数字图像处理

Digital Image Processing

信息工程学院

School of Information Engineering

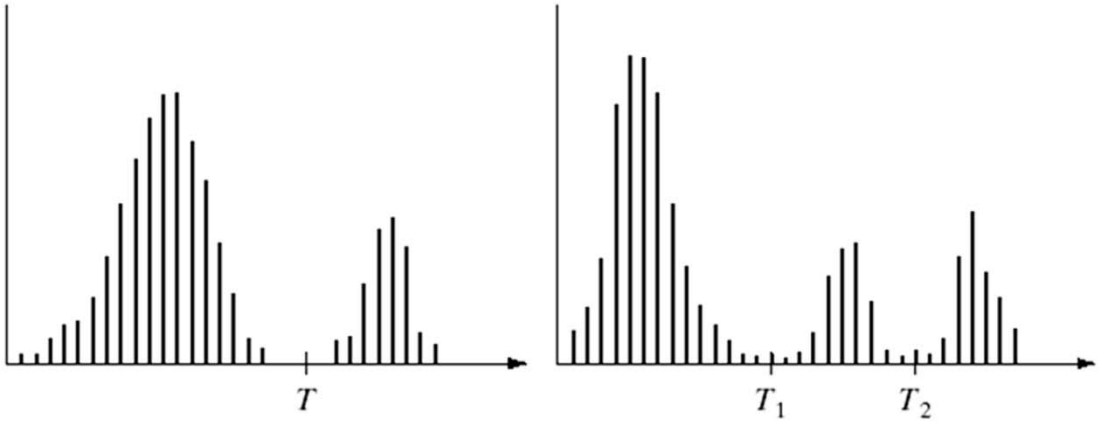
8.4 阈值分割

王 昱 主讲

8.4 阈值分割（Image Segmentation using Threshold）

- 阈值分割: 将所有灰度值大于或等于某阈值的像素都被判属于物体;将所有灰度值小于该阈值的像素被排除在物体之外。
- 适用于物体与背景有较强对比的景物的分割。
- 由于阈值分割的直观性和易于实现的性质,使它在图像分割应用中处于中心地位。

基本原理



a b

上图(a)为一幅图像的灰度级直方图,其由亮的对象和暗的背景组成. 对象和背景的灰度级形成两个不同的模式. 选择一个门限值T, 可以将这些模式分开. (b)包含3个模式.

基本原理

- 原始图像——f(x,y)
- 灰度阈值——T
- 阈值运算得二值图像——g(x,y)

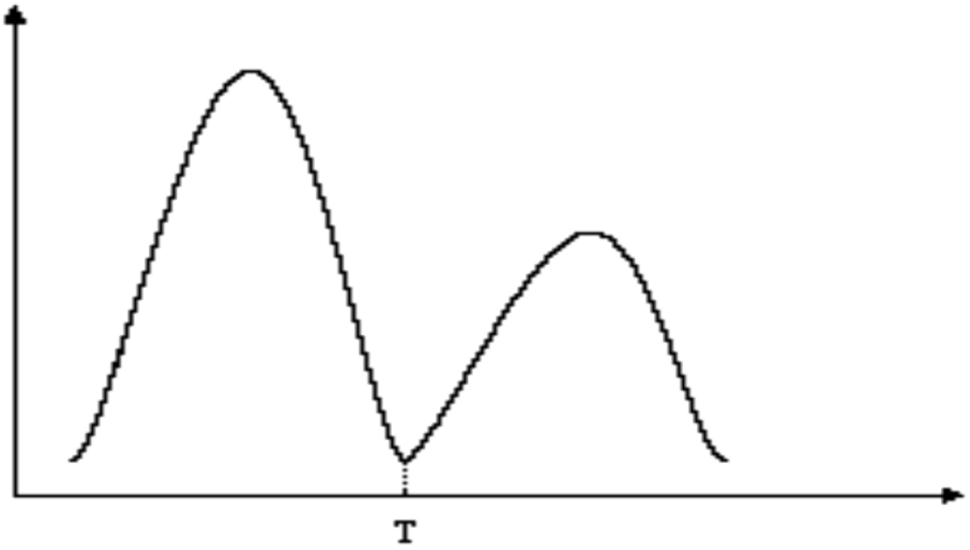
$$g[x,y]=\begin{cases}1 & \text{如果 } f[x,y]\geq T \\ 0 & \text{如果 } f[x,y]<T\end{cases}$$

