



第6章 图像的分割

6.1 图像分割的定义和分类

6.2 基于边缘的图像分割方法

6.3 基于区域的分割

6.4 运动图像目标分割



第6章 图像的分割

学习目标：

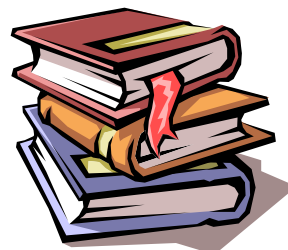
- 1.能够利用**Roberts**算子、**Prewitt**算子、**Sobel**算子、**LoG**算子及**Canny**算子等进行边缘检测；能够分析各类算法应用的优缺点；
- 2.能够描述霍夫曼变换直线检测的原理；
- 3.能够利用**MALAB**函数进行边缘检测以及边缘跟踪；
- 4.理解自适应阈值、最佳全局阈值的分割方法；
- 5.理解区域生长，分割和合并的分割方法。

重点：

边缘检测算子；边缘跟踪算法思想；阈值分割法；区域生长法

难点

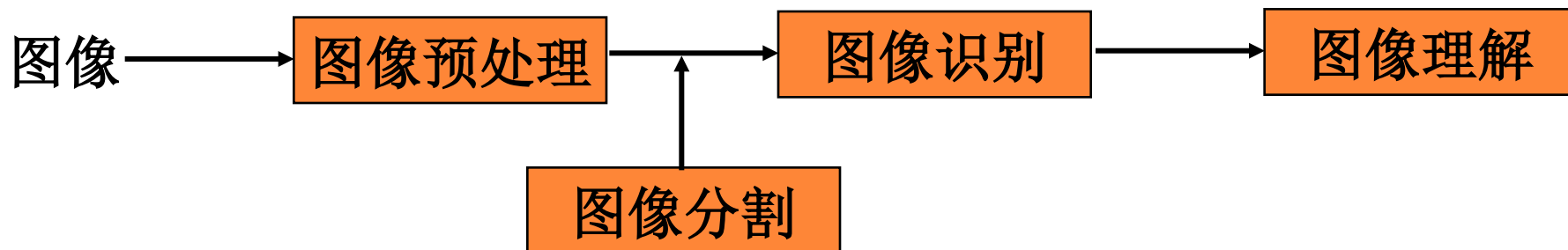
边缘跟踪算法思想；区域生长法





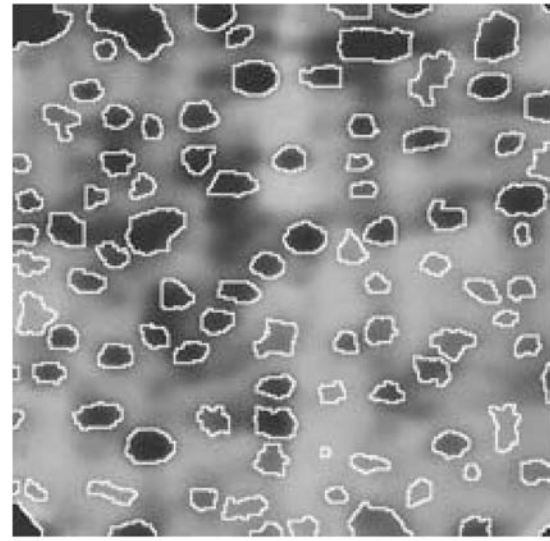
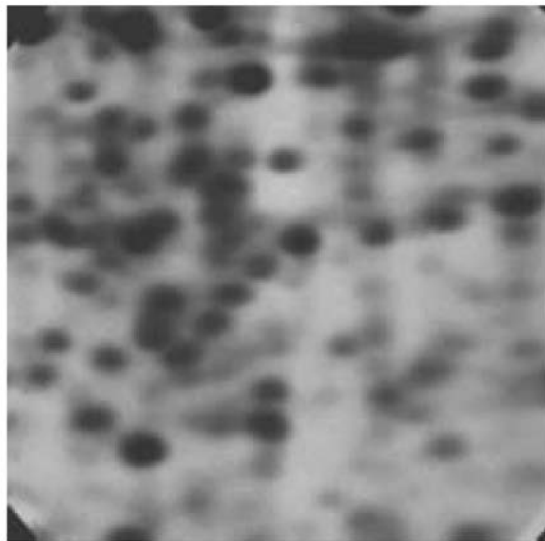
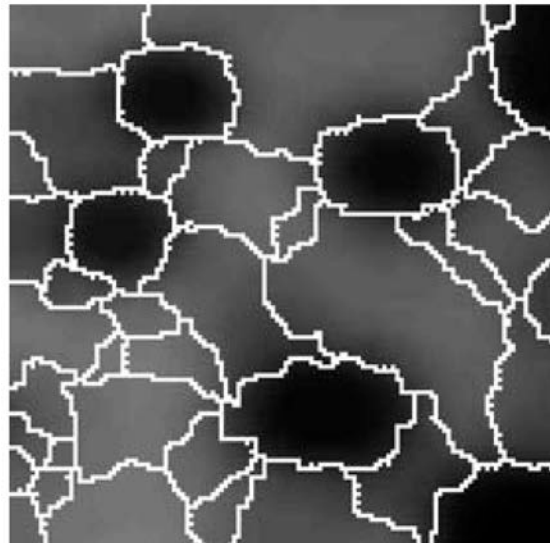
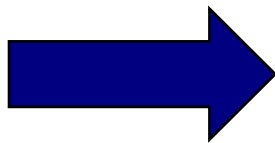
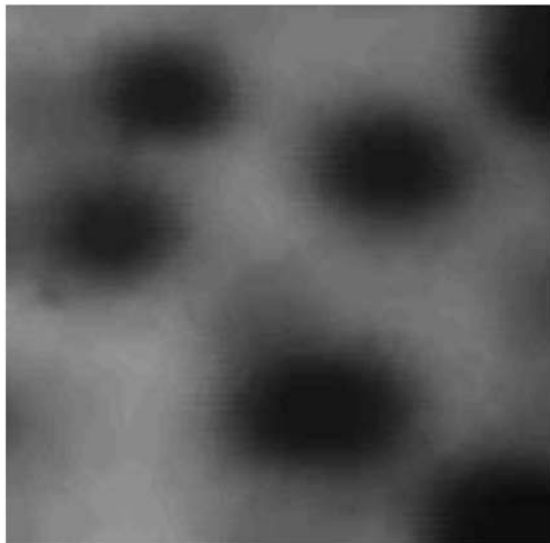
6.1 图像分割的定义和分类

- 图像分割的目标是重点根据图像中的物体将图像的像素分类，并提取感兴趣目标。
- 图像分割是图像识别和图像理解的基本前提步骤





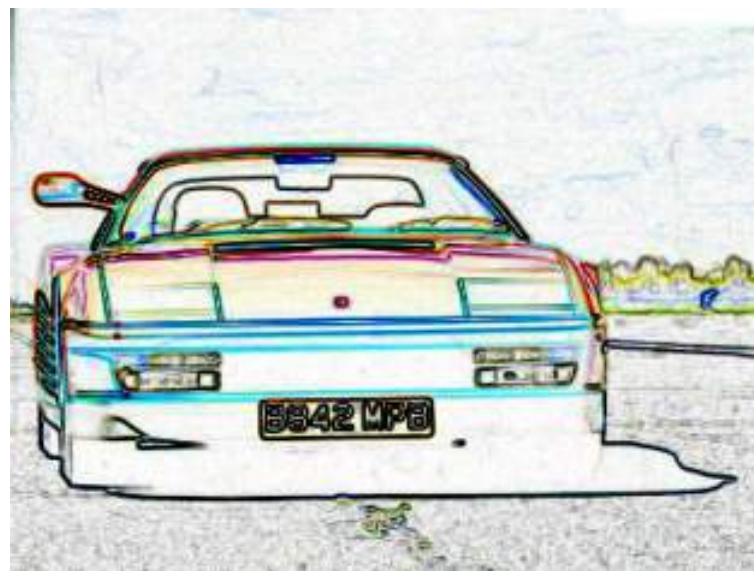
图像分割举例





图像分割举例

- 图像分割是把图像分解成构成的部件和对象的过程
- 把焦点放在增强感兴趣对象
 - 汽车牌照
- 排除不相干图像成分：
 - 非矩形区域





6.1 图像分割的定义和分类

○ 分类—分割依据

- 相似性分割：将相似灰度级的像素聚集在一起。形成图像中的不同区域。这种基于相似性原理的方法也称为**基于区域相关的分割技术**
- 非连续性分割：首先检测局部不连续性，然后将它们连接起来形成边界，这些边界把图像分以不同的区域。这种基于不连续性原理检出物体边缘的方法**称为基于边缘的分割技术**
- 两种方法是互补的。有时将它们地结合起来，以求得到更好的分割效果。



6.1 图像分割的定义和分类

■ 分类—连续性与处理策略

■ 连续性:

- 不连续性: 边界
- 相似性: 区域

■ 处理策略: 早期处理结果是否影响后面的处理

- 并行: 不
- 串行: 结果被其后的处理利用

■ 四种方法

- 并行边界; 串行边界; 并行区域; 串行区域



6.1 图像分割的定义和分类

○ 问题

- 不同种类的图像、不同的应用要求所要求提取的区域是不相同的。分割方法也不同，目前没有普遍适用的最优方法。
- 人的视觉系统对图像分割是相当有效的，但十分复杂，且分割方法原理和模型都未搞清楚。这是一个很值得研究的问题。

○ 研究层次

- 图像分割算法
- 图像分割算法的评价和比较
- 对分割算法的评价方法和评价准则的系统研究



6.2 基于边缘的图像分割方法

➤ 边缘的定义：

图像中像素灰度有阶跃变化或屋顶变化的那些像素的集合。

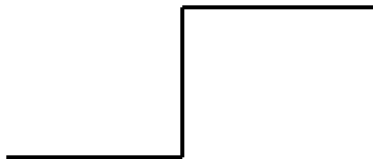
➤ 边缘的分类

- 阶跃状
- 屋顶状

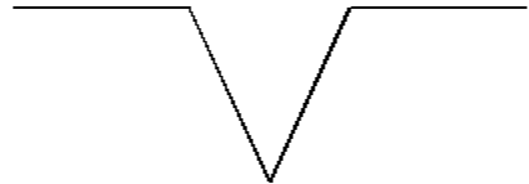
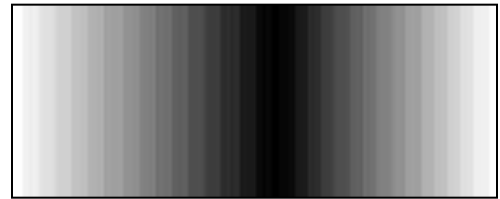
图像：



