) = \begin{cases} P_{k-1}(x,y) + 1 & [R(x,y) - f(x,y,k)] > T \\ P_{k-1}(x,y) & \text{其他} \end{cases} \tag{10.6.3}$$

和

$$N_k(x,y) = \begin{cases} N_{k-1}(x,y) + 1 & [R(x,y) - f(x,y,k)] < -T \\ N_{k-1}(x,y) & \text{其他} \end{cases} \tag{10.6.4}$$

这里 $A_k(x,t)$, $P_k(x,y)$和 $N_k(x,y)$分别为遇到图像序列中的第 k 幅图像后的绝对 ADI、正 ADI 和负 ADI。

这些 ADI 是从所有计数值为零时开始这一点不难理解。同时要注意ADI如序列中的图像一样大小是相同的。如前面提到的那样，序列中的图像都被假设为同样大小。最后，我们注意到如果背景像素的灰度级大于运动对象的灰度级，则式(10.6.3)和式(10.6.4)中的不等式的顺序以及门限的符号是相反的。

例 10.19　绝对、正和负值积累差异图像的计算

图 10.49 显示了以 3 个亮度图像显示的 ADI，图中的矩形物体，大小为 75 × 50 像素，其中以每帧 $5\sqrt{2}$个像素的速度向东南方向运动。图像的大小为 256 × 256 像素。我们注意到下列几点：(1)正ADI的非零区域等于运动对象的大小；(2)正ADI的位置对应于基准帧中运动对象的位置；(3)当运动对象在基准帧中相对于同一对象完全被移动时，正ADI停止计数；(4)绝对 ADI 包括正的和负的 ADI；(5)运动对象的速度和方向可以由绝对的和负的 ADI 决定。

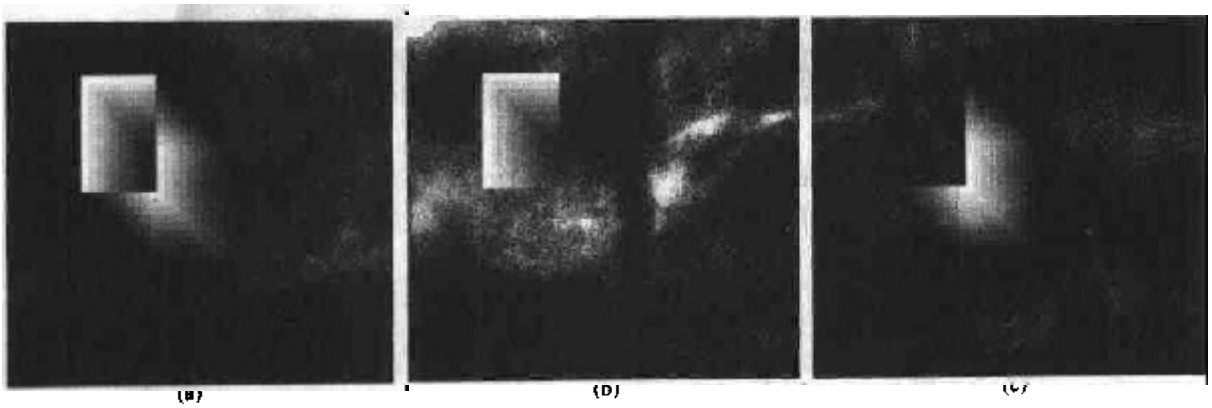

图 10.49　带有一个向东南方向移动的矩形目标的 ADI。(a)绝对 ADI，(b)正 ADI，(c)负 ADI

设置基准帧

前两节中讨论的技术成功的关键在于找到一幅可以与后续图像进行比较的基准图像。如上述指出的那样，在动态成像问题中两幅图像的差异倾向于消除图像中所有固定部分，而只留下对应于噪声和运动对象的图像元素。噪声问题可以通过使用前面提到的过滤方法或通过前面一节中提到的构造积累差异图像来解决。

