第4章 二值图像处理

计算机科学系

第4章 二值图像处理

- 二值图像是指经过算法处理得到的表示目标和背景的图像。
- 将一般灰度图像转换成二值图像的过程，称为二值化。
- 二值图像的所有目标区域都是相同的像素值，需进一步处理，即二值图像处理

第4章 二值图像处理

4.1 灰度图像二值化

4.2 二值图像处理

第4章 二值图像处理

4.1.1 基本概念

- **灰度图像二值化（Image Binarization）**
  - **将n个亮度等级的灰度图像，通过适当转换而得到仍然可以反映图像整体和局部特征的二值化图像的过程。**
    - **二值化结果：**
    - **就是将图像上的像素点的灰度值设置为0和255，或者0和1，即将整个图像呈现出明显的黑白效果的过程。**
    - **二值化作用：**
    - **在数字图像处理中，二值图像占有非常重要的地位，图像的二值化使图像中数据量大为减少，从而能凸显出目标的轮廓**
    - **作为数字图像处理应用的一种预处理，简化处理过程**

4.1.1 基本概念

- 二值化-阈值分割
  - 利用图像中目标物与其背景在灰度特性上的差异，把图像作为具有不同灰度级的两类区域（目标和背景）的组合，选取一个合适的阈值，以确定图像中每一个像素点应该属于目标还是背景区域，从而产生相应的二值图像。
  - 要从复杂的景物中分辨出目标，并将其形状完整地提取出来，阈值的选取是阈值分割技术的关键。
    - » 如果阈值选取过高，过多的目标点被误认为背景，即漏检；
    - » 阈值选得过低，则会出现非目标像素当做目标像素的情况，即误检。
  - 至今还未有一种对所有图像都能有效分割的阈值选取方法。

4.1.1 基本概念

- **二值化-阈值分割**
  - **以一定的准则在原始图像$f(x,y)$中找出一合适的灰度值作为阈值，分割后的图像是$g(x,y)$**
  - **一般表示为：**

$$g(x,y)=\begin{cases} Z_E & f(x,y)\in Z \\ Z_B & otherwise \end{cases}$$

- $$g(x,y)=\begin{cases} 1 & t_1 \le f(x,y) \le t_2 \\ 0 & otherwise \end{cases}$$
- $$g(x,y)=\begin{cases} f(x,y) & t_1 \le f(x,y) \le t_2 \\ 0 & otherwise \end{cases}$$

第4章 二值图像处理

4.1.1 基本概念

4.1.2 全局阈值

4.1.3 局部阈值

4.1.4 基于边缘方法

4.1.5 文本图像二值化

4.1.2 全局阈值

- 全局阈值-直方图阈值分割
- 如果灰度级直方图呈明显的双峰状，则选取两峰之间的谷底所对应的灰度级作为阈值

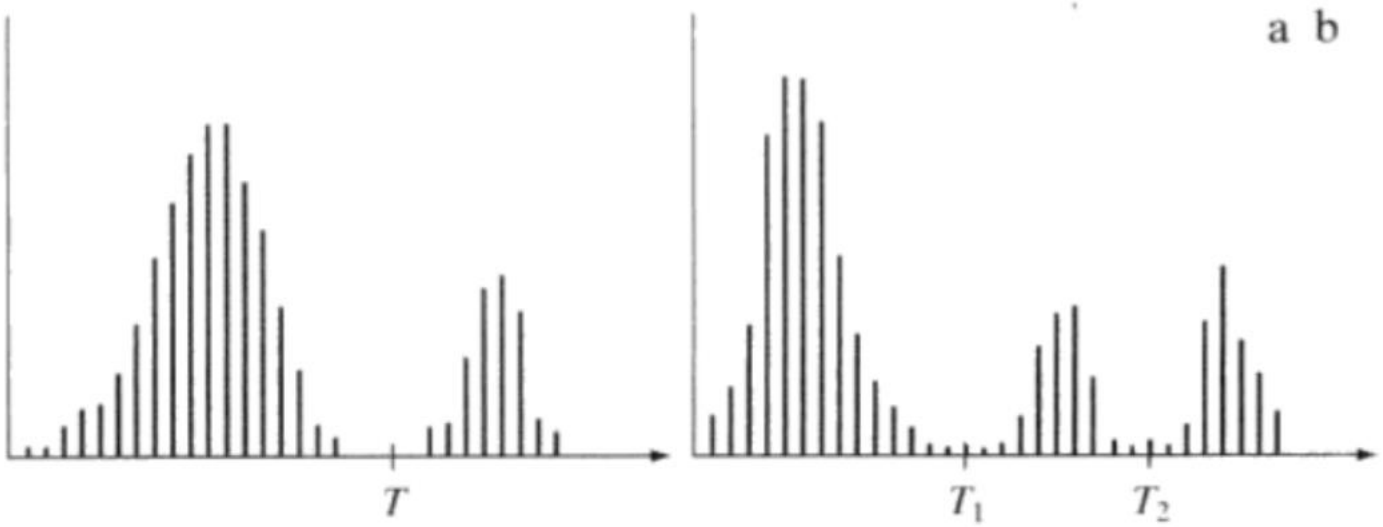
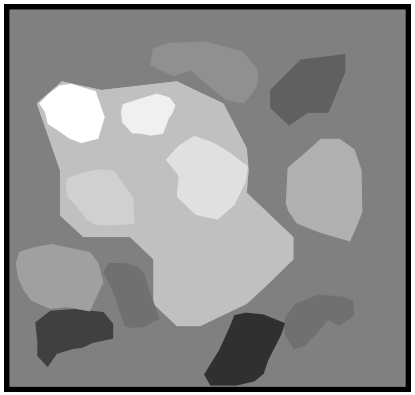
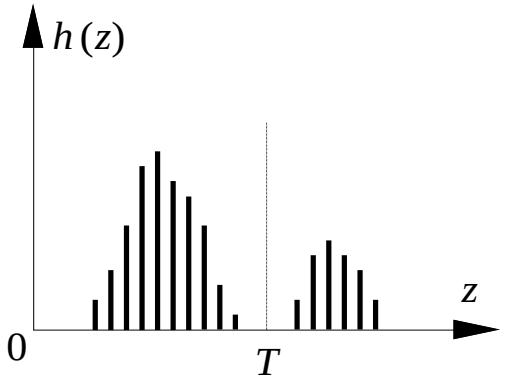