剖面：



阶跃状



屋顶状



6.2 基于边缘的图像分割方法

各种边缘其一阶、二阶导数特点

图像:



剖面:



一阶
导数:



二阶
导数:



(a)

(b)

(c)

(d)



6.2 基于边缘的图像分割方法

说明：对阶跃边缘，其一阶导数在图像由暗变明的位置处有**1个向上的阶跃**，而其它位置都为**0**，这表明可用一阶导数的幅度值来检测边缘的存在，幅度峰值一般对应边缘位置。

其二阶导数在一阶导数的阶跃上升区有**1个向上的脉冲**，而在一阶导数的阶跃下降区有**1个向下的脉冲**，在这两个脉冲之间有**1个过0点**，它的位置正对应原图像中边缘的位置，所以可用二阶导数的**过0点检测边缘**位置，而用二阶导数在过0点附近的符号确定边缘像素在图像边缘的暗区或明区。

对(c)而言，脉冲状的剖面边缘与(a)的一阶导数形状相同，所以(c)的一阶导数形状与(a)的二阶导数形状相同，而它的**2个二阶导数过0点**正好分别对应脉冲的上升沿和下降沿，通过**检测脉冲剖面的2个二阶导数过0点**就可确定脉冲的范围。

对(d)而言，屋顶状边缘的剖面可看作是将脉冲边缘底部展开得到，所以它的一阶导数是将(c)脉冲剖面的一阶导数的上升沿和下降沿展开得到的，而它的二阶导数是将脉冲剖面二阶导数的上升沿和下降沿拉开得到的，通过检测屋顶状边缘**剖面的一阶导数过0点**，可以确定屋顶位置。



6.2 基于边缘的图像分割方法

线检测

- 通过比较典型模板的计算值，确定一个点是否在某个方向的线上
- 你也可以设计其它模板：
 - 模板系数之和为0
 - 感兴趣的方向系数值较大

-1	-1	-1
2	2	2
-1	-1	-1

水平模板

-1	-1	2
-1	2	-1
2	-1	-1

45度模板

-1	2	-1
-1	2	-1
-1	2	-1

垂直模板

2	-1	-1
-1	2	-1
-1	-1	2

135度模板



线检测

- 用4种模板分别计算
 - $R_{\text{水平}} = -6 + 30 = 24$
 - $R_{45^\circ} = -14 + 14 = 0$
 - $R_{\text{垂直}} = -14 + 14 = 0$
 - $R_{135^\circ} = -14 + 14 = 0$
- 从这些值中寻找绝对值最大值，确定当前点更加接近于该模板所对应的直线

[illegible]



6.2 基于边缘的图像分割方法

1、边缘检测

梯度算子是一阶导数算子

梯度算子

$$\nabla f(x, y) = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

👉 幅值

$$mag(f) = (G_x^2 + G_y^2)^{1/2}$$

👉 方向角

$$\alpha(x, y) = \arctan\left(\frac{G_y}{G_x}\right)$$



6.2 基于边缘的图像分割方法

1、边缘检测

-1	1
----	---

-1
1

近似计算

梯度算子

👉 $M_1 = |G_x| + |G_y|$

👉 $M_2 = G_x^2 + G_y^2$

👉 $M_\infty = \text{Max}(G_x, G_y)$

💣 数字图像处理中用差分代替微分



6.2 基于边缘的图像分割方法

1、边缘检测

Roberts算子

Z_1	Z_2	Z_3
Z_4	Z_5	Z_6
Z_7	Z_8	Z_9

-1	0
0	1

0	-1
1	0

$$G_x = Z_9 - Z_5$$

$$G_y = Z_8 - Z_6$$



6.2 基于边缘的图像分割方法

1、边缘检测

Prewitt
算子

Z_1	Z_2	Z_3
Z_4	Z_5	Z_6
Z_7	Z_8	Z_9

-1	-1	-1
0	0	0
1	1	1

-1	0	1
-1	0	1
-1	0	1

$$G_x = (Z_7 + Z_8 + Z_9) - (Z_1 + Z_2 + Z_3)$$

$$G_y = (Z_3 + Z_6 + Z_9) - (Z_1 + Z_4 + Z_7)$$

★ 能检测边缘点，而且能抑制噪声。



6.2 基于边缘的图像分割方法

1、边缘检测

Sobel
算子

Z_1	Z_2	Z_3
Z_4	Z_5	Z_6
Z_7	Z_8	Z_9

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

$$G_x = (Z_7 + 2Z_8 + Z_9) - (Z_1 + 2Z_2 + Z_3)$$
$$G_y = (Z_3 + 2Z_6 + Z_9) - (Z_1 + 2Z_4 + Z_7)$$

★ 能检测边缘点，进一步抑制噪声，但检测的边缘较宽。



6.2 基于边缘的图像分割方法

1、边缘检测

[g,t]=edge(image,method,threshold,directio)

image---输入图像

Method—采用的方法类型

threshold—阈值，若给定阈值，则**t=threshold**；否则，由**edge**函数自动计算并把其值返回给**t**

direction—寻找边缘的方向，其值可为

horizontal、vertical、both

g—返回的二值图像

edge
函数



6.2 基于边缘的图像分割方法

1、边缘检测



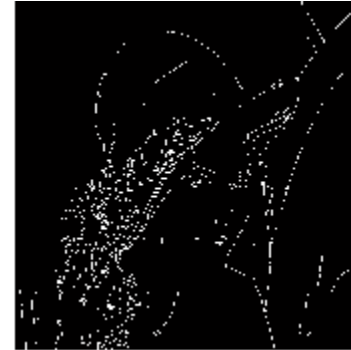
原图



Roberts算子



Sobel算子



Prewitt算子

梯度算子



6.2 基于边缘的图像分割方法

拉普拉斯算子

二阶导数算子

微分

$$\nabla^2 f(x, y) = \frac{\partial^2 f(x, y)}{\partial x^2} + \frac{\partial^2 f(x, y)}{\partial y^2}$$

差分

$$\nabla^2 f(x, y) = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$



6.2 基于边缘的图像分割方法

1、边缘检测

拉普拉斯算子

0	1	0
1	-4	1
0	1	0

1	1	1
1	-8	1
1	1	1

图9.4 两种常用的拉普拉斯算子模板

二阶导数算子对噪声敏感，需要先去噪。



6.2 基于边缘的图像分割方法

1、边缘检测

拉普拉斯算子



LOG算子分割图像结果



6.2 基于边缘的图像分割方法

将Gaussian滤波器和Laplacian边缘检测结合在一起，形成了LoG (Laplacian of Gaussian)算法。即先用高斯函数对图像进行平滑，然后再用拉普拉斯算子进行运算，形成Laplacian-Gauss算法，墨西哥草帽函数形式。

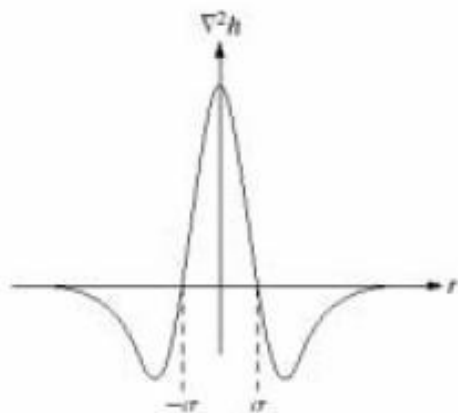
LOG算子

$$\begin{aligned} LOG(x, y) &= \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \right) \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(x^2 + y^2)}{2\sigma^2} \right) \\ &= \frac{-1}{2\pi\sigma^4} \left(2 - \frac{x^2 + y^2}{\sigma^2} \right) \exp\left(-\frac{x^2 + y^2}{2\sigma^2} \right) \end{aligned}$$



LoG 函数的三维曲线、横截面

三维曲线



横截面

0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

模板

近似的 5×5 模板：一个正的中心项，周围是一个相邻的负值区域，并被一个零值的外部区域包围。系数的总和为零



6.2 基于边缘的图像分割方法

Canny算子

👉 好的检测结果：对边缘的错误检测率要尽可能低，在检测出图像真实的边缘的同时要避免检测出现虚假的边缘。

👉 好的边缘定位精度：标记出的边缘位置要和图像上真正边缘的位置尽量接近。

👉 对同一边缘要有低的响应次数：有的算子会对一个边缘回产生多个响应。也就是说图像上本来只有一个边缘点的，可是检测出来就会出现多个边缘点。

👉 克服噪声的影响



6.2 基于边缘的图像分割方法

Canny算子

算法步骤

- 👉 用高斯滤波器平滑图像
- 👉 计算滤波后图像梯度的幅值和方向
- 👉 对梯度幅值应用非极大值抑制，其过程为找出图像梯度中的局部极大值点，把其它非局部极大值点置零以得到细化的边缘
- 👉 用双阈值算法检测和连接边缘，使用两个阈值 T_1 和 T_2 ($T_1 > T_2$)， T_1 用来找到每条线段， T_2 用来在这些线段的两个方向上延伸寻找边缘的断裂处，并连接这些边缘。



6.2 基于边缘的图像分割方法

Canny算子

实例



Canny算子分割图像结果



练习题

简述梯度法和Laplacian算子检测边缘的异同点。



练习题

答：梯度算子和Laplacian检测边缘对应的模板分别为

-1
1

-1	1
----	---

	1	
1	-4	1
	1	

梯度算子是利用阶跃边缘灰度变化的一阶导数特性，认为极大值点对应于边缘点；而Laplacian算子检测边缘是利用阶跃边缘灰度变化的二阶导数特性，认为边缘点是零交叉点。两者都能用于检测边缘，且都对噪声敏感。



1、边缘检测

算子比较

👉 **Roberts算子**：Roberts算子利用局部差分算子寻找边缘，边缘定位精度较高，但容易丢失一部分边缘，同时由于图像没经过平滑处理，因此不具备抑制噪声能力。该算子对具有陡峭边缘且含噪声少的图像效果较好。

