第3章　典型分割算法

图像分割的技术一直在不断发展，已有很多种方法提了出来。本章仅介绍几种具有比较特殊思路的典型方法作为例子。在2.1节中曾将图像分割技术分为4大类，本章以下各节介绍的内容也分别对应这4大类分割技术。

根据上述的讨论，本章各节将安排如下。

3.1节先介绍利用二阶导数对兴趣点的检测，再讨论既可用于边缘检测也可用于角点检测的最小核同值区算子的具体方法和特点，最后介绍哈里斯兴趣点算子，它们都属于并行边界类技术。

3.2节介绍一种结合了图像整体信息和局部信息，并利用了图结构的特殊方法——图割方法，它属于串行边界类技术。

3.3节介绍3种不同的属于并行区域类技术的方法，分别是借助小波变换的多分辨率特性阈值化方法、借助过渡区的阈值化方法和借助均移确定聚类的分割方法。

3.4节介绍一种结合了图像整体信息和局部信息的特殊方法——基于分水岭的方法，除了基本原理还讨论了对基本算法的改进和扩展。虽然分水岭本身对应边界，但这种方法更多地是利用边界所包围的区域性质来工作的，所以是一种串行区域类技术。

3.1　兴趣点检测

兴趣点或感兴趣点泛指图像中或目标上具有特定几何性质或属性性质的点，例如角点、拐点、梯度极值点等。对兴趣点的检测有很多方法，下面先讨论利用二阶导数来检测角点，再介绍两种比较有特色的检测算子，可检测多种类型的兴趣点。

3.1.1　二阶导数检测角点

如果一个像素在其小邻域中具有两个明显不同的边缘方向则常可被看作一个角点（或者说小邻域中有两个边缘段，其朝向更偏于互相垂直），也有人将局部曲率（离散曲率计算可参见10.4节）较大的边缘点看作**角点**。典型的角点检测器仍多基于像素灰度差。

利用二阶导数算子检测角点的原理与利用一阶导数算子检测边缘有些类似。在2-D图像中，由各个二阶导数组成的对称矩阵可写成（下标指示偏导数的方向）：

$$\boldsymbol{I}_{(2)}=\begin{bmatrix} I_{xx} & I_{xy} \\ I_{yx} & I_{yy} \end{bmatrix},\quad I_{xy}=I_{yx} \tag{3.1.1}$$

它给出在被检测点（原点）的局部曲率的信息。上述矩阵有两个特征值，可考虑它们的3种组合情况：①两个特征值都很小，表示被检测点局部的邻域平坦，检测点非边缘点或角点；

②两个特征值一个很小而另一个很大，表示被检测点局部的邻域呈脊线状，沿一个方向平坦而沿另一个方向变化迅速，被检测点为边缘点；③两个特征值都很大，表示被检测点为角点。这种检测方法对平移和旋转变换不敏感，也对光照和视角变化有一定的稳定性。

如果旋转坐标系统，可将 $\boldsymbol{I}_{(2)}$ 变换成对角形式：

$$\widetilde{\boldsymbol{I}}_{(2)} = \begin{bmatrix} I_{\tilde{x}\tilde{x}} & 0 \\ 0 & I_{\tilde{y}\tilde{y}} \end{bmatrix} = \begin{bmatrix} K_1 & 0 \\ 0 & K_2 \end{bmatrix} \tag{3.1.2}$$

此时的二阶导数矩阵给出了在原点的主曲率，即 K_1 和 K_2。

再回到如 $\boldsymbol{I}_{(2)}$ 的矩阵，其秩和行列式都是旋转不变的。进一步可分别得到拉普拉斯值和海森值：

$$\text{Laplacian} = I_{xx} + I_{yy} = K_1 + K_2 \tag{3.1.3}$$

$$\text{Hessian} = \det(\boldsymbol{I}_{(2)}) = I_{xx}I_{yy} - I_{xy}^2 = K_1K_2 \tag{3.1.4}$$

基于一个函数的二阶偏导数构成的方阵，**拉普拉斯算子**和**海森算子**分别计算拉普拉斯值和海森值。前者对边缘和直线都给出较强的响应，所以不太适合用来检测角点。后者描述了函数的局部曲率，虽然对边缘和直线没有响应，但是在角点的邻域有较强的响应，所以比较适合用来检测角点。不过海森算子在角点自身处响应为零而且在角点两边的符号是不一样的，所以需要较复杂的分析过程以确定角点的存在性并准确地对角点定位。为避免这个复杂的分析过程，可先计算曲率与局部灰度梯度的乘积：

$$C = Kg = K\sqrt{I_x^2 + I_y^2} = \frac{I_{xx}I_y^2 - 2I_{xy}I_xI_y + I_{yy}I_x^2}{I_x^2 + I_y^2} \tag{3.1.5}$$

再沿边缘法线方向利用非最大消除来确定角点的位置。

3.1.2 最小核同值区算子

最小核同值区(**SUSAN**)算子是一种很有特色的检测算子[Smith 1997]，它既可以检测边缘点，也可以检测角点。

1. 核同值区

先借助图 3.1.1 来解释检测的原理。这里设有一幅图像，取其中一个矩形区域，其上部为亮区域，下部为暗区域，分别代表背景和目标。现在考虑有一个圆形的**模板**(其大小由模板边界所限定)，其中心称为“核”，用“+”表示。图 3.1.1 中给出该模板放在图像中 6 个不同位置的示意情况，从左边数过去，第 1 个模板全部在亮区域，第 2 个模板大部在亮区域，第 3 个模板一半在亮区域(另一半在暗区域)，第 4 个模板大部在暗区域，第 5 个模板全部在暗区域，第 6 个模板的 1/4 在暗区域。

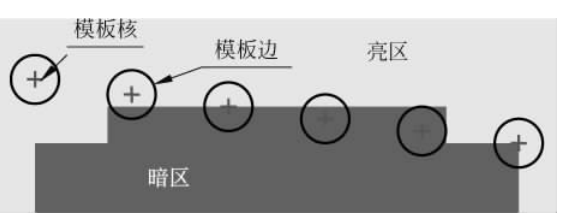

图 3.1.1 圆形模板在图像中的不同位置

如果将模板中各个像素的灰度都与模板中心核像素的灰度进行比较，那么就会发现总有一部分模板像素的灰度与核像素的灰度相同或相似。这部分区域可称为**核同值区**(**USAN**)，即与核灰度具有相同数值的区域。核同值区包含了很多与图像结构有关的信息。利用这种区域的尺寸、重心、二阶矩等各种特征可以帮助检测图像中的边缘和角点。从图 3.1.1 可见，当核像素处在图像中的灰度一致区域时，核同值区的面积会达到最大，第 1 个模板和第 5 个模板就属于这种情况。在第 2 个模板和第 4 个模板的情况下，核同值区的面积超过一半。这个面积当核处在直边缘处约为最大值的一半，第 3 个模板就属于这种情况。这个面积当核处在角点位置时更小，约为最大值的 1/4，第 6 个模板就属于这种情况。

利用核同值区的面积具有上述变化的性质可检测边缘或角点。具体说来，核同值区面积较大(超过一半)时表明核像素处在图像中的灰度一致区域，在模板核接近边缘时减少为一半，而在接近角点时减少得更多，即在角点处取得最小值(约 1/4)。如果将核同值区面积的倒数作为检测的输出，则可以通过计算面积极大值来方便地确定角点的位置。

由上讨论可见，使用核同值区面积作为特征可起到增强边缘和角点的效果。基于这种模板的检测方式与其他常用的检测方式有许多不同之处，最明显的就是不需要计算微分因而对噪声不很敏感。

2. 最小核同值区算子角点检测

在核同值区区域的基础上可讨论最小核同值区算子。最小核同值区算子采用圆形模板来得到各向同性的响应。在数字图像中，圆可用一个含有 37 个像素的模板来近似，见图 3.1.2。这 37 个像素排成 7 行，分别有 3,5,7,7,7,5,3 个像素。这相当于一个半径为 3.4 个像素的圆。如考虑到计算量，也有用简单的 3×3 模板来粗略近似圆形模板的。

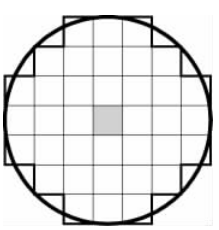

图 3.1.2　37 个像素的模板

设模板用 $N(x,y)$表示，将其依次放在图像中每个点的位置。在每个位置，将模板内每个像素的灰度值与核的灰度值进行比较，计算其响应：

$$C(x_0,y_0;x,y)=\begin{cases}1 & \text{当}\ |f(x_0,y_0)-f(x,y)|\leqslant T\\ 0 & \text{当}\ |f(x_0,y_0)-f(x,y)|>T\end{cases} \tag{3.1.6}$$

式中，(x_0,y_0)是核在图像中的位置坐标；(x,y)代表模板 $N(x,y)$中除核以外的其他位置；$f(x_0,y_0)$和 $f(x,y)$分别是在(x_0,y_0)和(x,y)处像素的灰度；T 是一个灰度差的阈值；函数 $C(\cdot;\cdot)$代表输出的比较结果。一般当检测到角点时，该函数类似一个门函数，如图 3.1.3 所示，设这里的阈值 T 取为 10。

上述比较计算要对模板中的每个像素进行，由此可得到一个输出的**游程和**：

$$S(x_0,y_0)=\sum_{(x,y)\in N(x_0,y_0)}C(x_0,y_0;x,y) \tag{3.1.7}$$

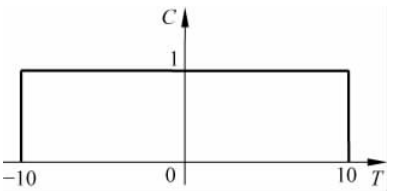

图 3.1.3　函数 $C(\cdot\,;\,\cdot)$示例

这个总和其实就是核同值区中的像素个数，或者说它给出了核同值区的面积。由上讨论可见，这个面积在角点处会达到最小。结合式(3.1.6)和式(3.1.7)可知，阈值 T 既可用来帮助检测核同值区面积的最小值，也可以确定可消除的噪声的最大值。

实际应用最小核同值区算子时，需要将游程和 S 与一个固定的几何阈值 G 进行比较以做出判断。该阈值可设为 $3S_{\max}/4$，其中 $S_{\max}$ 是 S 所能取得的最大值(对 37 个像素的模板，最大值为 36，所以 $3S_{\max}/4=27$)。初始的边缘响应 $R(x_0,y_0)$根据下式得到：

$$R(x_0,y_0)=\begin{cases}G-S(x_0,y_0) & S(x_0,y_0)<G\\ 0 & \text{其他}\end{cases} \tag{3.1.8}$$

上式是根据核同值区原理获得的，即核同值区的面积越小，边缘的响应就越大。

当图像中没有噪声时，完全不需要几何阈值。但当图像中有噪声时，将阈值 G 设为 $3S_{\max}/4$ 可给出最优的噪声消除性能。考虑有一个垂直的阶跃状边缘，则 S 的值总会在其某一边小于 $S_{\max}/2$。如果边缘是弯曲的，小于 $S_{\max}/2$ 的 S 值会出现在凹的一边。如果边缘不是理想的阶跃状边缘(有一定坡度)，S 的值会更小(R 的值会更大)，这样边缘检测不到的可能性更小。

上面介绍的方法一般已可以给出相当好的结果，但一个更稳定的计算 $C(\cdot\,;\,\cdot)$的公式是

$$C(x_0,y_0;x,y)=\exp\left\{-\left[\frac{f(x_0,y_0)-f(x,y)}{T}\right]^2\right\} \tag{3.1.9}$$

这个公式对应的曲线如图 3.1.4 中的曲线 b(曲线 a 对应式(3.1.6))所示。可见，式(3.1.9)给出了式(3.1.6)的一个平滑版本。它允许像素的灰度有一定的变化而不会对 $C(\cdot\,;\,\cdot)$造成太大的影响。

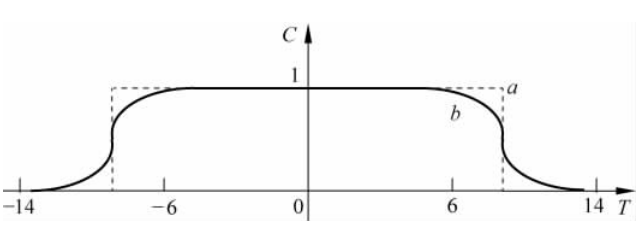

图 3.1.4　不同的 $C(\cdot\,;\,\cdot)$函数曲线示例

例 3.1.1　角点检测示例

图 3.1.5 给出两个用最小核同值区算子检测图像中角点得到的结果。

图 3.1.5　用最小核同值区算子检测到的角点　□

3. 边缘方向的检测

在对边缘的检测中，很多情况下不仅需要考虑边缘的强度也需要考虑边缘的方向。首先，如果要除去非最大边缘值，就必须要借助边缘的方向。另外，如果需要确定边缘的位置到亚像素精度，也常常需要利用边缘方向的信息。最后，许多应用中常把边缘的位置、强度和方向结合使用。对大多数现有的边缘检测算子，边缘方向是借助边缘强度来确定的。根据核同值区的原理确定边缘方向需根据边缘点的种类采用不同的方法。