对象点

背景点

阈值选择直接影响分割效果，通常可以通过对灰度直方图的分析来确定它的值。

阈值选择



利用灰度直方图求双峰或多峰
选择两峰之间的谷底作为阈值

人工阈值

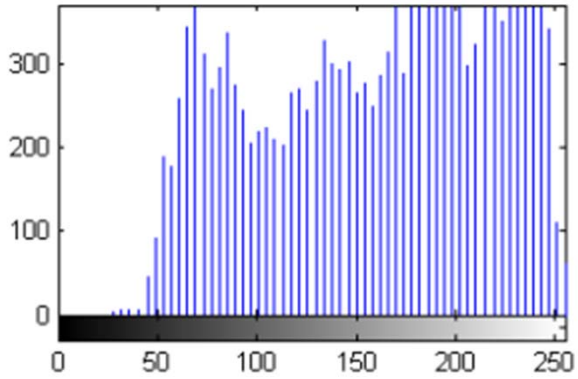
- 人工选择法是通过人眼的观察，应用人对图像的知识，在分析图像直方图的基础上，人工选出合适的阈值。
- 也可以在人工选出阈值后，根据分割效果，不断的交互操作，从而选择出最佳的阈值。

人工阈值

原始图像

T=155的二值化图像

T=210的二值化图像



图像直方图

全局阈值

- ✓ 采用阈值确定边界的最简单做法是在整个图像中将灰度阈值的值设置为常数。
- ✓ 如果背景的灰度值在整个图像中可合理地看作为恒定，而且所有物体与背景都具有几乎相同的对比度，那么，只要选择了正确的阈值，使用一个固定的全局阈值一般会有较好的效果。

自适应阈值

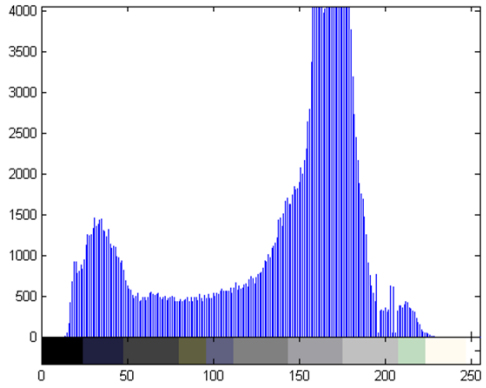
- 在许多的情况下，背景的灰度值并不是常数，物体和背景的对比度在图像中也有变化，这时，一个在图像中某一区域效果良好的阈值在其它区域却可能效果很差。
- 在这种情况下，把灰度阈值取成一个随图像中位置缓慢变化的函数值是适宜的。

最佳阈值的选择

需要一个最佳的，或至少是具有一致性的方法确定阈值。

求阈值-直方图技术

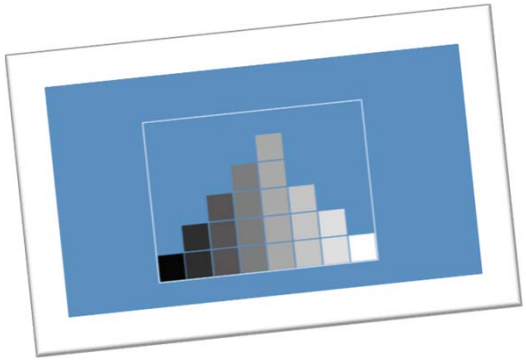
- 一幅含有一个与背景明显对比的物体的图像，其有包含双峰的灰度直方图（如图）。
- 两个尖峰对应于物体内部和外部较多数目的点。
- 两峰间的谷对应于物体边缘附近相对较少数目的点。
- 在类似这样的情况下，通常使用直方图来确定灰度阈值的值。



直方图生成　imhist(a)

