

2022春



数字图像处理



# 图像分割

曹劲舟

助理教授

深圳技术大学

大数据与互联网学院

2022年4月20日

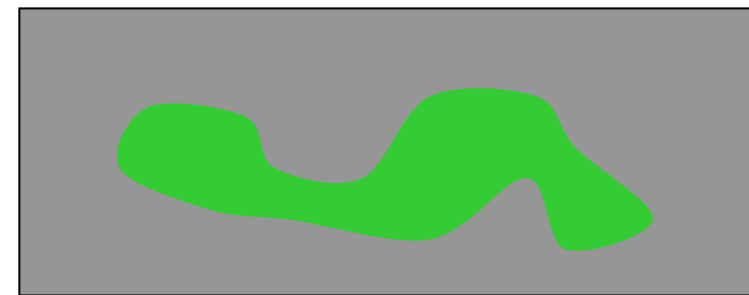
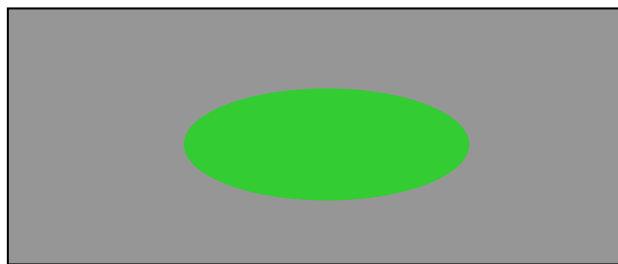
- 图像分割的概念
- 点、线、边缘检测
- 霍夫变换
- 阈值处理
- 基于区域的分割
- 使用分水岭变换的分割

- 在对图像的研究和应用中，人们往往仅对图像中的某些部分感兴趣，这些部分一般称为**目标或前景**。
- 为了辨识和分析目标，需要将有关区域分离提取出来，在此基础上对目标进一步利用，如进行**特征提取和测量**。
- **图像分割**就是指把图像分成各具**特性**的区域并提取出感兴趣目标的技术和过程。
- **特性**可以是灰度、颜色、纹理等，目标可以对应单个区域，也可以对应多个区域。



# 图像分割算法的基本策略

- 分割算法基于**图像灰度值**的两个基本特性：**不连续性**和**相似性**；
- 基于不连续性：基于灰度的突变（如**边缘**），检测图像像素灰度级的**不连续性**，找到点、线（宽度为1）、边（不定宽度）；
- 相似性：根据一组**预定义的规则**，检测图像像素的灰度值的**相似性**，通过选择阈值，将图像分割成相似的区域，区域的外轮廓就是对象的边（阈值处理、区域生长、区域分裂和聚合）



## – 基于边缘的分割方法

- 先提取区域边界，再确定边界限定的区域；

## – 区域分割

- 确定每个像素的归属区域，从而形成一个区域图；

## – 区域生长

- 将属性接近的连通像素聚集成区域；

## – 分裂 - 合并分割

- 综合利用前两种方法，既存在图像的划分，又有图像的合并；

- 图像分割的概念
- 点、线、边缘检测
- 霍夫变换
- 阈值处理
- 基于区域的分割
- 使用分水岭变换的分割

- 间断检测
  - 点检测
  - 线检测
  - 边缘检测

# 间断检测——灰度局部剧烈变化的检测

- 就像局部平均平滑一幅图像那样，假设**平均处理**类似于**积分**，对于**灰度的突变**，局部变化可以用**微分**来检测。
- 由于变化非常短促，因此一阶微分和二阶微分特别适合。
- 数字函数的导数可用差分定义
  - 一维函数 $f(x)$ 在点 $x$ 处的导数的近似：数字差分

$$\frac{\partial f}{\partial x} = f'(x) = f(x+1) - f(x)$$

- 关于点 $x$ 处的二阶导数的近似：

$$\frac{\partial^2 f}{\partial x^2} = f''(x) = f(x+1) + f(x-1) - 2f(x)$$

- 计算图像中每个像素位置处的一阶导数和二阶导数的另一种方法是**使用空间模板滤波器**

$$R = w_1 z_1 + w_2 z_2 + \cdots + w_9 z_9 = \sum_{k=1}^9 w_k z_k$$

$z_1$	$z_2$	$z_3$
$z_4$	$z_5$	$z_6$
$z_7$	$z_8$	$z_9$



- 使用如图所示的模板，如果  $R \geq T$ ，则在模板中心位置检测到一个点

其中， $T$ 是阈值， $R$ 是模板计算值。

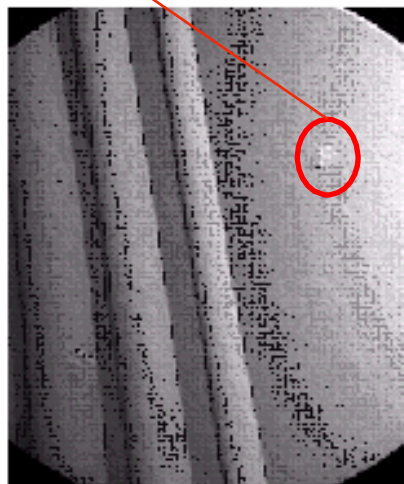
- **基本思想：**如果一个孤立点与它周围的点不同，则可以使用模板进行检测。模板响应的绝对值超过了指定的阈值，则在  $(x, y)$  处的点已被检测到。在输出图像中，该点标为1，否则为0。

-1	-1	-1
-1	8	-1
-1	-1	-1

孔中嵌有一个黑点

应用检测模板的结果

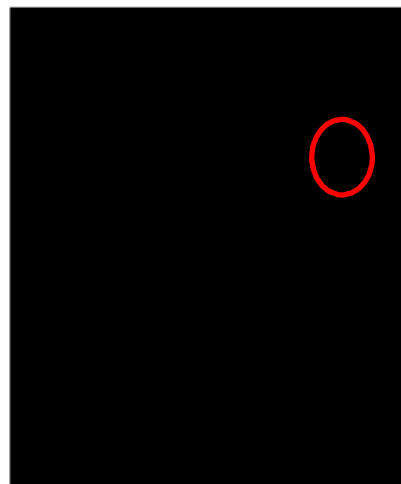
使用图b中像素最高值的  
90% 作为阈值



图a



图b



图c

结论：孤立点可以通过检测模板并设置阈值进行检测

**$g = \text{abs}(\text{imfilter}(\text{tofloat}(f), w)) \geq T;$**

其中， $f$  是输入图像， $w$  是适合点检测的模板， $g$  是包含检测点的图像。

```
f = imread( 'xxx.tif');  
w = [-1 -1 -1;-1 8 -1;-1 -1 -1]; % 点检测掩模  
g = abs(imfilter(double(f),w));  
T = max(g(:));  
g = g >= T;  
subplot(1,2,1);imshow(f);title('(a)原图像');  
subplot(1,2,2);imshow(g);title('(b)点检测');
```

- 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。
- **4个线检测模板：**每个模板的系数之和为0，表明恒定灰度区域中的响应为0
  - 第一个模板对水平线有最大响应
  - 第二个模板对45°方向线有最大响应
  - 第三个模板对垂直线有最大响应
  - 第四个模板对-45°方向线有最大响应