实际上，用固定不动的元素构造一幅基准图像并不总是可行的，所以有必要根据一组包含

一个或多个运动对象的图像构造一个基准。特别是在描述变化频繁的场景或图像需要频繁更新的情况。生成基准图像的过程如下所述。考虑将图像序列中的第一幅图像作为基准图像。当一个非恒定的图像分量完全离开了它在基准帧中所处的位置时,当前帧中对应的背景可以被复制到在基准帧中由运动对象占据的位置。当所有的运动对象都离开了它们原先的位置时,就生成了只包含恒定分量的基准图像。正如前节所说明的那样,可以通过监控正的ADI中产生的变化完成对象的移动。

例 10.20 构造一个基准图像

图 10.50(a)和(b)显示了两幅关于一个交通十字路口的图像帧。第一幅图像被作为基准,第二幅在一段时间后描绘了同一场景。目的是去掉基准图像中主要的运动对象,生成一幅静态图像。尽管还有其他更小的运动对象,但主要的运动特征是十字路口上从左到右运动的汽车。为了说明,我们将注意力主要放在这个对象上。通过监视正的ADI的变化,可以如前所解释的那样确定运动对象的初始位置。一旦确定这个区域被此对象占据,可以通过剪影法将对象从图像中除去。通过观察图像序列中正的ADI停止变化的那一帧图像,可以从这幅图像中复制在初始帧中原来由运动对象占据的区域。然后这个区域就被贴到被挖去对象的图像中,从而恢复这个区域的背景。如果对所有运动的对象都进行这样的操作,结果正如在前两节中解释的那样,得到了一幅可以和后续帧相比,对于运动检测只包含静态图像部分的基准图像。此处,图 10.50(c)中显示了去掉向东运动的汽车后得到的结果。

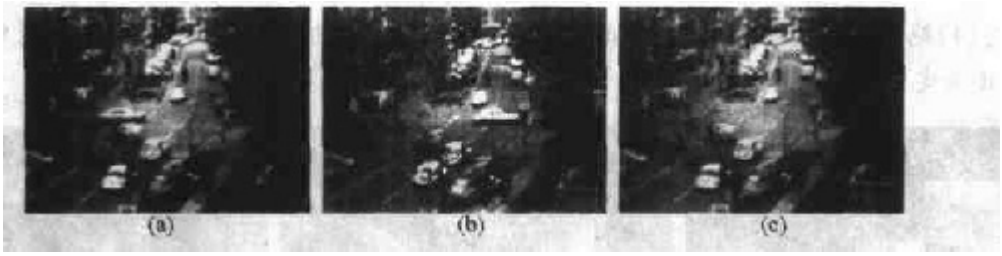

图 10.50 构造一幅静态基准图像,在图像序列中的(a)和(b)两帧图像,(c)从(a)中去掉向东运动的汽车并根据(b)中对应的区域恢复背景(Jain和Jain)

10.6.2 频域技术

本节中,我们考虑通过傅里叶变换确定对运动的估计的问题。考虑一个序列 $f(x,y,t)$,$t=0,1,\cdots,K-1$,由一台固定的照相机拍摄 K 帧大小为 $M\times N$ 的数字图像。通过假设所有帧有零亮度均一的背景开始讨论。一个例外是单一的,1 个像素的对象,它具有单位亮度并以恒定速度运动。假设对第一帧($t=0$),将图像平面投影到 x 轴上;即,将每一列像素的亮度相加。这样做会产生一个带有 M 个零值项的一维阵列,而对象的投影位置则是非零项。用 $\exp[j2\pi a_1 x\Delta t]$,$x=0,1,2,\cdots,M-1$,乘以阵列中的每个分量,包括在此帧所在的那一瞬间位于坐标(x',y')的对象在内,生成一个等于 $\exp[j2\pi a_1 x'\Delta t]$的和。在这个表示符号中,$a_i$ 是一个正整数,Δt 是不同帧之间的时间间隔。

假设在第二帧中($t=1$),对象运动到坐标$(x'+1,y')$;即,对象在平行于 x 轴的方向上移动了一个像素的位置。然后重复上一节中提到的投影过程,得到和 $\exp[j2\pi a_1(x'+1)\Delta t]$。如