根据核同值区的特点，可将边缘点分成两类。参见图 3.1.6，其中左图给出图像中一个典型的直线边缘。图中方格里的阴影用来近似地区分具有不同灰度的像素。对 3 个感兴趣点的局部核同值区分别显示在右边的 3 个 3×3 模板中。其中“○”代表核同值区的重心，而“＋”代表模板的核。区域 A 和 B 中都是同一类的边缘点，边缘都通过核同值区的重心，只是模板的核分别落在了边缘的两边。区域 C 中是另一类边缘点，这里模板的核与核同值区的重心重合。

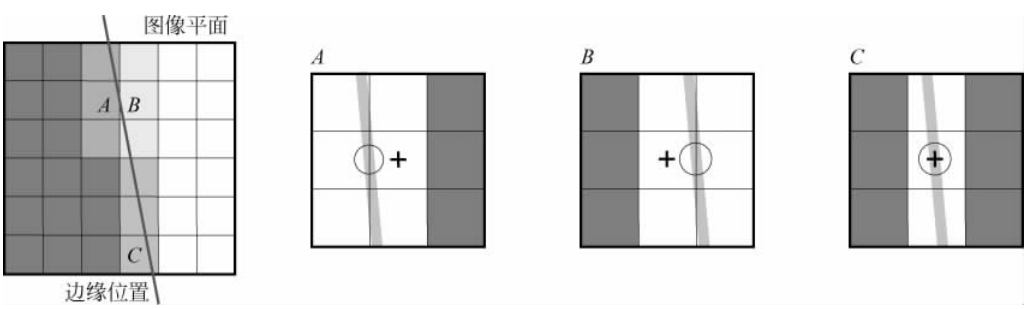

图 3.1.6　主要的边缘种类示意图

在区域 A 和区域 B 中，边缘落在像素之间，从核同值区的重心到模板核的矢量与边缘的局部方向（几乎）垂直。在区域 C 中，边缘通过像素的中心而不是像素之间，且边缘两边有较高的反差。这种情况对应像素内部边缘。在这种情况下，所获得的核同值区是沿边缘方向的细条，如图 3.1.6 所示。通过寻找最长的对称轴就可以确定边缘方向。具体可通过如下求和所得到的结果来估计：

$$F_x(x_0,y_0)=\sum_{(x,y)\in N(x_0,y_0)}(x-x_0)^2C(x_0,y_0;x,y) \tag{3.1.10}$$

$$F_y(x_0,y_0)=\sum_{(x,y)\in N(x_0,y_0)}(y-y_0)^2C(x_0,y_0;x,y) \tag{3.1.11}$$

$$F_{xy}(x_0,y_0)=\sum_{(x,y)\in N(x_0,y_0)}(x-x_0)(y-y_0)C(x_0,y_0;x,y) \tag{3.1.12}$$

使用 $F_y(x_0, y_0)$ 和 $F_x(x_0, y_0)$ 的比值就可以确定边缘的朝向，而用 $F_{xy}(x_0, y_0)$ 的符号可帮助区别对角朝向的边缘具有正的或负的梯度。

剩下来的问题就是如何自动地确定哪个图像点属于哪种情况。首先，如果核同值区的面积比模板的直径小，那么就应该对应像素内部边缘的情况。如果核同值区的面积比模板的直径大，那么就可以确定出核同值区的重心，并可用来根据像素之间边缘的情况来计算边缘的方向。当然，如果重心处在离核不到一个像素的位置，那么这更有可能属于像素内部边缘的情况。如果用较大的模板使得处于中间灰度的条带比一个像素还宽时，这种情况就会发生。

4. 最小核同值区算子的特点

与其他边缘和角点检测算子相比，最小核同值区算子有一些独特的地方。

(1) 在用最小核同值区算子对边缘和角点进行检测增强时不需要计算微分，这可帮助解释为什么在有噪声时最小核同值区算子的性能会较好。这个特点再加上最小核同值区算子的非线性响应特点都有利于减少噪声。为理解这一点，可考虑一个混有独立分布的高斯噪声的输入信号。只要噪声相对于核同值区面积比较小，就不会影响基于核同值区面积所做的判断。这里在面积计算中，对各个像素值的求和操作进一步减少了噪声的影响。

(2) 最小核同值区算子的另一个特点可以从图 3.1.1 中的核同值区看出。当边缘变得模糊时，在边缘中心的核同值区的面积将减少。所以，对边缘的响应将随着边缘的平滑或模糊而增强。这个有趣的现象对一般的边缘检测算子是不常见的，但它在实际中很有用。

(3) 大多数的边缘检测算子会随图像或模板尺度的变化而改变所检测出来的边缘的位置，但最小核同值区检测算子能提供不依赖于模板尺寸的边缘精度。换句话说，最小核同值区面积的计算是个相对的概念，与模板的绝对尺寸无关，所以最小核同值区算子的性能不受模板尺寸影响。这是一个很有用的期望特性。

(4) 最小核同值区算子的另一个优点是控制参数的选择很简单，且任意性比较小(只取决于模板的尺寸)，所以比较容易实现自动化选取。

最后，如果对边缘位置的精度要求比用整个像素作为计算单元所能获得的精度还要高，则可采用下面的方法来改进以获得亚像素的精度(还可参见 4.2 节)。对每个边缘点，先确定在该点的边缘方向，然后在与该点边缘垂直的方向上细化边缘。对这样剩下来的边缘点用 3 个点的二阶曲线来拟合初始的边缘响应，在这条拟合线上的转向点(应该和细化后边缘点的中心距离小于半个像素)可取作边缘的准确位置。

3.1.3 哈里斯兴趣点算子

哈里斯兴趣点算子也称哈里斯兴趣点检测器。检测器的表达矩阵可借助图像中局部模板里两个方向的梯度 I_x 和 I_y 来定义。一种常用的哈里斯矩阵可写成

$$\boldsymbol{H} = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \tag{3.1.13}$$

1. 角点检测

哈里斯角点检测器通过计算像素邻域中灰度值平方差的和来检测角点[Sonka 2008]。在检测角点时，可用下式计算角点强度(注意行列式 det 和秩 trace 都不受坐标轴旋转的

影响)：

$$C = \frac{\det(\boldsymbol{H})}{\mathrm{trace}(\boldsymbol{H})} \tag{3.1.14}$$

理想情况下考虑圆形的局部模板。对模板中只有直线的情况，$\det(\boldsymbol{H})=0$，所以 $C=0$。如果模板中有一个锐角(两条边之间的夹角小于 90°)的角点，如图 3.1.7(a)所示，则哈里斯矩阵可写成：

$$\boldsymbol{H} = \begin{bmatrix} l_2 g^2 \sin^2\theta & l_2 g^2 \sin\theta\cos\theta \\ l_2 g^2 \sin\theta\cos\theta & l_2 g^2 \cos^2\theta + l_1 g^2 \end{bmatrix} \tag{3.1.15}$$

其中，l_1 和 l_2 分别为角的两条边的长度，g 表示边两侧的对比度，在整个模板中为常数。

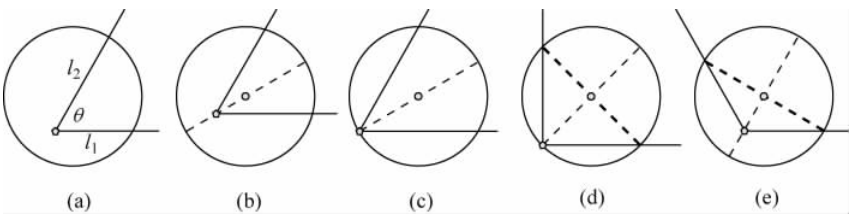

图 3.1.7　角点与模板的各种位置关系

根据式(3.1.15)可算得：

$$\det(\boldsymbol{H}) = l_1 l_2 g^4 \sin^2\theta \tag{3.1.16}$$

$$\mathrm{trace}(\boldsymbol{H}) = (l_1 + l_2) g^2 \tag{3.1.17}$$

代入式(3.1.14)得到角点强度：

$$C = \frac{l_1 l_2}{l_1 + l_2} g^2 \sin^2\theta \tag{3.1.18}$$

其中包括 3 项：依赖于模板中边长度的强度因子 $\lambda = l_1 l_2/(l_1 + l_2)$，对比度因子 g^2，依赖于锐角度数的形状因子 $\sin^2\theta$。

前面已指出，对比度因子是个常数。形状因子依赖于夹角 θ。在 $\theta=\pi/2$ 时，形状因子取得最大值 1；而在 $\theta=0$ 和 $\theta=\pi$ 时，形状因子都取得最小值=0。由式(3.1.18)可知，对直线，角点强度为零。

强度因子与模板中两条边的长度都有关。如果设 l_1 与 l_2 之和为一个常数 L，则强度因子 $l=(Ll_2-l_2^2)/L$，并在 $l_1=l_2=L/2$ 时取得极大值。这表明为获得大的角点强度，需要把角点的两条边对称地放入模板区域，如图 3.1.7(b)所示，即角点落在角的中分线(也是直径)上。而为了获得最大的角点强度，要让角点的两条边在模板区域中都最长，这种情况如图 3.1.7(c)所示，即将角点沿中分直径线移动，直到角点落在圆形模板的边界上。

根据类似上面的分析可知，如果角点是个直角的角点，则角点强度最大的角点位置也是在圆形模板的边界上，如图 3.1.7(d)所示。此时，角点两条边与圆形模板边界的两个交点间的直径是与角平分线垂直的。进一步，对钝角角点也可进行类似的分析，而结论也是角点两条边与圆形模板边界的两个交点间的直径是与角平分线垂直的，如图 3.1.7(e)所示。

2. 交叉点检测

哈里斯兴趣点算子除了可帮助检测各种角点外，还可帮助检测其他兴趣点，如交叉点和 T 形交点。这里交叉点可以是两条互相垂直的直线的交点(如图 3.1.8(a)所示)，也可以是

两条互相不垂直的直线的交点(如图 3.1.8(b)所示)。类似地,构成 T 形交点的两条直线可以互相垂直(如图 3.1.8(c)所示),也可以不互相垂直(如图 3.1.8(d)所示)。在图 3.1.8 中,相同的数字表示所指示的区域具有相同的灰度,而不同的数字表示所指示的区域具有不同的灰度。

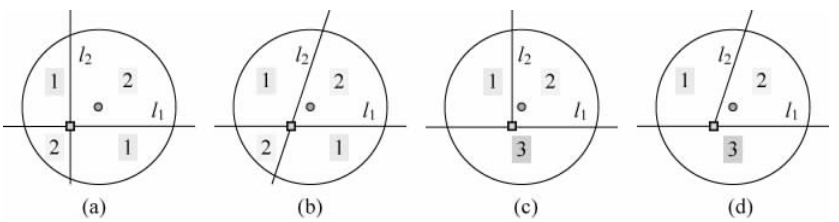

图 3.1.8　交叉点和 T 形交点

在计算交叉点的强度时,仍可以使用式(3.1.18),只是这里 l_1 和 l_2 的值分别是两个方向直线的总长度(交叉点两边线段之和)。另外要注意,在交叉点处,沿两个方向的对比度符号都会反转。但这对交叉点强度的计算没有影响,因为在式(3.1.18)中使用了 g 的平方作为对比度因子。所以,如果将交叉点与模板中心点重合,l_1 和 l_2 的值分别是角点时的两倍,这也是交叉点强度最大的位置。顺便指出,这个位置是二阶导数的过零点。

T 形交点可以看作是比角点和交叉点更一般的兴趣点,因为它涉及 3 个具有不同灰度的区域。为考虑交点处有两种对比度的情况,需要将式(3.1.18)推广为

$$C = \frac{l_1 l_2 g_1^2 g_2^2}{l_1 g_1^2 + l_2 g_2^2} \sin^2\theta \tag{3.1.19}$$