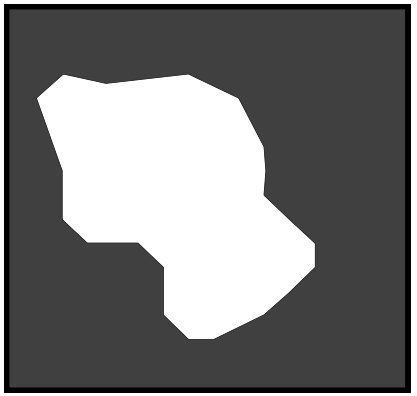
图 10.35　可被(a)单阈值和(b)双阈值分隔的灰度直方图

4.1.2 全局阈值

- 全局阈值-直方图阈值分割







4.1.2 全局阈值

- 全局阈值-直方图阈值分割
- 噪声影响

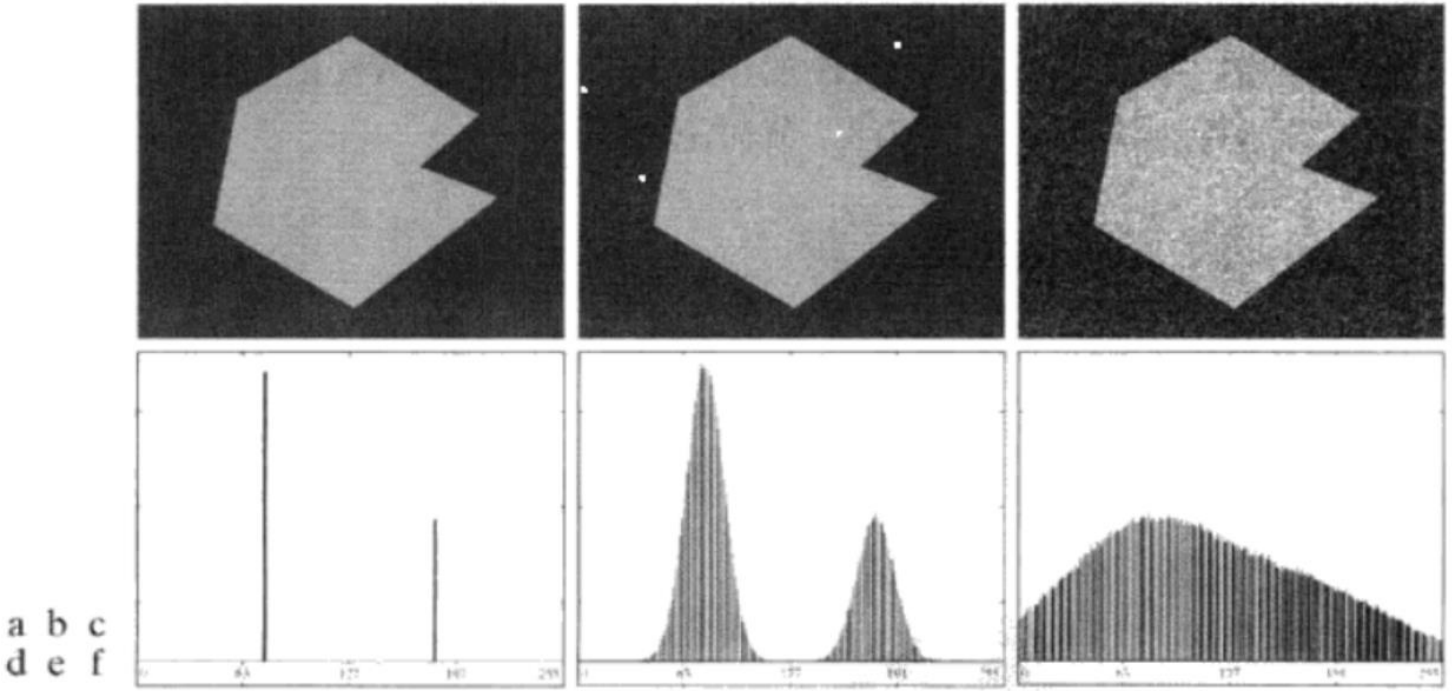


图 10.36 (a)无噪声的 8 比特图像；(b)带有均值为零、标准差为 10 个灰度级的加性高斯噪声的图像；(c)带有均值为零、标准差为 50 个灰度级的加性高斯噪声的图像；(d)~(f)相应的直方图