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

R1 水平

R2 -45°

R3 垂直

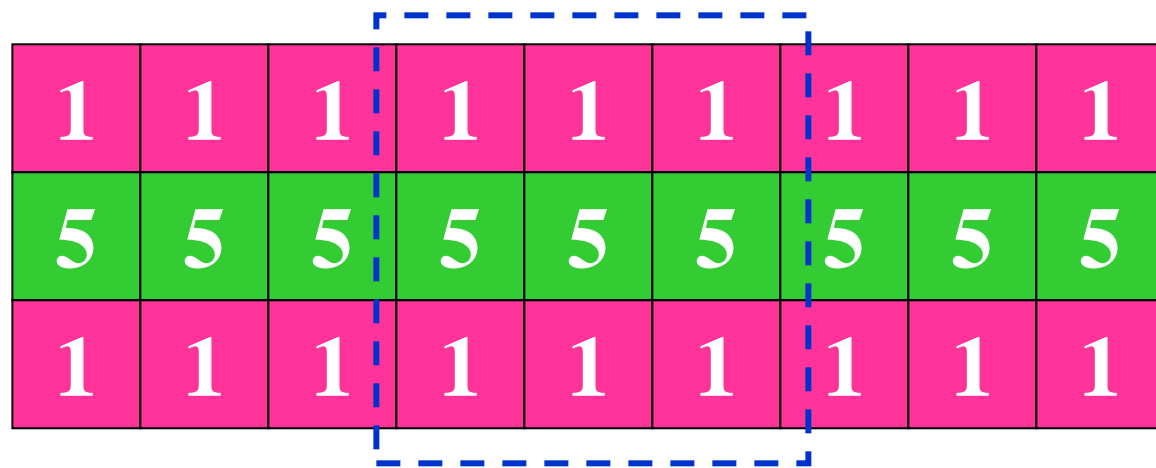
R4 +45°

在图像中心的点，如果

$$|R_i| > |R_j|, j \neq i$$

则此点被认为与在模板  $i$  方向上的线更相关。

例：如果  $|R_1| > |R_j|, j = 2, 3, 4$   
则该点与水平线有更大的关联。



1	1	1	1	1	1	1	1	1
5	5	5	5	5	5	5	5	5
1	1	1	1	1	1	1	1	1

$$R_1 = -6 + 30 = 24$$

$$R_2 = -14 + 14 = 0$$

$$R_3 = -14 + 14 = 0$$

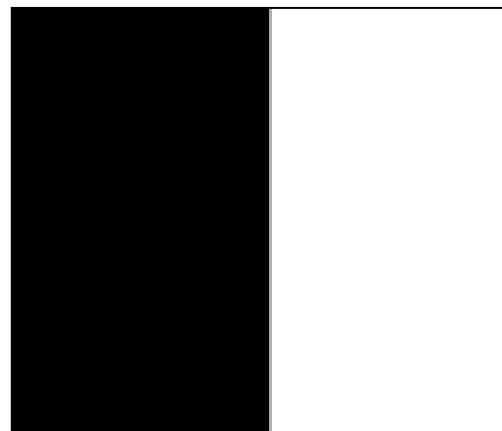
$$R_4 = -14 + 14 = 0$$

## ■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
- 存在于目标与背景、目标与目标、区域与区域、基元与基元之间。

## ■边缘的分类

- 阶跃状
- 屋顶状



阶跃状



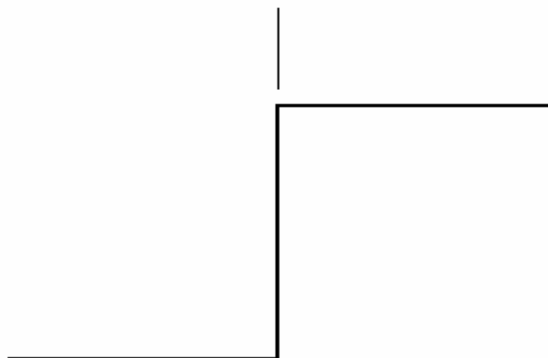
屋顶状

## 理想数字边缘模型

Model of an ideal digital edge



Gray-level profile  
of a horizontal line  
through the image

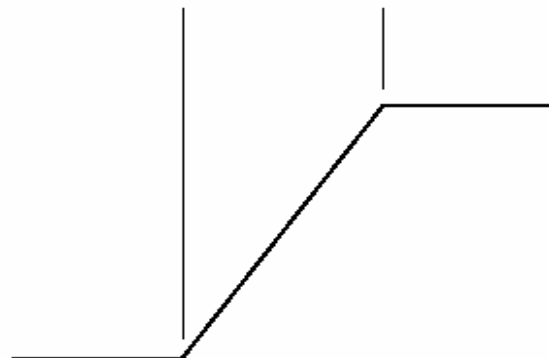


## 斜坡数字边缘模型

Model of a ramp digital edge



Gray-level profile  
of a horizontal line  
through the image



a b

FIGURE 10.5

(a) Model of an ideal digital edge.  
(b) Model of a ramp edge. The slope of the ramp is proportional to the degree of blurring in the edge.

**斜坡的产生**是由光学系统、取样和图像采集系统的不完善带来的边缘模糊造成的。

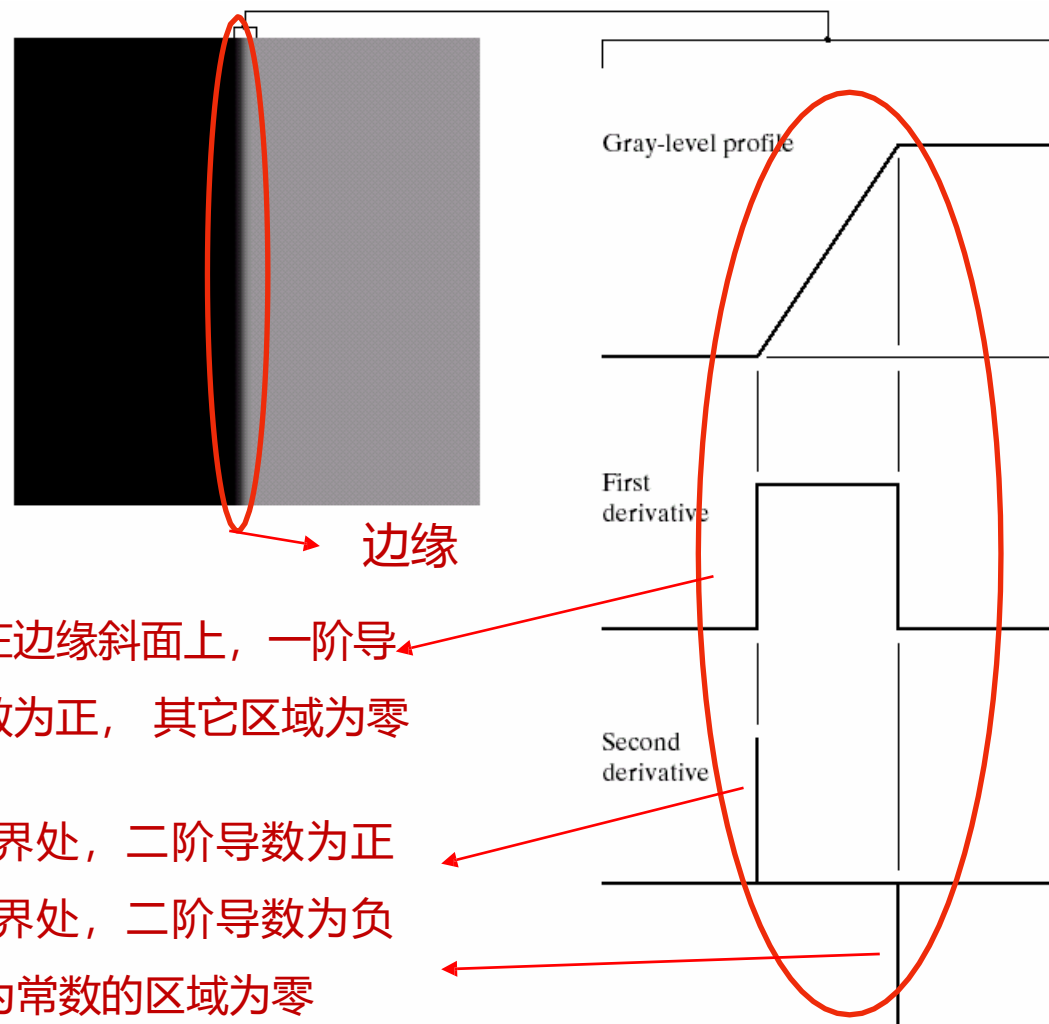
## ■ 边缘检测是基于灰度突变来分割图像的常用方法

- **一阶导数的幅度**可用于检测图像中的某个点处是否存在一个边缘。
- **二阶导数的符号**用于确定一个边缘像素是位于该边缘的暗侧还是位于该边缘的亮侧。

a b

FIGURE 10.6

(a) Two regions separated by a vertical edge.  
(b) Detail near the edge, showing a gray-level profile, and the first and second derivatives of the profile.





