第10章 图像分割

主讲教师　张涛

电子信息与电气工程

第10章 图像分割

问题的提出：

- 在对图像的研究和应用中，人们往往仅对图像中的某些目标感兴趣，这些目标通常对应图像中具有特定性质的区域。
- 图像分割（**Image Segmentation**）是指把一幅图像分成不同的具有特定性质区域的图像处理技术，将这些区域分离提取出来，以便进一步提取特征和理解
- 如何实现图像分割？

主要内容

10.1 阈值分割

10.2 边界分割

10.3 区域分割

10.4 基于聚类的图像分割

10.1 阈值分割

- 阈值分割原理

$$g(x,y)=\begin{cases}1 & f(x,y)\geq T\\ 0 & f(x,y)<T\end{cases}$$

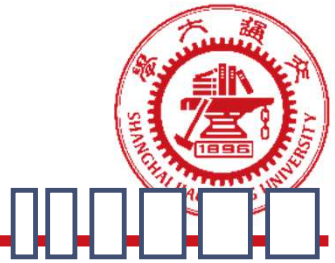
其中，$f(x,y)$为原始图像，$g(x,y)$为结果图像（二值），T为阈值。

显然，阈值的选取决定了二值化效果的好坏。

10.1 阈值分割

- 阈值化概念
  - 上阈值化：灰度值大于等于阈值的所有像素作为前景像素，其余像素作为背景像素
  - 下阈值化：灰度值小于等于阈值的所有像素作为前景像素
  - 内阈值化：确定一个较小的阈值和一个较大的阈值，灰度值介于二者之间的像素作为前景像素
  - 外阈值化：灰度值介于小阈值和大阈值之外的像素作为前景像素

10.1 阈值分割



10.1.1 基于灰度直方图的阈值选择

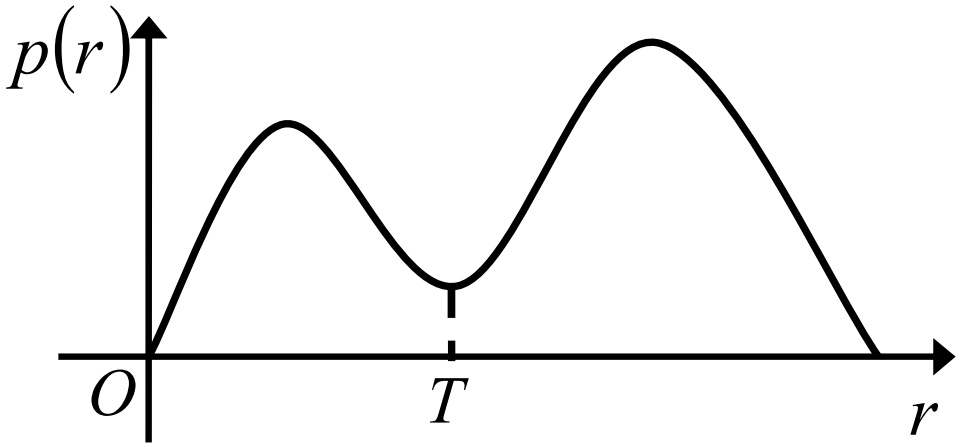
10.1.2 基于模式分类思路的阈值选择

10.1.3 其他阈值分割方法

10.1.1 基于灰度直方图的阈值选择

（1）原理

- 若图像的灰度直方图为双峰分布，表明图像的内容大致为两部分，分别为灰度分布的两个山峰的附近。



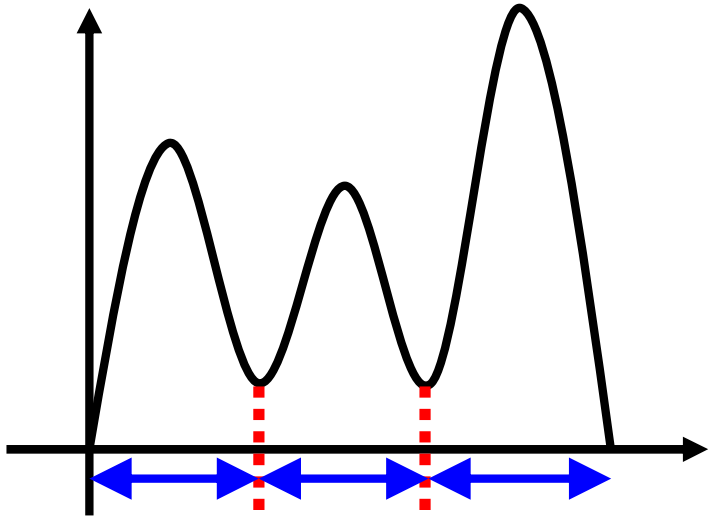
- 选择阈值为两峰间的谷底点对应灰度值

适用于图像中前景与背景灰度差别明显，且各占一定比例的情形，是一种特殊的方法。若图像整体直方图不具有双峰或多峰特性，可以考虑局部范围内应用。

10.1.1 基于灰度直方图的阈值选择

（2）实例

- 一幅图像的直方图如图所示，根据直方图对其进行分割，分成几个区域？阈值点是什么？
  - 分成3个区域，阈值点为3峰之间的谷点对应灰度级
  - 或内阈值化，灰度值介于二者之间的像素作为前景像素



  - 或外阈值化：灰度值介于小阈值和大阈值之外的像素作为前景像素

10.1.1 基于灰度直方图的阈值选择

（3）例程

实现基于双峰分布的直方图选择阈值，分割图像

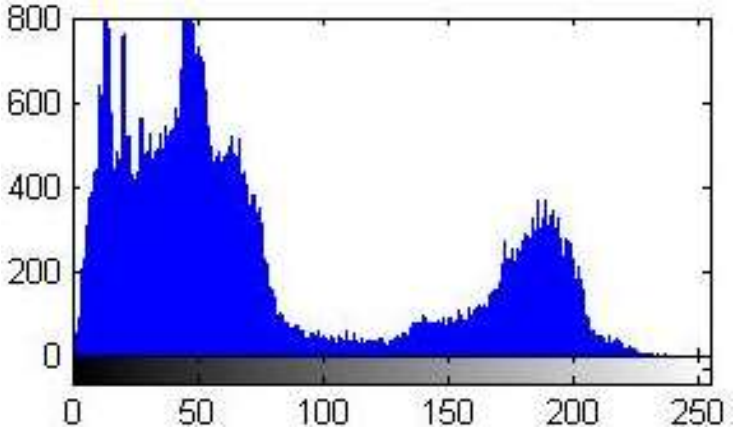
- 分析
  - 重点在于找到直方图的峰和波谷，但直方图通常是不平滑的，因此，首先要平滑直方图，再去搜索峰和谷。
  - 程序设计中，将直方图中相邻三个灰度的频数相加求平均作为中间灰度对应的频数，不断平滑直方图，直至成为双峰分布。
  - 也可以采用其他方法确定峰谷。