4.1.2 全局阈值

- 全局阈值-直方图阈值分割
- 光照影响

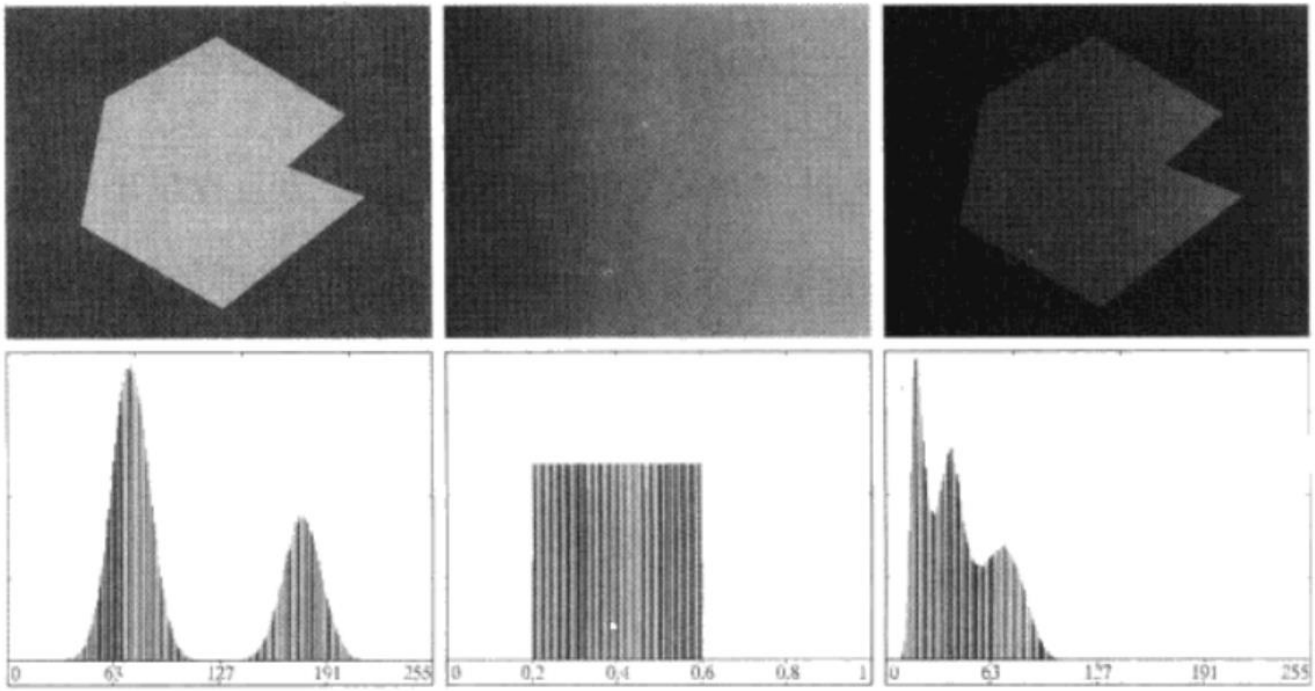


图 10.37 (a)带有噪声的图像；(b)在[0.2, 0.6]范围内的灰度斜坡图像；(c)图(a)和图(b)的乘积；(d)~(f)相应的直方图

4.1.2 全局阈值

- **全局阈值-迭代阈值分割**

通过迭代的方法产生阈值。具体方法如下：

1.用图像的平均灰度值作为初始阈值T

2.通过初始阈值，把图像的像素按灰度分成两组R_1和R_2

3.计算两组像素的平均灰度值，记为μ_1和μ_2

4.重新计算阈值$T=(\mu_1+\mu_2)/2$

5.重复2~5，直到前后两个阈值之间的差小于一个预定义的参数ΔT为止

4.1.2 全局阈值

- 全局阈值-迭代阈值分割

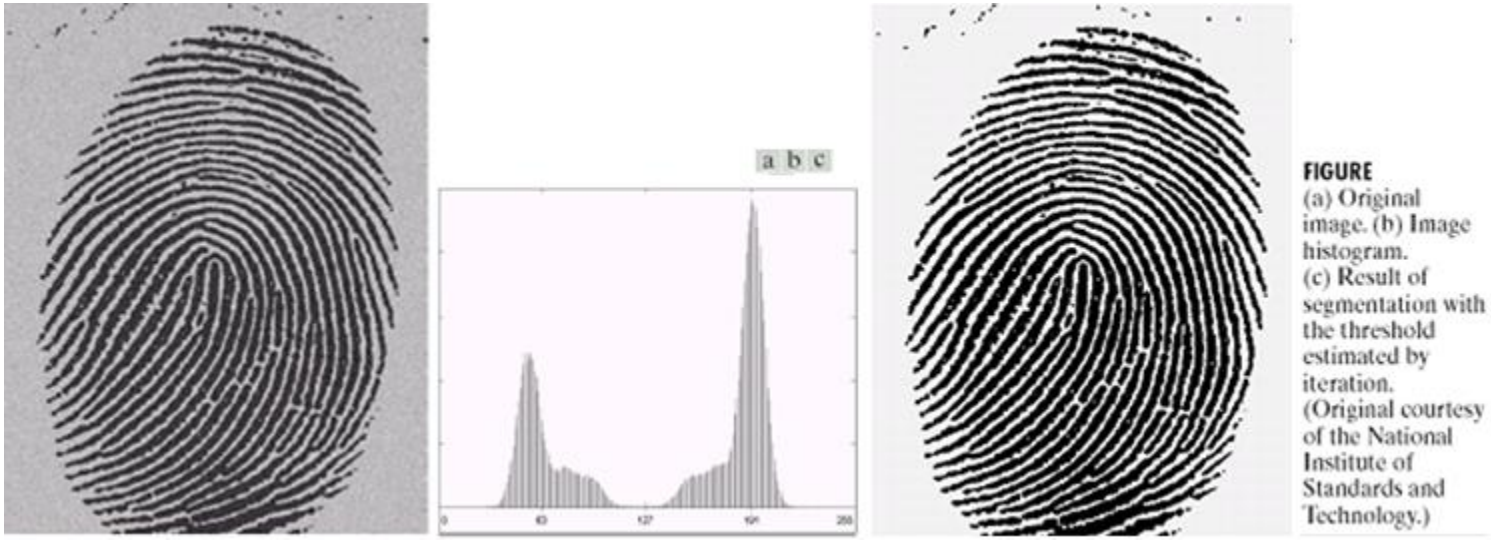


FIGURE (a) Original image. (b) Image histogram. (c) Result of segmentation with the threshold estimated by iteration. (Original courtesy of the National Institute of Standards and Technology.)

初值: $T = mean$， $\Delta T = 0$

迭代3次，T=125.4

4.1.2 全局阈值

- 全局阈值-用Otsu (大津)方法求最佳阈值分割
- **最佳阈值:**是指使图像中能给出最好的类间分离的阈值。
- Otsu方法的阈值在类间方差最大情况下最优。

4.1.2 全局阈值

- 设图像有L级灰度，n_i是第i级灰度的像素数，总像素数为MN，则归一化直方图为：$p_i = n_i/MN$
- 若选择阈值为：$T(k)$，$0 < k < L-1$
- 把图像分成两类：C_1，$[0, k]$；C_2，$[k+1, L-1]$
- 分到C_1的概率为：$P_1(k) = \sum_{i=0}^{k} p_i$
- 分到C_2的概率为：$P_2(k) = \sum_{i=k+1}^{L-1} p_i = 1 - P_1(k)$
- 分到C_1的灰度均值为：$m_1(k) = \frac{1}{P_1(k)} \sum_{i=0}^{k} ip_i$
- 分到C_2的灰度均值为：$m_2(k) = \frac{1}{P_2(k)} \sum_{i=k+1}^{L-1} ip_i$

4.1.2 全局阈值

- 整个图像的灰度均值（全局均值）为： $m_G = \sum_{i=0}^{L-1} ip_i$
- 得： $P_1 m_1 + P_2 m_2 = m_G$
- $P_1 + P_2 = 1$
- 其中： P_1是$P_1(k)$的简写
- 累加均值： $m = \sum_{i=0}^{k} ip_i$

4.1.2 全局阈值

- 图像全局灰度方差：
- $\sigma_G^2 = \sum_{i=0}^{L-1}(i - m_G)^2 p_i$
- 划分的C_1和C_2两类的类间方差，定义为：
- $\sigma_B^2 = P_1(m_1 - m_G)^2 + P_2(m_2 - m_G)^2$
- $\frac{P_1 m_1 + P_2 m_2 = m_G}{P_1 + P_2 = 1} P_1 P_2 (m_1 - m_2)^2 = \frac{(m_G P_1 - m)^2}{P_1(1 - P_1)}$
- 当m_1和m_2差距越大，则σ_B^2越大，所以类间方差可以度量两个划分的好坏。
- 计算使得σ_B^2最大的灰度值就是最优阈值。

4.1.2 全局阈值

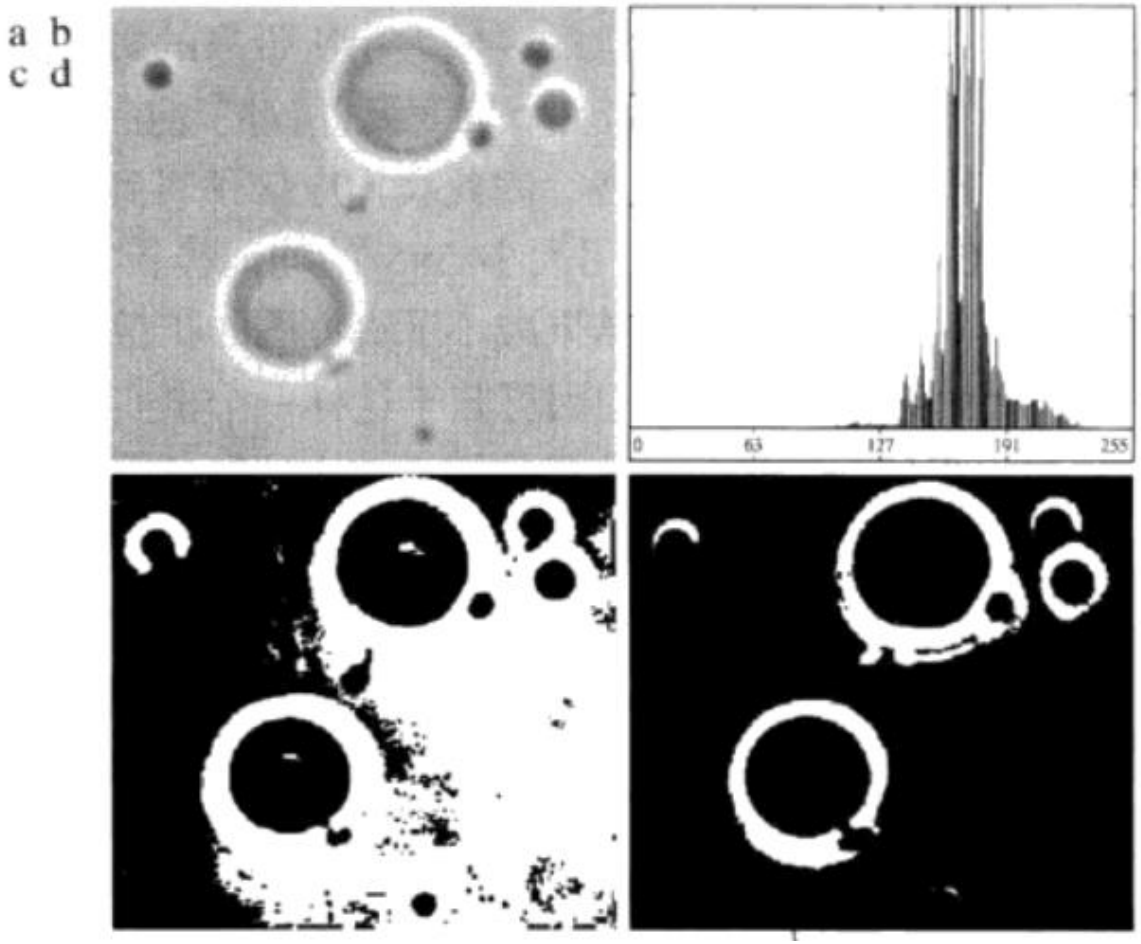
- 根据类间方差定义
- $\sigma_B^2(k) = \frac{\left(m_G P_1(k) - m(k)\right)^2}{P_1(k)\left(1 - P_1(k)\right)}$
- 最优阈值是使得$\sigma_B^2(k)$最大的阈值：
- $\sigma_B^2(k^*) = \max\limits_{0<k<L-1} \sigma_B^2(k)$
- 由于是数字图像，灰度值是量化后的，所以不能使用数学上的求导取零的方法
- 对所有k的取值，计算对应的$\sigma_B^2(k)$，进行比较，找最大的$\sigma_B^2(k)$，对应的k为所求最有阈值