## ■ 结论

- 一阶导数可用于检测图像中的一个点是否在边缘上
- 二阶导数可以判断一个边缘像素是在边缘亮的一边还是暗的一边
- 一条连接二阶导数正值和负值的虚构直线将在边缘中点附近穿过零点
- 一阶导数使用**梯度算子**，二阶导数使用**拉普拉斯算子**

□图像 $f(x,y)$ 在 $(x,y)$ 处的梯度定义为一个向量:

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

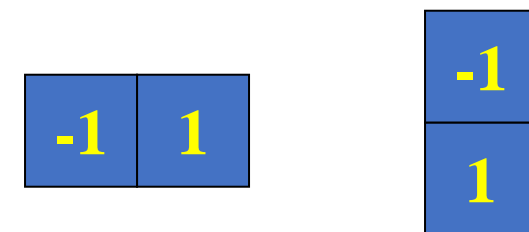
□计算这个向量的大小为:

$$\nabla f = \text{mag}(\nabla F) = \left[ G_x^2 + G_y^2 \right]^{\frac{1}{2}}$$

□可以近似为:  $G \approx |\partial f_x| + |\partial f_y|$  或  $G \approx \max(|\partial f_x|, |\partial f_y|)$

□梯度的方向角为:  $\alpha(x, y) = \arctan \left( \frac{G_y}{G_x} \right)$

□梯度向量指向 $f$ 变化率最大方向。



模板表示

- 为了检测边缘点，选取适当的阈值 $T$ ，对梯度图像进行二值化，则有：

$$g(x, y) = \begin{cases} 1 & \text{Grad}(x, y) \geq T \\ 0 & \text{其它} \end{cases}$$

- 这样形成了一幅边缘二值图像 $g(x, y)$
- **特点：**仅计算相邻像素的灰度差，对噪声比较敏感，无法抑制噪声的影响。

## ■ 几种常用的边缘检测算子

- Sobel算子

- Prewitt算子

- Roberts算子

- Laplacian算子

- LoG算子

## ■ Sobel梯度算子——3x3的梯度模板

□ 权值2用于通过增加中心点的重要性而实现某种程度的平滑效果

□ 公式

$$\nabla f \approx |G_x| + |G_y| = \left| (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3) \right| + \left| (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7) \right|$$

□ 模板  $\nabla f > T$  (T是指定阈值)，则在该位置的像素是边缘像素

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

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

$z_1$	$z_2$	$z_3$
$z_4$	$z_5$	$z_6$
$z_7$	$z_8$	$z_9$

□ 特点

□ 对4邻域采用**带权方法**计算差分，每行/每列的中心像素用2来加权。

□ 能进一步抑止噪声，但检测的边缘较宽。

## ■ Prewitt梯度算子——3x3的梯度模板

### □ 公式

$$\nabla f \approx |G_x| + |G_y| = \left| (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3) \right| +$$

### □ 模板:

$$\left| (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7) \right|$$

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

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

$z_1$	$z_2$	$z_3$
$z_4$	$z_5$	$z_6$
$z_7$	$z_8$	$z_9$

### □ 特点:

□ 在检测边缘的同时，能抑止噪声的影响

## ■ Roberts交叉梯度算子

$[g, t] = \text{edge}(f, \text{'roberts'}, T, \text{dir})$

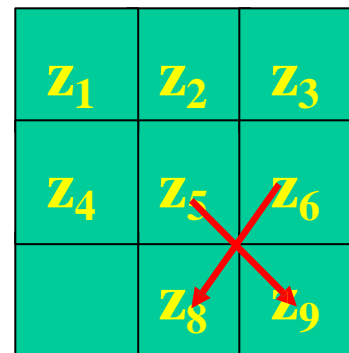
- ▣ 梯度计算由两个模板组成，第一个求得梯度的第一项，第二个求得梯度的第二项，然后求和，得到梯度。

▣ 公式:  $\nabla f \approx |G_x| + |G_y|$   
 $= |z_9 - z_5| + |z_8 - z_6|$

- ▣ 模板:

-1	0
0	1

0	-1
1	0



- ▣ 特点:

- ▣ 与梯度算子检测边缘的方法类似，对噪声敏感，但效果较梯度算子略好

## ■ 结论

- Prewitt和Sobel算子是计算数字梯度时最常用的算子
- Prewitt模板比Sobel模板简单, 但Sobel模板能够有效抑制噪声

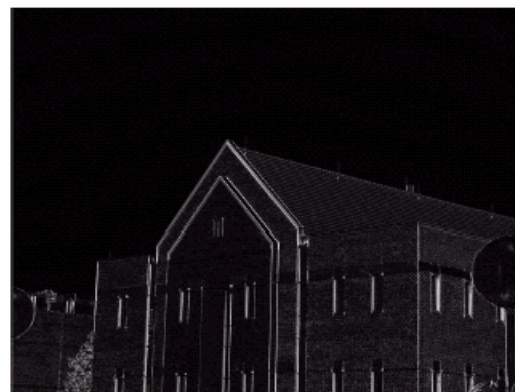
问题：砖墙等图像细节在边缘检测中通常是不符合需要的，因为它往往表现为噪声，使主要边缘的检测变得复杂处理

办法：对图像进行平滑处理

a b  
c d

**FIGURE 10.10**  
(a) Original image. (b)  $|G_x|$ , component of the gradient in the  $x$ -direction. (c)  $|G_y|$ , component of the gradient in the  $y$ -direction. (d) Gradient image,  $|G_x| + |G_y|$ .