👉 **Sobel算子和Prewitt算子**：都是对图像先做加权平滑处理，然后再做微分运算，所不同的是平滑部分的权值有些差异，因此对噪声具有一定的抑制能力，但不能完全排除检测结果中出现的虚假边缘。虽然这两个算子边缘定位效果不错，但检测出的边缘容易出现多像素宽度。



1、边缘检测


算子比较

👉 **Laplacian算子**：是不依赖于边缘方向的二阶微分算子，对图像中的阶跃型边缘点定位准确，该算子对**噪声非常敏感**，它使**噪声成分得到加强**，这两个特性使得该算子容易丢失一部分边缘的方向信息，造成一些不连续的检测边缘，同时抗噪声能力比较差。



1、边缘检测

算子比较

 **LOG算子**：该算子首先用高斯函数对图像作平滑滤波处理，然后才使用Laplacian算子检测边缘，因此克服了Laplacian算子抗噪声能力比较差的缺点，但是在抑制噪声的同时也可能将原有的比较尖锐的边缘也平滑掉了，造成这些**尖锐边缘无法检测**到。应用LOG算子，高斯函数中**方差参数的选择很关键**，对图像边缘检测效果有很大的影响。高斯滤波器为低通滤波器，越大，通频带越窄，对较高频率的噪声的抑制作用越大，避免了虚假边缘的检出，同时信号的边缘也被平滑了，造成某些边缘点的丢失。反之，越小，通频带越宽，可以检测到的图像更高频率的细节，但对噪声的抑制能力相对下降，容易出现虚假边缘。因此，应用LOG算子，为取得更佳的效果，对于不同图像应选择不同参数。



1、边缘检测

算子比较

👉 **Canny算子**：Canny算子虽然是基于最优化思想推导出的边缘检测算子，实际效果并不一定最优，原因在于理论和实际有许多不一致的地方。该算子同样采用高斯函数对图像作平滑处理，因此具有较强的抑制噪声能力，同样该算子也会将一些高频边缘平滑掉，造成边缘丢失。Canny算子其后所采用双阈值算法检测和连接边缘，采用的多尺度检测和方向性搜索较LOG算子要好。



2、边缘跟踪

为什么需要边缘跟踪

- ✓ 由于噪声、照明等产生边缘间断，使得一组像素难以完整形成边缘
- ✓ 因此，在边缘检测算法后，使用连接过程将间断的边缘像素组合成完整边缘



2、边缘跟踪

局部处理

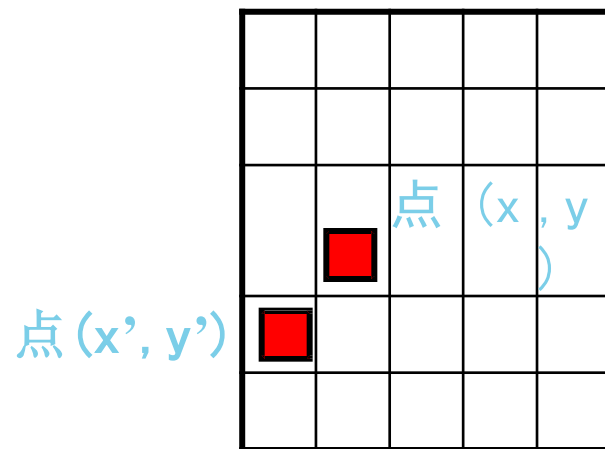
- ✓ 分析图像中每个边缘点 (x, y) 的一个邻域内的像素，根据某种准则将相似点进行连接，由满足该准则的像素连接形成边缘。
- ✓ 如何确定边缘像素的相似性
 - 边缘像素梯度算子的响应强度
 - 边缘像素梯度算子的方向



2、边缘跟踪

利用边缘像素梯度的相似性

- 对做过边缘检测的图像的每个点的特性进行分析
- 分析在一个小的邻域 (3×3 或 5×5) 中进行
- 所有相似的点被连接, 形成一个享有共同特性像素的边界。
- 用比较梯度值的幅度和梯度方向确定两个点是否同属一条边。





2、边缘跟踪

局部处理

- 边缘像素梯度算子的响应强度

如果 $\nabla f(x, y) - \nabla f(x_0, y_0) \leq E$

则 (x, y) 邻域内坐标为 (x_0, y_0) 的边缘像素，在幅度上相似于 (x, y) 的像素

- 边缘像素梯度算子的方向

如果 $|\alpha(x, y) - \alpha(x_0, y_0)| < A$, $\alpha(x, y) = \arctan\left(\frac{G_y}{G_x}\right)$

则 (x, y) 邻域内坐标为 (x_0, y_0) 的边缘像素，在角度上相似于 (x, y) 的像素



2、边缘跟踪

局部处理

连接算法描述：

- 1、设定 A 、 E 的阈值大小，确定邻域的大小
- 2、对图像上每一个像素的邻域点进行分析，判断是否需要连接。
- 3、记录像素连接的情况，给不同的边以不同的标记。
- 4、最后，连接断开的线段，删除孤立线段。



函数BWTRACEBOUNDARY

$B = \text{bwtraceboundary}(BW, P, FSTEP)$

BW为输入图像矩阵，值为0的元素为背景像素点，P为2*1
矢量两个元素分别对应起始点的行和列坐标，fstep为字符
串制定其实搜索方向，

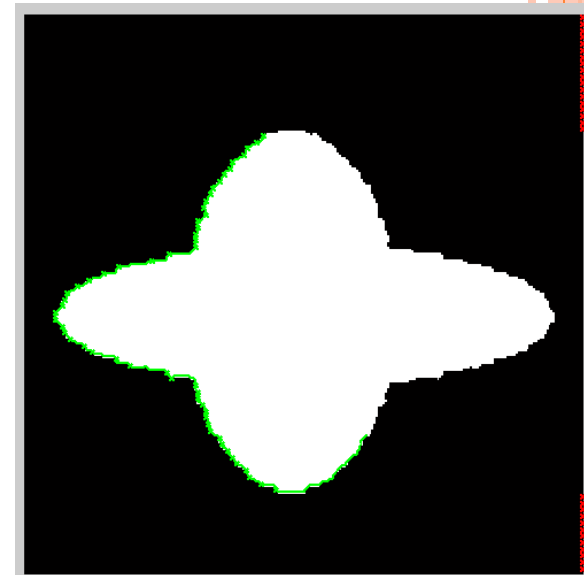
B为Q*2维矩阵其中Q为所提取的边界长度（即边界所含像
素点数目）B矩阵中存储边界像素点的行坐标和列坐标。



```
clear all;  
BW=imread('ellipse.bmp');  
BW=im2bw(BW);  
imshow(BW,[]);  
s=size(BW);  
for row=2:3:s(1)  
    for col=1:s(2)  
        if BW(row,col),break;  
    end  
end
```



```
coutour=bwtraceboundary(BW,[row,col],'W',8,50,'  
counterclockwise');  
if(~isempty(coutour));  
    hold on;  
    plot(coutour(:,2),coutour(:,1),'g','LineWidth',2);  
    hold on;  
    plot(col,row,'gx','LineWidth',2);  
else  
    hold on;  
    plot(col,row,'rx','LineWidth',2);  
end  
end
```





2、边缘跟踪

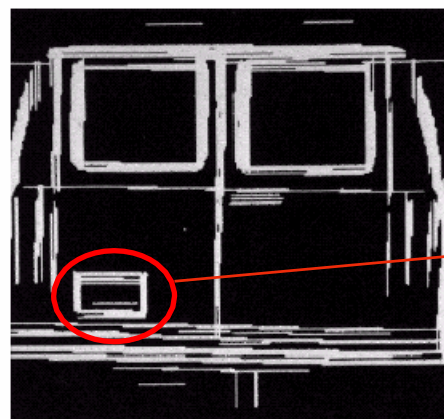
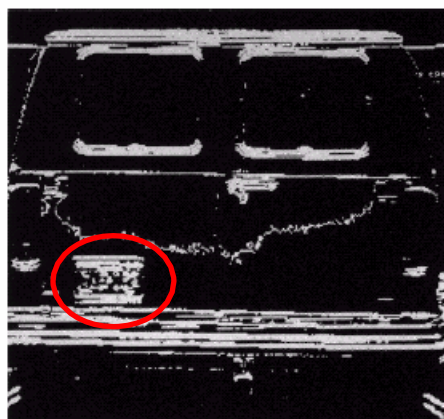
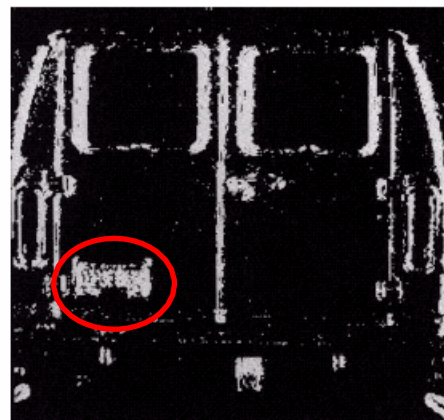
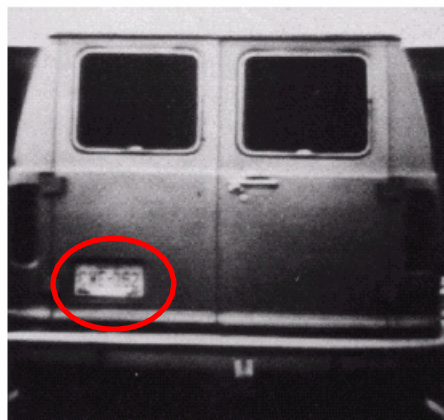
原始图像：检测车牌牌照

Sobel算子的 G_y 分量

a b
c d

FIGURE 10.16

(a) Input image.
(b) G_y component of the gradient.
(c) G_x component of the gradient.
(d) Result of edge linking. (Courtesy of Perceptics Corporation.)