T 形交点可以有许多不同的构型,这里仅考虑一种一个弱边缘接触到一个强边缘但没有穿过该强边缘的情况,如图 3.1.9 所示。

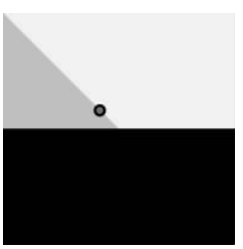

图 3.1.9　T 形交点示例和检测强度最大的点

在图 3.1.9 中,T 形交点是不对称的。由式(3.1.14)可知,其最大值在 $l_1|g_1| = l_2|g_2|$ 时取得。这表明检测强度最大的点处在弱边缘上但不处在强边缘上,如图 3.1.9 中的圆点所示。换句话说,检测强度最大的点并不处在 T 形交点的几何位置上,因为受到灰度的影响而产生了偏移。

3.2　图 割 方 法

图割方法是一类基于图论的图像分割技术,本质上采用了基于边缘的串行分割思路,所以属于串行边界类。下面先给出用图割方法进行图像分割的主要步骤,再具体对每个步骤

进行解释。

用图割方法进行图像分割的主要步骤为

(1) 将待分割图像 I 映射为一个对弧加权的有向图 G，G 在尺寸上和维数上都与 I 对应。

(2) 确定目标和背景的种子像素，并针对它们构建两个特殊的图结点，即源结点 s 和汇结点 t；然后将所有其他像素对应的图结点都与源结点或汇结点分别相连接。

(3) 计算弧代价函数，并对图 G 中的各个弧赋予一定的弧代价。

(4) 使用最大流图优化算法来确定对图 G 的图割，从而区分对应目标和背景像素的结点。

下面对 4 个步骤的一些原理和方法分别进行简单介绍。

1. 构建有向图 G

一个**图**结构可表示为 $G=[N,A]$，其中 N 是一个有限非空的结点集合，A 是一个无序结点对的集合。集合 A 中的每个结点对 (n_i,n_j) 称为一段弧 $(n_i\in N,n_j\in N)$。如果图中的弧是有向的，即从一个结点指向另一个结点，则称该弧为有向弧，称该图为有向图。对任一段弧 (n_i,n_j) 都可定义一个代价(或费用)，记为 $C(n_i,n_j)$，它可看作是对弧的加权。

对给定的待分割图像 I，要将其转化表示为一个对弧加权的图 G。其中，将图像 I 中每个像素看成图 G 中的一个结点，即结点集合 N 由所有像素构成；而将像素之间的邻接关系用图 G 中的弧来表示，即结点对集合 A 表示像素之间的(加权)联系。在图 G 中，需要确定目标种子结点和背景种子结点，它们应分别是最终分割结果中目标集合 O 和背景集合 B 的一部分。这个工作目前采用的方法常常借助人机交互来进行。根据所确定的种子，可以对应构建两个特殊的图结点：源结点 s 和汇结点 t(这里取源结点对应目标种子，汇结点对应背景种子)。初始时，其他非对应目标种子结点或背景种子结点的像素结点都分别与源结点和与汇结点相连接。

2. 图割分割

如上构建的弧加权图 $G_{st}=[N\cup\{s,t\},A]$，其中结点集 N 对应图像 I 中的像素，s 和 t 是两个特殊的终端结点。在 G_{st} 中的弧集合 A 中的元素可以分为两类：连接一对相邻像素的弧与将像素和终端结点连接起来的弧。可使用割(cut)将其穿过/跨越的弧切断。在 G_{st} 中的一个割集合可将图中的结点分成两组，它的代价是这个割集合所对应的弧集合的代价之和。代价最小的割集合被称为最小 s-t 割，它将结点分成两组不重叠的子集 S(所有与源连接的结点，$s\in S$)和 T(所有与汇连接的结点，$t\in T$)，且从 s 到 t 没有有向的通路。

图 3.2.1 给出一个解释上面构建有向图并利用图割技术进行图像分割的示意。图(a)给出一幅待分割的图像 I，且已确定了目标种子 o(属于源集合 S)和背景种子 b(属于汇集合 S)。图(b)是所构建的图 G_{st}(其中，采用虚线表示连接相邻像素的弧，采用不同的实线表示连接像素和终端结点的弧)，其中，$o\subset N$，$b\subset N$，$o\cap b=\varnothing$。在分割开始前，所有结点都同时与源集合和汇集合相连接。图(c)在图 G_{st} 中给出一个 s-t 割(将相应的弧割断了)，它将结点分为两组，其中所有目标结点都仅与源集合相连而所有背景结点都仅与汇集合相连。图(d)给出根据这个 s-t 割将各个结点映射回图像而得到的分割结果，其中所有的目标像素都标记为白色，而所有的背景像素都标记为黑色。

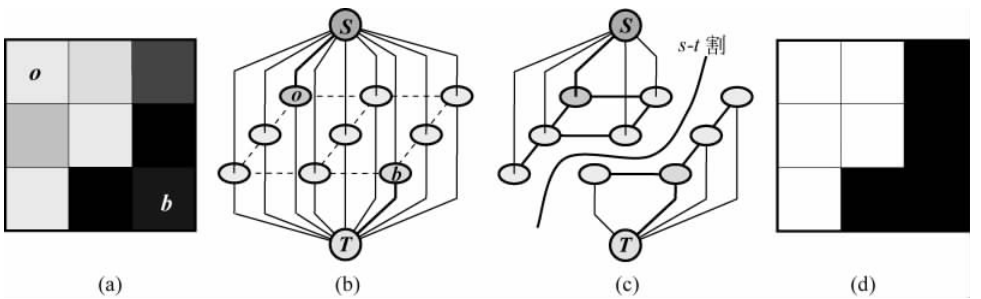

图 3.2.1　图割分割示意图

3. 弧的代价

代价最小 *s-t* 割的代价是其所对应的所有弧的代价之和。下面考虑各种弧的**代价函数**的计算。

给各个像素 i_p 一个二值的标号 $L_p \in \{o,b\}$，其中 o 和 b 分别代表目标和背景的标号。标号矢量 $\boldsymbol{L}=\{L_1,L_2,\cdots,L_p,\cdots,L_{|I|}\}$ 定义所得到的二值分割结果。为计算一段弧的代价，既要考虑该弧两个端结点所对应像素的灰度，也要考虑该弧两个端结点所对应像素之间的灰度差。这里，需要最小化以获得最优标号的代价函数可以定义为一个用 λ 加权的区域性质项 $R(\boldsymbol{L})$ 和边界性质项 $F(\boldsymbol{L})$ 的组合

$$C(\boldsymbol{L}) = \lambda R(\boldsymbol{L}) + F(\boldsymbol{L}) \tag{3.2.1}$$

其中

$$R(\boldsymbol{L}) = \sum_{p\in I} R_p(L_p) \tag{3.2.2}$$

$$F(\boldsymbol{L}) = \sum_{(p,q)\in A} F_{(p,q)}\delta(L_p,L_q) \tag{3.2.3}$$

$$\delta(L_p,L_q) = \begin{cases} 1 & L_p \neq L_q \\ 0 & \text{其他} \end{cases} \tag{3.2.4}$$

先考虑连接一对相邻像素的弧。一方面，给定一个像素，根据其灰度将其标为目标或背景都会有一定的代价。这里，$R_p(o)$ 可看作是将像素 p 标为目标的代价，而 $R_p(b)$ 可看作是将像素 p 标为背景的代价。当亮的目标处在暗背景上时，$R_p(o)$ 的值在暗像素（低 I_p 值）处大而在亮像素处小。反之，当暗的目标处在亮背景上时，$R_p(b)$ 的值在亮像素（高 I_p 值）处大而在暗像素处小。另一方面，对两个相邻的像素 p 和 q，根据其灰度将其赋予不同的标号也会有一定的代价。如果它们都属于目标或背景，则弧 (p,q) 的代价 $F_{(p,q)}$ 应比较大；如果它们一个属于目标而另一个属于背景，即跨越目标和背景的边界，则弧 (p,q) 的代价 $F_{(p,q)}$ 应比较小。例如，可以取两个相邻像素 p 和 q 之间弧 (p,q) 的代价 $F_{(p,q)}$ 与它们之间的梯度幅度（边缘两边的绝对灰度差 $|f(p)-f(q)|$）成反比。

再考虑将像素和终端结点连接起来的弧。两个终端结点之间借助通路上对应相邻像素之间各段弧连接起来的总代价应是这些相邻像素之间弧的代价之和。实际中常常可再加一个 1 以使弧没有饱和（B 和 O 分别代表背景像素集合和目标像素集合）：

$$K = 1 + \max_{p\in B} \sum_{q;(p,q)\in O} F(p,q) \tag{3.2.5}$$

将上述分析结果结合起来，赋予各种弧的代价函数如表 3.2.1 所示。

表 3.2.1　各种弧的代价函数

弧	(p,q)	(s,p)		(p,t)	
代价	$C_{(p,q)}$　$(p,q)\in N$	$\lambda R_p(b)$	$p\in I, p\notin (O\cup B)$	$\lambda R_p(o)$	$p\in I, p\notin (O\cup B)$
		K	$p\in O$	0	$p\in O$
		0	$p\in B$	K	$p\in B$

4. 最小 *s-t* 割的计算

计算最小 *s-t* 割的问题可借助它的对偶——计算最大流问题来进行，即可通过搜索从源 s 到汇 t 的最大流来解决。式(3.2.5)的弧代价也可解释成从源 s 到 $p\in O$(或从 $p\in B$ 到汇 t)的最大流容量。在最大流算法中，从源 s 到汇 t 的“最大量的水流”通过图中的各段弧构成的通路来输送，而通过各段弧的流量由它的容量或弧代价决定。从源 s 到汇 t 的最大流能使图中的一组弧达到饱和，这些饱和的弧(对应最小割)会将结点分为不重合的两个集合 S 和 T。最大流的值对应最小割的代价。

最小 *s-t* 割问题和它的对偶最大流问题都是传统的**组合问题**，可用多项式时间的算法解决。有许多算法可用来解决这个组合优化的问题，如增强通路算法，优化最大流的压入和重标记(push-relabel)算法等[Sonka 2014]。

增强通路算法[Ahuja 1993]考虑推动从源 s 到汇 t 的流直至达到最大流量以计算最短的 $s\rightarrow t$ 通路。算法开始时将流的状态初始化为 0，在将流增强并使通路逐渐饱和的过程中，把流分布的当前状态保存在一个**残留图** G_r 中。当 G_r 的拓扑与 G_{st} 相同时，弧的值在当前流的状态下保留了余下弧的流量。在各个迭代步骤，算法沿着残留图中未饱和的弧所构成的通路来确定最短的 $s\rightarrow t$ 通路。流过一条通路的流量借助推动最大可能的流而增加，直到这条通路上的至少一个弧能达到饱和。每个流量增加的步骤都能增加从源 s 到汇 t 的总流量。增加每条通路的流量直到都不能再增加(不能定义新的 $s\rightarrow t$ 通路组成非饱和的弧)，则达到最大流，最优化过程结束。此时的饱和弧集合(对应最小 *s-t* 割)将分别属于 S 和 T 的图结点分了开来。在确定最短的 $s\rightarrow t$ 通路时可以采用宽度优先搜索的策略。开始先搜索长度最短的 *s-t* 通路，一旦所有长度为 k 的通路都饱和了，则接着搜索长度为 $k+1$ 的通路，直到穷尽所有长度的通路。可以证明该算法具有收敛性，算法的复杂度是 $O(|N||A|^2)$，其中 $|N|$ 是结点总数而 $|A|$ 是弧总数。

图 3.2.2 给出一个使用增强通路最大流算法来确定最小割的各个步骤的示例。其中，图(a)所示为与图 3.2.1(a)对应的原始图像；图(b)所示为根据 4-连接计算得到的图像灰度的梯度幅度($|f(p)-f(q)|$)；图(c)所示为构建图 G_{st} 而使用的结点标记，其中源结点记为 s，汇结点记为 t，其余结点按扫描顺序依次记为 a 到 g。图(d)中用粗线表示出饱和图的弧；图(e)加上了分开 S 和 T 结点的最小 *s-t* 割；图(f)是最终的图像分割结果(一旦获得了 *s-t* 割，则将结点映射回到图像中就可)。

图(g)所示为根据图(c)的标记而画出的图 G_{st}，弧的代价标在弧旁；图(h)所示为将图(b)的梯度幅度值画在图 G_{st} 中的结果；图(i)所示为根据表 3.2.1 构建的图 G_{st}，其中 $\lambda=0$，$F_{(p,q)}=\max|f(p)-f(q)|-|f(p)-f(q)|$(由于有 3 条弧的流量为 0，所以实际上该图中只有一条从 s 到 t 的通路)；图(j)所示为当唯一具有非饱和 $s\rightarrow t$ 连接的最短通路被确定

且饱和后的残余图G_r，此时已没有非饱和的$s\rightarrow t$通路了，分割结束。这里，图(i)中沿最短通路的流量增加2后从结点a到结点b的弧饱和，则图(j)中该通路上各段弧的残留流量就都减少2。注意，图(j)是与图(d)相对应的。