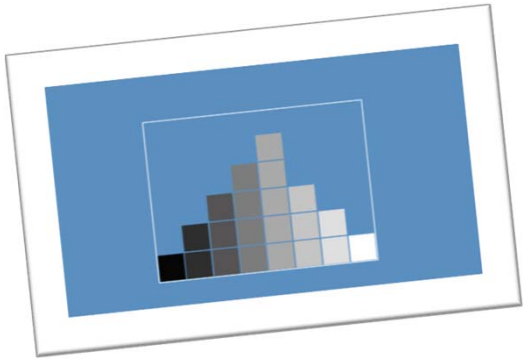
求阈值-直方图技术

- 一种更可靠的方法是把阈值设在相对于两峰的某个固定位置，如中间位置上，这两个峰分别代表物体内部和外部点典型（出现最频繁）的灰度值。
- 一般情况下，对这些参数的估计比对最少出现的灰度值，即直方图的谷的估计更可靠。



求阈值-最大类间法

- 又称为OTSU（大津）算法，该算法是在灰度直方图的基础上用最小二乘法原理推导出来的，具有统计意义上的最佳分割阈值。
- 基本原理是以最佳阈值将图像的灰度直方图分割成两部分，使两部分之间的方差取最大值，即分离性最大。



求阈值-最大类间法

设*X*是一幅具有L级灰度级的图像，其中第i级像素为N_i个，其中i的值在0到L-1之间，图像的总像素点个数为：

$$N = \sum_{i=0}^{L-1} N_i$$

第i级出现的概率为：$P_i = \frac{N_i}{N}$

以阈值k将所有的像素分为目标和背景两类。其中C_0类的像素灰度级为0到k-1，C_1类的像素灰度级为k到L-1。

图像的总平均灰度级为：$\mu = \sum_{i=0}^{L-1} iP_i$

求阈值-最大类间法

C_0类像素所占面积的比例为：$w_0 = \sum_{i=0}^{k-1} P_i$

C_1类像素所占面积的比例为：$w_1 = 1 - w_0$

C_0类像素的平均灰度为：$\mu_0 = \mu_0(k) / w_0$

C_1类像素的平均灰度为：$\mu_1 = \mu_1(k) / w_1$

其中：

$$\mu_0(k) = \sum_{i=0}^{k-1} iP_i$$

$$\mu_1(k) = \sum_{i=k}^{L-1} iP_i = 1 - \mu_0(k)$$

则类间方差公式为：$\delta^2(k) = w_0(\mu - \mu_0)^2 + w_1(\mu - \mu_1)^2$

令k从0到L-1变化，计算在不同k值下的类间方差，使得最大时的那个k值就是所要求的最优阈值。

求阈值-迭代法

迭代式阈值选择的基本步骤如下:

(1) 选择图像灰度的中值作为初始阈值$T_i=T_0$。

(2) 利用阈值T_i把图像分割成两部分区域，R_1和R_2，并计算其灰度均值。

$$\mu_1=\frac{\sum_{i=0}^{T_i} in_i}{\sum_{i=0}^{T_i} n_i},\quad \mu_2=\frac{\sum_{i=T_i}^{L-1} in_i}{\sum_{i=T_i}^{L-1} n_i}$$

适用于背景和对象在图像中占据的面积相近的情况.