原图像



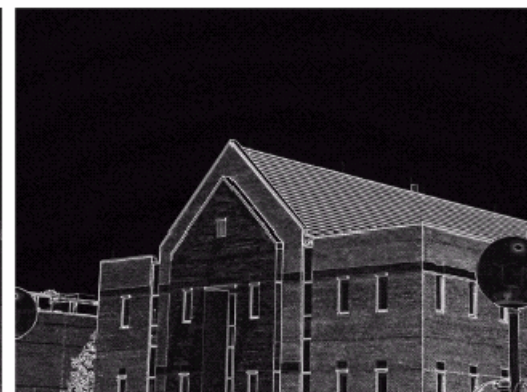
$|G_y|$ ,  $y$ 方向上的梯度分量, 和垂直细节非常清楚

$|G_x|$ ,  $x$ 方向上的梯度分量

, 水平细节非常清楚



梯度图像  $|G_x| + |G_y|$ , 水平和垂直细节都非常清楚

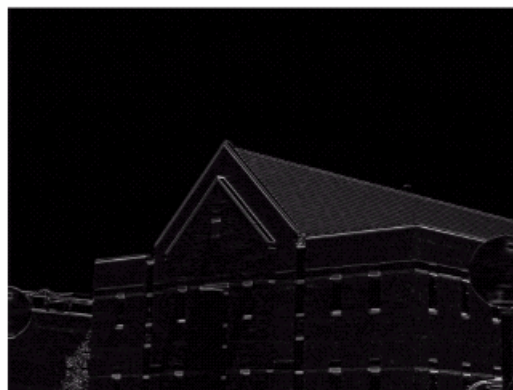




原图像经过 $5 \times 5$ 的均值滤波器进行平滑处理



$|G_x|$ , x方向上的梯度分量  
, 突出水平细节

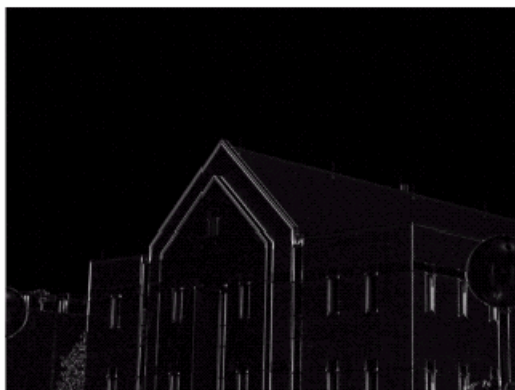


a b  
c d

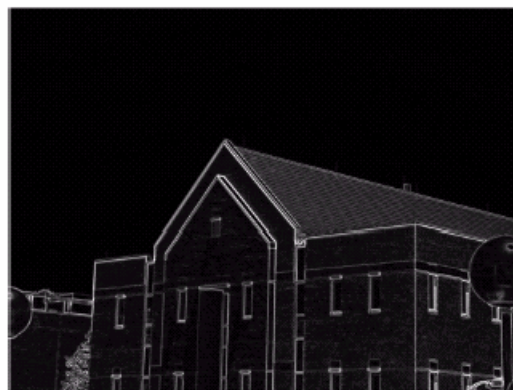
**FIGURE 10.11**

Same sequence as in Fig. 10.10, but with the original image smoothed with a  $5 \times 5$  averaging filter.

$|G_y|$ , y方向上的梯度分量  
, 突出垂直细节



梯度图像 $|G_x| + |G_y|$ ,  
突出水平和垂直细节



- 高斯型函数的目的是对图像进行平滑处理，拉普拉斯算子的目的是**提供一幅用零交叉确定边缘位置的图像**

## □定义

□二维函数 $f(x,y)$ 的拉普拉斯是一个二阶的微分定义为：

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

## □离散形式：

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

0	1	0
1	-4	1
0	1	0

- 模板：作用于中心像素的系数是一个负数，而且其周围像素的系数是正数，系数之和为0。

## ■拉普拉斯算子的特点

### □优点:

- 可以利用零交叉的性质进行边缘定位
- 可以确定一个像素是在边缘暗的一边还是亮的一边
- 对细线和孤立点检测效果较好。

### □缺点:

- 拉普拉斯算子对噪声具有敏感性
- 拉普拉斯算子的幅值产生双边缘
- 拉普拉斯算子不能检测边缘的方向

由于梯度算子和Laplace算子都对噪声敏感，因此一般在用它们检测边缘前要先对图像进行平滑。

- 在原始图像上进行边缘检测，由于噪声的影响，可能把噪声当 边缘点检测出来了，而真正的边缘又没被检测出来--噪声点对 边缘检测有较大的影响。
- 马尔算子(Marr-Hildreth)思想
  - 在拉普拉斯算子的基础上实现。由于拉普拉斯算子对噪声比较敏感，为了减少噪声影响，可先对待检测图进行平滑然后再用拉普拉斯算子检测边缘。
- 特点：把高斯平滑滤波器和拉普拉斯锐化滤波器结合起来，先平滑掉噪声，再进行边缘检测，所以效果更好。

$[g, t] = \text{edge}(f, 'log', T, \text{sigma})$

## ■ 步骤:

Step1: 平滑滤波器采用高斯滤波器;

Step2: 边缘增强用二阶导数 (二维拉普拉斯函数) ;

Step3: 边缘检测判据是二阶导数零交叉点;

Step4: 采用线性插值的方法估计边缘的位置。

因为采用Laplacian算子, 故又称LoG (Laplacian of Gaussian) 滤波器。

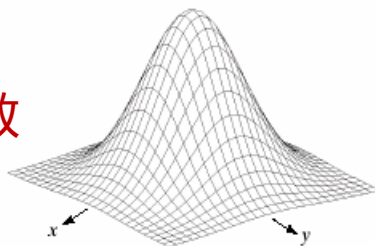
# 通过零交叉检测边缘

原图



Sobel算子检测结果

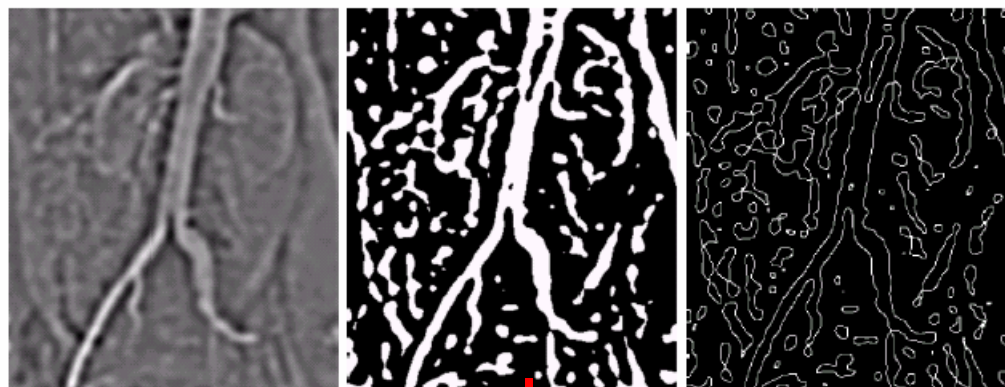
空间高斯型平滑函数



-1	-1	-1
-1	8	-1
-1	-1	-1

拉普拉斯算子模板

LoG检测结果



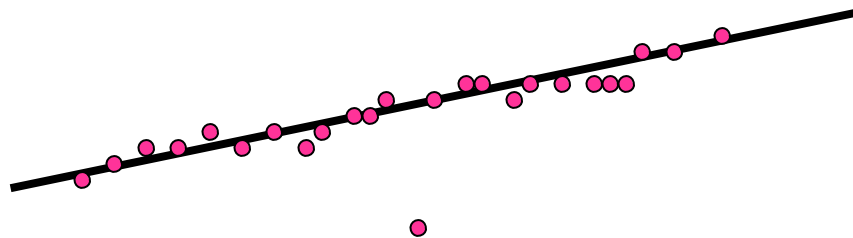
检测边缘：寻找零交叉点，检测黑色和白色区域之间的过渡点

对LoG图像设置阈值的结果，所有正值区域为白色，所有负值区域为黑色

- 图像分割的概念
- 点、线、边缘检测
- 霍夫变换
- 阈值处理
- 基于区域的分割
- 使用分水岭变换的分割

## ■ Hough变换问题的提出

- 边缘检测得到的边缘像素，由于存在噪声、不均匀照明引起的边缘断裂，以及引入杂散灰度不连续等，导致不能完全表征边缘。
- 在执行边缘检测算法后，通常需要使用连接过程来把边缘像素组装成有意义的边缘图形描述。
- **霍夫变换**可以用于将边缘像素连接起来得到边界曲线，它的主要优点在于受噪声和曲线间断的影响较小。
- 例如：给定一幅图像中的 $n$ 个点，假设我们希望找到这些点中一个位于直线上的子集。