图割方法的一个重要特性是提供了一种借助交互以有效地改进先前获得的分割结果的能力。设用户确定了初始的种子，且有了代价函数，但图割产生的结果还未达到要求(所得到的割还不能将目标结点和背景结点完全分开)。用户可通过增加目标或背景的种子进行改进。此时不需要从头计算，可以使用先前已经得到的结果来初始化下一个优化过程并继续分割。

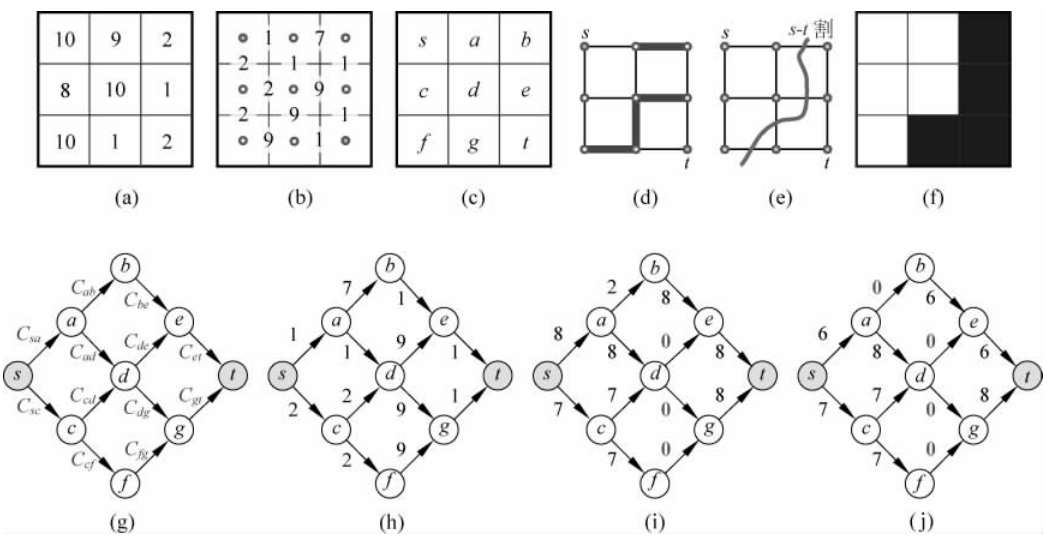

图3.2.2　利用图割和最大流优化的图像分割过程

例3.2.1　增强通路算法示例

使用增强通路算法来计算最大流和最小割的过程可以借助图3.2.3来示例说明。图中给出一个图$G=\{N,A\}$，即结点集合N和有向弧集合A构成的图。每条连接结点p和q的弧c上的(最大)流量为c_{pq}(非负值)，与弧代价$C(p,q)$对应。结点集合N有两个特殊的结点，分别是源(起点)s和汇(终点)t。这是一个有向图，在一般情况下允许有些结点之间存在两个方向的弧。最大流计算问题可描述成在图中找到一条从源到汇的通路，这条通路要在各条弧所允许的流量约束下能通过最大的流。换句话说，可以考虑不断增加流量来获得需要的能通过最大流的通路。

这样找到的从源到汇的通路有可能不止一条。但在每条这样的通路上，至少有一条弧的流量会达到饱和，否则就还可以增加流量，而该通路就还不是最大流通路。根据这个分析，最大流计算问题也可从考虑弧的饱和来解决。设“割”是图中将源和汇结点分离开的弧集合，则如果把这个弧集合从图中除去，从源结点到汇结点就没有通路了。每个弧集合对应一个代价，即集合中所有弧的代价(这里对应流量c_{pq})之和。现在，最大流计算问题就成为找到代价最小的最小割的计算问题了。

图3.2.4所示是增强通路算法在计算过程中各个步骤的一个示例，所用图$G=\{N,A\}$与图3.2.3相同，对弧的标记形式为“当前流/饱和流”。它首先从图中任意选择一条从源到

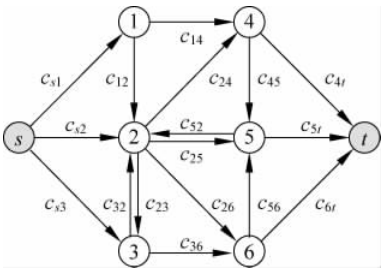

图 3.2.3　最大流计算示意图

汇的通路，并不断增加其中的流量，直到最大流量被通路中允许流量最小的弧所限制（该弧达到饱和）。此时，将这个流量从这条通路中的其他弧的允许流量中减去，饱和弧的允许流量会成为 0。重复这个过程，选择第二条从源到汇的通路，增加流量直到有一条弧饱和，更新各条弧的允许流量。这样重复进行，直到再也找不到不含饱和弧的、从源到汇的通路时为止。这些通路的总流量就是从源到汇的最大流量，而这些饱和的弧合起来就构成最小割。如果在选择通路时每次都选择能使剩下的流量达到最大的通路，则该算法可保证收敛。

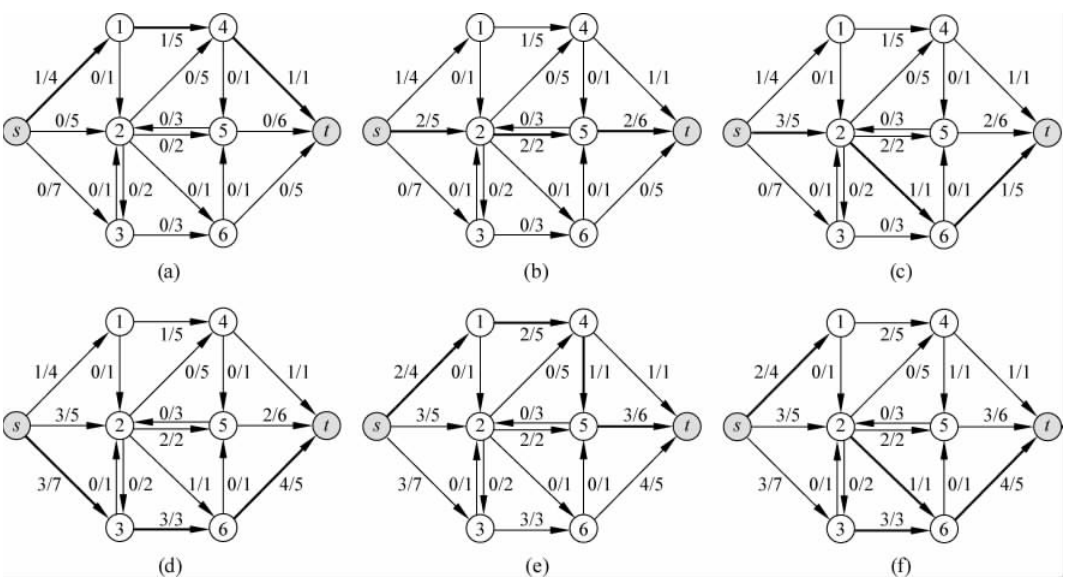

图 3.2.4　增强通路算法计算过程

在图 3.2.4 中，图(a)中画出了第一条从源到汇的通路（上部），包括三条（粗线）弧，即从 s 到①到④到 t。增加流量，直到第三条弧（④到 t）饱和（红色）。图(b)中画出了第二条从源到汇的通路（中部），也包括三条（粗线）弧，即从 s 到②到⑤到 t。同样增加流量，这次直到第二条弧（②到⑤）饱和（红色）。图(c)中，将刚才已饱和的弧用另一条弧（②到⑥）替换，得到一条新的通路），也包括三条弧，即从 s 到②到⑥到 t。增加流量，使②到⑥的弧饱和，此时第一条弧（s 到②）上的流量是原饱和弧（②到⑤）与现饱和弧（②到⑥）的总和。图(d)中画出了第四条从源到汇的通路（下部），包括三条弧，即从 s 到③到⑥到 t。增加流量，使③到⑥的

弧饱和，则第三条弧(⑥到 t)上的流量是原经过②到⑥的流量与现经过③到⑥的流量的总和。图(e)给出第五条通路，也是唯一的长度为 4 的通路(前面四条通路的长度均为 3)。该通路包括的四条弧为从 s 到①到④到⑤到 t，其中第三条弧(④到⑤)先饱和。图(f)中共有五条饱和的弧，截断这些弧的割如图中条带所示。该条带将所有从 s 通往 t 的通路全部割断，对应最小割。它将所有结点分成了两组：一组包括源结点 s 不包括汇结点 t，另一组包括汇结点 t 不包括源结点 s。 □

3.3 特色的阈值化和聚类技术

第 2 章介绍并行区域分割方法时主要讨论了基本的阈值化和聚类方法。本节介绍两种有一定代表性的借助特殊理论和方法确定阈值的技术以及一种近年常用的确定空间聚类以分割图像的技术。

(1) 借助小波变换的多分辨率特性来帮助进行阈值选取。

(2) 借助过渡区选取阈值进行分割。

(3) 借助均移计算特征空间的聚类中心来进行分割。

3.3.1 多分辨率阈值选取

利用图像的**直方图**帮助选取阈值是常用的方法，其中的关键是确定峰点和谷点。由于场景的复杂性，图像成像过程中各种干扰因素的存在等原因，峰点和谷点的有无检测和位置确定常比较困难。峰点和谷点的检测与直方图的尺度有密切的联系。一般在较大尺度下常能较可靠地检测到真正的峰点和谷点，但在大尺度下对峰点和谷点的定位不易准确。相反，在较小尺度下对真正峰点和谷点的定位常比较准确，但在小尺度下误检或漏检的比例会增加。

图像在小波变换后可分解为一系列尺度不同的分量。对小波变换后的图像直方图也可进行多分辨率分析，具体就是先在较大尺度下检测出真正的峰点和谷点，再在较小尺度下对这些峰点和谷点进行较精确的定位。这里主要有如下两个步骤。

1. 确定分割区域的类数

首先利用在较大尺度(粗分辨率)下的直方图细节信息确定分割区域的类数。引入相当于低通滤波器的尺度函数 $\phi(x)$，则图像直方图 $H(x)$ 的低通分量可表示为

$$S_{2^i}[H(x)] = H(x) \otimes \phi_{2^i}(x) \tag{3.3.1}$$

设原始图像直方图信号的分辨率为 1，最低分辨率尺度为 2^I，则尺度在 2^1 和 2^I 之间的各个阶小波变换为 $\{W_{2^i}[H(x)], 1 \leqslant i \leqslant I\}$。可以证明，对信号在尺度为 2^I 时被平滑掉的高频部分可以用尺度在 2^1 和 2^I 之间的小波变换来恢复。这里集合 $\{S_{2^i}[H(x)], W_{2^i}[H(x)], 1 \leqslant i \leqslant I\}$ 就是直方图的多分辨率小波分解表示。

先在分辨率为 2^1 时确定初始的分割区域类数，即判断直方图中独立的峰的个数。这里要求独立的峰应满足三个条件：①具有一定的灰度范围；②具有一定的峰下面积；③具有一定的峰谷差。

2. 确定最优阈值

确定类数后，可利用多分辨率的层次结构在直方图的相邻峰之间确定**最优阈值**。这个过程首先在最低分辨率一层进行，然后逐渐向高层推进，直到最高分辨率层。选高斯函数作

为平滑函数 $\theta(x)$，令小波函数为 $\psi(x)$，有

$$\psi(x)=\frac{\mathrm{d}^2\theta(x)}{\mathrm{d}x^2} \tag{3.3.2}$$

则 $\psi(x)$对应的二进小波变换为

$$W_{2^i}[H(x)]=(2^i)^2\cdot\frac{\mathrm{d}^2}{\mathrm{d}x^2}[H(x)\otimes\theta_{2^i}(x)] \tag{3.3.3}$$

由上式可知小波变换的零交叉位置对应在分辨率 2^i 时的低通信号 $H(x)\otimes\theta_{2^i}(x)$的剧烈变化点。当尺度 2^i减小时，信号的局部细节增多；而当尺度 2^i增加时，信号中结构较大的轮廓比较明显。

对图像的直方图来说，式(3.3.1)中的 $S_{2^i}[H(x)]$是一个最低分辨率下的近似信号，式(3.3.3)中的 $W_{2^i}[H(x)]$代表不同分辨率下的细节信号，它们联合构成直方图的多分辨率小波分解表示。给定直方图，可考虑其多分辨率小波分解表示的零交叉点和极值点来确定直方图的峰值点和谷点。具体可利用以下四个准则(参见图 3.3.1 的直方图)。