美国牌照的长宽比例是2:1

Sobel算子的 G_x 分量

边缘连接的结果



2、边缘跟踪

Hough 变换

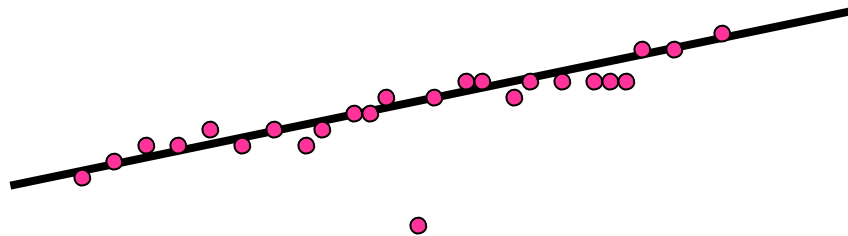
- ✓ 问题的提出
- ✓ Hough变换的基本思想
- ✓ 算法实现
- ✓ Hough变换的扩展



2、边缘跟踪

问题的提出

- ✓ 在找出边界点集之后，需要连接，形成完整的边界图形描述





2、边缘跟踪

基本思想

- Hough变换的基本思想
 - ✓ 对于边界上的 n 个点的点集，找出共线 的点集和直线方程。
 - ✓ 对于任意两点的直线方程： $y = ax + b$ ，构造一个参数 a ， b 的平面，从而有 如下结论：



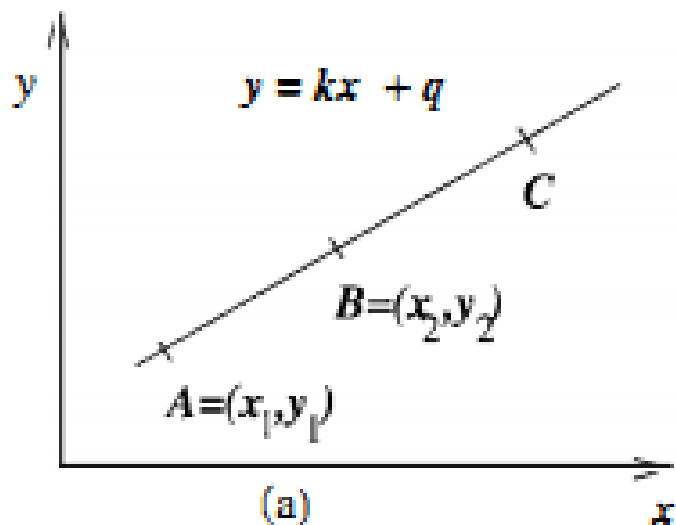


2、边缘跟踪

基本思想

- Hough变换的基本思想

一个直线可由两个点 $A = (x_1, y_1)$ 和 $B = (x_2, y_2)$ 确定。（笛卡尔坐标）



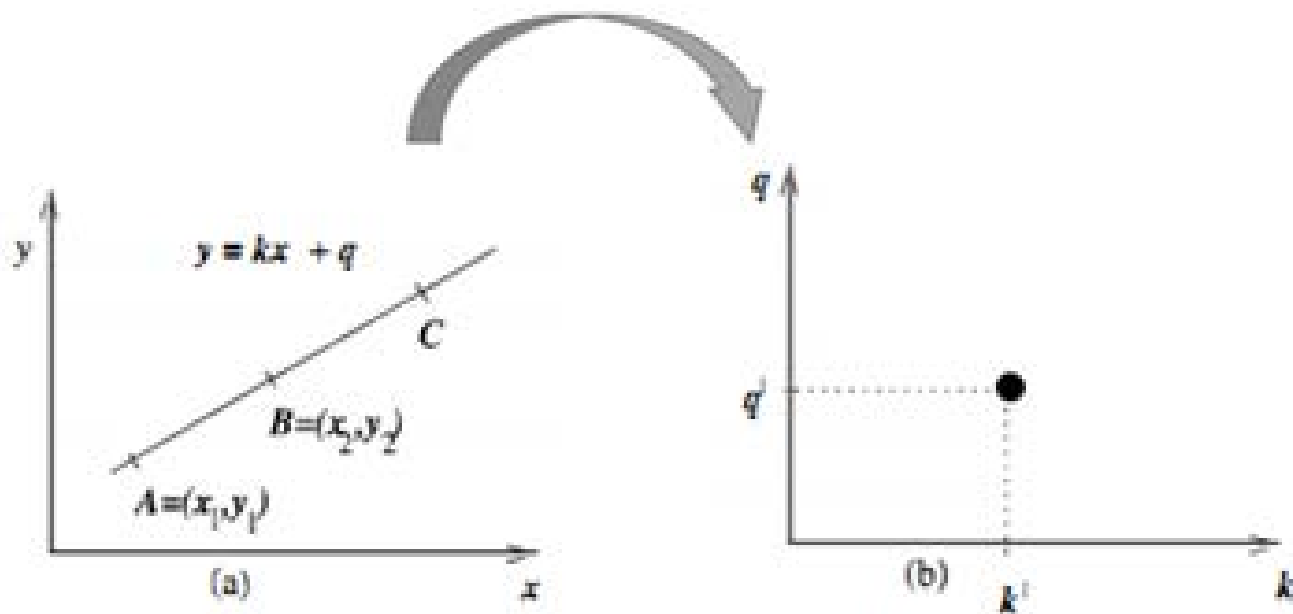
另外， $y = kx + q$ 也可以写成关于 (k, q) 的函数表达式。（霍夫空间）