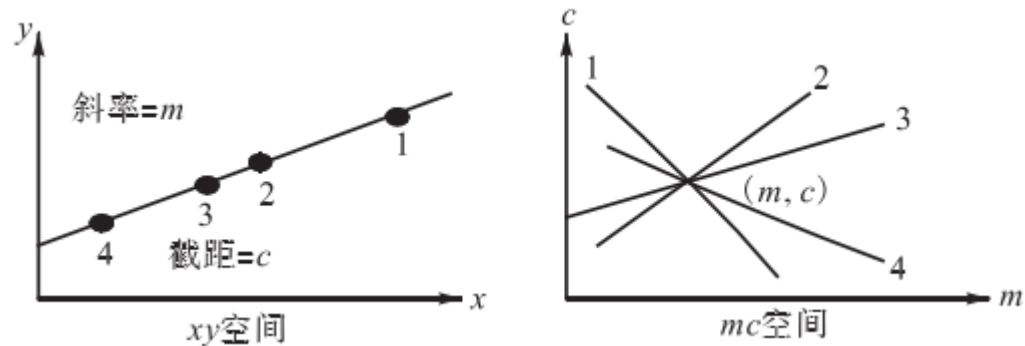




# 通过Hough变换进行整体处理

## ■ Hough变换的基本思想

- 霍夫变换本质上是**坐标变换**。
- 左半部分表示直线的x-y空间，直线方程为 $y=mx+c$ ，其中斜率为 $m$ ，截距为 $c$ 。
- 右半部分表示将直线从x-y空间变换到m-c空间，取直线在x-y空间上的四点1、2、3、4，在m-c空间就是不同斜率和截距的四条直线。
- 在m-c空间中四条直线的交点处的 $m$ 值和 $c$ 值就对应x-y空间中直线。
- 过x-y平面一个点 $(x,y)$ 的所有直线，构成参数m-c平面上的一条直线。
- 同理，对图片中的直线来说，将x-y空间的直线上的每个点都变换到m-c空间，找出在参数m-c平面上相交直线最多的点，对应的xy平面上的直线就是我们的解。



## ■基本思想

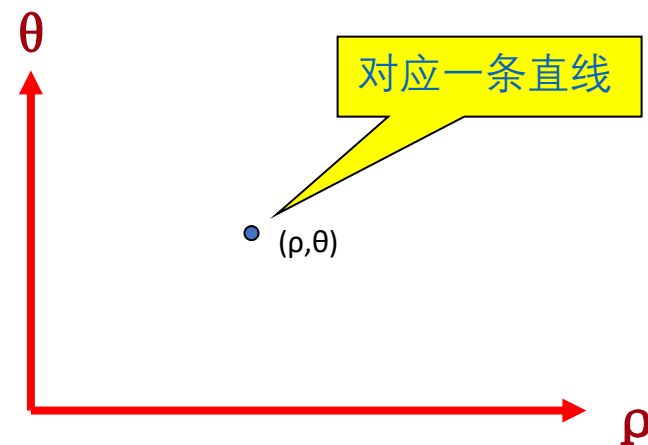
- 在实际应用中是将xy空间变换到极坐标系。
- 对于直角坐标系中的一条直线l，可用 $\rho$ 、 $\theta$ 来表示该直线，且直线方程为：

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

其中， $\rho$ 为原点到该直线的垂直距离， $\theta$ 为垂线与x轴的夹角，这条直线是唯一的。

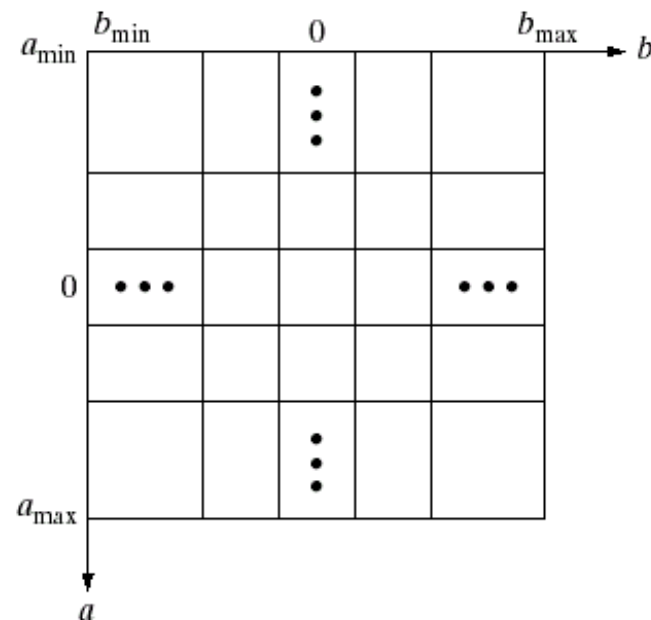
- 构造一个参数 $(\rho, \theta)$ 的平面，从而有如下结论：

直角坐标系中的一条直线对应极坐标系中的一点，这种线到点的变换就是**Hough变换**。



- 算法步骤:

- 1. 在 $\rho$ 、 $\theta$ 的极值范围内对其分别进行 $m$ 、 $n$ 等分, 设一个二维数组的下标与 $\rho_i$ 、 $\theta_j$ 的取值对应;
- 2. 对图像上的所有边缘点作Hough变换, 求每个点在 $\theta_j$  ( $j = 0, 1, \dots, n$ ) Hough变换后的 $\rho_i$ , 判断 $(\rho_i, \theta_j)$ 与哪个数组元素对应, 则让该数组元素值加1;
- 3. 比较数组元素值的大小, 最大值所对应的 $(\rho_i, \theta_j)$ 就是这些共线点对应的直线方程的参数。



# 通过Hough变换进行整体处理

图像处理工具箱提供了 3 个与霍夫变换有关的函数。

1、 **[H,T,R] = hough(BW, 'Theta' ,20:0.1:75);**

(实现霍夫变换的概念。输入二值图像BW，角度范围与步进（最大，[-90, 90)），返回 H-霍夫空间，T-theta，R-p);

2、 **PEAKS = houghpeaks(H,NUMPEAKS);**

(寻找霍夫变换中的峰值（高计数累加器单元）。输入霍夫空间和极值数量，返回极值的坐标)

3、 **LINES=houghlines(BW,T,R,Peaks);**

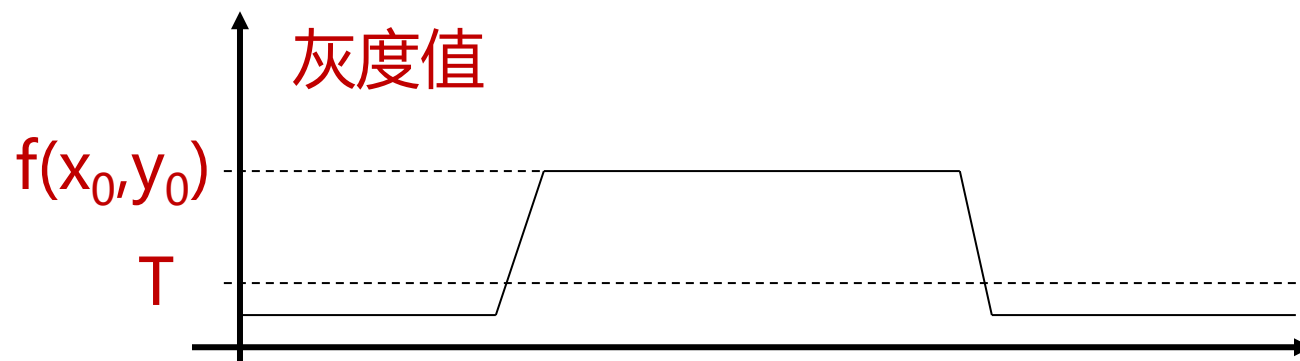
(基于前两个函数的结果，提取原始图像中的线段。返回lines是一个包含图像中线段首末点、p、theta的结构体)