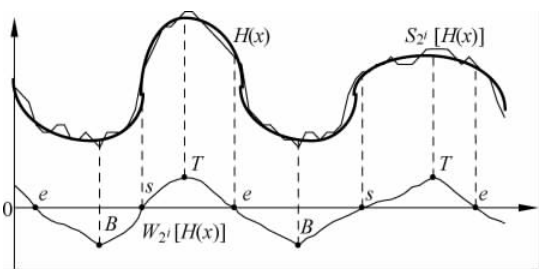

图 3.3.1　直方图的峰点和谷点的确定

(1) 用从负值变化到正值的零交叉点确定峰的起点(图中各个 s 点)；

(2) 用从正值变化到负值的零交叉点确定峰的终点(图中各个 e 点)；

(3) 用起点和终点间的最大值点确定峰的位置(图中各个 T 点)；

(4) 用前一个峰的终点和后一个峰的起点间的最小值点确定这两个峰之间谷点的位置(图中各个 B 点)。

随着分辨率的逐渐增加，阈值数目也会逐渐增加。这里可用最小距离判据来解决两相邻尺度之间各阈值并非一一对应的问题。设在两相邻尺度 2^{i+1} 和 2^i所对应的阈值分别为 T_j^{i+1}和 T_k^i，考察下列条件(N^i是在尺度 2^i所具有的阈值数目)

$$\mathrm{dis}(T_j^{i+1},T_k^i)=\min\{\mathrm{dis}(T_j^{i+1},T_l^i),\ l=0,1,\cdots,N^i\} \tag{3.3.4}$$

当 $l=k$ 时取得最小值，这表明在尺度 2^{i+1}的阈值 T_j^{i+1}对应在尺度 2^i的阈值 T_k^i。所以，可先对在最低分辨率一层选取的所有阈值逐层跟踪，最后选取在最高分辨率一层的对应阈值作为最优阈值。

3.3.2　借助过渡区选择阈值

一般在讨论基于区域和基于边界的算法时认为区域的并集覆盖了整个图像而边界本身是没有宽度的。然而实际数字图像中的边界是有宽度的，它本身也是图像中的一个区域，一个特殊的区域。一方面它将不同的区域分隔开来，具有边界的特点；另一方面，它面积不为零，具有区域的特点；可将这类特殊区域称为过渡区。下面介绍一种先计算图像中目标和

背景间的**过渡区**，再进一步选取阈值进行分割的方法[Zhang 1991a]。

1. 过渡区和有效平均梯度

过渡区可借助对图像**有效平均梯度**(EAG)的计算和对图像灰度的**剪切操作**来确定。设以 $f(i,j)$代表 2-D 空间的图像函数，其中 i,j 表示像素空间坐标，f 表示像素的灰度值，它们都属于整数集合 $\mathbf{Z}$。再设 $g(i,j)$代表 $f(i,j)$的梯度图(可用梯度算子作用于 $f(i,j)$得到)，则 EAG 可定义为

$$\mathrm{EAG}=\frac{\mathrm{TG}}{\mathrm{TP}} \tag{3.3.5}$$

其中

$$\mathrm{TG}=\sum_{i,j\in\mathbf{Z}} g(i,j) \tag{3.3.6}$$

为梯度图的总梯度值，而

$$\mathrm{TP}=\sum_{i,j\in\mathbf{Z}} p(i,j) \tag{3.3.7}$$

为非零梯度像素的总数，因为这里 $p(i,j)$定义为

$$p(i,j)=\begin{cases}1 & \text{当 } g(i,j)>0\\ 0 & \text{当 } g(i,j)=0\end{cases} \tag{3.3.8}$$

由此定义可知，在计算 EAG 时只用到了具有非零值梯度的像素。因为除去了零梯度像素的影响，所以称为“有效”梯度。EAG 是图中非零梯度像素的平均梯度，它代表了图像中一个有选择的统计量。

进一步，为了减少各种干扰的影响，定义以下特殊的**剪切变换**。它与一般剪切操作的不同之处是它把被剪切了的部分设成剪切值，这样避免了一般剪切在剪切边缘造成大的反差而产生的不良影响。根据剪切部分的灰度值与全图灰度值的关系，这类剪切可分为低端剪切与高低端剪切两种。设 L 为剪切值，则低端剪切与高端剪切后的图可分别表示为

$$f_{\mathrm{low}}(i,j)=\begin{cases}f(i,j) & \text{当 } f(i,j)>L\\ L & \text{当 } f(i,j)\leqslant L\end{cases} \tag{3.3.9}$$

$$f_{\mathrm{high}}(i,j)=\begin{cases}L & \text{当 } f(i,j)\geqslant L\\ f(i,j) & \text{当 } f(i,j)<L\end{cases} \tag{3.3.10}$$

如果对这样剪切后的图像求梯度，则其梯度函数必然与剪切值 L 有关，由此得到的 EAG 也变成剪切值 L 的函数 EAG(L)。注意 EAG(L)与剪切的方式也有关，对应低端和高端剪切的 EAG(L)可分别写成 $\mathrm{EAG}_{\mathrm{low}}(L)$和 $\mathrm{EAG}_{\mathrm{high}}(L)$。

2. 有效平均梯度的极值点和过渡区边界

典型的 $\mathrm{EAG}_{\mathrm{low}}(L)$和 $\mathrm{EAG}_{\mathrm{high}}(L)$曲线都是单峰曲线，即它们都各有一个极值，这可以借助对 TG 和 TP 的分析得到。这里仅考虑 $\mathrm{EAG}_{\mathrm{low}}(L)$，参见图 3.3.2[Zhang 1993a]，其中 $\mathrm{EAG}_{\mathrm{low}}(L)$是 $\mathrm{TG}_{\mathrm{low}}(L)$与 $\mathrm{TP}_{\mathrm{low}}(L)$的比。$\mathrm{TG}_{\mathrm{low}}(L)$和 $\mathrm{TP}_{\mathrm{low}}(L)$都随 L 的增加而减少。$\mathrm{TP}_{\mathrm{low}}(L)$减少是因为大的 L 会剪切掉更多的像素，而 $\mathrm{TG}_{\mathrm{low}}(L)$的减少有两个原因，一是像素个数的减少，二是剩下的像素间的对比度也会减少。当 L 从 0 开始增加，$\mathrm{TG}_{\mathrm{low}}(L)$和 $\mathrm{TP}_{\mathrm{low}}(L)$曲线都从它们各自的最大值下降。开始时，$\mathrm{TG}_{\mathrm{low}}(L)$曲线下降得比较慢，因为剪切掉的像素都属于背景(梯度较小)；而 $\mathrm{TP}_{\mathrm{low}}(L)$曲线下降得相对较快，因为此时剪切掉的像

素个数较多。这两个因素的共同作用会使 $EAG_{low}(L)$ 值逐步增加并达到一个极大值。然后，$TG_{low}(L)$ 曲线会比 $TP_{low}(L)$ 曲线下降得快，因为更多的具有大梯度的像素会被剪切掉，结果 $EAG_{low}(L)$ 值减少，并最后在 L_{max} 处趋向 0。

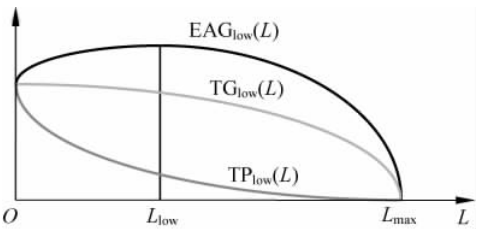

图 3.3.2　对 $EAG_{low}(L)$ 曲线是单峰曲线的解释

设 $EAG_{low}(L)$ 和 $EAG_{high}(L)$ 曲线的极值点分别为 L_{low} 和 L_{high}，则

$$L_{low} = \arg\{\max_{L}[EAG_{low}(L)]\} \tag{3.3.11}$$

$$L_{high} = \arg\{\max_{L}[EAG_{high}(L)]\} \tag{3.3.12}$$

上面计算出的 $EAG_{low}(L)$ 和 $EAG_{high}(L)$ 曲线的极值点对应图像灰度值集合中的两个特殊值，由它们可确定过渡区。事实上过渡区是一个由两个边界圈定的 2-D 区域，其中像素的灰度值是由两个 1-D 灰度空间的边界灰度值所限定的(见图 3.3.3)。这两个边界的灰度值分别是 L_{high} 和 L_{low}，它们在灰度值上限定了过渡区的灰度范围。

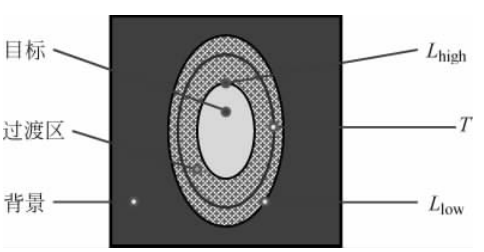

图 3.3.3　过渡区示例

这两个极值点有 3 个重要的性质(严格的证明可见[章 2001c]或[Zhang 1991a])：

(1) 对每个过渡区，L_{low} 和 L_{high} 总是存在并且各只存在一个；

(2) L_{low} 和 L_{high} 所对应的灰度值都具有明显的像素特性区别能力；

(3) 对同一个过渡区，L_{high} 不会比 L_{low} 小，在实际图像中 L_{high} 总大于 L_{low}。

由于过渡区处于目标和背景之间，而目标和背景之间的边界又在过渡区之中，所以可借助过渡区来帮助选取阈值。首先因为过渡区所包含像素的灰度值一般在目标和背景区域内部像素的灰度值之间，所以可根据这些像素确定一个阈值以进行分割。例如可取过渡区内像素的平均灰度值或过渡区内像素的直方图的极值。其次，由于 L_{high} 和 L_{low} 限定了边界线灰度值的上下界，阈值还可直接借助它们来计算[Zhang 1993a]。

前面指出的两个极值点的 3 个重要性质在图像中有不止一个过渡区时也成立。如图 3.3.4 所示，其中图 3.3.4(a)中的剖面有两个阶跃(对应两个过渡区)，反映在梯度曲线上有两个峰。图 3.3.4(b)给出由此得到的有两组极值点的 $EAG_{high}(L)$ 和 $EAG_{low}(L)$ 曲线。可以看出，对同一个过渡区，极值点的上面 3 个重要性质仍成立[章 1996c]。所以，上面基

于过渡区的方法不仅可用于确定单个阈值对图像进行二值分割也可以确定多个阈值对图像进行多阈值分割。另外，由图 3.3.4(b)可见，如果将两个过渡区混淆了，那就有可能出现 $L_{high}<L_{low}$ 的反常情况。而根据前面关于极值点性质的讨论，对同一个过渡区 L_{high} 不会比 L_{low} 小。

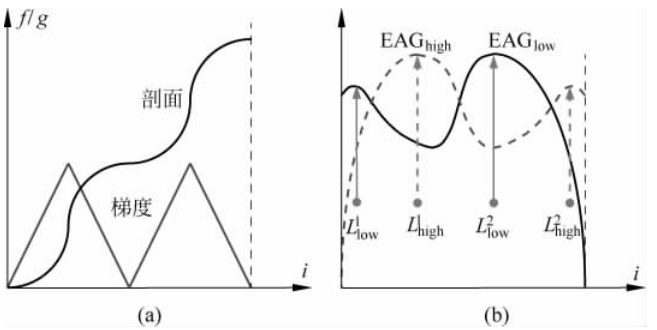

图 3.3.4　多过渡区时的情况

3.3.3　借助均移确定聚类

在空间聚类方法中，需要确定聚类的均值或聚类的中心。作为 ***K*-均值法**的一种替代方法，**均移**方法采用寻找点密度的最频值(峰)的技术，它的一个潜在好处是还可自动地选择聚类的数目。均移代表偏移的均值向量，是一种非参数技术，可用于分析复杂的多模特征空间并确定特征聚类。它假设聚类在其中心部分的分布要密，通过迭代计算密度核的均值(对应聚类重心，也是给定窗口中的最频值)来达到目的。

下面借助图 3.3.5 来介绍均移方法的原理和步骤，其中各图中的圆点表示 2-D 特征空间(实际可更高维)中的特征点。首先随机选择一个初始的感兴趣区域(初始窗)并确定其重心(如图(a)所示)。也可看作以该点为中心画个球(2-D 时画个圆)。该球或圆的半径应能包含一定数量的数据点，但不能把所有数据点都包进来。接下来，搜索周围点密度更大的感兴趣区域并确定其重心(相当于移动球的中心到一个新的位置，该位置是在这个半径中所有点的平均位置)，然后将窗移动到该重心确定的位置，这里原重心和新重心间的位移矢量就对应均移(如图(b)所示)。重复上面的过程不断地将均值移动(结果就是球体会逐步向具有较大密度的区域靠近)直到收敛(如图(c)所示)。这里最后的重心位置确定了局部密度的极大值，即局部概率密度函数的最频值。