$$\begin{cases} q = -kx_1 + y_1 \\ q = -kx_2 + y_2 \end{cases} \quad 47$$



2、边缘跟踪

基本思想



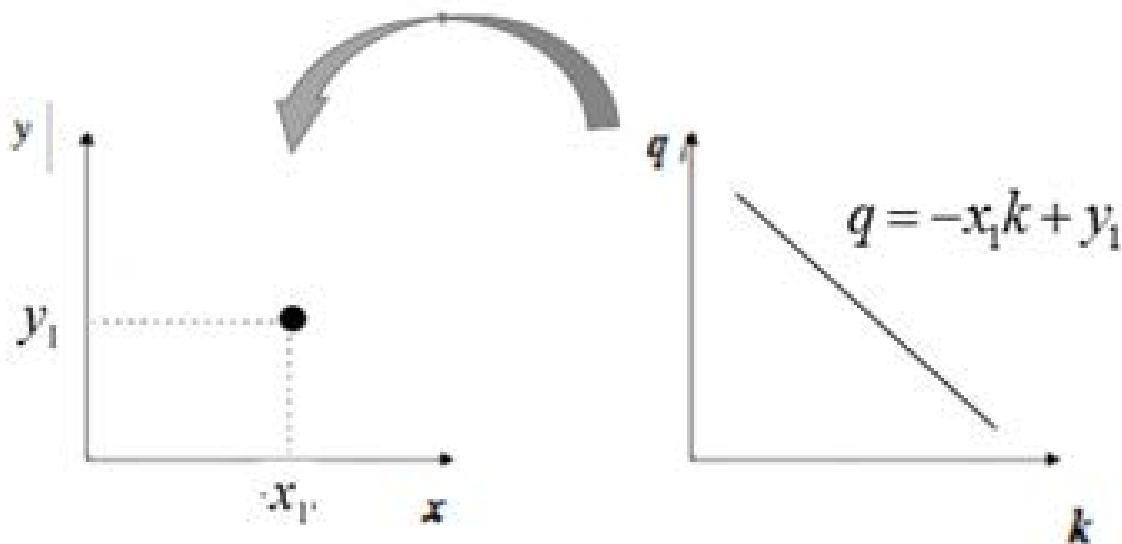
变换后的空间成为霍夫空间。即：笛卡尔坐标系中一条直线，对应霍夫空间的一个点。



2、边缘跟踪

基本思想

反过来同样成立（霍夫空间的一条直线，对应笛卡尔坐标系的一个点）：

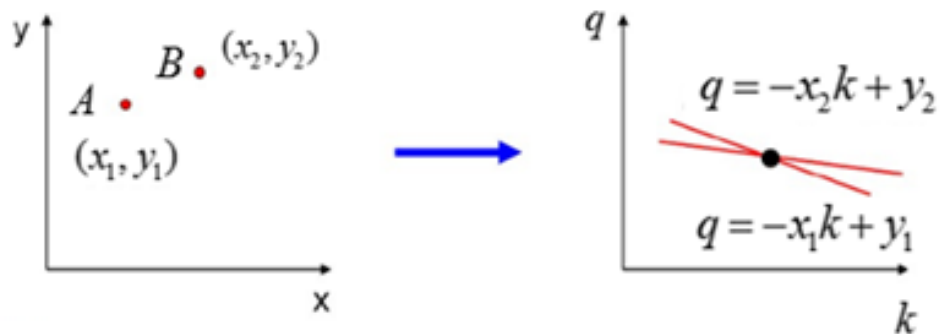




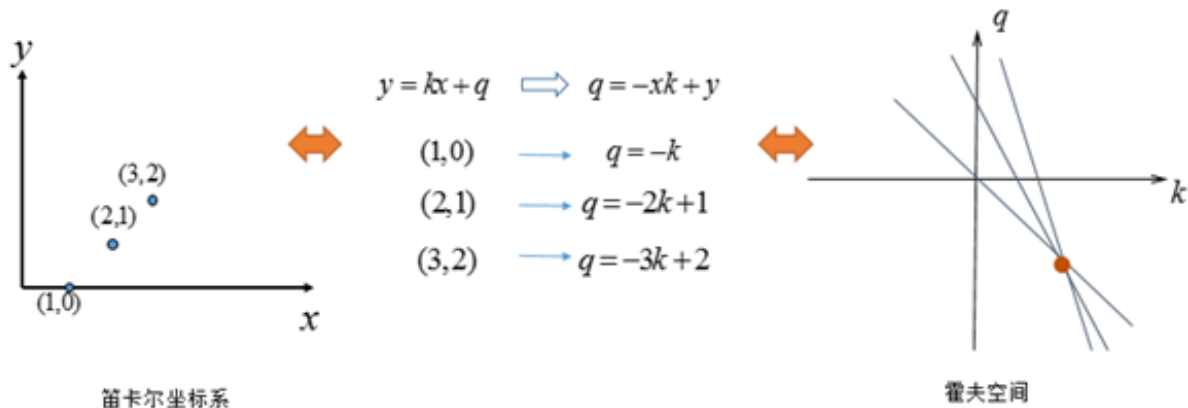
2、边缘跟踪

基本思想

再来看看A、B两个点，对应霍夫空间的情形：



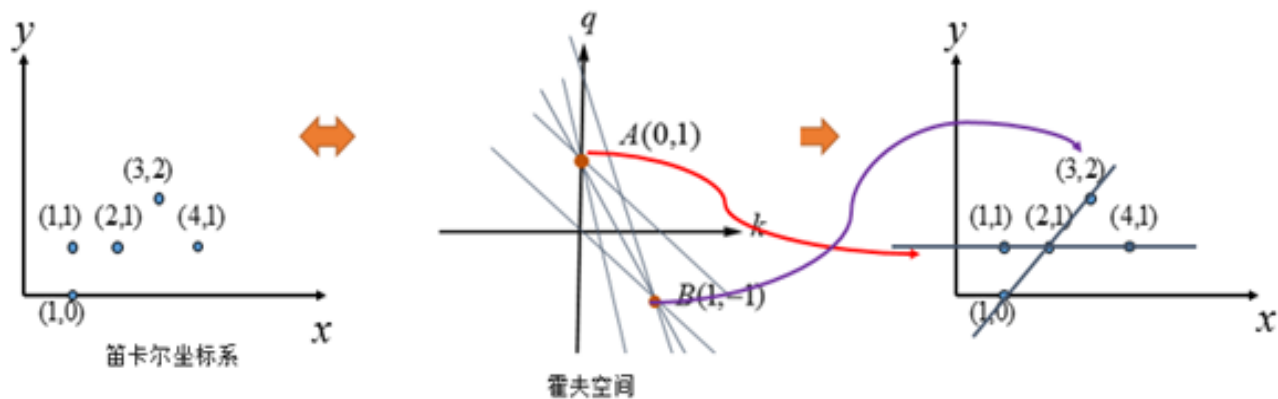
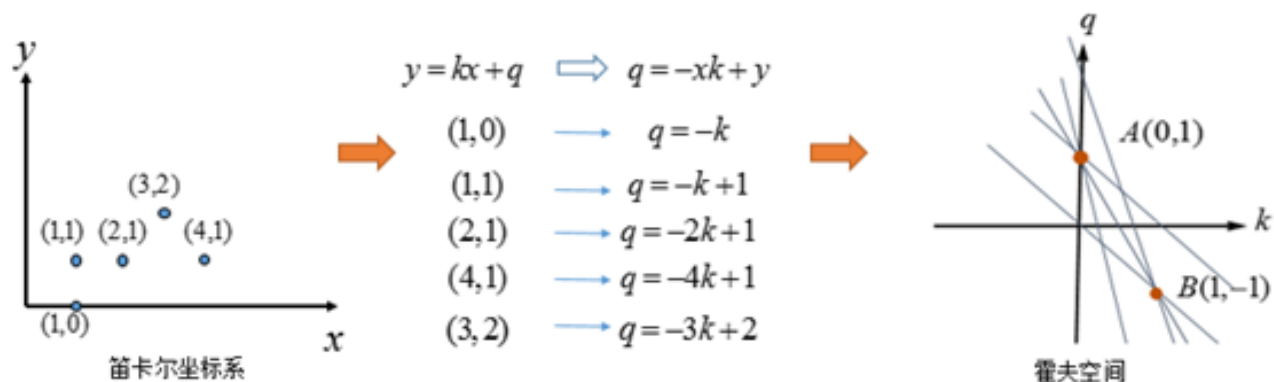
三个点共线的情况：





2、边缘跟踪

基本思想





2、边缘跟踪

算法实现

- ✓ 由于垂直直线 a 为无穷大，我们改用极坐标形式： $x \cos \theta + y \sin \theta = \rho$
- ✓ 参数平面为 θ, ρ ，对应不是直线而是正弦曲线
- ✓ 使用交点累加器，或交点统计直方图，找出相交线段最多的参数空间的点
- ✓ 然后找出该点对应的 xy 平面的直线线段



2、边缘跟踪

Hough扩展

- ✓ Hough变换不只对直线，也可以用于圆：

$$(x - c_1)^2 + (y - c_2)^2 = c_3^2$$

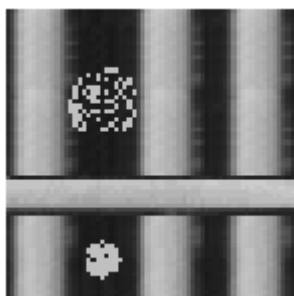
- ✓ 这时需要三个参数的参数空间



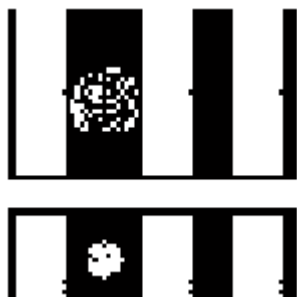
2、边缘跟踪

Hough 变换

直线检测



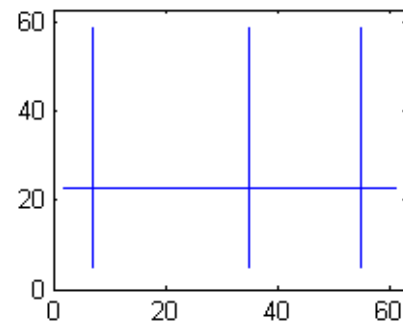
原始图像



二值化图像



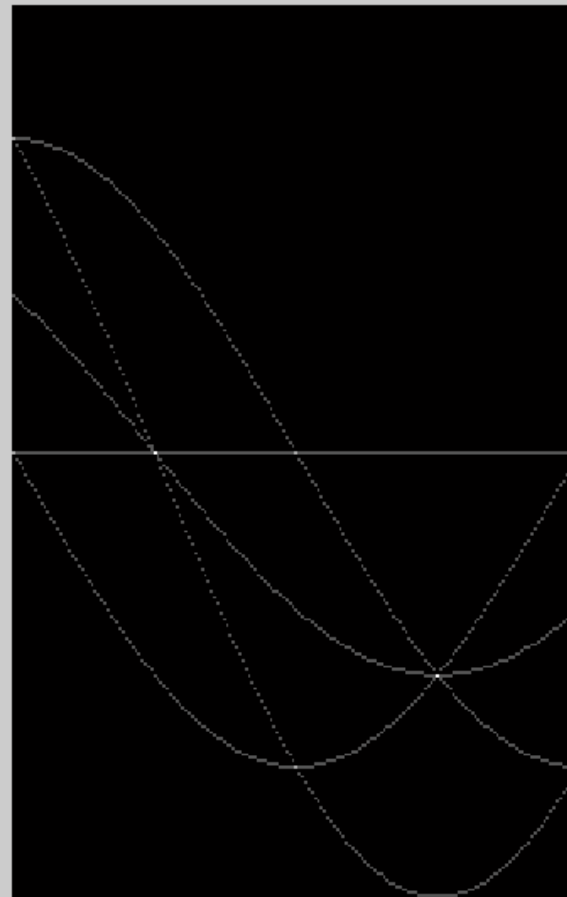
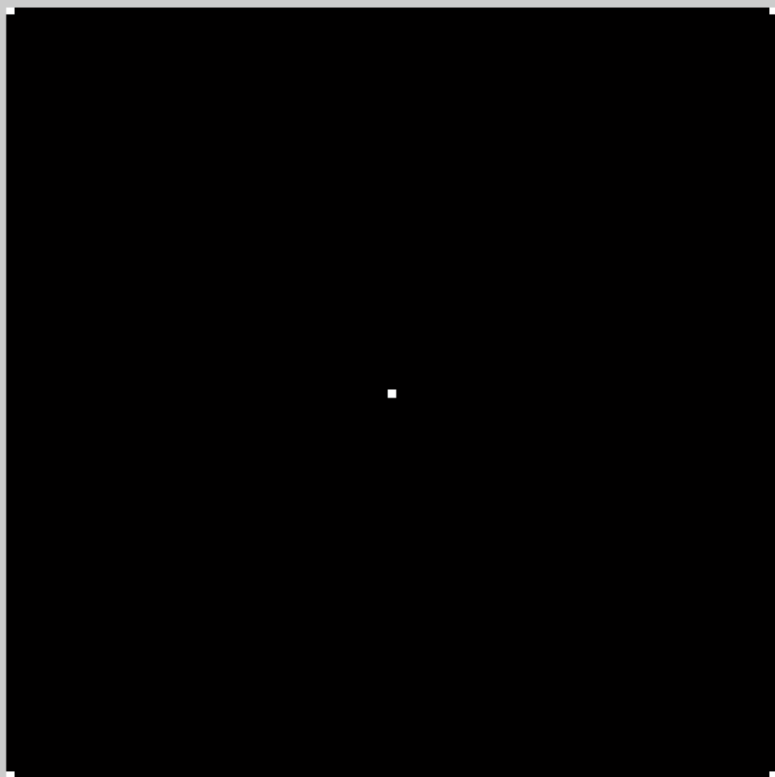
细化图像



Hough 变换检测出的直线



```
clear all;  
% 创建合成图像  
f = zeros(101, 101);  
f(1, 1) = 1;  
f(101, 1) = 1;  
f(1, 101) = 1;  
f(101, 101) = 1;  
f(51, 51) = 1;  
  
% 显示合成图像  
figure;  
subplot(121), imshow(f);  
  
% 采用默认值进行hough变换,  
并显示hough变换的结果  
H = hough(f);  
subplot(122), imshow(H, []);
```