4.1.2 全局阈值

- 算法过程
- 1.计算输入图像归一化直方图$p_i = n_i/MN$，$i = 0, 1, \cdots, L-1$
- 2.对$k = 0, 1, \cdots, L-1$，计算累计和$P_1(k) = \sum_{i=0}^{k} p_i$
- 3.对$k = 0, 1, \cdots, L-1$，计算累积均值$m(k) = \sum_{i=0}^{k} ip_i$
- 4.计算全局灰度均值 $m_G = \sum_{i=0}^{L-1} ip_i$
- 5.对$k = 0, 1, \cdots, L-1$，计算类间方差$\sigma_B^2(k) = \frac{\left(m_G P_1(k) - m(k)\right)^2}{P_1(k)\left(1 - P_1(k)\right)}$
- 6.查找其中最大的$\sigma_B^2(k)$，对应的k就是最优阈值
- 7.二值化

4.1.2 全局阈值

- 实例
- A原图
- B原图直方图
- C基本全局阈值结果
- D用Otsu方法结果



第4章 二值图像处理

4.1.1 基本概念

4.1.2 全局阈值

4.1.3 局部阈值

4.1.4 基于边缘方法

4.1.5 文本图像二值化

4.1.3 局部阈值

- 局部阈值(自适应阈值)
- 全局阈值的问题
- 在实际的情况中，当照明不均匀，有突发噪声或者背景灰度变化较大时，整幅图像分割时将没有合适的单一阈值，如果仍采用单一的阈值去处理每一个像素，可能会将目标区域和背景区域错误的划分。

4.1.3 局部阈值

- 局部阈值(自适应阈值)
- 局部阈值法思想，即图像中的每个像素对应的阈值可能不相同，每个像素单独计算阈值。
- 基本思路：每个像素的阈值由自身为中心的邻域窗口确定，依据邻域窗口中像素的灰度中值、均值或其他统计信息的组合作为阈值，再或者在此基础上加个常量值做调整。

4.1.3 局部阈值

- 局部阈值(自适应阈值)-Niblack算法与Sauvola算法
- 两种算法的输入是灰度图像，它以当前像素点为中心，根据当前像素点邻域内的灰度均值与标准方差来动态计算该像素点的阈值。

4.1.3 局部阈值

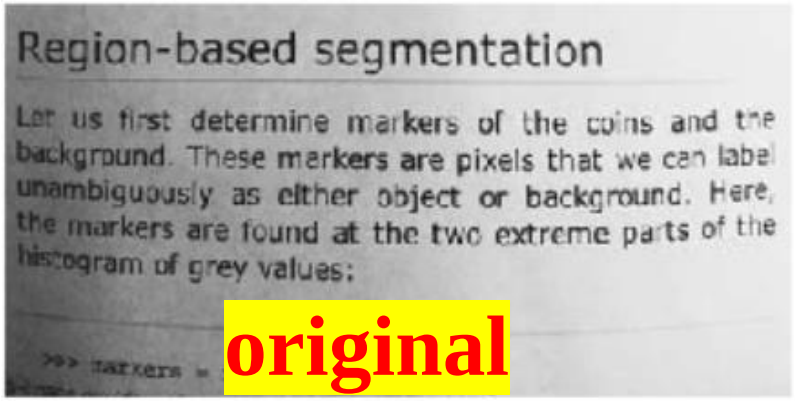
- 局部阈值(自适应阈值)-Niblack算法
- 1.计算$r \times r$邻域内的灰度均值$m(x,y)$与标准方差$s(x,y)$
- $$m(x,y) = \frac{1}{r^2}\sum_{i=x-\frac{r}{2}}^{x+\frac{r}{2}}\sum_{j=x-\frac{r}{2}}^{x+\frac{r}{2}} g(i,j)$$
- $$s(x,y) = \sqrt{\frac{1}{r^2}\sum_{i=x-\frac{r}{2}}^{x+\frac{r}{2}}\sum_{j=x-\frac{r}{2}}^{x+\frac{r}{2}}\left(g(i,j)-m(x,y)\right)^2}$$
- 2.计算像素点(x,y)的阈值$T(x,y)$
- $T(x,y) = m(x,y) + t \times s(x,y)$
- 其中，t是使用者自定义的一个修正参数，通常$t = -0.2$

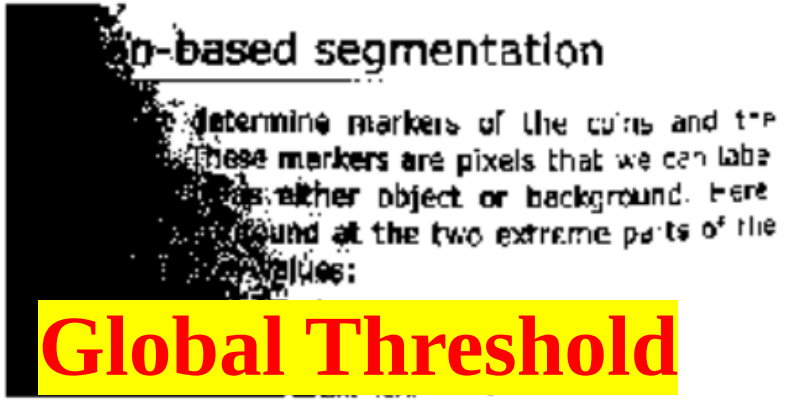
4.1.3 局部阈值

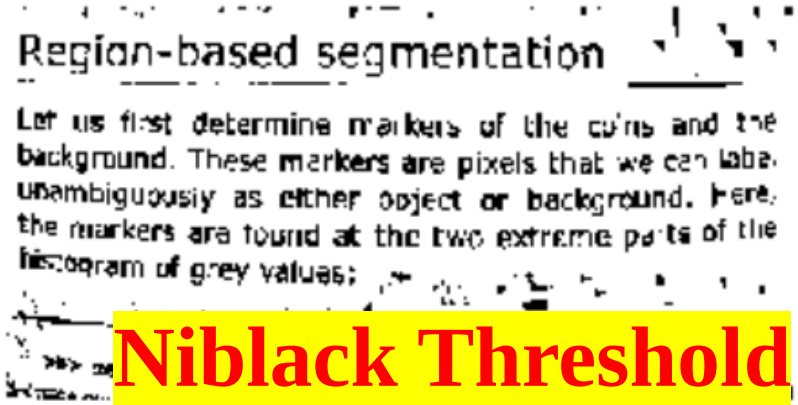
- **局部阈值(自适应阈值)-Sauvola算法**
- **1.计算$r \times r$邻域内的灰度均值$m(x,y)$与标准方差$s(x,y)$**
- $$m(x,y) = \frac{1}{r^2}\sum_{i=x-\frac{r}{2}}^{x+\frac{r}{2}}\sum_{j=x-\frac{r}{2}}^{x+\frac{r}{2}} g(i,j)$$
- $$s(x,y) = \sqrt{\frac{1}{r^2}\sum_{i=x-\frac{r}{2}}^{x+\frac{r}{2}}\sum_{j=x-\frac{r}{2}}^{x+\frac{r}{2}}\left(g(i,j)-m(x,y)\right)^2}$$
- **2.计算像素点(x,y)的阈值$T(x,y)$**
- $$T(x,y) = m(x,y) \times \left[1 + k \times \left(\frac{s(x,y)}{R} - 1\right)\right]$$
- **其中，R是标准方差的动态范围，当输入图像为8位灰度图像，灰度级256，则$R = 128$；k是使用者自定义的一个修正参数，k的取值对算法的结果影响不显著，通常$0 < k < 1$**