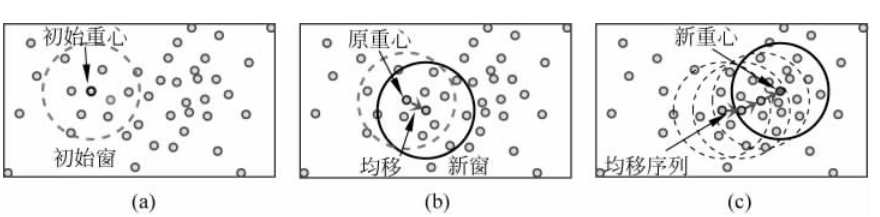

图 3.3.5　均移方法的原理示意图

在均移方法中，需要确定**多变量密度核估计器**。这里**核函数**的作用是要使得特征点随着与均值的距离 $\boldsymbol{x}$ 不同，对均值偏移的贡献也不同。实际中使用的是放射对称的核 $K(\boldsymbol{x})$，

它满足

$$K(\boldsymbol{x}) = ck(\|\boldsymbol{x}\|^2) \tag{3.3.13}$$

其中，c 是一个大于 0 的常数，用来使对 $K(\boldsymbol{x})$的求和为 1；k 为剖面核函数。两个典型的核是正态核 $K_N(\boldsymbol{x})$和 Epanechnikov 核 $K_E(\boldsymbol{x})$。正态核定义为（它常被对称性地截断以获得有限支撑的核）

$$K_N(\boldsymbol{x}) = c\exp\left(-\frac{1}{2}\|\boldsymbol{x}\|^2\right) \tag{3.3.14}$$

它的核剖面是

$$k_N(\boldsymbol{x}) = \exp\left(-\frac{1}{2}x^2\right) \quad x \geqslant 0 \tag{3.3.15}$$

Epanechnikov 核定义为

$$K_E(\boldsymbol{x}) = \begin{cases} c(1-\|\boldsymbol{x}\|^2) & \|\boldsymbol{x}\| \leqslant 1 \\ 0 & \text{其他} \end{cases} \tag{3.3.16}$$

它的核剖面在边界处不可微分

$$k_E(\boldsymbol{x}) = \begin{cases} 1-x & 0 \leqslant x \leqslant 1 \\ 0 & x > 1 \end{cases} \tag{3.3.17}$$

给定 d-D 空间的 n 个数据点 $\boldsymbol{x}_i$，在点 $\boldsymbol{x}$ 算得的多变量密度核估计器为

$$\tilde{f}_{h,k}(\boldsymbol{x}) = \frac{1}{nh^d}\sum_{i=1}^{n} K\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right) \tag{3.3.18}$$

其中，h 表示核尺寸，也称为**核带宽**。

由图 3.3.5 可见，需要确定 $f_{h,k}(\boldsymbol{x})$的梯度为零的点，即确定 $\boldsymbol{x}$ 以使$\nabla f_{h,k}(\boldsymbol{x})=0$。均移方法就是不用估计概率密度函数而确定这些点位置的一种有效方法。这里把要解决的问题由估计密度变成了估计密度梯度：

$$\nabla \tilde{f}_{h,k}(\boldsymbol{x}) = \frac{1}{nh^d}\sum_{i=1}^{n} \nabla K\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right) \tag{3.3.19}$$

使用剖面为 $k(x)$的核形式，并设它的导数 $k'(x)=-g(x)$对所有 $x\in[0,\infty]$存在（除去有限个点），则

$$K\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right) = c_k k\left(\left\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right\|^2\right) \tag{3.3.20}$$

其中 c_k是归一化常数。如果 $K(\boldsymbol{x})=K_E(\boldsymbol{x})$，则剖面 $g_E(\boldsymbol{x})$是均匀/各向同性的；如果 $K(\boldsymbol{x})=K_N(\boldsymbol{x})$，则剖面 $g_N(x)$由与 $K_N(\boldsymbol{x})$相同的指数表达所定义。使用 $g(x)$作为剖面的核 $G(\boldsymbol{x})=c_g g(\|\boldsymbol{x}\|^2)$，式(3.3.19)变成

$$\begin{aligned} \nabla \tilde{f}_{h,k}(\boldsymbol{x}) &= \frac{2c_k}{nh^{(d+2)}}\sum_{i=1}^{n}(\boldsymbol{x}-\boldsymbol{x}_i)k'\left(\left\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right\|^2\right) \\ &= \frac{2c_k}{nh^{(d+2)}}\sum_{i=1}^{n}(\boldsymbol{x}-\boldsymbol{x}_i)g\left(\left\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right\|^2\right) \\ &= \frac{2c_k}{nh^{(d+2)}}\left(\sum_{i=1}^{n} g_i\right)\left(\frac{\sum_{i=1}^{n}\boldsymbol{x}_i g_i}{\sum_{i=1}^{n} g_i} - \boldsymbol{x}\right) \end{aligned} \tag{3.3.21}$$

其中，$\sum_{i=1}^{n} g_i$ 设计为正，$g_i = g(\| (\boldsymbol{x}-\boldsymbol{x}_i)/h \|^2)$。

式(3.3.21)中的第一项正比于用核 G 算得的密度估计

$$\tilde{f}_{h,G}(\boldsymbol{x}) = \frac{c_g}{nh^{(d)}} \sum_{i=1}^{n} g\left(\left\| \frac{\boldsymbol{x}-\boldsymbol{x}_i}{h} \right\|^2 \right) \tag{3.3.22}$$

第二项代表均移矢量 $m_{h,G}(\boldsymbol{x})$

$$m_{h,G}(\boldsymbol{x}) = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i g(\| (\boldsymbol{x}-\boldsymbol{x}_i)/h \|^2)}{\sum_{i=1}^{n} g(\| (\boldsymbol{x}-\boldsymbol{x}_i)/h \|^2)} - \boldsymbol{x} \tag{3.3.23}$$

对核 G 依次确定的位置 $\{\boldsymbol{y}_j\}_{j=1,2,\cdots}$ 为

$$\boldsymbol{y}_{j+1} = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i g(\| (\boldsymbol{y}_j-\boldsymbol{x}_i)/h \|^2)}{\sum_{i=1}^{n} g(\| (\boldsymbol{y}_j-\boldsymbol{x}_i)/h \|^2)} - \boldsymbol{x} \tag{3.3.24}$$

其中 $\boldsymbol{y}_1$ 是核 G 的初始位置。

用核 K 计算得到的密度估计序列为

$$\tilde{f}_{h,K}(j) = \tilde{f}_{h,K}(\boldsymbol{y}_j) \tag{3.3.25}$$

如果核 K 具有凸的和单减的剖面，序列 $\{\boldsymbol{y}_j\}_{j=1,2,\cdots}$ 和 $\{\tilde{f}_{h,K}(j)\}_{j=1,2,\cdots}$ 都收敛，其中 $\{\tilde{f}_{h,K}(j)\}_{j=1,2,\cdots}$ 是单增的，收敛速度依赖于所用的核。在离散数据上使用 Epanechnikov 核，在有限步迭代后就可以收敛。当考虑对数据加权时(如使用正态核)，均移过程无穷收敛。此时可用步长变化的下限值来停止计算。

均移方法的优缺点都与它对数据的全局表达有关。最主要的优点就是它的通用性。由于对噪声鲁棒，所以可用于各种实际场合。它可以处理任何聚类的形状和特征空间。它仅有的一个选择参数(核尺寸 h)具有物理上可理解的意义。不过对 h 的选择也是对它应用的一个限制，因为确定一个合适的 h 并不是一件简单的事情。过大会导致最频值被聚合起来，而过小又会引入不重要的最频值并人为使得聚类被分裂。

3.4 分水岭分割算法

分水岭(也称分水线/**水线**)图像分割算法借助**地形学**概念进行图像分割，近年得到广泛使用。该算法的实现可借助一些数学形态学的方法(参见第 13 章)。该算法的计算过程是串行的，虽然直接得到的是目标的边界，但在计算过程中利用的是区域一致性。

3.4.1 基本原理和步骤

先介绍分水岭的概念，再给出对分水岭的计算方法。

1. 分水岭

分水岭是一个地形学的概念。图像也可看成是 3-D 地形的表示，即 2-D 的地基(对应图像坐标空间 xy)加上第 3 维的高度(对应图像灰度 f)。

参见图 3.4.1，设想有一个简单的圆形目标，其灰度从边缘向中间逐步增加，如把图像

看成 3-D 地形，则该目标类似一座山峰（圆锥体），如图(a)所示。如果从上向下成像则得到一幅有圆形目标的图像（参见图(b)）。现在考虑有两个圆形的目标，对应两个圆锥体，它们相距很近且有部分重叠，如图(c)所示。如果对这两个圆锥体从上向下成像则得到一幅有两个重叠圆形目标的图像，见图(d)。在两个圆的两个相交点画一条直线，该直线的位置为从两个山峰上流下来的水汇聚的地方，该直线可称为分水线。在该直线位置将两个重叠圆形目标分开应可给出一种最优的结果。

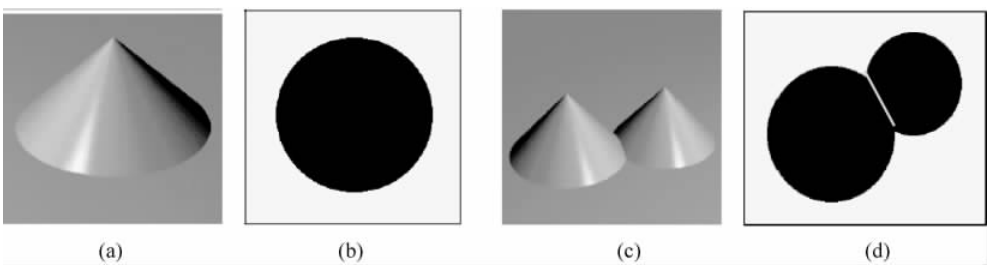

图 3.4.1　将图像看成地形图

上面借助降水法（水从高向下降）引出了分水岭的概念。实际中建立不同目标间的分水岭的过程常借助涨水法（水从低向上涨）来讨论。参见图 3.4.2（可看作灰度图像的一个剖面），这里为简便，仅画出了各个目标的 1-D 剖面。假设有水从各谷底孔涌出并且水位逐渐增高。如果从两个相邻谷底涌出的水的水位高过其间的山峰，这些水就会汇合。如要阻止这些水汇合，就需在该山峰上修筑水坝，且水坝的高度要随水位的上升而增高。这个过程随着全部山峰都被水浸没而结束。在这个过程中修筑的各个水坝将整片土地分割成很多区域，这些水坝就构成了这片土地的分水岭。

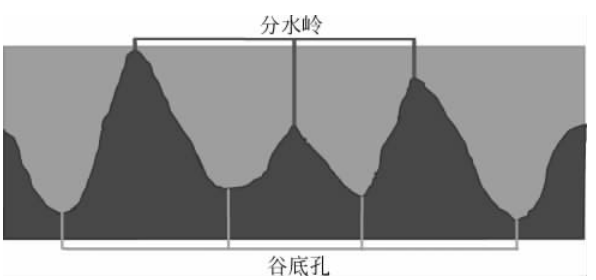

图 3.4.2　涨水法分水岭示意图

由上可见，如果能确定出分水岭的位置，就能将图像用一组各自封闭的曲线分割成不同的区域。分水岭图像分割算法就是通过确定分水岭的位置而进行图像分割的。一般考虑到各区域内部像素的灰度比较接近，而相邻区域像素间的灰度差距比较大，所以可先计算一幅图像的梯度图，再寻找梯度图的分水岭。在梯度图中，小梯度值像素对应区域的内部，而大梯度值像素对应区域的边界，分水岭算法要寻找大梯度值像素的位置，也就是寻找分割边界的位置。

例 3.4.1　用分水岭算法分割接触目标

图 3.4.3 给出使用分水岭算法对相接触的圆形目标进行分割的示例。图(a)是原始图像，而图(b)是用分水岭算法分割的结果。需要指出，由于目标有不同的尺寸，所以如果仅

使用形态学方法(见第 13 章)中的膨胀和腐蚀并不能很好地将所有目标分割开,而分水岭算法则对不同尺寸的目标均能分割。

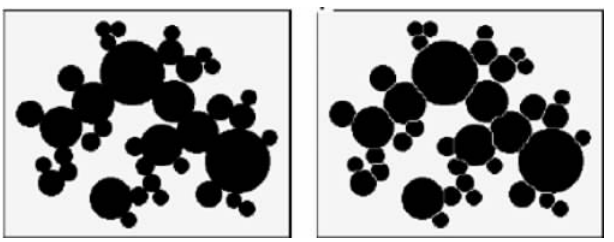

图 3.4.3　对不同尺寸的接触圆目标的分割　□

2. 分水岭计算步骤

上面的讨论实际上也指出了分水岭计算的思路,即逐渐增加一个灰度阈值,每当它大于一个局部极大值,就把当时的二值图像(只区分陆地和水域,即大于灰度阈值和小于灰度阈值两部分)与前一个时刻(即灰度阈值上一个值的时刻)的二值图像进行逻辑异或(XOR)操作,从而确定出灰度局部极大值的位置。根据所有灰度局部极大值的位置集合就可确定分水岭。