- 图像分割的概念
- 点、线、边缘检测
- 霍夫变换
- 阈值处理
- 基于区域的分割
- 使用分水岭变换的分割

在前面的章节中，我们采用首先寻找边缘线段，然后将这些线段连接为边界的方法来识别区域。本节讨论基于灰度值或灰度值的特性来将图像**直接划分为区域**的技术。

## ■ 阈值分割法的基本思想

- 确定一个合适的阈值 $T$ （阈值的好坏是此方法成败的关键）。
- 将**大于等于阈值**的像素作为物体或背景，生成一个二值图像。
- 适用于物体与背景有较强对比的情况，重要的是背景或物体的灰度比较单一。可通过先求背景，然后求反得到物体)
- 这种方法总可以得到封闭且连通区域的边界。



# 图像阈值处理

## ■ 阈值处理操作

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

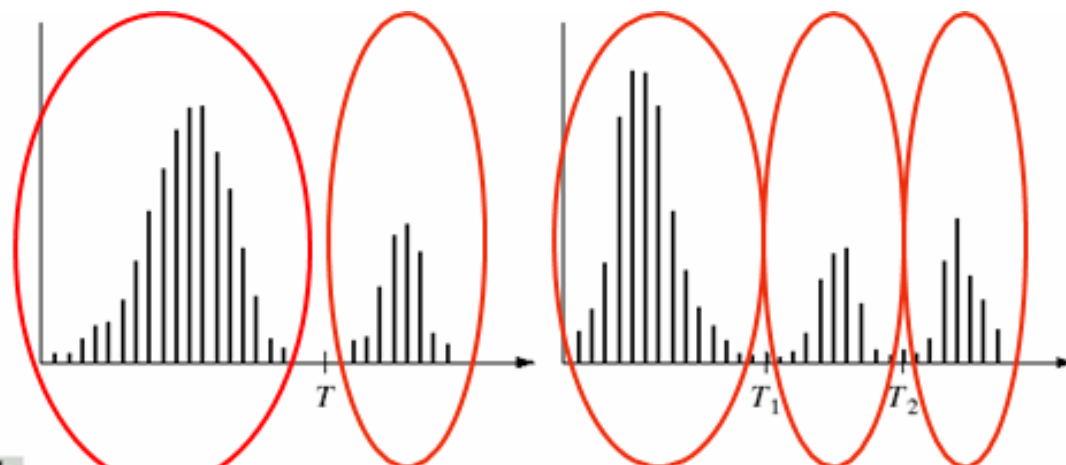
$f(x, y)$ 是点 $(x, y)$ 的灰度级,  $p(x, y)$ 表示该点的局部性质, 如以 $(x, y)$ 为中心的邻域的平均灰度级

## ■ 阈值处理后的图像 $g(x, y)$ 定义为

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

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**



a b

FIGURE 10.26 (a) Gray-level histograms that can be partitioned by (a) a single threshold, and (b) multiple thresholds.

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:  $f(x, y) > T_2$

## ■ 阈值分割方法

- 基本全局阈值
- Otsu方法进行全局阈值处理
- 图像平滑改进全局阈值处理
- 通过边缘改进选择阈值
- 基于局部统计的可变阈值处理
- 基于移动平均的阈值处理



# 基本全局阈值

## 基本全局阈值算法——自动选择阈值

1. 选择一个 $T$ 的初始估计值。
2. 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。
3. 对区域 $G_1$ 和 $G_2$ 中的所有像素计算平均灰度值 $m_1$ 和 $m_2$ 。
4. 计算新的阈值  $T = \frac{1}{2}(m_1 + m_2)$
5. 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参数 $\Delta T$

参数 $\Delta T$ 用于控制迭代的次数。通常， $\Delta T$ 越大，则算法执行的迭代次数少。  
图像的平均灰度对于 $T$ 来说是较好的初始选择。

# 利用基本全局阈值算法的例子

$g = \text{im2bw}(f, T/\text{den})$

其中, den 是整数(例如一幅8比特图像的255)。

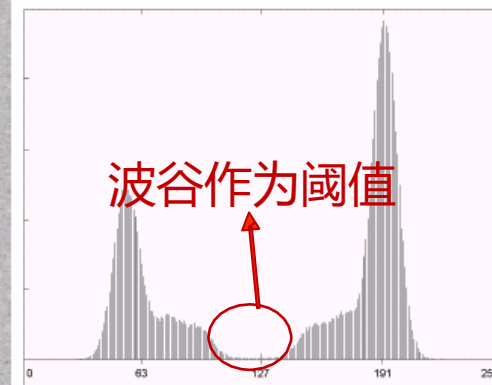
```
f = imread( 'xxx.tif');
count=0;
T=mean2(f);
done=false;
while ~done
    count=count+1;
    g=f>T;
    Tnext=0.5*(mean(f(g))+mean(f(~g)));
    done=abs(T-Tnext)<0.5;
    T=Tnext;
end
disp('count的值为: ');
disp(['count=',num2str(count)]) %打印输出count的值
disp('T的值为: ');
disp(['T=',num2str(T)]) %打印输出T的值
g=im2bw(f,T/255);

figure;subplot(1,3,1);imshow(f);title('(a)带噪声的指纹');
subplot(1,3,2);imhist(f);title('(b)直方图');
subplot(1,3,3);imshow(g);title('(c)用全局阈值分割的结果');
```

原图



原图的直方图



波谷作为阈值



基本全局阈值算法处理的结果  $\Delta T = 0.5$ , 2次迭代得到值为125.38  
最后确定  $T = 125$

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.)

# 用Otsu方法的最佳全局阈值处理

## ■Otsu方法进行全局最佳阈值处理

□取一个最优阈值把原图像分为前景色（A部分）与背景色（B部分），两部分的类间方差越大，说明两部分差别越大，便能有效的分割图像，所以该算法最关键的是找到最优阈值。

□ Otsu算法小节如下：

- 计算输入图像的归一化直方图。使用 $p_i, i = 0, 1, 2, \dots, L - 1$ 表示该直方图的各个分量
- 计算累积和： $P_1(k) = \sum_{i=0}^k p_i, (k = 0, 1, 2, \dots, L - 1), P_2(k) = 1 - P_1(k)$
- 计算累积均值： $m(k) = \sum_{i=0}^k ip_i$ ，对于 $k = 0, 1, 2, \dots, L - 1$ ，
- 计算全局灰度均值： $m_G = \sum_{i=0}^{L-1} ip_i$
- 用 $\sigma_B^2(k) = \frac{[m_G P_1(k) - m(k)]^2}{P_1(k)[1 - P_1(k)]}$ ，对于 $k = 0, 1, 2, \dots, L - 1$ ，计算类间方差 $\sigma_B^2(k)$
- 得到Otsu阈值 $k^*$ ，即使得 $\sigma_B^2(k)$ 最大的 $k$ 值。如果最大值不唯一，用相应检测到的各个最大值的 $k$ 的平均得到 $k^*$
- 在 $k = k^*$ 处计算 $\eta(k) = \frac{\sigma_B^2(k)}{\sigma_G^2}$ ，得到可分性测度 $\eta^*$

# 用Otsu方法的最佳全局阈值处理

## ■Otsu方法进行全局最佳阈值处理

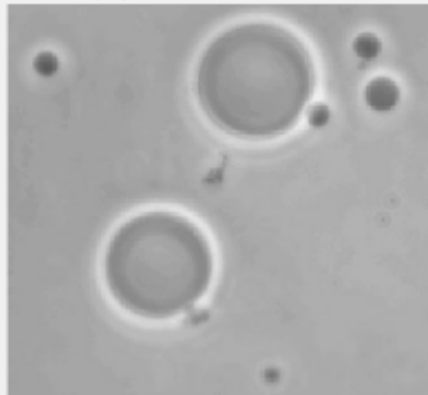
```
f = imread('XXX.tif');
count=0;
T=mean2(f);
done=false;
while ~done
    count=count+1;
    g=f>T;
    Tnext=0.5*(mean(f(g))+mean(f(~g)));
    done=abs(T-Tnext)<0.5;
    T=Tnext;
end
disp('count的值为: ');
disp(['count=',num2str(count)]) %打印输出count的值
disp('T的值为: ');
disp(['T=',num2str(T)]) %打印输出T的值
g=im2bw(f, T/255); %T=169.4

[T, SM] = graythresh(f);
g1 = im2bw(f, T); % T=181
figure;subplot(2,2,1);imshow(f);title('(a)带噪声的指纹');
subplot(2,2,2);imhist(f);title('(b)直方图');
subplot(2,2,3);imshow(g);title('(c)用全局阈值分割的结果');
subplot(2,2,4);imshow(g1);title('(d)使用 Otsu s算法得到的结果');
```