10.1.1 基于灰度直方图的阈值选择

（3）例程

- 效果

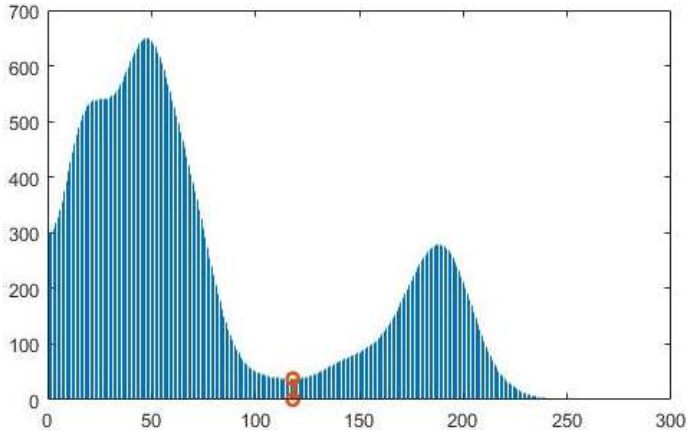
原图



灰度直方图

分割图

T=118



平滑后的直方图和波谷

10.1.2基于模式分类思路的阈值选择

（1）原理

- 认为像素值（或像素特征值）为待分类的数据，寻找合适的阈值，把数据分为不同类别，从而实现图像分割。
- 把所有的像素分为两组（类），属于“同一类别”的对象具有较大的一致性，“不同类别”的对象具有较大的差异性。
- 如何衡量同类的一致性和类间的差异性，采用不同的衡量方法对应不同的算法

10.1.2基于模式分类思路的阈值选择

（2）最大类间方差法（OTSU）

- 准则

各级灰度出现的概率为：$p_i = \frac{n_i}{M \times N}, i = 0,1,2,\mathrm{L}\ ,L-1$

两类像素在图像中的分布概率：$p_O = \sum_{i=0}^{T} p_i \quad p_B = \sum_{i=T+1}^{L-1} p_i$

两类像素值均值和总体灰度均值：

$$\mu_O = \frac{1}{p_O}\sum_{i=0}^{T} i \times p_i \quad \mu_B = \frac{1}{p_B}\sum_{i=T+1}^{L-1} i \times p_i \quad \mu = p_O \times \mu_O + p_B \times \mu_B$$

（2）最大类间方差法（OTSU）

- 准则

两类方差：$\sigma_O^2 = \frac{1}{p_O}\sum_{i=0}^{T} p_i\left(i-\mu_O\right)^2$　$\sigma_B^2 = \frac{1}{p_B}\sum_{i=T+1}^{L-1} p_i\left(i-\mu_B\right)^2$

类内方差和类间方差：

$$\sigma_{in}^2 = p_O\cdot\sigma_O^2 + p_B\cdot\sigma_B^2$$

$$\sigma_1^2 = p_O\times\left(\mu_O-\mu\right)^2 + p_B\times\left(\mu_B-\mu\right)^2$$

使得类内方差最小或类间方差最大、或者类内和类间方差比值最小的阈值T为最佳阈值。

10.1.2基于模式分类思路的阈值选择

（2）最大类间方差法（OTSU）

- 函数
  - **LEVEL = graythresh(I)**：采用OTSU方法计算图像I的全局最佳阈值LEVEL。
  - **BW = im2bw(I,LEVEL)**：采用阈值LEVEL实现灰度图像I的二值化。
  - **BW = imbinarize(I)**：采用基于OTSU方法的全局阈值实现灰度图像I的二值化。
  - **BW = imbinarize(I, METHOD)**：采用METHOD指定的方法获取阈值实现灰度图像I的二值化

10.1.2基于模式分类思路的阈值选择

（2）最大类间方差法（OTSU）

- 程序

```
Image=rgb2gray(imread('lotus1.jpg'));
figure,imshow(Image),title('原始图像');
T=graythresh(Image);
result=im2bw(Image,T);
figure,imshow(result),title('OTSU方法二值化图像 ');
```

10.1.2基于模式分类思路的阈值选择

（2）最大类间方差法（OTSU）

- 效果

原图

分割图，阈值T=109

（2）最大类间方差法（OTSU）

- 效果

原图

阈值T=80

10.1.2基于模式分类思路的阈值选择

（3）最大熵法

- 准则

熵作为分类的标准：若两类的平均熵之和为最大时，从图像中获得最大信息量，对应阈值是最佳阈值。

$$H_O(T) = -\sum_{i=0}^{T} \frac{p_i}{p_O} \log \frac{p_i}{p_O} \qquad H_B(T) = -\sum_{i=T+1}^{L-1} \frac{p_i}{p_B} \log \frac{p_i}{p_B}$$