根据这个思路,可用不同的方法实现分水岭算法。下面介绍一种借助**数学形态学**(参见第 13 章)中的膨胀运算迭代计算分水岭的方法[Gonzalez 2008]。

设给定一幅待分割图像 $f(x,y)$,其梯度图像为 $g(x,y)$,分水岭的计算是在梯度空间进行的。用 $M_1,M_2,\cdots,M_R$ 表示 $g(x,y)$ 中各局部极小值的像素位置,$C(M_i)$ 为与 M_i 对应的(分水岭所围绕的)区域中的像素坐标集合。用 n 表示当前的梯度阈值,$T[n]$ 代表记为 (u,v) 的像素集合,$g(u,v)<n$,即

$$T[n]=\{(u,v)\mid g(u,v)<n\} \tag{3.4.1}$$

梯度阈值从图像梯度范围的最低值整数增加。在梯度阈值为 n 时,算法统计处于平面 $g(x,y)=n$ 以下的像素集合 $T[n]$。对 M_i 所在的区域,其中满足条件的坐标集合 $C_n(M_i)$ 可看作一幅二值图像

$$C_n(M_i)=C(M_i)\cap T[n] \tag{3.4.2}$$

即在 $(x,y)\in C(M_i)$ 且 $(x,y)\in T[n]$ 的地方,有 $C_n(M_i)=1$,其他地方 $C_n(M_i)=0$。也可这样说,对处于平面 $g(x,y)=n$ 以下的像素,用"与"操作可将与最低点 M_i 对应的那些像素提取出来。

图 3.4.4 用 1-D 的形式(可看作梯度图像的一个剖面)给出对上面讨论的各个概念的一个直观解释,$C_n(M_i)$ 可由 $C(M_i)$ 和 $T[n]$ 求交集得到,$C_n(M_{i+1})$ 可由 $C(M_{i+1})$ 和 $T[n]$ 求交集得到。

如果用 $C[n]$ 代表在梯度阈值为 n 时图像中所有满足梯度值小于 n 的像素集合

$$C[n]=\bigcup_{i=1}^{R}C_n(M_i) \tag{3.4.3}$$

那么 $C[\max+1]$ 将是所有区域的并集($\max$ 是图像灰度范围的最高值):

$$C[\max+1]=\bigcup_{i=1}^{R}C_{\max+1}(M_i) \tag{3.4.4}$$

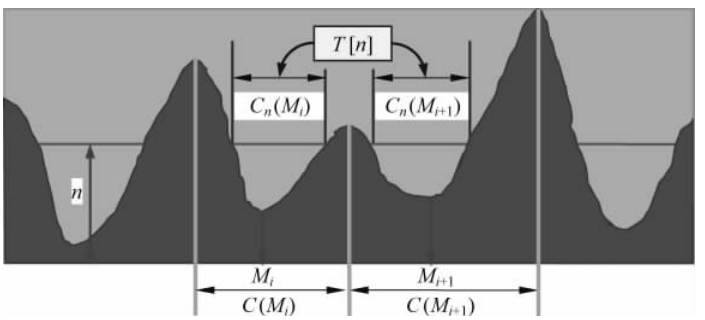

图 3.4.4　计算 $C_n(M_i)$的示意图

可以证明，集合 $C_n(M_i)$和 $T[n]$里的所有元素在分割算法进行中，即在将 n 逐渐增加期间始终保持在原集合中。随着 n 的增加，这两个集合中的元素个数是单增不减的。所以 $C[n-1]$是$C[n]$的子集。根据式(3.4.2)和式(3.4.3)，$C[n]$是 $T[n]$的子集，所以 $C[n-1]$又是 $T[n]$的子集。由此可知，每个 $C[n-1]$中的连通组元都包含在一个 $T[n]$的连通组元中。

分水岭算法先初始化 $C[\min+1]=T[\min+1]$，然后算法逐步迭代进行。设在步骤 n 时已建立了 $C[n-1]$，下面考虑从 $C[n-1]$得到 $C[n]$的计算过程。令 S 代表 $T[n]$中的连通组元集合，对每个连通组元 $s\in S[n]$，有如下 3 种可能性(参见图 3.4.5)：

(1) $s\cap C[n-1]$是一个空集(如图 3.4.5(a))；

(2) $s\cap C[n-1]$里包含 $C[n-1]$中的一个连通组元(如图 3.4.5(b))；

(3) $s\cap C[n-1]$里包含 $C[n-1]$中一个以上的连通组元(如图 3.4.5(c))。

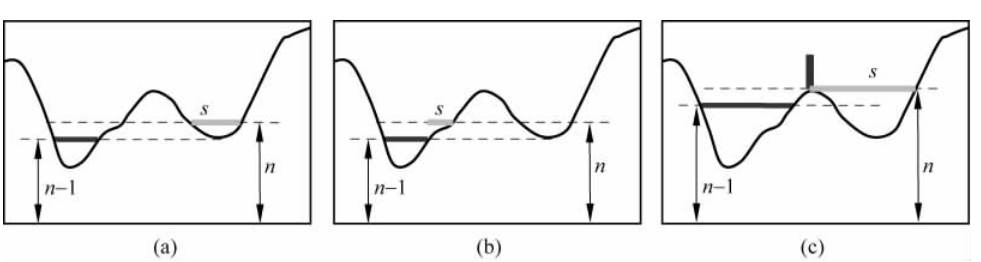

图 3.4.5　计算 $C_n(M_i)$的示意图

从 $C[n-1]$得到 $C[n]$的计算过程取决于上面 3 个条件中的哪个条件可以成立。条件(1)在遇到一个新的极小值时成立，在这种情况下，$C[n]$可由把连通组元 s 加到 $C[n-1]$中得到。条件(2)在 s 属于某些极小值的区域时成立，在这种情况下，同样 $C[n]$可由把连通组元 s 加到 $C[n-1]$中得到。条件(3)在遇到部分或整个区分两个或以上区域的分界线时成立。如果 n 继续增加，不同的区域将会连通，这时就需要在 s 中建分水岭。实际中，通过用全是 1 的 3×3 结构元素(见 12.3.1 小节)对 $s\cap C[n-1]$进行膨胀(限定在 s 中)就可获得宽度为 1 的边界。

例 3.4.2　分水岭算法分割示例

图 3.4.6 给出一组对实际沙粒图像用分水岭算法分割的结果。图(a)是原始图像，图(b)是阈值化后的结果(相邻沙粒混在一起)，图(c)是应用分水岭算法分割的结果，图(d)

是将图(c)得到的轮廓线叠加到原始图像上的效果。

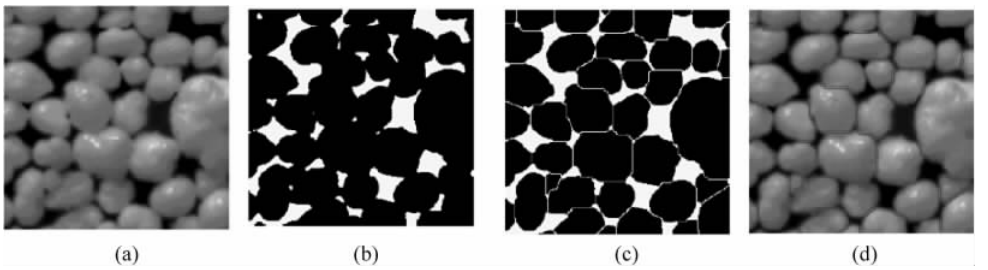

图 3.4.6　分水岭算法分割示例 □

3.4.2　算法改进和扩展

上面介绍的基本分水岭算法已得到广泛应用，在实用中，还有一些改进和推广。

1. 利用标号控制分割

分水岭分割算法的主要优点是对图像的变化很敏感，既可以检测出感兴趣区域的轮廓又可检测出均匀区域中的低对比度变化。但直接使用前述的算法步骤在实际中由于受到图像中噪声和其他局部不规则结构的影响(在梯度图中这些问题更明显)，常常会导致**过分割**，即将图像分割得过细。直观地说，由于梯度噪声、量化误差及目标内部细密纹理的影响，在平坦区域内部可能会产生许多局部的“谷底”和“山峰”，经分水岭变换后形成小的区域，很容易导致过分割，使希望得到的正确轮廓被大量不相关轮廓所淹没。

为解决上述过分割的问题，一般可在分水岭分割算法前先加一个预处理步骤(如可借助先验知识)，限定允许的分割区域的数目。

有一种常用的控制过分割的方法是利用**标号**。一个标号本身是图像中的一个连通组元，可以区分内部标号和外部标号，前者对应目标而后者对应背景。内部标号的像素应有相同灰度并组成一个连通的局部极小值区域。运用分水岭算法，将得到的分水岭作为外部标号。这些外部标号将图像分成多个区域，每个区域仅包含一个内部标号和一部分背景。这样分割问题简化成将这些区域一分为二的问题：一个目标和它的背景。

选择标号有许多方法，简单的仅考虑灰度和连通性，复杂的还可考虑尺寸、形状、纹理、相互位置和距离等。这些可根据对具体应用的先验知识来帮助决定。利用标号可将先验知识加进分割过程，从这个角度说，分水岭算法提供了一个借用先验知识帮助分割的框架。

下面介绍一种利用标号克服过分割的具体方法，称为**标号控制的分割**[Jähne 2000]。该技术的基本思路是除了对输入图像中的轮廓进行增强以获得分割函数图像(前面求梯度图就是一种特例)外，还用**强制最小值**技术(见下)先对输入图像进行滤波处理。具体就是借助特征检测确定一个标号函数，这个标号函数标示出目标和对应的背景(也可看作标号图像)。将这些标号强制加到分割函数中作为极小值，然后计算分水岭得到分割结果。整个流程可参见图 3.4.7，其中细线框代表与外界交互的模块。

这里目标的标号可从图像中用特征检测的方法提取出来。对合适特征的选取与先验知识或对图像目标性质的假设有关，常用的包括图像极值、平坦区域等，必要时也可手工确定标号。对每个目标区域都要确定一个标号，因为标号和最终的分割区域是一对一的。标号

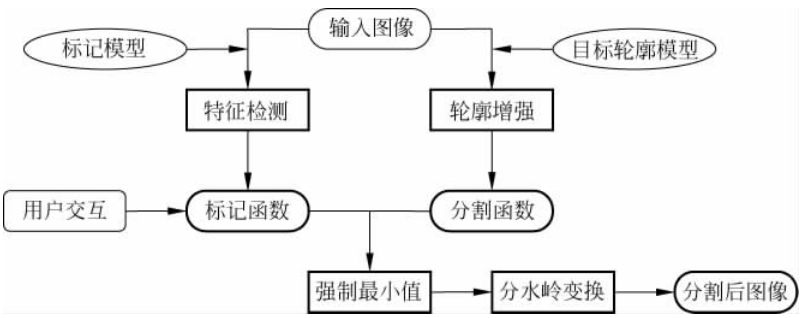

图 3.4.7　标号控制分割方法的流程框图

区域的尺寸可大可小(最小一个像素),如果处理噪声较大的图像,标号区域也要较大。目标标号确定后,与分割函数图像结合使用。最后通过计算滤波后的分割函数图像的分水岭而得到目标的边界。

前面提到的强制最小值技术是一种基于测绘算子的方法。**测绘算子**也是一类形态学算子(更多介绍见第 12 章和第 13 章),其主要特点是使用两幅输入图像。先将一个形态学算子(基本的膨胀和腐蚀算子)作用于第一幅图像,再要求结果保持大于或等于第二幅图像。

在使用强制最小值技术时,认为与图像特征对应的标号已知。对每个像素(x,y),其标号图像 $f_m(x,y)$可定义为

$$f_m(x,y)=\begin{cases}0 & (x,y)\text{ 属于标记}\\ 255 & \text{其他}\end{cases} \tag{3.4.5}$$

强制最小值的计算分两个步骤进行:

(1) 首先逐点计算输入图像 $f(x,y)$和标号图像 $f_m(x,y)$的最小值:$f_{\min}(x,y)=\min\{f(x,y),f_m(x,y)\}=f\wedge f_m$。这里 $\wedge$ 表示取最小值的操作,并在对应标号的地方保留最小值,这样得到的图像将总是小于或等于标号图像。如果需要强制的两个最小值都已经属于 $f(x,y)$在 0 级时的一个最小值,这时就需要考虑 $f_{\min}(x,y)+1=\min\{f(x+1,y+1),f_m(x,y)\}+1=(f+1)\wedge f_m$ 而不是 $f\wedge f_m$。