(3) 计算新的阈值T_{i+1}

$$T_{i+1}=\frac{1}{2}(\mu_1+\mu_2)$$

(4) 重复步骤2、3，直到T_{i+1}和T_i的值差别小于某个给定值

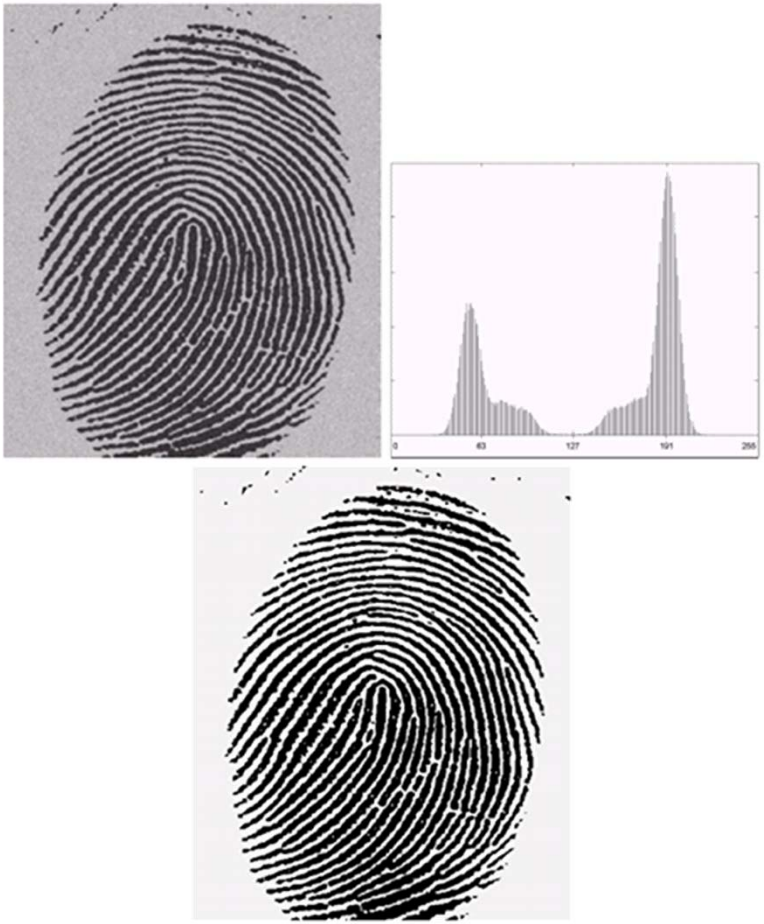
例8.2 全局阈值、OTSU及迭代求阈值算法。

```
I =
imread('i_boat_gray.bmp');
[width,height] = size(I);

%otsu algorithm
level = graythresh(I);
BW = im2bw(I,level);
figure
imshow(BW)
```

```
%global threshold
for i=1:width
    for j=1:height
        if(I(i,j) < 80)
            BW1(i,j) = 0;
        else
            BW1(i,j) = 1;
        end
    end
end
figure
imshow(BW1)
```

```
%迭代求阈值
I = double(I);
T = (min(I(:))+max(I(:)))/2;
done = false;
i=0;
while ~done
    r1 = find(I<=T);
    r2 = find(I>T);
    Tnew =
(mean(I(r1))+mean(I(r2)))/2;
    done = abs(Tnew -T)<1;
    T = Tnew;
    i=i+1;
end
I(r1) = 0;
I(r2) = 1;
Figure;
imshow(I)
```



a b
c

FIGURE 8.9
(a) Original image. (b) Image histogram. (c) Result of segmentation with the threshold estimated by iteration. (Original courtesy of the National Institute of Standards and Technology.)

a) 原图

b) 图像的直方图

c) 通过用迭代估计的门限对图像进行分割的结果

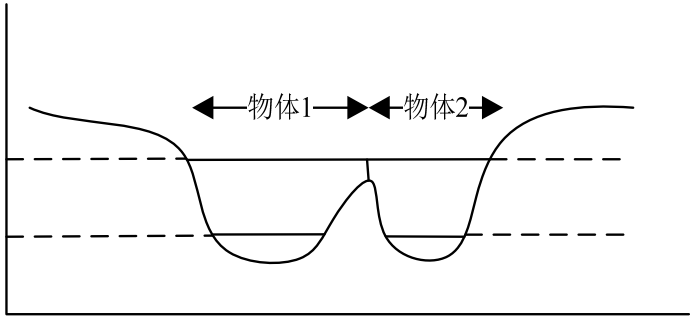
分水岭算法(Watershed Algorithm)

最常用的分水岭算法是F.Meyer在90年代早期提出的。

- 分水岭算法(watershed)是一种借鉴了形态学理论的分割方法，它将一幅图象看成为一个拓扑地形图，其中灰度值被认为是地形高度值。高灰度值对应着山峰，低灰度值处对应着山谷。将水从任一处流下，它会朝地势底的地方流动，直到某一局部低洼处才停下来，这个低洼处被称为**吸水盆地**，最终所有的水会分聚在不同的吸水盆地，吸水盆地之间的山脊被称为**分水岭**，水从分水岭流下时，它朝不同的吸水盆地流去的可能性是相等的。
- 将这种想法应用于图像分割，就是要在灰度图像中找出不同的吸水盆地和分水岭，由这些不同的吸引盆地和分水岭组成的区域即为我们要分割的目标。