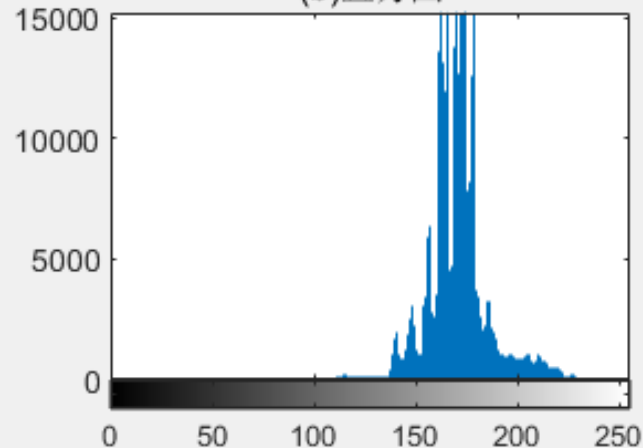
$[T, SM] = \text{graythresh}(f)$

T:阈值,被归一化; SM: 可分性测度

(a)带噪声的指纹



(b)直方图



(c)用全局阈值分割的结果



(d)使用 Otsu s算法得到的结果

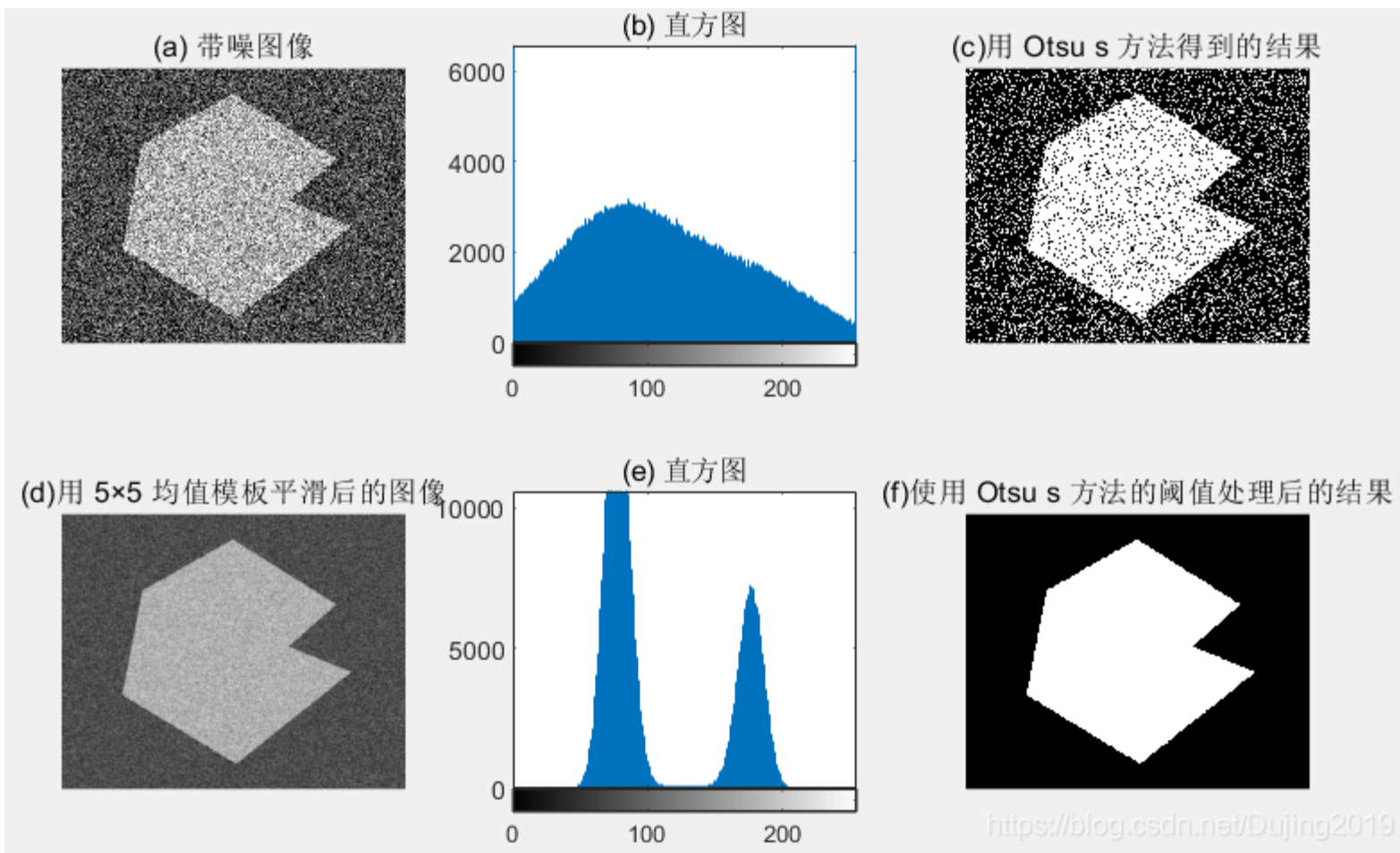


# 用图像平滑改善全局阈值处理

## ■用图像平滑改善全局阈值处理

- 噪声会将简单的阈值处理问题变为不可解决的问题
- 解决方法：在阈值处理之前平滑图像

```
f = imread('xxx.tif');  
fn = imnoise(f,'gaussian',0,0.038);  
subplot(2,3,1),imshow(fn);title('(a) 带噪图像');  
subplot(2,3,2),imhist(fn);title('(b) 直方图');  
Tn = graythresh(fn);  
gn = im2bw(fn,Tn);  
subplot(2,3,3),imshow(gn);title('(c)用 Otsu s 方法得到的结果');  
w = fspecial('average',5);  
fa = imfilter(fn,w,'replicate');  
subplot(2,3,4),imshow(fa);title('(d)用 5×5 均值模板平滑后的图像');  
subplot(2,3,5),imhist(fa);title('(e) 直方图');  
Ta = graythresh(fa);  
ga = im2bw(fa,Ta);  
subplot(2,3,6),imshow(ga);title('(f)使用 Otsu s 方法的阈值处理后的结果');
```





# 利用边缘改进全局阈值处理

## ■利用边缘改进全局阈值处理

### □基本思想：

- 1) 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。
- 2) 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。
- 3) 用**微分算子**处理图像，使图像只剩下边界中心两边的值。