$$J(T) = H_O(T) + H_B(T)$$

（3）最大熵法

- 例程

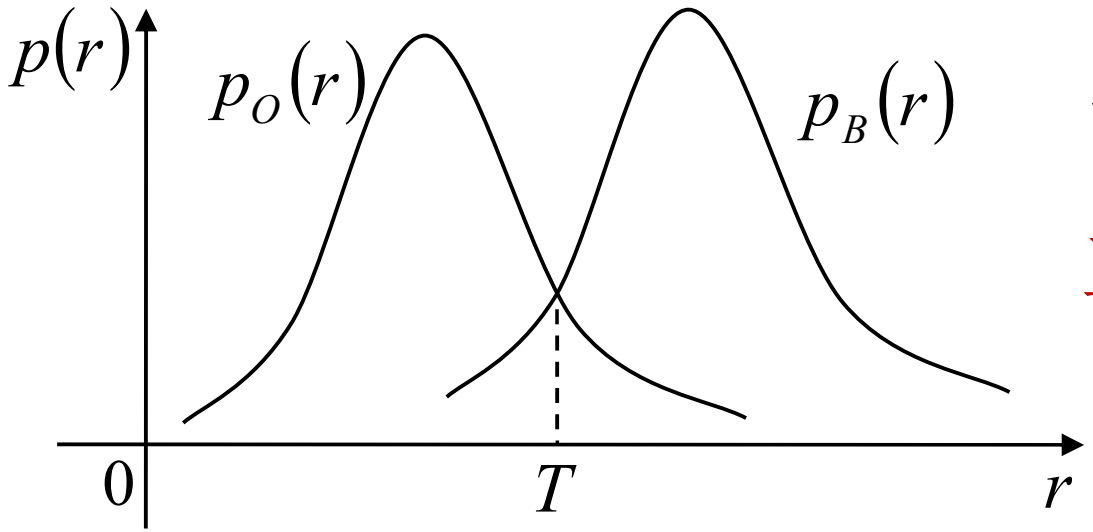
原图

阈值T=120

10.1.2基于模式分类思路的阈值选择

（4）最小误差法

- 准则　分类的错误率最小时对应的阈值为最佳阈值



目标和背景的概率密度分布

目标均值和标准差：μ_O　σ_O

背景均值和标准差：μ_B　σ_B

正态分布概率密度：

$$p_O(r)=\frac{1}{\sqrt{2\pi}\sigma_O}e^{\left[-(r-\mu_O)^2/2\sigma_O^2\right]}$$

$$p_B(r)=\frac{1}{\sqrt{2\pi}\sigma_B}e^{\left[-(r-\mu_B)^2/2\sigma_B^2\right]}$$

10.1.2基于模式分类思路的阈值选择

（4）最小误差法

背景误判为目标的概率：$\varepsilon_B(T)=\int_{-\infty}^{T} p_B(r)dr$

目标误判为背景的概率：$\varepsilon_o(T)=\int_{T}^{+\infty} p_O(r)dr=1-\int_{-\infty}^{T} p_O(r)dr$

误判的概率：$J(T)=\alpha\varepsilon_o(T)+(1-\alpha)\varepsilon_B(T)$

α为目标占图像的比例。$J(T)$最小时对应T为最佳阈值

求解极值点，可得：$(1-\alpha)p_B(r)-\alpha p_O(r)=0$

需要已知目标在图像中所占比例，且目标和背景的灰度概率密度符合正态分布，因此，往往需要用已知的正态分布来拟合直方图的分布，实现较为复杂。

（4）最小误差法

- 例程

原图

阈值T=111

10.1.3其他阈值分割方法

（1）基于迭代运算的阈值选择

- 基本思路：先选择一个阈值作为初始值，然后进行迭代运算，按照某种策略不断改进阈值，直到满足给定的准则为止。
- 关键技术：阈值改进策略的选择，应能使算法快速收敛且每次迭代产生的新阈值优于上一次的阈值。

10.1.3其他阈值分割方法

（1）基于迭代运算的阈值选择

- 一种常用的基于迭代运算的阈值分割算法
  - 求出图像中的最小和最大灰度值r_1和r_2，令阈值初值为：$T^0=\frac{r_1+r_2}{2}$
  - 根据阈值T^k将图像分割成背景和目标两部分，求出两部分的平均灰度值r_B和r_O

$$r_O=\frac{\sum_{f(x,y)<T^k} f(x,y)}{N_o} \qquad r_B=\frac{\sum_{f(x,y)\geq T^k} f(x,y)}{N_B}$$

10.1.3其他阈值分割方法

（1）基于迭代运算的阈值选择

- 求出新的阈值 $T^{k+1}=\frac{r_B+r_O}{2}$
- 如果$T^k=T^{k+1}$，则结束，否则k增加1，转入第二步

10.1.3其他阈值分割方法

（1）基于迭代运算的阈值选择

- 程序

```
Image=im2double(rgb2gray(imread('lotus1.jpg')));
T=(max(Image(:))+min(Image(:)))/2;
equal=false;
while ~equal
   NewT=(mean(Image(Image>=T))
         +mean(Image(Image<T)))/2;
   equal=abs(NewT-T)<1/256;
   T=NewT;
end
result=im2bw(Image,T);
figure,imshow(result),title('迭代方法二值化图像 ');
```

10.1.3其他阈值分割方法

（1）基于迭代运算的阈值选择

- 效果

原图

阈值T=109.7

10.1.3其他阈值分割方法

（2）基于模糊理论的阈值选择

- 基本思路：模糊度表示一个模糊集的模糊程度，模糊熵是一种度量模糊度的数量指标，用模糊熵作为目标函数，求解最佳阈值。
- 图中的每一点对于目标和背景均有一定的隶属程度，隶属度函数为

$$\mu_f\left(f_{xy}\right)=\begin{cases}\dfrac{1}{1+\left|f_{xy}-\mu_O\right|/C} & f_{xy}\le T\\ \dfrac{1}{1+\left|f_{xy}-\mu_B\right|/C} & f_{xy}>T\end{cases}$$

C是一个常数，保证

$$\mu_f\left(f_{xy}\right)\in[0.5,1]$$

10.1.3其他阈值分割方法

（2）基于模糊理论的阈值选择

- 模糊熵

$$H(f)=\frac{1}{MN\ln 2}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}S\left(\mu_f\left(f_{xy}\right)\right)$$

Shannon函数：$$S(k)=\begin{cases}-k\ln k-(1-k)\ln(1-k) & k\in(0,1)\\ 0 & k=0,1\end{cases}$$

模糊熵取最小值时对应的阈值为最佳阈值

10.1.3其他阈值分割方法

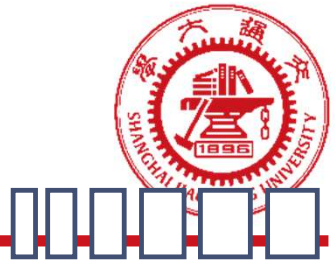
（2）基于模糊理论的阈值选择

- 例程

原图

阈值T=135

10.2 边界分割



- 边界分割：通过检测区域的边界轮廓来实现图像分割的方法
- 分割过程：
  - 边界检测：通过各种边缘检测算子从图像中抽取边缘线段
  - 边界改良：执行各种改良边缘的处理：如边界细化，边缘闭合等
  - 边界跟踪：跟踪边缘形成边界曲线