(2) 从标号图像 $f_m(x,y)$出发腐蚀 $f_{\min}(x,y)$直到稳定(类似 13.4 节中图像聚类快速分割时用到的最终腐蚀),这可表示为

$$R^{*}_{[(f+1)\wedge f_m]}(f_m) \tag{3.4.6}$$

图 3.4.8 给出对一个 1-D 信号计算强制最小值的示例。图(a)中的信号 $f(x)$有 7 个极小值,而标号信号 $f_m(x)$有两个极小值,这两个极小值借助腐蚀加到 $f(x)$上。图(b)给出逐点计算 $f(x)$和 $f_m(x)$的最小值得到的结果。图(c)给出从标号图像 $f_m(x)$出发腐蚀$(f+1)\wedge f_m$ 直到稳定的结果。

例 3.4.3　使用标号控制分割

图 3.4.9 给出一个使用标号控制分割技术的示例。生物细胞在图像上呈现为块状区域,且常接触或覆盖。为将这样的区域区分开,可使用标号控制的分割技术。图 3.4.9(a)表示部分覆盖的两个区域。图 3.4.9(b)表示经过距离变换的结果(即 1.4 节中的**距离图**),因为它的局部极小值和需要分开的区域有一对一的关系,所以可用作标号函数(这里取围绕局部极小值的小区域为标号)。从图 3.4.9(b)中检测出的分水岭可将两个部分覆盖的区域

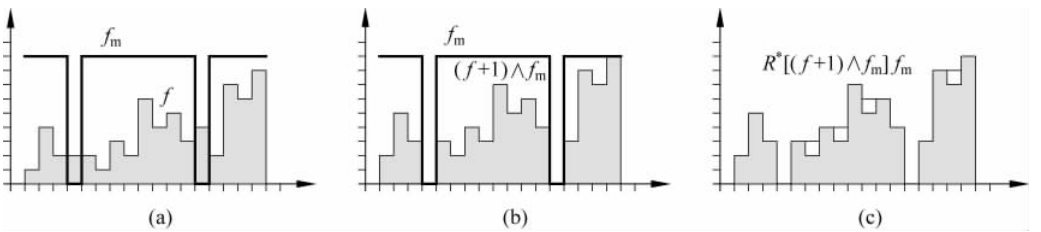

图 3.4.8　强制最小值技术示例

分割开来，结果见图 3.4.9(c)。

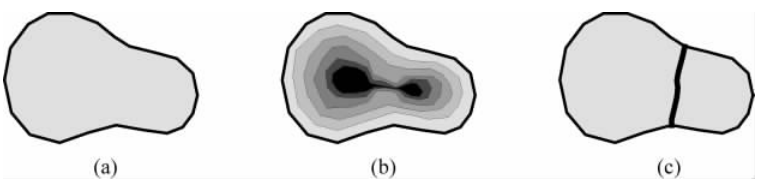

图 3.4.9　分割互相覆盖的块状区域　□

2. 分水岭算法的扩展

分水岭算法不仅可以在图像域(灰度图和梯度图)中使用，还可以在特征域中使用。下面介绍一种在 3-D 颜色直方图中用分水岭算法找出颜色聚类进行**彩色图像分割**的方法。其主要步骤为[Dai 2003]：

(1) 选择合适的颜色空间(这里选了 $L^*a^*b^*$，主要考虑颜色空间中各颜色间的相对欧氏距离应与人的视觉特性有尽可能接近正比的关系)，做出待分割图像的 3-D 颜色直方图。实际中为减少计算量，可将颜色空间量化以减少 3-D 颜色直方图中直方柱的数量。

(2) 将 3-D 颜色直方图进行反转变换，使其中极小值的位置对应颜色空间中像素个数最多的颜色。对自然图像来说，同一个区域内部的颜色通常比较接近，反映在直方图上会形成山峰形状。当将直方图反转过来时，区域内部的颜色将对应局部极小值。

(3) 在不同的颜色聚类之间建立分水岭，将颜色直方图分割开来，获得颜色聚类。分水岭算法所确定的分水岭位置是直方图中对应图像中不同颜色区域的边界位置，在这些位置能将各个对应不同颜色区域的颜色聚类区分开。

(4) 将聚类结果映射回图像域中，得到各个分割后的区域。在直方图上找到了颜色聚类的边界，那么在原始图像上也就找到了不同颜色区域的轮廓。

(5) 对上面的结果进行后处理，最终得到分割图像。对颜色丰富的自然图像，常会出现过分割的情况，有些颜色区域非常小，不对应有意义的目标。此时可先确定一个面积阈值将这些过小区域除去。对属于这些小区域的像素可利用诸如最高置信度优先算法等将它们分到其他区域中。

上述第(3)步中建立分水岭的方法是一种迭代的方法。先将反转直方图中的极小值点依次计算标号，然后取出尚没有标号的最小的直方柱，对它根据其邻域(这里需采用 26-邻域)的情况计算标号。如果所考虑的最小直方柱的 26-邻域中出现了多于一种的标号，则可判定它对应颜色聚类之间的分水岭；否则将其邻域中已标记直方柱的标号值赋给它。这个过程反复进行，最后就可找出所有分水岭的位置。

例 3.4.4 分水岭算法分割彩色图像

图 3.4.10 给出几组用分水岭算法分割彩色图像得到的结果[Dai 2006]。第 1 列是原始图像,第 2 列是处理前的结果,第 3 列是处理后的结果。

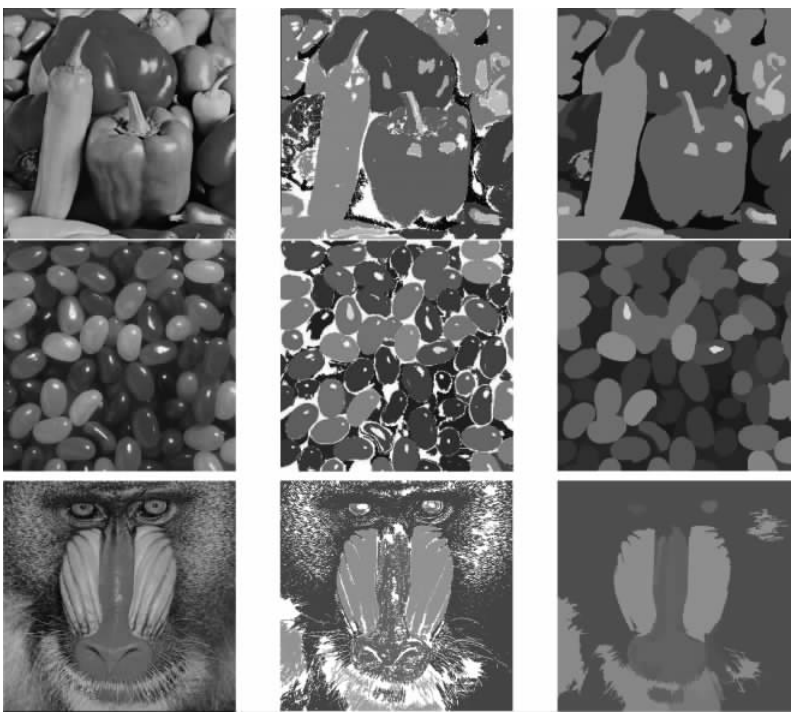

图 3.4.10 分水岭算法分割彩色图像的结果 □

总结和复习

为更好地学习,下面对各小节给予概括小结并提供一些进一步的参考资料;另外给出一些思考题和练习题以帮助复习(文后对加星号的题目还提供了解答)。

1. 各节小结和文献介绍

3.1 节详细介绍了一种很有特色的边缘角点检测算子——最小核同值区算子。与 2.2 节所介绍的微分算子不同,最小核同值区算子利用了一些积分的性质。它通过对模板覆盖像素的统计,在一定程度上有利于减少噪声的影响。角点检测的方法还有很多,如 Moravec 检测器、Zunigh-Haralick 检测器、Kitchen-Rosenfeld 检测器、哈里斯角点检测器等[Sonka 2008],以及非对称闭合方法[Shih 2010]。这些检测器从原理上讲,有些是很接近的,虽然步骤顺序不完全一样[Davies 2012]。

3.2 节介绍的图割方法与传统的图搜索方法有类似的框架,虽然在具体代价的选择上和解优化问题的方法上有所不同。图搜索方法借助梯度定义代价,求解 A * 算法来计算最优路径。图割方法在代价上考虑了像素的灰度以及像素间的灰度差,在求解中使用的最大流问题解法也是一种优化方法。它们作为串行方法,都利用了全局信息,或者说采用了整体

决策。对图割方法的更一般的介绍见文献[Boykov 2006a],利用图割技术分割高维图像也可参见文献[Boykov 2006b]。有关最大流和最小割相关定理的一些数学证明可见[Schrijver 2003]。

3.3 节讨论并行区域分割技术。在取阈值分割时,除了可根据对图像灰度统计的直方图来进行外,还有许多种有特色的方法。本节介绍的借助小波变换的方法仍利用直方图但结合了多分辨率特性来帮助进行阈值选取。本节介绍的借助过渡区的方法则先利用轮廓区域性质缩小阈值的可能范围再进行选取。基于过渡区选择阈值的思路简捷直观,为许多基于过渡区的分割方法提供了基础。加速计算过渡区边界线灰度值的上下界的一种方法见文献[Zhang 1991a]。另外,文献[章 2014b]、[章 2015c]和[Zhang 2018a]对第一个基于过渡区的图像分割方法提出四分之一个世纪来的研究工作借助谷歌学术工具进行了搜索,并以搜索到的文献为基础对近年来相关的研究进展进行了全面的统计和回顾。均移技术也常常被称为均移滤波,不仅可用于空间聚类来对图像进行分割,也可用于视频分割[Gu 2006]。

3.4 节介绍的分水岭分割算法可看作一种串行区域技术,虽然它直接得到的是区域的轮廓(如果参照图 3.4.5,这一点就更清楚了)。它可借助地形学概念进行分析,比较直观。它不仅可以在图像域(灰度图和梯度图)中使用,还可以在特征域中使用。教材[Gonzalez 2008]和[Sonka 2008]中对分水岭分割算法都有较详细的介绍。其他借助区域层次结构进行串行分割的技术也得到较多关注,一个应用示例可见文献[Shen 2009a]。

2. 思考题和练习题

3-1 拉普拉斯算子和海森算子都是基于二阶导数的算子,但效果为什么不同?它们可用于交叉点的检测吗?为什么?

3-2 从微分运算和积分运算的特点出发,比较最小核同值区算子与第 2 章介绍的梯度算子的优缺点。

3-3 哈里斯兴趣点算子与拉普拉斯算子和海森算子有什么联系?

3-4 图割方法与图搜索方法均是基于图结构进行图像分割的方法,它们的主要区别是什么?所导致的两种方法的主要优缺点各是什么?

3-5 分析和讨论如何将 3.2 节介绍的本质上实现二值分割的图割技术推广到多个不同灰度目标区域的多值分割中去。

3-6 试讨论 3.3.1 节的多分辨率阈值选取方法受图像中噪声影响的情况。

*__3-7__ 除了可根据过渡区中所有像素来确定一个阈值以进行分割外,还可直接利用 L_{high} 和 L_{low} 的值来直接计算阈值。试列举几种方法。

*__3-8__ 试画出典型的 $EAG_{high}(L)$ 曲线以及 $TP_{high}(L)$ 和 $TG_{high}(L)$ 曲线并分析它们形成的原因。

3-9 给定一幅具有单一目标的灰度图像,可以先计算其梯度图,然后利用均移技术在灰度值和梯度值散射图上确定聚类来对图像进行分割。列出这个过程中的主要步骤,讨论其中可能会遇到的问题,并分析解决方法。

3-10 图 3.4.4 画出的实际是 2-D 图像的一个剖面,要求

(1) 画出对应的 2-D 图像的示意图,标出 $C_n(M_i)$ 和 $C_n(M_{i+1})$;

(2) 在示意图上添加 $n+1$ 时的情况。设此时 $C_{n+1}(M_i)$ 和 $C_{n+1}(M_{i+1})$ 汇合,画出分

水岭。

3-11 为克服分水岭分割算法有可能产生过分割的问题，人们采用过将小区域合并，对图像梯度阈值化，先用小波变换对图像进行多分辨率分析等方法。试查阅相关文献，并对比讨论各种方法的优缺点。

3-12 图像分割问题的求解过程在很多情况下可看作一个优化的过程。以本章介绍的各种方法为例，分别分析和讨论它们的优化手段和目标。