分水岭算法(Watershed Algorithm)

假定图中的物体灰度值低，而背景的灰度值高。图中显示了沿一条扫描线的灰度分布，该线穿过两个靠得很近的物体。

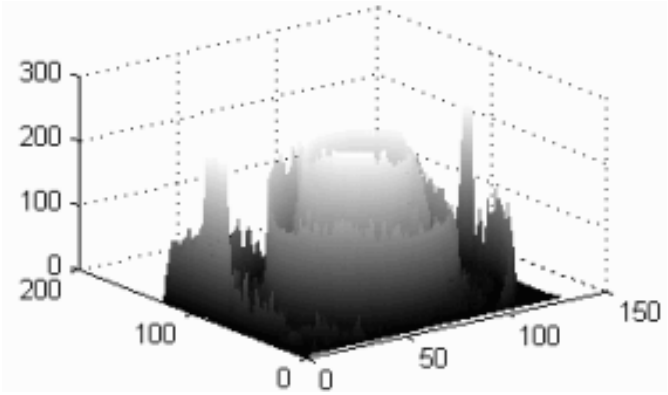


图像最初在一个低灰度值上二值化。该灰度值把图像分割成正确数目的物体，但它们的边界偏向物体内部。

随后阈值逐渐增加，每一次增加一个灰度级。物体的边界将随着阈值增加而扩展。当边界相互接触时，这些物体并没有合并。因此，这些初次接触的点变成了相邻物体间的最终边界。这个过程在阈值达到背景的灰度级之前终止。也就是说，在被恰当分割的物体的边界正确地确定时终止。

分水岭算法(Watershed Algorithm)

(a)原始图像



(b)图像对应的拓扑地形图

图8.23 图像对应的拓扑表面图

分水岭算法(Watershed Algorithm)

分水岭阈值选择算法可以看成是一种自适应的多阈值分割算法。

分水岭对应于原始图像中的边缘

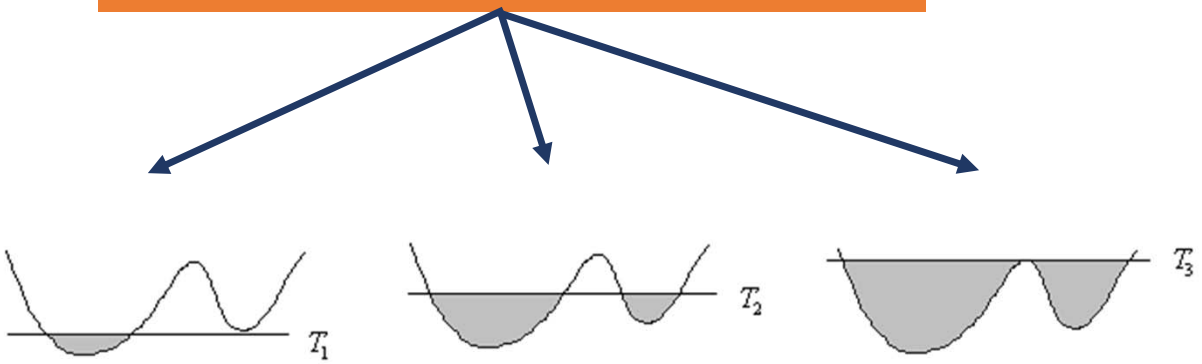


图8.24 分水岭形成示意图

分水岭算法(Watershed Algorithm)

- 分水岭算法不是简单地将图像在最佳灰度级进行阈值处理，而是从一个偏低但仍然能正确分割各个物体的阈值开始。然后随着阈值逐渐上升到最佳值，使各个物体不会被合并。这个方法可以解决那些由于物体靠得太近而不能用全局阈值解决的问题。只要也只有所采用最初的阈值进行分割的结果是正确的，那么，最后的分割也是正确的（即图像中每个实际物体都有相应的边界）。
- 最初和最终的阈值灰度级都必须很好地选取。
- 如果初始的阈值太低，那么低对比度的物体开始时会被丢失，然后随着阈值的增加就会和相邻的物体合并。
- 如果初始阈值太高，物体一开始便会被合并。最终的阈值决定了最后的边界与实际物体的吻合程度。


谢 谢
THANK YOU