10.2 边界分割

10.2.1 基于梯度的边界闭合

10.2.2 Hough变换

10.2.3 边界跟踪

10.2.1基于梯度的边界闭合

（1）边界改良

- 目标的部分边界与相邻部分背景相近或相同时，提取出的目标区域边界线会出现断点、不连续或分段连续等情况；有噪声干扰时，也会使轮廓线断开。
- 要提取目标区域时，应使不连续边界闭合
- 方法多种多样：**Hough**变换、基于梯度的边界闭合技术、数学形态学等

（2）基于梯度的边界闭合

两像素(x_1,y_1)、(x_2,y_2)互为邻点，梯度幅度和方向满足

$$|G(x_1,y_1)-G(x_2,y_2)|\le T$$

$$|d(x_1,y_1)-d(x_2,y_2)|\le D$$

将两像素连接起来。对所有边缘像素进行同样的操作，则有希望得到闭合的边界。

10.2.2Hough变换

（1）概述

- 霍夫变换（**Hough transform**）是检测图像中直线和曲线的一种方法。
- 核心思想：
  - 建立一种点—线对偶关系，将图像从图像空间变换到参数空间，确定曲线的参数，进而确定图像中的曲线。
  - 若边界线形状已知，通过检测图像中离散的边界点，确定曲线参数，在图像空间重绘边界曲线，进而改良边界。

（2）检测直线

- 原理

以截距式方程为例：$y = kx + b$

- 一条确定的直线对应一组确定的数据k、b

xy空间一条确定的直线对应参数空间的一个点

- 直线变形为关于k和b的直线：$b = -xk + y$

参数空间的一条直线对应xy空间的一个点

综上所述，xy空间一条直线上的n个点，对应参数kb空间经过一个公共点的n条直线

（2）检测直线

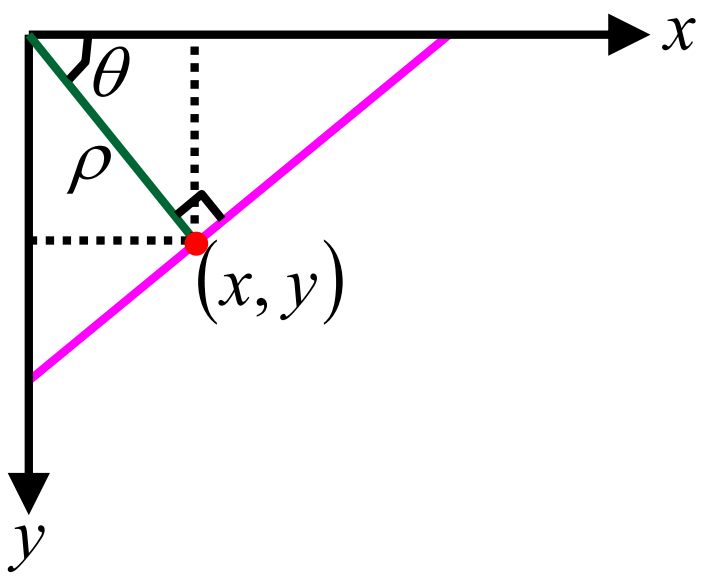
- 算法
  - 假设原图像已经处理为二值边缘图像，扫描图中的每一个像素点：
    - 背景点，不作任何处理
    - 目标点，确定直线：$b=-xk+y$，参数空间上的对应直线上所有的值累加1
  - 循环扫描所有点，重复上述操作
  - 参数空间上累计值最大的点(k^*,b^*)为所求直线参数，按照该参数绘制直线

（2）检测直线

- 极坐标形式
  - 直线方程$y=kx+b$对垂直直线不起作用，采用极坐标形式

$$\rho = x\cos\theta + y\sin\theta$$

$$\begin{cases} x = \rho\cos\theta \\ y = \rho\sin\theta \end{cases}$$



（2）检测直线

- 极坐标形式
  - xy空间一条确定的直线对应$\rho\theta$参数空间的一个点
  - 参数空间的一条正弦曲线对应xy空间的一个点
  - xy空间一条直线上的n个点，对应$\rho\theta$参数空间经过一个公共点的n条正弦曲线
  - 对于原图中每一点，在参数空间确定一条正弦曲线，经过曲线最多的点为原图中直线的参数

（2）检测直线

- 函数
  - [H, THETA, RHO] = hough(BW)
  - [H, THETA, RHO] = hough(BW,PARAM1,VAL1, PARAM2,VAL2)：对输入图像BW进行Hough变换
  - LINES = houghlines(BW, THETA, RHO, PEAKS)
  - LINES = houghlines(...,PARAM1,VAL1, PARAM2,VAL2)

    根据Hough变换的结果提取图像BW中的线段
  - PEAKS = houghpeaks(H,NUMPEAKS)
  - PEAKS = houghpeaks(...,PARAM1,VAL1, PARAM2,VAL2)

    提取Hough变换后参数平面的峰值点

（2）检测直线

- 例程

```
Image=rgb2gray(imread('houghsource.bmp'));
bw=edge(Image,'canny');
figure,imshow(bw);
[h,t,r]=hough(bw,'RhoResolution',0.5,
                        'ThetaResolution',0.5);
figure,imshow(imadjust(mat2gray(h)),'XData',t,
                    'YData',r,'InitialMagnification','fit');
xlabel('\theta'),ylabel('\rho');
axis on,axis normal,hold on;
P=houghpeaks(h,2);    x=t(P(:,2));      y=r(P(:,1));
plot(x,y,'s','color','r');
```