果对象继续每一帧移动一个像素的位置，在任何整数的瞬间，结果是 $\exp[j2\pi a_1(x'+t)\Delta t]$，使用欧拉公式，可以用如下形式表示：

$$e^{j2\pi a_1(x'+t)\Delta t} = \cos[2\pi a_1(x'+t)\Delta t] + j\sin[2\pi a_1(x'+t)\Delta t] \tag{10.6.5}$$

$t=0,1,\cdots,K-1$。换句话说，这一过程得到一个频率为 a_1 的复正弦曲线。如果对象以每帧 v_1 个像素的速度运动(在 x 轴方向上)，正弦曲线的频率就是 $v_1 a_1$。因为 t 在 0 到 $K-1$ 之间以整数增量变化，限制 a_1 只取整数值，使复正弦曲线的离散傅里叶变换具有了两个波峰(一个位于频率 $v_1 a_1$ 的位置上，另一个位于 $K - v_1 a_1$ 的位置)。后一个波峰可能会被忽略掉。它是由于离散傅里叶变换的对称性而生成的，这一点在 4.6 节中进行了讨论。因此，在傅里叶频谱中找到一个波峰得到 $v_1 a_1$。用 a_1 除这个量得到 v_1，它就是在 x 轴方向上的速度分量，这里假设帧频是已知的。同样可以通过这种方法得到 y 轴上的速度分量 v_2。

没有运动发生的帧序列产生同样的指数项，这些帧的傅里叶变换结果在频率为 0 处(一个单一的直流项)由单峰组成。所以，由于迄今为止讨论的都是线性操作，在静态背景中涉及一个或多个运动对象的一般情况下，对应于静态图像分量的直流的傅里叶变换结果为一个波峰，而在与对象的速度成比例的位置上有多个波峰。

上述这些概念可以总结如下。对一个有 K 幅大小为 $M \times N$ 的数字图像的序列，在任何值为整数的瞬时点上，图像加权投影于 x 轴上的和是：

$$g_x(t,a_1) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y,t)e^{j2\pi a_1 x\Delta t} \quad t = 0,1,\cdots,K-1 \tag{10.6.6}$$

同样，y 轴上的投影之和为：

$$g_y(t,a_2) = \sum_{y=0}^{N-1}\sum_{x=0}^{M-1} f(x,y,t)e^{j2\pi a_2 y\Delta t} \quad t = 0,1,\cdots,K-1 \tag{10.6.7}$$

这里，a_1 和 a_2 是正整数。

式(10.6.6)和式(10.6.7)的一维傅里叶变换分别为：

$$G_x(u_1,a_1) = \frac{1}{K}\sum_{t=0}^{K-1} g_x(t,a_1)e^{-j2\pi u_1 t/K} \quad u_1 = 0,1,\cdots,K-1 \tag{10.6.8}$$

和

$$G_y(u_2,a_2) = \frac{1}{K}\sum_{t=0}^{K-1} g_y(t,a_2)e^{-j2\pi u_2 t/K} \quad u_2 = 0,1,\cdots,K-1 \tag{10.6.9}$$

实际上，如 4.6 节中讨论的那样，这些变换的分量是应用 FFT 算法进行计算的。

频率-速度关系是：

$$u_1 = a_1 v_1 \tag{10.6.10}$$

和

$$u_2 = a_2 v_2 \tag{10.6.11}$$

在这个公式中，速度单位是用每总帧时间内的像素数来表示的。例如，$v_1 = 10$ 可以解释为在 K 幅图像帧中运动对象经过了 10 像素。对于匀速摄取的图像帧，真实的物理速度取决于帧频和像素之间的距离。因此，如果 $v_1 = 10$，$K = 30$，帧频是每秒两幅图像，像素之间的距离是 0.5 m，则在 x 轴方向上的真实的物理速度是：

$$v_1 = (10\text{ 像素})(0.5\text{ m/ 像素})(2\text{ 帧/s})/(30\text{ 帧}) = 1/3\text{ m/s}$$

速度的 x 分量的符号通过计算下式得到：