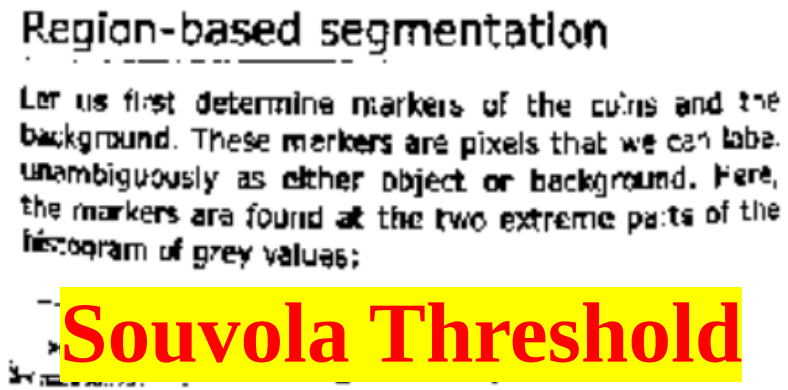
4.1.3 局部阈值

- 实例


Region-based segmentation
Let us first determine markers of the coins and the background. These markers are pixels that we can label unambiguously as either object or background. Here, the markers are found at the two extreme parts of the histogram of grey values:
original


Global Threshold


Region-based segmentation
Niblack Threshold


Region-based segmentation
Souvola Threshold

第4章 二值图像处理

4.1.1 基本概念

4.1.2 全局阈值

4.1.3 局部阈值

4.1.4 基于边缘方法

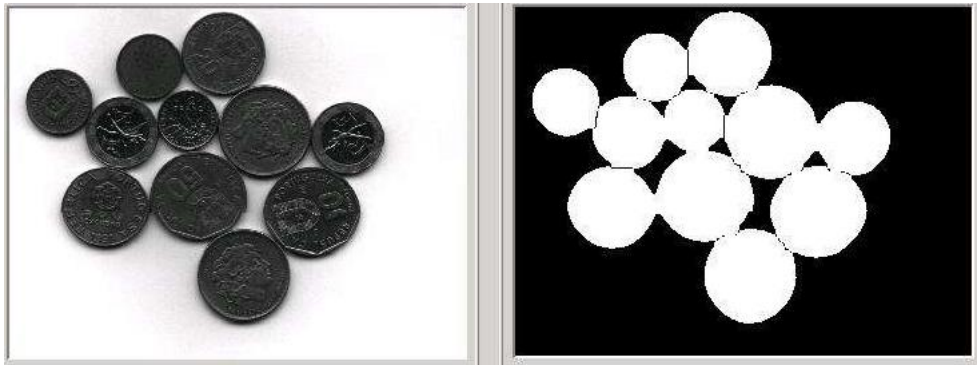
4.1.5 文本图像二值化

4.1.4 基于边缘方法

二值化-基于边缘检测的方法

1.使用边缘像素确定阈值

基本思想：二值化的阈值是图像中目标和背景之间边缘上像素的灰度值。



4.1.4 基于边缘方法

算法过程：

1.使用边缘检测方法提取图像的边缘；

可以使用Laplace算子，梯度算子，以及后续介绍的其他方法

2.对边缘图像进行边缘跟踪，找到图像边缘位置；

3.在原图像中，根据边缘位置找到边缘像素集合，取其均值作为二值化阈值。

4.1.4 基于边缘方法

实例：

原始图像

使用全局阈值效果

使用边缘阈值效果

使用Otsu阈值效果

第4章 二值图像处理

4.1.1 基本概念

4.1.2 全局阈值

4.1.3 局部阈值

4.1.4 基于边缘方法

4.1.5 文本图像二值化

4.1.5 文本图像二值化

二值化-与具体应用结合的方法-文本图像二值化

分析：

1.全局阈值法

对于图像有噪声或不均匀光照，适用性不好

2.局部阈值法

对于文本图像进行二值化，会出现伪影

局部阈值法的局部大小选择不容易

4.1.5 文本图像二值化

3.局部阈值法存在问题原因

- ✓ 局部阈值Bernsen算法
  - ✓ 使用局部窗口中最大最小值的均值作考察点的阈值
- ✓ 当窗口中没有目标点
  - 个别噪声点将引起阈值的突变，出现伪影现象
  - 背景灰度非均匀性也影响局部阈值变换
- ✓ 当考察窗口内均为目标点
  - 局部阈值被拉伸，使得本应同类的部分像素被二值化为背景，从而出现断笔现象
- ✓ 所以，要选合适的窗口大小

4.1.5 文本图像二值化

4. 窗口大小选择

- 文本图像二值化
- 窗口大小跟笔划宽度相关
- 如果窗口大小小于笔划宽度，则会出现窗口中全部是笔划像素，进而产生断笔现象
- 要求：窗口大小大于笔划宽度

4.1.5 文本图像二值化

解决思路：

结合全局阈值与局部阈值，根据实际问题（文本图像二值化）自动确定局部阈值的局部窗口大小

4.1.5 文本图像二值化

算法步骤：

1.预处理，对文本图像进行滤波，滤除图像中噪声。

- **均值滤波，中值滤波，或结合**

2.计算全局阈值*T*

- **迭代法，Otsu法**

3.计算笔划宽度

- **结合实际问题，进行计算**

4.文本图像逐点二值化

4.1.5 文本图像二值化

4.文本图像逐点二值化

根据像素(x,y)灰度值$g(x,y)$将像素分为两类

条件是：$T(1-\alpha) < g(x,y) < T(1+\beta)$

1）不满足该条件，进行全局阈值处理，利用下式

$$b(x,y)=\begin{cases}0 & g(x,y)<T(1-\alpha)\\ 1 & g(x,y)>T(1+\beta)\end{cases}$$

其中，$\alpha,\beta\in(0,1)$，通常取0.2~0.4

2）满足该条件，采用Bernsen算法处理

(1)利用前面计算的笔划宽度d，设计窗口宽度$w=2d+1$

(2)计算窗口阈值，进行二值化

第4章 二值图像处理

4.1 灰度图像二值化

4.2 二值图像处理

第4章 二值图像处理

4.2.1 基本概念

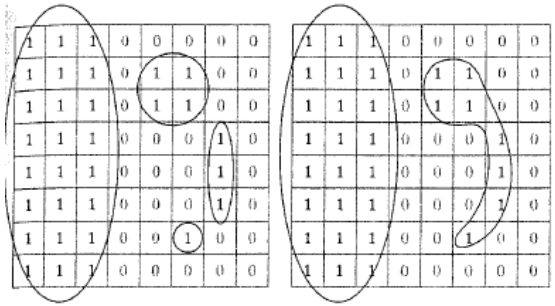
一、连接关系

指目标像素之间的连接。

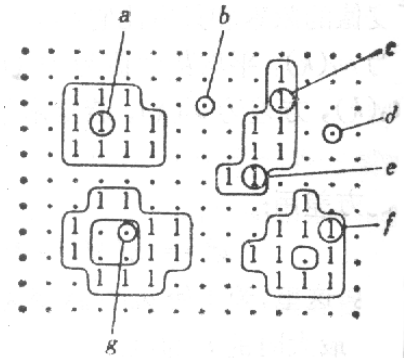
像素间的邻接、连接同之前定义，此处限指反映目标上的像素间的关系。

对于二值图像中目标上的两个像素A和B，若$p_0(=A),p_1,p_2,\cdots,p_{n-1},p_n(=B)$存在，并且$p_{i-1}$和$p_i$互为4-/8-/m邻接，那么像素A和B叫做4-/8-/m连接，以上的像素序列叫4-/8-/m连通。

连通区域，区域内所有像素存在连接关系。



a b
Connected components (a) Four 4-connected components. (b) Two 8-connected components.