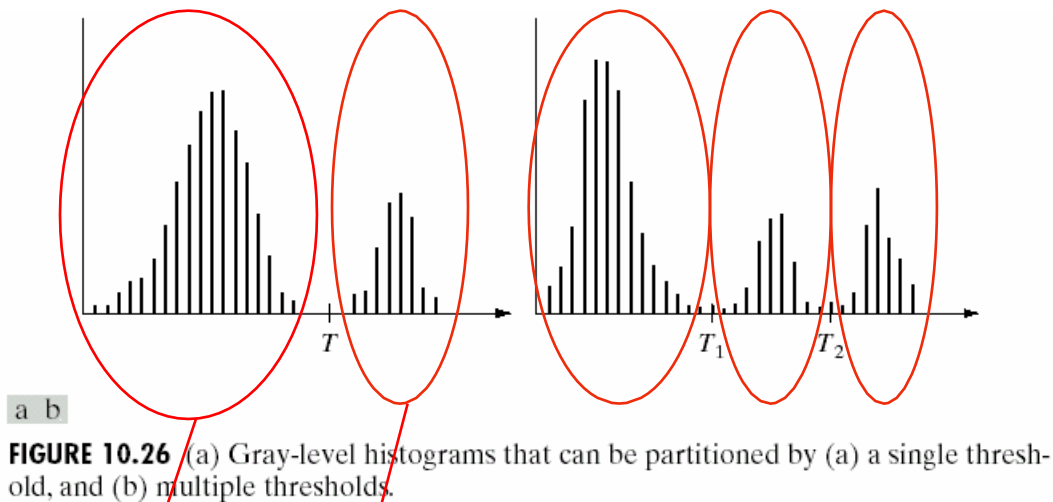




6.3 基于区域的分割

基础

1、阈值分割



暗的背景: $f(x,y) \leq T$

亮的对象: $f(x,y) > T$

暗的背景: $f(x,y) \leq T_1$

亮的一个对象: $T_1 < f(x,y) \leq T_2$

亮的另一个对象: $f(x,y) > T_2$



1、 阈值分割

✓ 阈值处理操作

$$T = T[x, y, p(x, y), f(x, y)]$$

$f(x, y)$ 是点 (x, y) 的灰度级， $p(x, y)$ 表示该点的局部性质，如以 (x, y) 为中心的邻域的平均灰度级

✓ 阈值处理后的图像 $g(x, y)$ 定义为

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



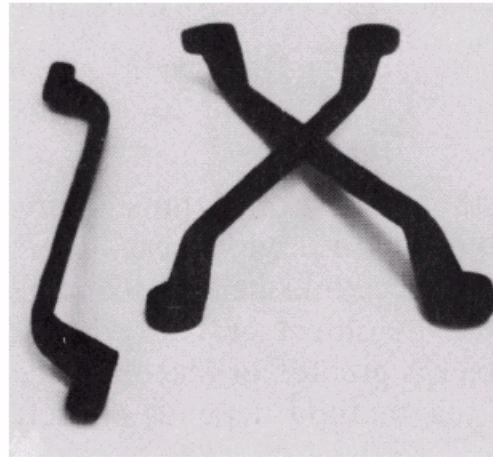
1、 阈值分割

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

- ✓ 标记为1的像素对应于对象，标记为0的像素对应于背景
- ✓ 当T仅取决于 $f(x, y)$ ， 阈值称为全局的
- ✓ 当T取决于 $f(x, y)$ 和 $p(x, y)$ ， 阈值是局部的
- ✓ 当T取决于空间坐标 x 和 y ， 阈值就是动态的或自适应的



1、阈值分割



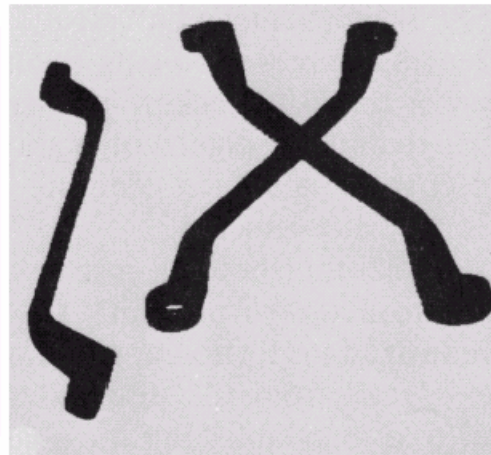
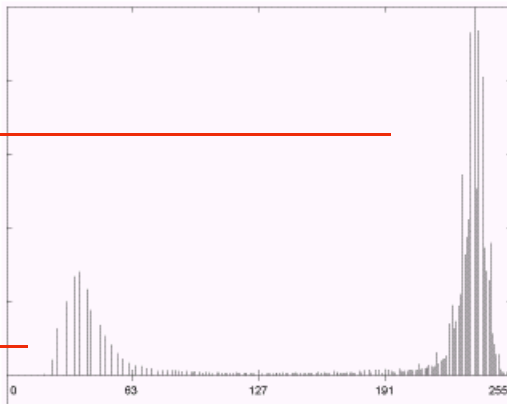
a
b c

FIGURE 10.28

(a) Original image. (b) Image histogram. (c) Result of global thresholding with T midway between the maximum and minimum gray levels.

亮背景

暗对象







```
clear all;  
G=imread('finger.png');  
BW=rgb2gray(G);  
imshow(BW);  
for i=1:304  
    for j=1:300  
        if BW(i,j)<200  
            BI(i,j)=0;  
        else BI(i,j)=1;  
        end  
    end  
end  
figure;  
imshow(BI);
```



1、 阈值分割

- 计算基本全局阈值算法

- a 选择一个T的初始估计值
- b 用T分割图像，生成两组像素： G_1 由所有灰度值大于T的像素组成，而 G_2 由所有灰度值小于或等于T的像素组成
- c 对区域 G_1 和 G_2 中的所有像素计算平均灰度值 μ_1 和 μ_2
- d 计算新的阈值 $T=1/2(\mu_1 + \mu_2)$
- e 重复步骤2到4，直到逐次迭代所得的T值之差小于事先定义的参数 T_0

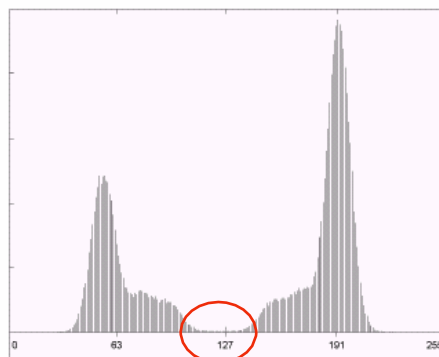


1、阈值分割

原图



原图的直方图



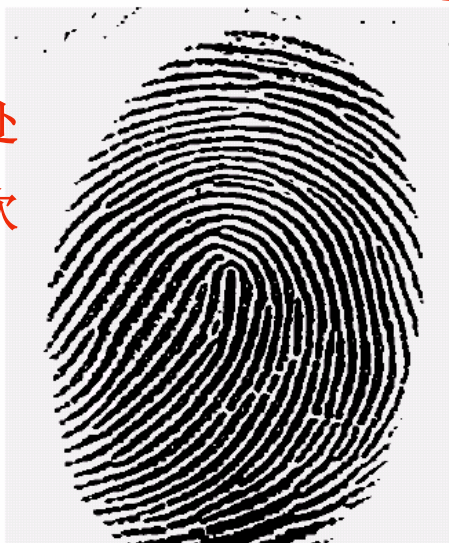
a b
c

FIGURE 10.29

(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.)

基本全局阈值算法处理的结果 $T_0=0$ ，3次迭代得到值为125.4
最后确定 $T=125$



波谷作为阈值



1、 阈值分割

基本自适应阈值

✓ 单一全局阈值存在的问题

不均匀亮度图像无法有效分割

✓ 方法

将图像进一步细分为子图像，并对不同的子图像使用不同的阈值处理

✓ 解决的关键问题：

如何将图像进行细分和如何为得到的子图像估计阈值

➤ 自适应阈值：取决于像素在子图像中的位置



1、 阈值分割

原图

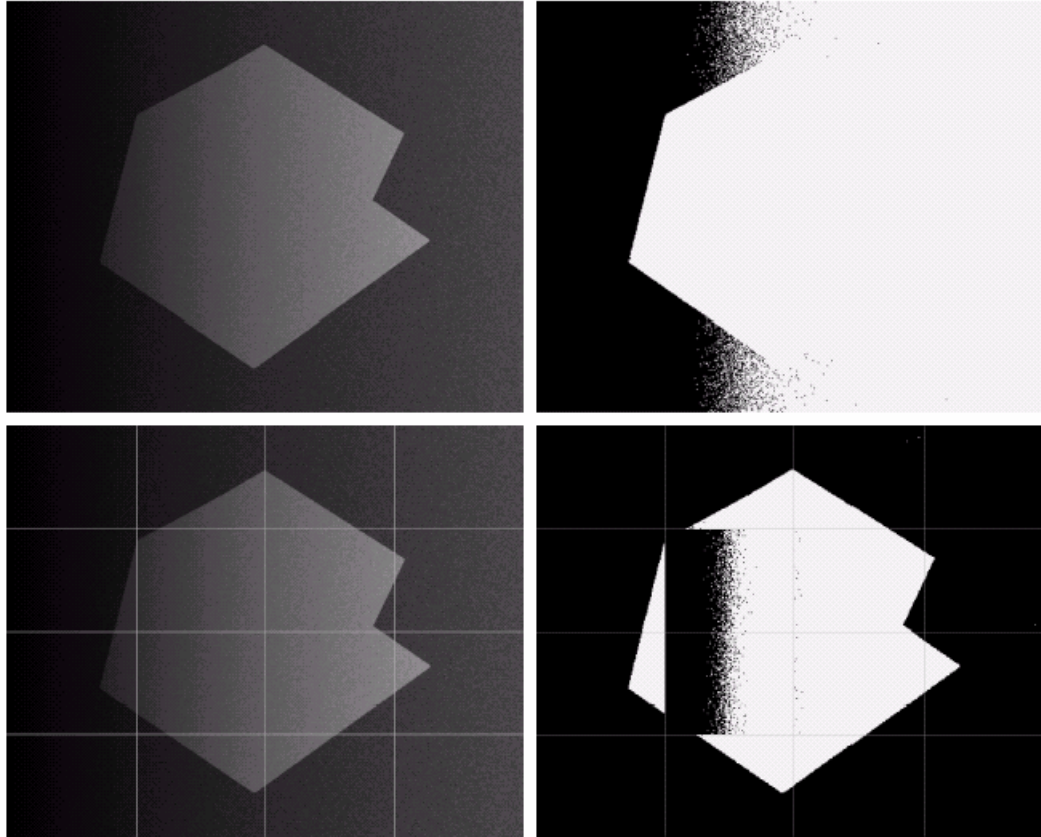
一个全局阈值处理后的结果:

人工设置直方图的波谷为阈值

a b
c d

FIGURE 10.30

(a) Original image. (b) Result of global thresholding. (c) Image subdivided into individual subimages. (d) Result of adaptive thresholding.