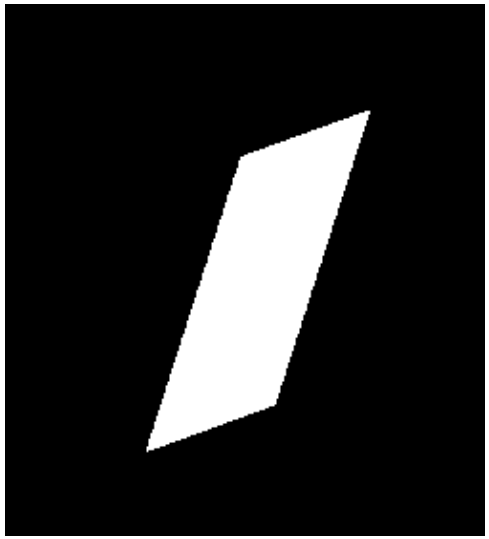
（2）检测直线

- 例程

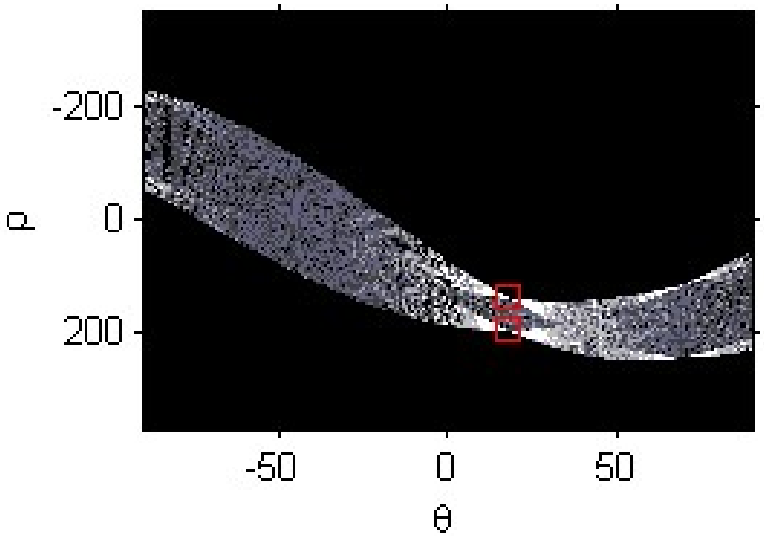
```
lines=houghlines(bw,t,r,P,'FillGap',5,'Minlength',7);
figure,imshow(Image);
hold on;
max_len=0;
for i=1:length(lines)
   xy=[lines(i).point1;lines(i).point2];
   plot(xy(:,1),xy(:,2),'LineWidth',2,'Color','g');
   plot(xy(1,1),xy(1,2),'x','LineWidth',2,'Color','y');
   plot(xy(2,1),xy(2,2),'x','LineWidth',2,'Color','r');
end
```

（2）检测直线

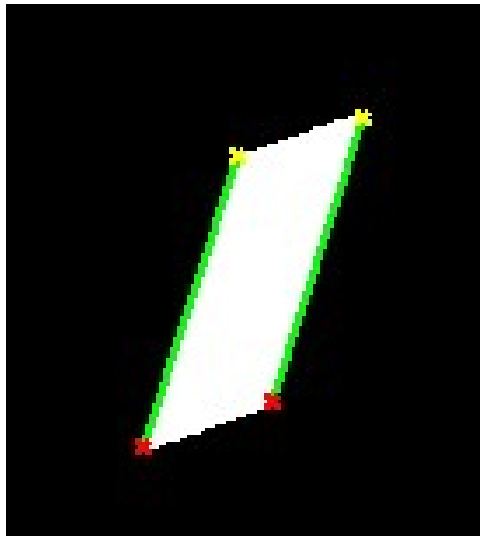
- 例程



原始图像



ρθ参数空间



检测结果

（3）检测圆

- 原理

$$\text{圆方程}\ (x-a)^2+(y-b)^2=r^2$$

  - xy空间一个圆对应三维参数空间一个点（a,b,r)
  - xy空间圆上一个点(x,y)对应参数空间一条曲线
  - xy空间圆上n个点对应参数空间n条相交于一点的曲线

对于原图中每一点，在参数空间确定一条曲线，经过曲线最多的点为原图中圆的参数

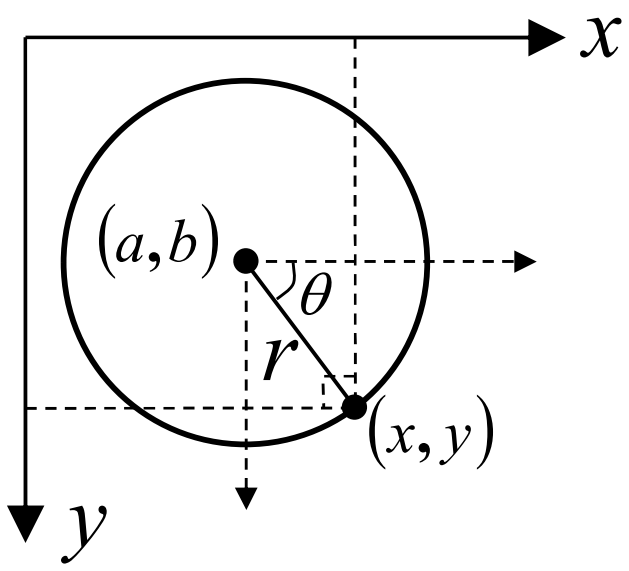
（3）检测圆

- 算法
  - 假设原图像已经处理为二值边缘图像，扫描图中的每一个像素点：
    - 背景点，不作任何处理
    - 目标点，确定曲线：参数空间上的对应曲线上所有的值累加1
  - 循环扫描所有点
  - 参数空间上累计值为最大的点(a^*,b^*,r^*)为所求圆参数
  - 按照该参数与原图像同等大小的空白图像上绘制圆