4.2.1 基本概念

二、内部点与边界点

1. 目标区域：边界点与内部点组成
2. 内部点：与连接定义相关

   8连接：在8近邻中没有背景像素的点；

   4连接：在4近邻中没有背景像素的点；

3. 实例

	1	1	1	
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
	1	1	1	

图像

	1	1	1	
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
	1	1	1	

内点8连接定义

	1	1	1	
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
	1	1	1	

内点4连接定义

4.2.1 基本概念

三、连接数

1. 定义： 像素连接数是指进行包括孔的所有的边界线追踪时，通过目标上该像素近邻（4近邻或8近邻）像素的次数，叫做该像素的连接数。

2. 计算。【区分不同属性像素】

x_4	x_3	x_2
x_5	x_0	x_1
x_6	x_7	x_8

4近邻

$$N_c^{(4)}(x_0)=\sum_{k\in C}\left[f(x_k)-f(x_k)\cdot f(x_{k+1})\cdot f(x_{k+2})\right]$$

8近邻

$$N_c^{(8)}(x_0)=\sum_{k\in C}\left[\overline{f}(x_k)-\overline{f}(x_k)\cdot \overline{f}(x_{k+1})\cdot \overline{f}(x_{k+2})\right]$$

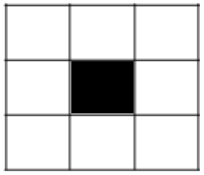
其中：$C=\{x_1,x_3,x_5,x_7\}$；$\overline{f}=1-f$；$x_9=x_1$。

4.2.1 基本概念

3. 连接数与像素属性

8近邻下的连接数与像素属性关系

连接数	像素特征
0	孤立点或内部点
1	端点或边界点
2	连接点
3	分支点
4	交叉点

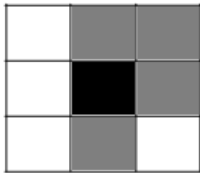


孤立点

内部点

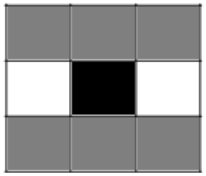
端点

边界点



边界点

连接点



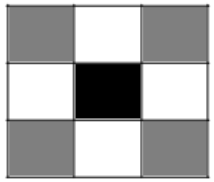
连接点

分支点

$$N_c^{(8)}(x_0)=\sum_{k\in C}\left[\bar{f}(x_k)-\bar{f}(x_k)\cdot\bar{f}(x_{k+1})\cdot\bar{f}(x_{k+2})\right]$$

其中：$C=\{x_1,x_3,x_5,x_7\}$；$\bar{f}=1-f$；$x_9=x_1$。

x_4	x_3	x_2
x_5	x_0	x_1
x_6	x_7	x_8



交叉点

第4章 二值图像处理

4.2.1 基本概念

4.2.2 贴标签

4.2.3 边界跟踪

4.2.4 细化

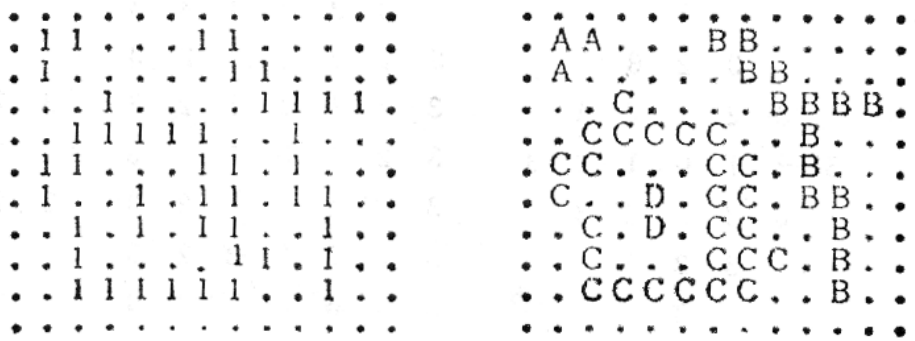
4.2.2 贴标签

一、定义

贴标签处理(Labeling)：是对二值图像的每个不同的连通区域进行编号。目的是为了区分不同的目标区域。

不同的对象被不同的整数值来标记，即每个区域被贴上标签以便于辨识。

通常设置一个与原图大小相同的标签矩阵，用于描述对二值图像不同连通区域的划分结果。



(a) 输入图象　　(b) 标记结果

4.2.2 贴标签

二、关键步骤

1. 按照从上到下，从左向右的顺序扫描所有值为1的像素，判断其是否与已经贴过的标签属于同一个连通区域，如果是则贴相同的标签，否则暂时认为是不同的区域，贴上新标签。

2. 校正贴好的标签：对在下方连接在一起的，已经贴为不同标签的像素归并为同一标签，并对整体的标签号进行调整。

判断像素是否属于同一个区域利用4或8连接判断。

1	1	1	0	0	0	0
1	1	1	0	1	1	0
1	1	0	0	1	1	0
1	0	0	1	0	0	1
0	0	0	0	0	1	1

二值图像

1	1	1	0	0	0	0
1	1	1	0	2	2	0
1	1	0	0	2	2	0
1	0	0	3	0	0	4
0	0	0	0	0	4	4

4连接标签

1	1	1	0	0	0	0
1	1	1	0	2	2	0
1	1	0	0	2	2	0
1	0	0	2	0	0	2
0	0	0	0	0	2	2

8连接标签

4.2.2 贴标签

三、具体步骤

设二值图像为f，标签矩阵为L

1. 设标签λ=0，已贴标签数N=0，按照从上到下，从左向右的顺序扫描，寻找值为1的目标像素。

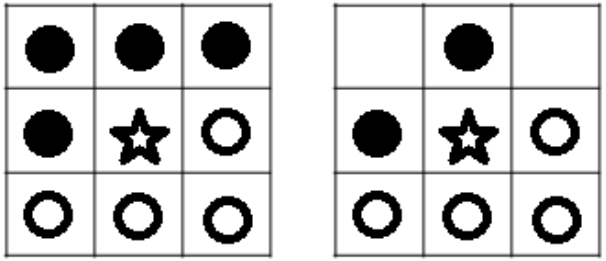
2. 对于尚未贴标签的目标像素，搜索其邻域已经扫描过的像素（如下图），对于已经扫描过的点，如果所有点的标签值为0，则λ=λ+1，L(i,j)= λ，N=N+1;如果所有点的标签值相同且不为0，则L(i,j)= λ；如果标签值不同，设标签值为λ、 λ’，且0<λ< λ’，则当前目标像素的标签值L(i,j)= λ，并将标签值为λ’所有像素的标签值改为λ，并令标签数N=N-1。

3. 将图像中所有像素按第2步处理。

4.2. 判断是否λ=N，若是则处理结束；

否则，说明已贴标签为不连续编号，

进行编号调整，使其标签号连续排列。●已扫描像素 ○未扫描像素 ☆当前



4.2.2 贴标签

四、实例

(a)

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	0	1	1	0	1	0
0	1	1	1	1	1	1	0	0	1	0	0	1	0
0	0	0	0	1	0	1	0	0	0	0	0	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	0	1	1	1	1	1	1	1	1	1	0
0	1	1	0	0	0	1	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

0 Background

1 Foreground

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

初始标签

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	0	2	2	0	3	0
0	4	4	4	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	0	2	2	0	3	0
0	1	1	1	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

4.2.2 贴标签

四、实例

(a)

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	0	1	1	0	1	0
0	1	1	1	1	1	1	0	0	1	0	0	1	0
0	0	0	0	1	0	1	0	0	0	0	0	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	0	1	1	1	1	1	1	1	1	1	0
0	1	1	0	0	0	1	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