分割为子图像: 4等分
后 再4等分

自适应阈值处理的结果



1、阈值分割

- 最佳全局和自适应阈值
 - ✓ 假设一幅图像仅包含两个主要的灰度级区域。令 z 表示灰度级值，则两个灰度区域的直方图可以看作它们概率密度函数(PDF)的估计 $p(z)$
 - ✓ $p(z)$ 是两个密度的和或混合。一个是图像中亮区域的密度，另一个是暗区域的密度
 - ✓ 如果 $p(z)$ 已知或假设，则它能够确定一个最佳阈值（具有最低的误差）将图像分割为两个可区分的区域



1、阈值分割

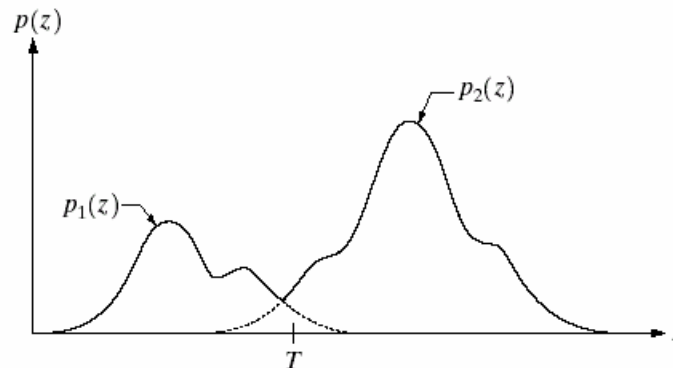
✓ 假设2个PDF中较大的一个对应背景的灰度级，较小的一个描述了图像中对象的灰度级，则混合PDF是

$$p(z) = P_1p_1(z) + P_2p_2(z)$$

P_1 是属于对象像素的概率， P_2 是属于背景像素的概率，假设图像只包括对象和背景，则

$$P_1 + P_2 = 1$$

FIGURE 10.32
Gray-level
probability
density functions
of two regions in
an image.





1、阈值分割

✓ 在区间 $[a, b]$ 内取值的随机变量的概率是它的概率密度函数从 a 到 b 的积分，即在这两个上下限之间PDF曲线围住的面积，因此，将一个背景点当作对象点进行分类时，错误发生的概率为：

$$E_1(T) = \int_{-\infty}^T p_2(z) dz$$

这是在曲线 $p_2(z)$ 下方位于阈值左边区域的面积

✓ 将一个对象点当作背景点进行分类错误发生的概率为

$$E_2(T) = \int_T^{\infty} p_1(z) dz$$

这是在曲线 $p_1(z)$ 下方位于阈值右边区域的面积



1、阈值分割

✓ 出错率的整体概率是

$$E(T) = P_2 E_1(T) + P_1 E_2(T)$$

✓ 为了找到出错最少的阈值，使用莱布尼兹法则把 $E(T)$ 对 T 求微分并令结果等于0，得到

$$P_2 p_1(T) = P_1 p_2(T)$$

✓ 上式解出 T ，即为最佳阈值

✓ 如果 $P_1 = P_2$ ，则最佳阈值位于曲线 $p_1(z)$ 和 $p_2(z)$ 的交点处



1、 阈值分割

✓ 高斯密度可以用两个参数均值和方差描述

$$p(z) = \frac{P_1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(z-\mu_1)^2}{2\sigma_1^2}} + \frac{P_2}{\sqrt{2\pi}\sigma_2} e^{-\frac{(z-\mu_2)^2}{2\sigma_2^2}}$$

✓ 出错最少的阈值T的解

$$AT^2 + BT + C = 0$$

$$A = \sigma_1^2 - \sigma_2^2$$

$$B = 2(\mu_1\sigma_2^2 - \mu_2\sigma_1^2)$$

$$C = \sigma_1^2\mu_2^2 - \sigma_2^2\mu_1^2 + 2\sigma_1^2\sigma_2^2 \ln(\sigma_2 P_1 / \sigma_1 P_2)$$



1、阈值分割

- ✓ 如果方差相等，则得到单一的阈值

$$T = \frac{\mu_1 + \mu_2}{2} + \frac{\sigma^2}{\mu_1 + \mu_2} \ln \left(\frac{P_2}{P_1} \right)$$

- ✓ 如果 $P_1 = P_2$ ，最佳阈值是均值的平均数



1、阈值分割

● 通过边界特性选择阈值

基本思想：

- 如果直方图的各个波峰很高、很窄、对称，且被很深的波谷分开时，有利于选择阈值
- 为了改善直方图的波峰形状，我们只把区域边缘的像素绘入直方图，而不考虑区域中间的像素
- 用微分算子处理图像，使图像只剩下边界中心两边的值



1、阈值分割

- 通过边界特性选择阈值

优点：

- 1) 在前景和背景所占区域面积差别很大时，不会造成一个灰度级的波峰过高，而另一个过低。
- 2) 边缘上的点在区域内还是区域外的概率是相等的，因此可以增加波峰的对称性
- 3) 基于梯度和拉普拉斯算子选择的像素，可以增加波峰的高度



1、阈值分割

● 通过边界特性选择阈值 算法的实现：

- 1) 对图像进行梯度计算，得到梯度图像。
 - 2) 得到梯度值最大的那一部分（比如10%）的像素直方图
 - 3) 通过直方图的谷底，得到阈值T
- ✓ 如果用拉普拉斯算子，不通过直方图，直接得到阈值，方法是使用拉普拉斯算子过滤图像，将 0_{75} 跨越点对应的灰度值为阈值T



1、阈值分割

● 基于不同变量的阈值

- ✓ 在某些情况下，传感器可以产生不止一个在图像中描述每一个像素的可利用的变量，因此，允许进行多谱段阈值处理
- ✓ 例如一幅有3个变量的图像(RGB分量)，每个像素有16种可能的灰度级，构成 $16 \times 16 \times 16$ 种灰度级（网格，立方体）
- ✓ 阈值处理就是在三维空间内寻找点的聚簇的过程。如在直方图中找到有效点簇K，可以对RGB分量值接近某一个簇的像素赋予一个任意值(如白色的值)，对其它像素赋予另一个值（如黑色的值）
- ✓ 彩色图像处理中的色调和饱和度易于图像分割



1、阈值分割

彩色照片的单色图像

对应于脸部色调的一个簇进行阈值处理得到

对应于红色轴的一个簇 进行阈值处理得到，红色的围巾和花出现在分割结果中



a b c

FIGURE 10.39 (a) Original color image shown as a monochrome picture. (b) Segmentation of pixels with colors close to facial tones. (c) Segmentation of red components.

- ✓ 原彩色图是由16位RGB图像组成的
- ✓ 围巾是鲜红色，头发和脸部的颜色很浅



2、区域生长法

基本概念

✓ 目标：将区域R划分为若干个子区域
 R_1, R_2, \dots, R_n ，这些子区域满足5个条件：

1) 完备性：

$$\bigcup_{i=1}^n R_i = R$$

2) 连通性：每个 R_i 都是一个连通区域

3) 独立性：对于任意 $i \neq j$ ， $R_i \cap R_j = \Phi$



2、区域生长法

基本概念

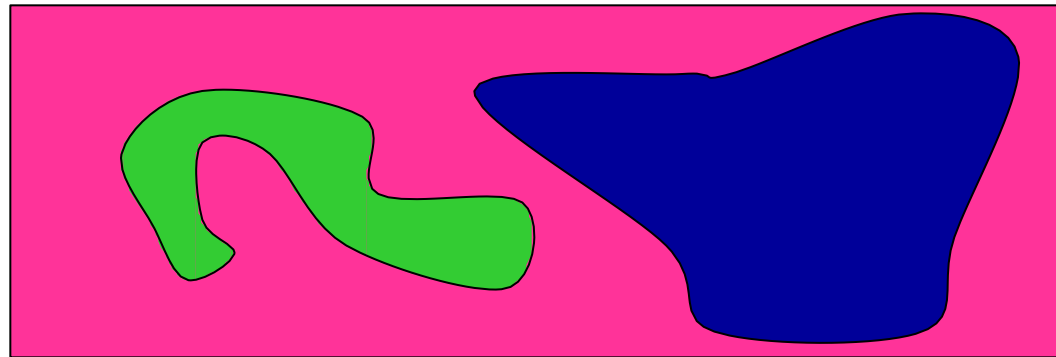
- 基本概念

4) 单一性：每个区域内的灰度级相等，

$$P(R_i) = \text{TRUE}, i = 1, 2, \dots, n$$

5) 互斥性：任两个区域的灰度级不等， P

$$(R_i \cup R_j) = \text{FALSE}, i \neq j$$





2、区域生长法

基本概念

● 区域增长的算法实现:

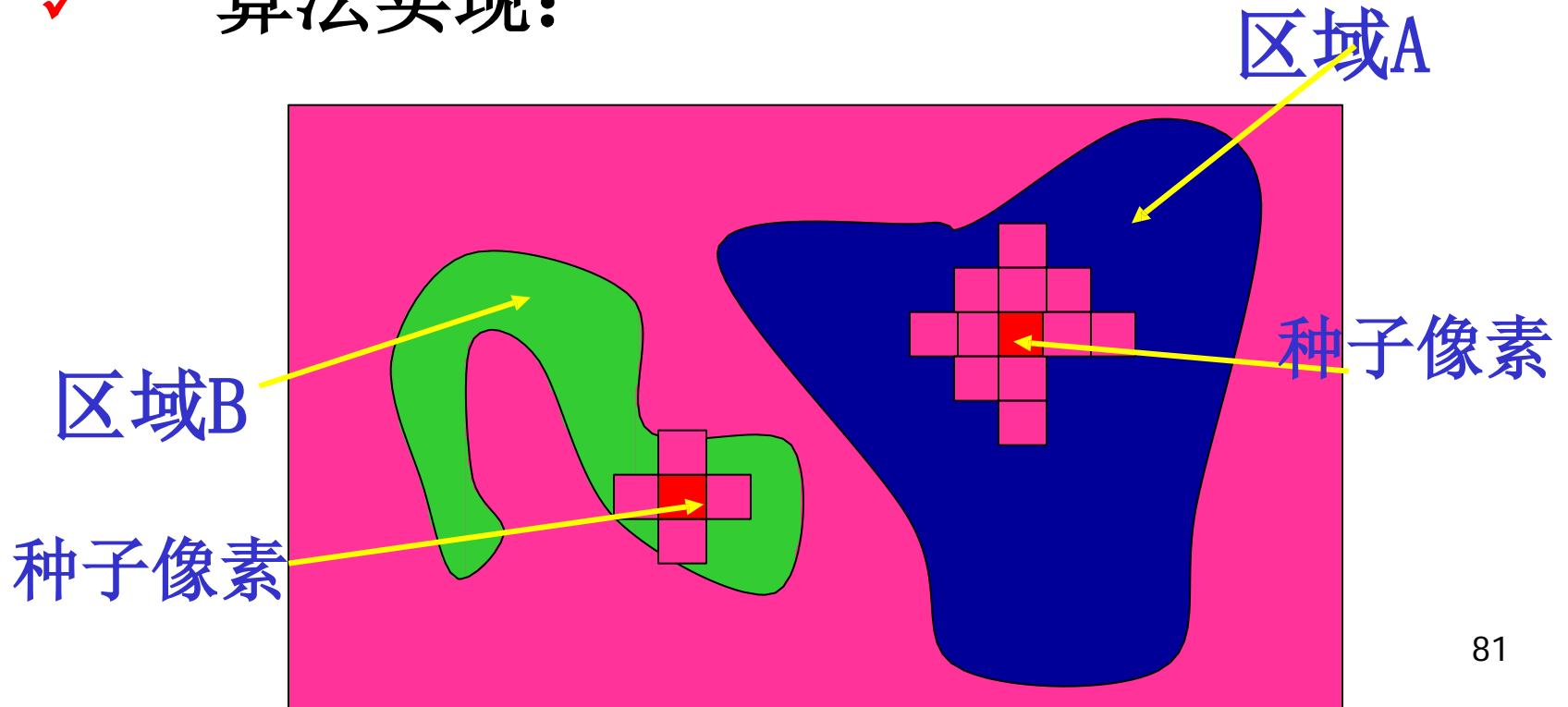
- 1) 根据图像的不同应用选择一个或一组种子，它或者是最亮或最暗的点，或者是位于点簇中心的点
- 2) 选择一个描述符（条件）
- 3) 从该种子开始向外扩张，首先把种子像素加入结果集合，然后不断将与集合中各个像素连通、且满足描述符的像素加入集合
- 4) 上一过程进行到不再有满足条件的新结点加入集合为止