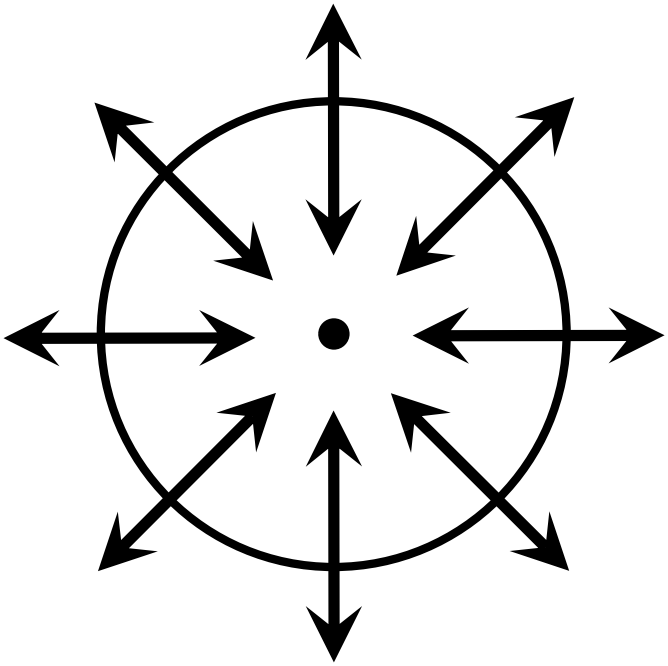
（3）检测圆

- 简化运算

三维参数空间，计算量大，可以采样其他形式，如极坐标式，进行进一步简化。



$$\begin{cases} x = a + r\cos\theta \\ y = b + r\sin\theta \end{cases}$$



（4）检测任意曲线

Hough变换可以推广到具有解析形式$f(x,a)=0$的任意曲线，x表示图像点，a表示参数向量

- 过程
  - 初始化参数空间$A[a]$
  - 对每个边缘像素x确定a，使得$f(x,a)=0$，并令$A[a]+=1$
  - A的局部最大值对应图像中曲线参数

10.2.3边界跟踪

（1）跟踪方法

- 根据某些严格的“探测准则”找出目标物体轮廓上的像素，即确定边界的起始搜索点；
- 再根据一定的“跟踪准则”找出目标物体上的其他像素，直到符合跟踪终止条件。
- 由二维图像变为一维的点序列

（2）函数

- **B = bwboundaries(BW)：搜索二值图像BW的外边界和内边界。**
- **B = bwboundaries(BW,CONN,OPTIONS)**
- **[B,L,N,A] = bwboundaries(...)**
- **B = bwtraceboundary(BW,P,FSTEP)：跟踪二值图像BW中目标轮廓**

（3）例程

读取一幅灰度图像，对其进行阈值分割，并对分割的二值图像进行边界跟踪

```
Image=im2bw(imread('algae.jpg'));
Image=1-Image;
[B,L]=bwboundaries(Image);
figure,imshow(L),title('划分的区域');
hold on;
for i=1:length(B)
    boundary=B{i};
    plot(boundary(:,2),boundary(:,1),'r','LineWidth',2);
end
```

10.2.3边界跟踪

（3）例程

原始图像

边界跟踪

10.3 区域分割

一般认为，同一个区域内的像素点具有某种相似性，如灰度、颜色、纹理等，区域分割即是根据特定区域与其他背景区域特性上的不同来进行图像分割的技术。

10.3.1 区域生长　　10.3.3 区域分裂

10.3.2 区域合并　　10.3.4 区域合并分裂

10.3.1区域生长

（1）原理

- 设计思路

拟把图像划分成一系列区域，确定每个区域区别于其他区域的特征，由此生成相似性判据；从图像某个像素开始，判断其应该属于哪个区域，使区域逐渐变大，直到被比较的像素与区域像素具有显著差异为止。

10.3.1区域生长

（1）原理

- 实现方法

在每个要分割的区域内确定一个种子点，判断种子像素周围邻域是否有与种子像素相似的像素，若有将新的像素包含在区域内，并作为新的种子继续生长，直到没有满足条件的像素点时停止生长。

10.3.1区域生长

（1）原理

- 关键技术
  - 种子点的选取

    选择待提取区域的具有代表性的点，可以是单个像素，也可以是包括若干个像素的子区域，根据具体问题，利用先验知识来选择。
  - 生长准则的确定（相似性准则）

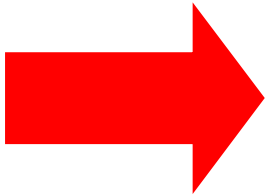
    一般根据图像的特点，采用与种子点的距离度量（彩色、灰度、梯度等量之间的距离）。
  - 区域停止生长的条件

    区域大小、迭代次数或区域饱和等条件。

10.3.1区域生长

（2）示例

1	0	4	6	5	1
1	0	4	6	6	2
0	1	5	5	5	1
0	0	5	6	5	0
0	0	1	6	0	1
1	0	1	2	1	1



1	1	2	2	2	1
1	1	2	2	2	1
1	1	2	2	2	1
1	1	2	2	2	1
1	1	1	2	1	1
1	1	1	1	1	1

种子点为：(2, 2)