0 Background

1 Foreground

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	0	2	2	0	3	0
0	1	1	1	1	1	1	0	0	2	2	0	3	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	0	2	2	0	3	0
0	1	1	1	1	1	1	0	0	2	2	0	3	0
0	0	0	0	1	0	1	0	0	0	0	0	3	0
0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	0	2	2	0	1	0
0	1	1	1	1	1	1	0	0	2	2	0	1	0
0	0	0	0	1	0	1	0	0	0	0	0	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	0	1	1	1	1	1	1	1	1	1	0
0	3	3	0	0	0	1	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

第4章 二值图像处理

4.2.1 基本概念

4.2.2 贴标签

4.2.3 边界跟踪

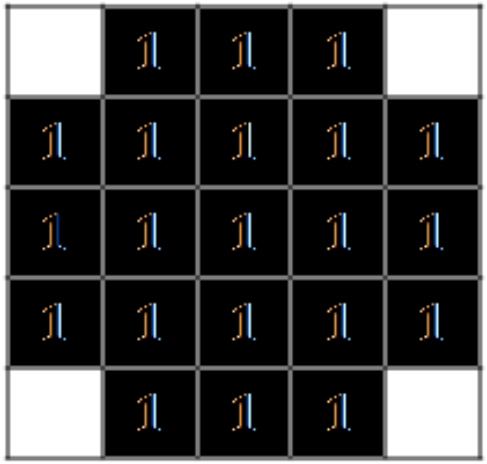
4.2.4 细化

4.2.3 边界跟踪

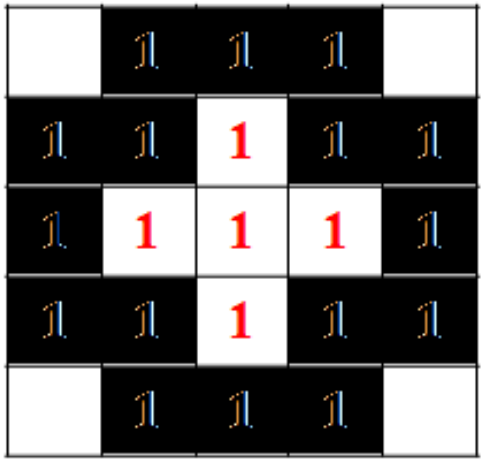
一、引言

边界跟踪：搜素单个连通区域的边界像素点集。

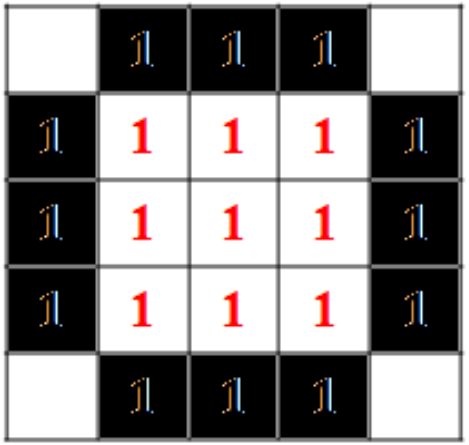
边界像素的连接关系有两种：8连接边界和4连接边界。常用8连接边界。



目标区域



4连接边界



8连接边界

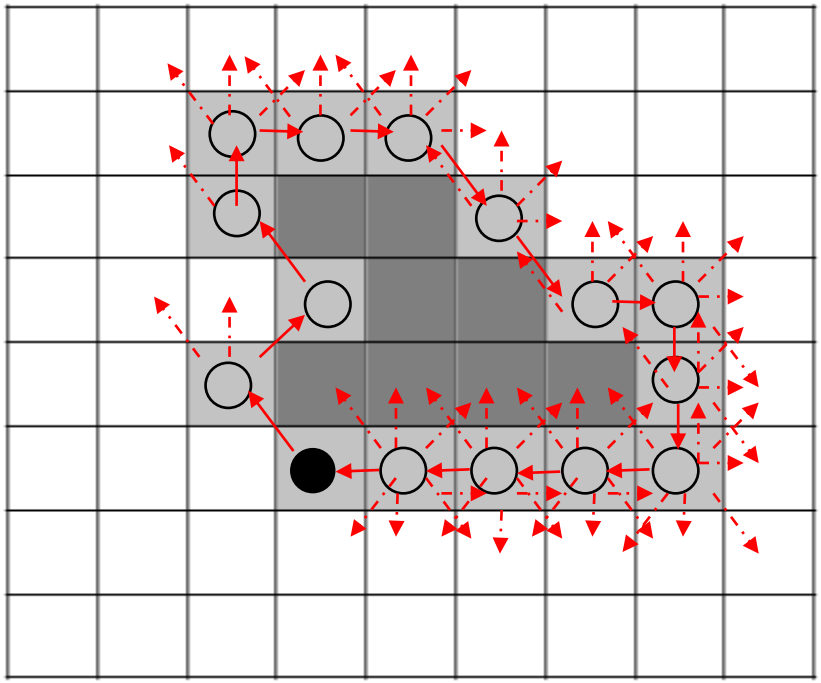
4.2.3 边界跟踪

二、跟踪算法

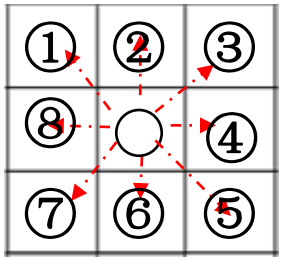
- 首先标记该目标区域的内部点，内部点为在4近邻中没有背景像素的点。保留下来的为边界点。
- 然后，找到该目标区域最左下方的边界点，作为跟踪的起点。
- 搜索：以这个边界点起始，定义搜索方向为沿左上方。如果左上方为边界点，则其为前一个边界点相邻的边界点，进行标记；否则，搜索方向顺时针旋转45°，继续寻找，直到寻找到下一个边界点为止。
- 结束：如果寻找到的相邻像素点为标记的起始点，则完成追踪。

4.2.3 边界跟踪

三、实例



搜索顺序



第4章 二值图像处理

4.2.1 基本概念

4.2.2 贴标签

4.2.3 边界跟踪

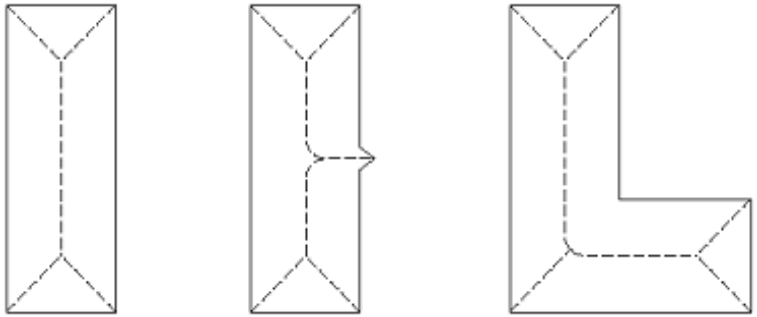
4.2.4 细化

4.2.4 细化

一、细化（Thinning）

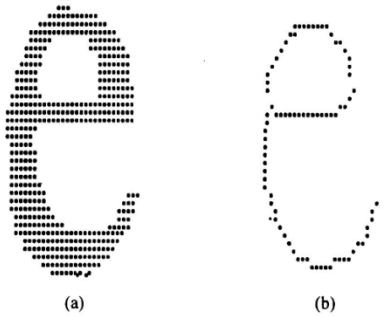
1. 骨架

目标区域的一种只有单像素宽的细化结构，反映了目标区域的形状特征。



2. 细化

是在不改变图像像素拓扑连接性关系的前提下，求一个目标区域骨架的过程。



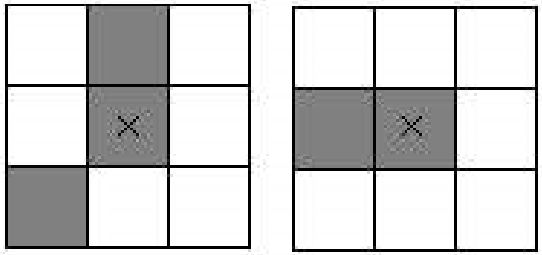
4.2.4 细化

二、细化规则

1. 在去除区域边界点时，不能消除破坏区域的连通性的点，如图(a)不能删除其中心像素。

2. 不能减小区域形状的的长度，也就是说迭代的过程中不能去掉端点(只有一个邻接点的点)。

3. 如果把边界分为上下左右四个方向，那么每次的迭代只能消除一个方向上的边界点，为了保持细化的结果尽量靠近骨架，也即位于中线附近，需要交替的对四个方向进行细化，比如采用上、下、左、右、上…的顺序。