2、区域生长法

- 通过像素集合的区域增长

✓ 算法实现:

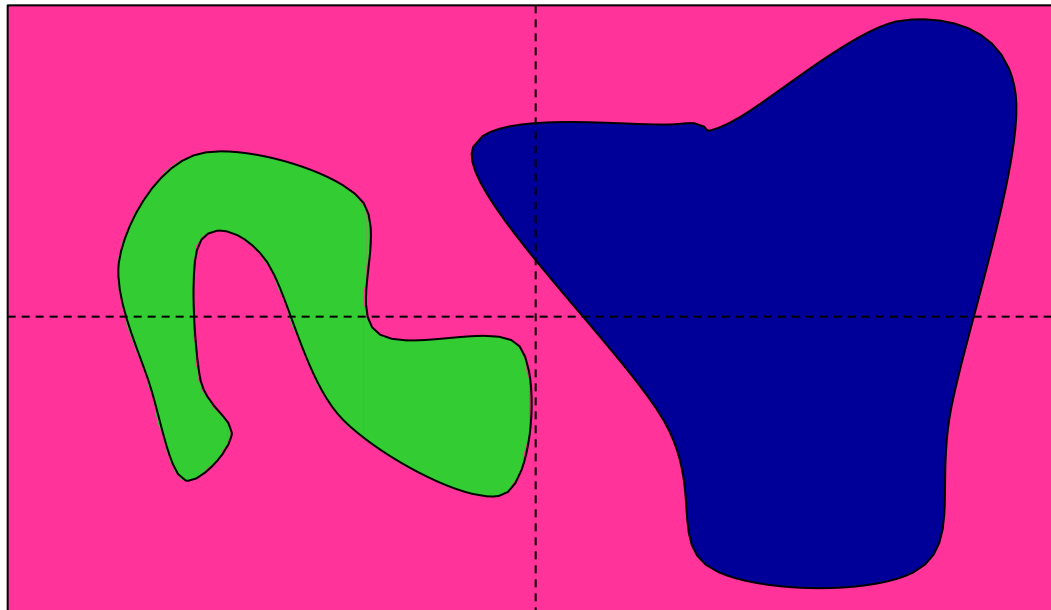




3、分裂合并法

✓ 算法实现：

1) 对图像中灰度级不同的区域，均分为四个子区域





3、分裂合并法

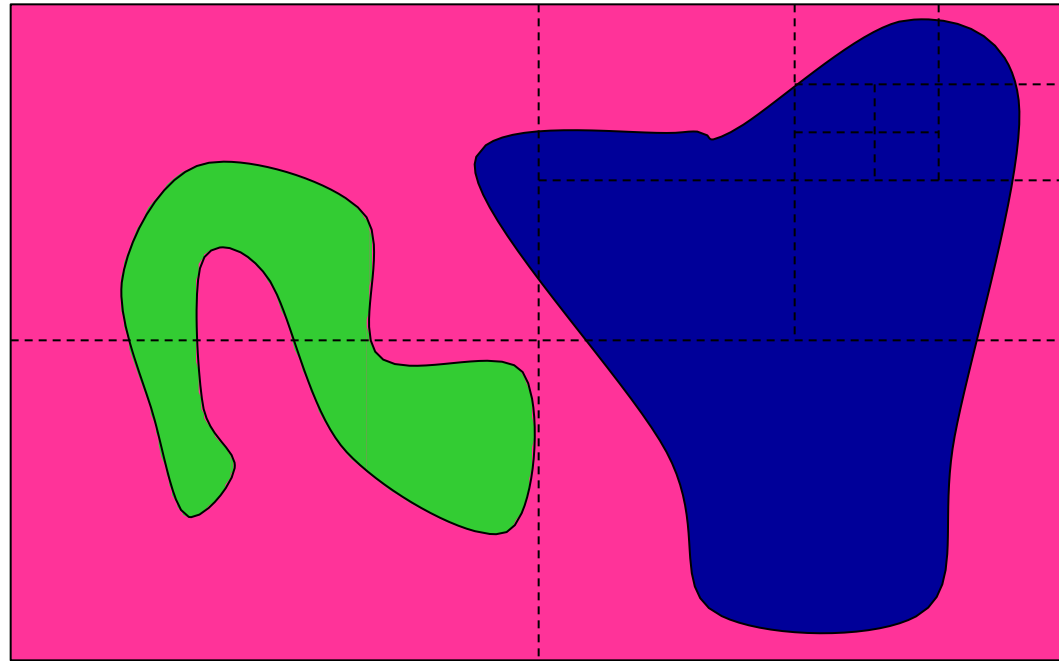
✓ 算法实现：

2) 如果相邻的子区域所有像素的灰度级相同，则将其合并

3) 反复进行上两步操作，直至不再有新的分裂与合并为止



3、分裂合并法





3、分裂合并法

- 区域分裂与合并

✓ 算法实现：实际应用中还可作以下修改：

$P(R_i)$ 的定义为：

1) 区域内多于80%的像素满足不等式

$|z_j - m_i| \leq 2\sigma_i$ ， 其中： z_j 是区域 R_i 中第 j 个点的灰度级， m_i 是该区域的平均灰度级， σ_i 是区域的灰度级的标准方差。

2) 当 $P(R_i)=\text{TRUE}$ 时，将区域内所有像素的灰度级置为 m_i



6.4 运动图像目标分割

- ✓ 使用两帧图像 $f(x, y, t_i)$ 和 $f(x, y, t_j)$ 相减的办法,

形成差值图像

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

- ✓ 在动态图像处理过程中, d_{ij} 中值为1的像素被认为是对象运动的结果
- ✓ 考虑图像帧序列 $f(x, y, t_1), f(x, y, t_2), \dots, f(x, y, t_n)$, 并令 $f(x, y, t_1)$ 为基本图像, 一幅累积差异 图像(ADI)由基准图像和图像序列的后续图像对比得到



6.4 运动图像目标分割

✓ 令 $R(x, y)$ 表示基准图像，绝对ADI，正ADI和负ADI 定义如下：

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

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

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



6.4 运动图像目标分割

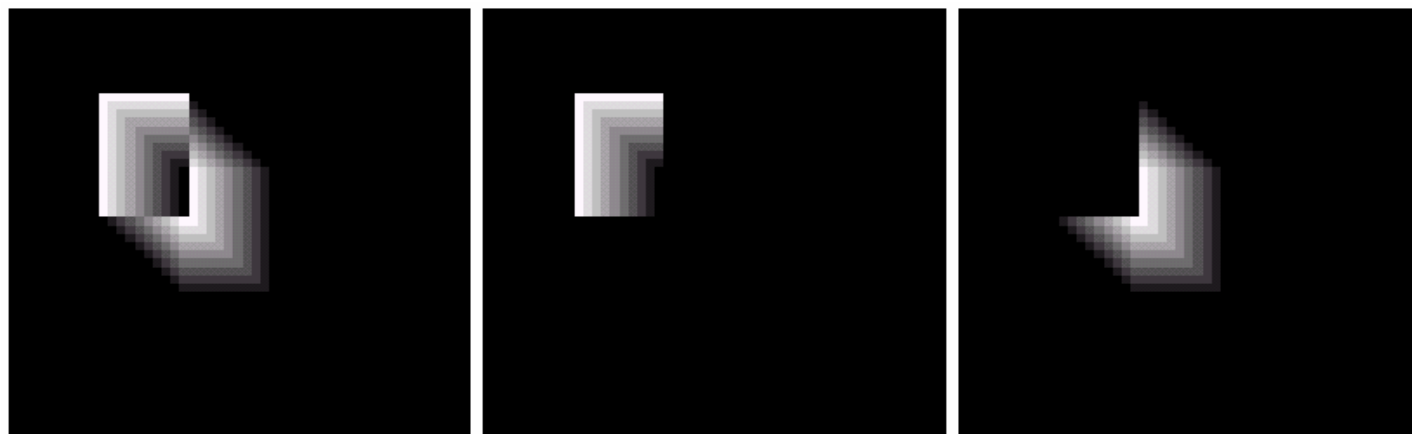
● 空间技术举例

向东南方向运动的矩形目标的ADI

绝对ADI

正ADI

负ADI



a b c

FIGURE 10.49 ADIs of a rectangular object moving in a southeasterly direction. (a) Absolute ADI. (b) Positive ADI. (c) Negative ADI.



练习题：

什么是阈值分割技术？该技术适用于什么场景下的图形分割？