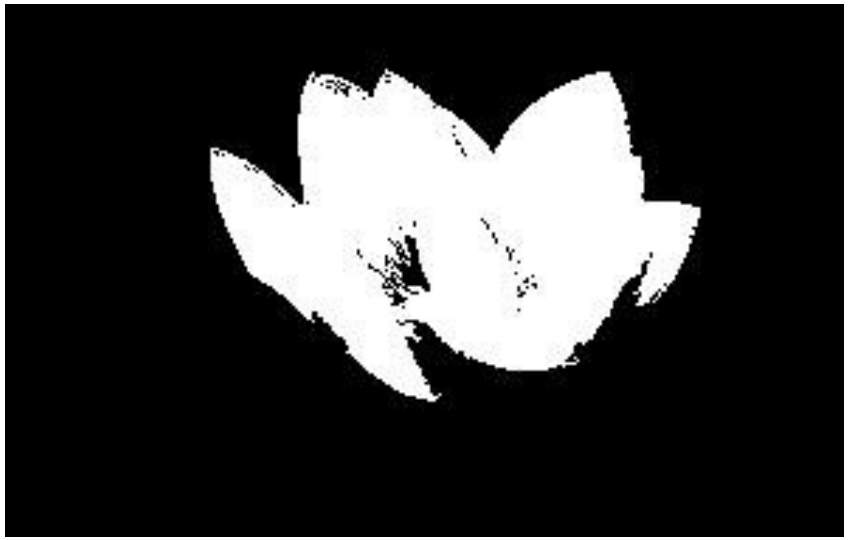
相似性准则：灰度值差小于2

邻域选选择：4邻域

（3）例程

对图像进行区域生长。交互式选取种子，生长准则采用“待测像素点与区域的平均灰度差小于40”，8邻域范围生长，停止生长条件为区域饱和。

原图



区域生长

10.3.2区域合并

（1）原理

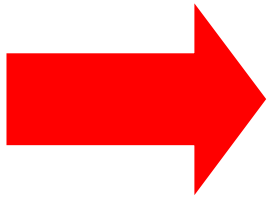
假设图像已经分为若干个小区域，合并具有相似性的相邻区域。

极端的情况，可以认为每个像素为一个小区域，把具有相似性的像素合并到一个区域内。

10.3.2区域合并

（2）示例

1	0	4	6	5	1
1	0	4	6	6	2
0	1	5	5	5	1
0	0	5	6	5	0
0	0	1	6	0	1
1	0	1	2	1	1



1	0	4	6	5	1
1	0	4	6	6	2
0	1	5	5	5	1
0	0	5	6	5	0
0	0	1	6	0	1
1	0	1	2	1	1

（3）例程

通过区域合并将示例中的小图像分割为两个区域

- 设计思路
  - 初始化：每个像素为一个小区域；相似性准则采用：相邻区域灰度均值差≤2；左上角第一个点设为区域1，其余为0，表示未标记。
  - 第一次扫描图像：从左到右，从上到下，判断每一点与其左上、上、左邻点的灰度距离，三个距离中最小的若符合合并规则，将对应邻点的标记赋予当前点；若没有相似的点，则赋予当前点新的标记。

（3）例程

- 设计思路
  - 再次扫描图像：若某一像素点上、左邻点标记不一致，但当前点和其中一个邻点标记一致，则判断两个区域是否是同一个，若是，则将两个区域标记修改为较小的一个，即区域合并。

10.3.3区域分裂

（1）原理

认为整幅图像是一个完整区域，检验整幅区域是否具有一致性，不具有时，分裂为几个小区域；然后再检测小区域的一致性，不具有时进一步分裂；重复这个过程直到每个区域都具有一致性。

每个区域可以具有不同的一致性

通常在分裂区域时，采用一分为4的方法

10.3.3区域分裂

（2）示例

$$\begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 2 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 6 & 7 & 1 & 0 & 0 & 1 \\ 1 & 6 & 7 & 5 & 6 & 7 & 1 & 1 \\ 0 & 7 & 6 & 6 & 6 & 0 & 1 & 1 \\ 0 & 7 & 6 & 5 & 7 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$$

- 初始化及准则、方法的确定
  - 区域内最大灰度值与最小灰度值之差≤ 2
  - 采用一分为四的分裂方法
- 分裂

①				②			
1	1	0	1	1	0	0	1
0	1	2	0	1	1	1	0
0	0	6	7	1	0	0	1
1	6	7	5	6	7	1	1
0	7	6	6	6	0	1	1
0	7	6	5	7	1	0	0
1	1	0	1	1	1	1	0
0	1	1	1	1	1	0	1
③				④			

10.3.3区域分裂

（2）示例

对四个小区域分别计算最大与最小灰度差，与阈值2比较，每个区域均需分裂

$$\left[\begin{array}{cc|cc|cc|cc}
1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 1 & 2 & 0 & 1 & 1 & 1 & 0 \\
\hline
0 & 0 & 6 & 7 & 1 & 0 & 0 & 1 \\
1 & 6 & 7 & 5 & 6 & 7 & 1 & 1 \\
\hline
0 & 7 & 6 & 6 & 6 & 0 & 1 & 1 \\
0 & 7 & 6 & 5 & 7 & 1 & 0 & 0 \\
\hline
1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 & 0 & 1
\end{array}\right]$$

依此类推，直至所有的区域都不能再分裂

$$\left[\begin{array}{cc|cc|cc|cc}
1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 1 & 2 & 0 & 1 & 1 & 1 & 0 \\
\hline
0 & 0 & 6 & 7 & 1 & 0 & 0 & 1 \\
1 & 6 & 7 & 5 & 6 & 7 & 1 & 1 \\
\hline
0 & 7 & 6 & 6 & 6 & 0 & 1 & 1 \\
0 & 7 & 6 & 5 & 7 & 1 & 0 & 0 \\
\hline
1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 & 0 & 1
\end{array}\right]$$

（3）例程

- 函数

```
S = qtdecomp(I)
S = qtdecomp(I,THRESHOLD)
S = qtdecomp(I,THRESHOLD,MINDIM)
S = qtdecomp(I,THRESHOLD,
                          [MINDIM MAXDIM])
S = qtdecomp(I,FUN)
```

（3）例程

- 程序

```
Image=imread('cameraman.jpg');
S=qtdecomp(Image,0.27);
blocks=repmat(uint8(0),size(S));
for dim=[256 128 64 32 16 8 4 2 1]
    numblocks=length(find(S==dim));
    if(numblocks>0)
        values=repmat(uint8(1),[dim dim numblocks]);
        values(2:dim,2:dim,:)=0;
        blocks=qtsetblk(blocks,S,dim,values);
    end
end
```

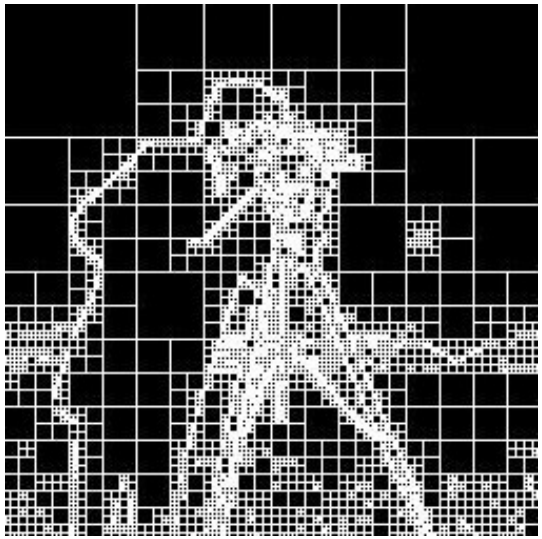
（3）例程

- 程序

```
blocks(end,1:end)=1;
imshow(Image);
```

```
blocks(1:end,end)=1;
figure,imshow(blocks,[]);
```