(a)破坏连通性　(b)减小形状长度

4.2.4 细化

三、细化方法

1. 细化的过程

在不破坏连通性且不减小区域形状长度的条件下消去R中不是端点的简单边界点，过程是按S的上（北）、下（南）、左（西）、右（东）四个方向顺序，反复进行扫描以消去可删除简单边界点，直到不存在可以消去的简单边界点为止。

4.2.4 细化

2. 一种算法（E.S. Deutsch提出）

已知二值图像中目标点标记为1，背景点标记为0。扫描像素p_1（x, y）时，建立以p_1为中心的3 × 3窗口，周围的8个点分别表示为p_2，p_3，p_4，p_5，p_6，p_7，p_8，p_9。

p_9	p_2	p_3
p_8	p_1	p_4
p_7	p_6	p_5

步骤：（1）首先标记满足下列条件的像素点（$p_1=1$）

$$(1.1)2 \le N(p_1) \le 6; (1.2)S(p_1)=1$$

$$(1.3)p_2 \cdot p_4 \cdot p_6 = 0; (1.4)p_4 \cdot p_6 \cdot p_8 = 0$$

其中$N(p_1)$是p_1的非零邻点的个数，$S(p_1)$是以p_2，p_3，…p_9为序时这些点的值从0到1变化的次数。扫描所有点后，将所有标记点删除掉；

（2）同第1步，修改(1.3)为 $(2.3)p_2 \cdot p_4 \cdot p_8 = 0;$ (1.4)为

$(2.4)p_2 \cdot p_6 \cdot p_8 = 0;$ 扫描所有点后，删除所有标记点；

（3）重复(1)和(2)，直到没有删除像素点为止。

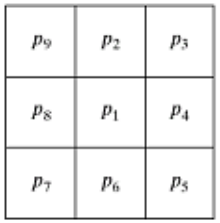
4.2.4 细化

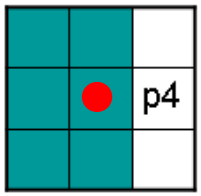
说明:

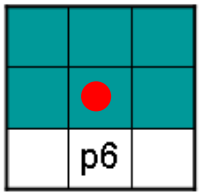
(1)条件(1.1)排除端点及内部点

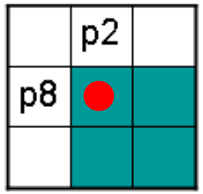
(2)条件(1.2)除去删除单像素宽的线段操作以避免割断骨架

(3)条件(1.3)和条件(1.4)删除了边界的右或下端点 (p_4=0或 p_6=0)或左上角点(p_2=0和p_8=0)

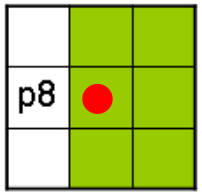


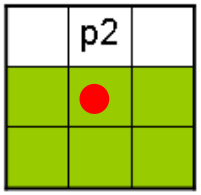


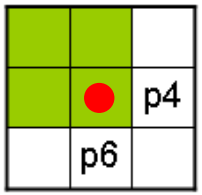




$$(1.1)2 \le N(p_1) \le 6; (1.2)S(p_1) = 1$$
$$(1.3)p_2 \cdot p_4 \cdot p_6 = 0; (1.4)p_4 \cdot p_6 \cdot p_8 = 0$$







$$(2.1)2 \le N(p_1) \le 6; (2.2)S(p_1) = 1$$
$$(2.3)p_2 \cdot p_4 \cdot p_8 = 0; (2.4)p_2 \cdot p_6 \cdot p_8 = 0$$

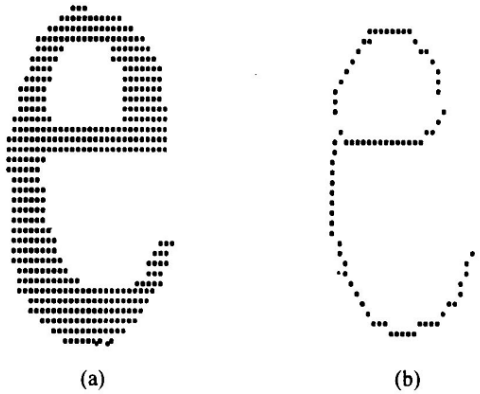
4.2.4 细化

五、细化实例

1. 指纹图像细化

(a) 指纹原图　(b) 二值化的结果　(c) 细化图

2. 字符细化



(a)　(b)

(a) 原始二值图　(b) 利用本节讨论的方法求得的细化结果

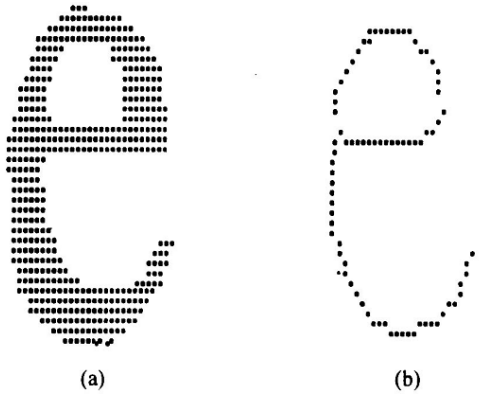
4.2.4 细化

五、细化实例

1. 指纹图像细化

(a) 指纹原图　(b) 二值化的结果　(c) 细化图

2. 字符细化



(a)　(b)

(a) 原始二值图　(b) 利用本节讨论的方法求得的细化结果

小 结

图像二值化、二值图像基本概念

全局阈值、局部阈值、基于边界方法

贴标签

边界跟踪

细化

Hough变换

作 业

4.2.1、在网上检索论文，请给出近3年中，至少3种以上Hough变换方法或应用的简述。

4.2.2、设图像 $f=\begin{bmatrix}0&0&0&1&0&0&0&0\\0&1&1&1&0&0&0&0\\0&0&1&1&1&1&0&0\\0&1&0&1&1&1&1&1\\0&1&0&1&1&1&0&0\\0&1&1&0&0&1&0&0\end{bmatrix}$ **，值为1是目标像素。**

（1）对该图像分别采用4连接和8连接进行贴标签处理。（需要有主要步骤）

（2）对各连通区域进行8邻域边界跟踪。

作 业

4.2.3、对下列图像进行细化，要求写出每步结果。

$$f=\begin{bmatrix}0&0&0&0&0&0&0&0&0&0\\0&1&1&1&0&0&1&1&1&0\\0&1&1&1&0&0&1&1&1&0\\0&1&1&1&1&1&1&1&1&0\\0&1&1&1&1&1&1&1&1&0\\0&1&1&1&1&1&1&1&1&0\\0&1&1&1&0&0&1&1&1&0\\0&1&1&1&0&0&1&1&1&0\\0&0&0&0&0&0&0&0&0&0\end{bmatrix}$$