原始图像



四叉树分解

10.3.4区域分裂合并

（1）原理

结合合并、分裂方法，将原图分成若干个子块，检测子块是否具有一致性，不具有则分裂该子块，若某些子块具有相似性，合并这些子块

10.3.4区域分裂合并

（2）步骤

- 将原图分为四个相等的子块，计算子块区域是否具有一致性（例如灰度均值或方差）
- 如果子块不具有一致性（例如：方差大于设定的阈值）分裂该块
- 对不需要分裂的子块进行比较，具有相似性的子块合并
- 重复上述过程，直到不再需要分裂或合并

10.3.4区域分裂合并

（3）示例

$$f=\begin{pmatrix}1&3&7&8\\1&2&8&9\\2&1&2&7\\3&8&8&9\end{pmatrix}$$

$$f_{11}=\begin{pmatrix}1&3\\1&2\end{pmatrix},\ f_{12}=\begin{pmatrix}7&8\\8&9\end{pmatrix}\quad \mu_{11}=1.75,\sigma_{11}^{\ 2}=0.6875,\ \mu_{12}=8,\sigma_{12}^{\ 2}=0.5$$

$$f_{13}=\begin{pmatrix}2&1\\3&8\end{pmatrix},\ f_{14}=\begin{pmatrix}2&7\\8&9\end{pmatrix}\quad \mu_{13}=3.5,\sigma_{13}^{\ 2}=7.25,\ \mu_{14}=6.5,\sigma_{14}^{\ 2}=7.25$$

$\sigma_{Th}=1$

1	3	7	8
1	2	8	9
2	1	2	7
3	8	8	9

$\mu_{Th}=2$

1	3	7	8
1	2	8	9
2	1	2	7
3	8	8	9

10.4 基于聚类的图像分割

（1）原理

把图像分割看做对像素进行分类的问题，把像素表示成特征空间的点，采用聚类算法把这些点划分为不同类别，对应原图则是实现对像素的分组，分组后利用“连通成分标记”找到连通区域。

（2）关键技术

- 如何把像素表示成特征空间中的点

用向量来代表像素或像素周围邻域，向量的元素为与像素相关的特征，根据图像的具体情况，判断待分割区域的共性来设计。

- 聚类方法

（3）K均值聚类

- 原理

首先确定K个初始聚类中心，然后根据各类样本到聚类中心的距离平方和最小的准则，不断调整聚类中心，直到聚类合理。

（3）K均值聚类

- 步骤
  - 任选K个初始聚类中心
  - 逐一将样本按最小距离原则分配给K个聚类中心

若$\|x-\mu_j(m)\|<\|x-\mu_i(m)\|, i=1,2,\mathrm{L}\ ,K, i\neq j$则$x\in\omega_j(m)$

$\omega_j(m)$为第m次迭代时，聚类中心为$\mu_j(m)$的聚类域

  - 计算新的聚类中心

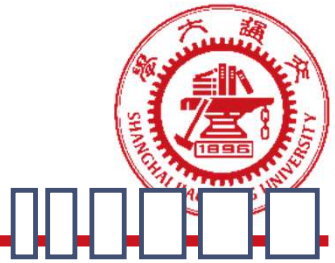
$$\mu_i(m+1)=\frac{1}{N_i}\sum_{x\in\omega_i(m)}x \qquad i=1,2,\mathrm{L}\ ,K$$

（3）K均值聚类

- 步骤
  - 判断算法是否收敛

    若 $\mu_i(m+1)=\mu_i(m)$　　$i=1,2,\mathrm{L}\ ,K$ 算法收敛

    否则，重新分配，进行下一次迭代。



（3）K均值聚类

- 函数

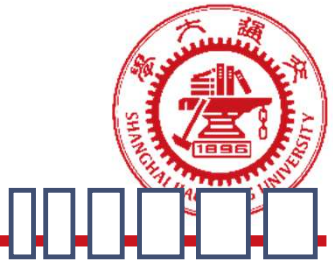
```
[IDX,C] = kmeans(X,K);
[IDX,C,SUMD,D] =kmeans(..., 'PARAM1',val1,
                            'PARAM2',val2, ...)
```

（3）K均值聚类

- 例程

  对一幅苹果图像，利用色彩信息，实现聚类分割

```
Image=imread('fruit.jpg');
figure,imshow(Image);
hsv=rgb2hsv(Image);
h=hsv(:,:,1);
h(h>330/360)=0;
training=h(:);
```



（3）K均值聚类

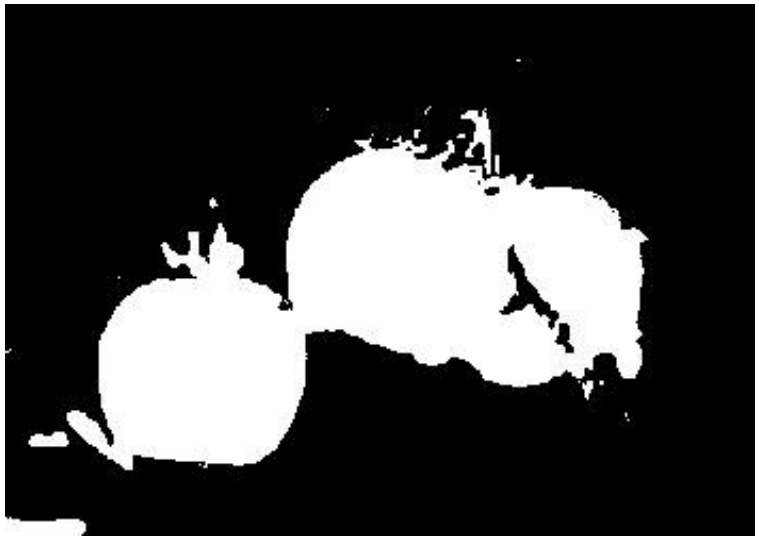
- 例程

```
startdata = [0;60/360;120/360;180/360;
                                240/360;300/360];
[IDX,C]= kmeans(training,6,'Start',startdata);
idbw = (IDX == 1);
template = reshape(idbw, size(h));
figure,imshow(template),title('K均值聚类分割');
```

（3）K均值聚类

- 例程

原始图像



K均值聚类分割

编程实践

- 10.1编写程序实现基于边界的图像分割，可以选择不同的边界改良算法。
- 10.2编写程序实现基于Hough变换检测图中硬币的个数。