### □方法优点：

- 1)在前景和背景所占区域面积差别很大时，不会造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。

# 利用边缘改进全局阈值处理

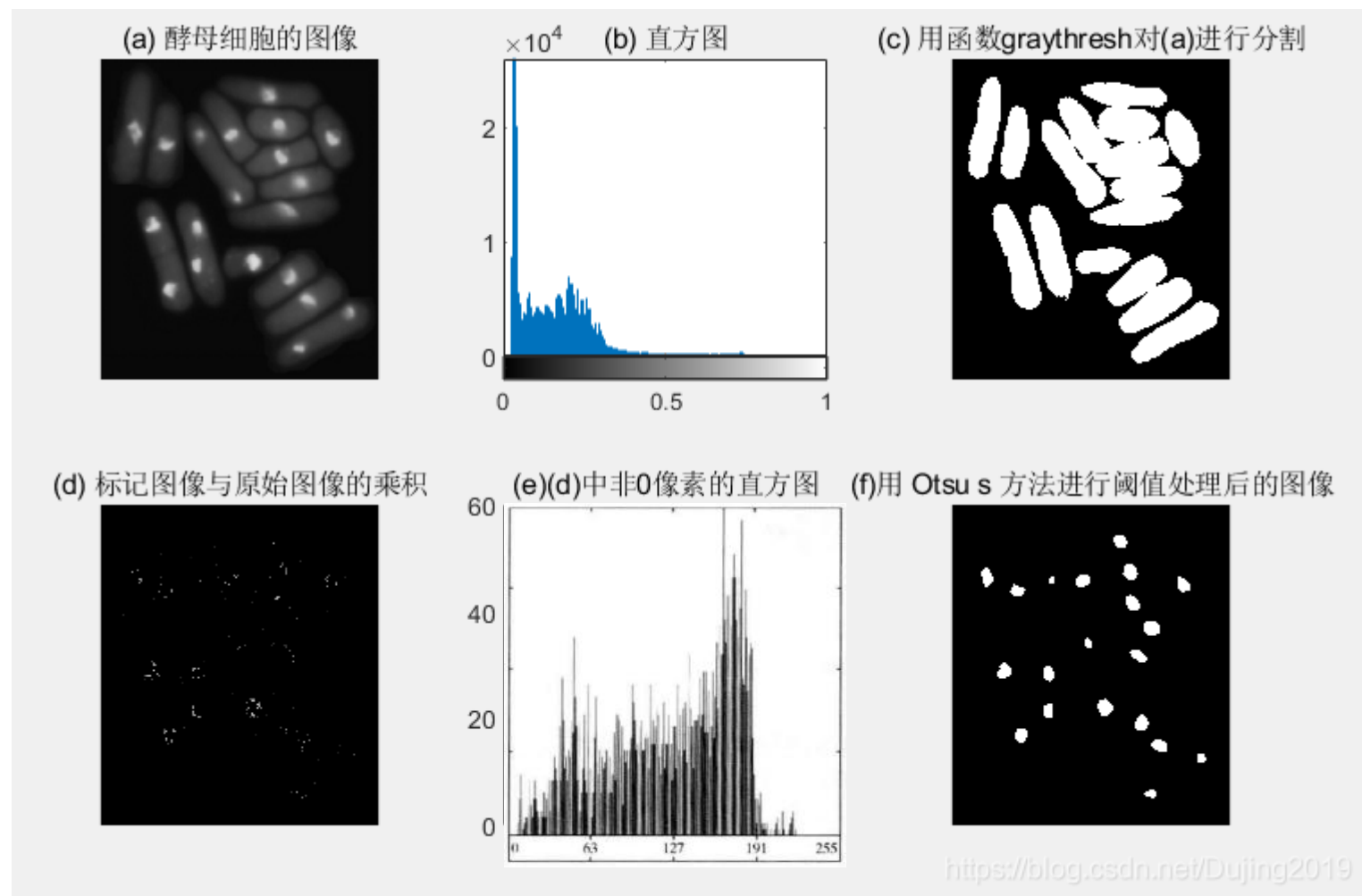
■该算法可小结如下，其中 $f(x,y)$ 是输入图像：

1. 计算一副边缘图像（梯度和拉普拉斯都可以）。
2. 指定一个阈值 $T$ 。
3. 用步骤2中的阈值对步骤1中产生的图像进行阈值处理，产生一副二值图像 $gT(x,y)$ 。  
在步骤4中，该图像作为一副模板图像。
4. 仅用 $f(x,y)$ 中对应于 $gT(x,y)$ 中像素为1的位置的像素（强边缘像素）计算直方图。
5. 用步骤4中的直方图全局地分割 $f(x,y)$ 。

# 利用边缘改进全局阈值处理

## 用以梯度为基础的边缘信息改进全局阈值处理

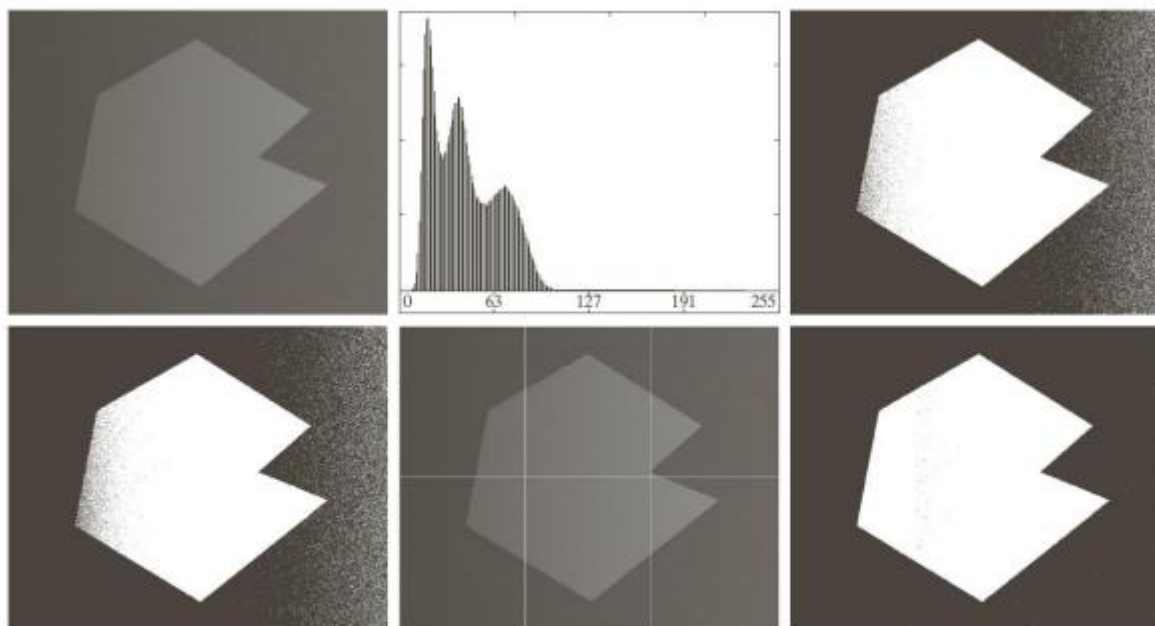
```
f = tofloat(imread('xxx.tif'));
subplot(2,3,1),imshow(f);title('(a) 酵母细胞的图像');
subplot(2,3,2),imhist(f);title('(b) 直方图');
hf = imhist(f);
[Tf SMf] = graythresh(f);
gf = im2bw(f, Tf);
subplot(2,3,3),imshow(gf);title('(c) 用函数graythresh对(a)进行分割');
w = [-1 -1 -1; -1 8 -1; -1 -1 -1]; %拉普拉斯算子
lap = abs(imfilter(f, w, 'replicate'));
lap = lap/max(lap(:));
h = imhist(lap);
Q = percentile2i(h, 0.995);
markerImage = lap > Q;
fp = f.*markerImage;
subplot(2,3,4),imshow(fp);title('(d) 标记图像与原始图像的乘积');
hp = imhist(fp);
hp(1) = 0;
subplot(2,3,5),bar(hp,0);title('(e)(d)中非0像素的直方图');
T = otsuthresh(hp);
g = im2bw(f, T);
subplot(2,3,6),imshow(g);title('(f)用 Otsu s 方法进行阈值处理后的图像');
后的图像);
```





## ■ 基于局部统计的可变阈值处理

- 可变阈值处理最简单的方法之一是，把一幅图像分成**不重叠的矩形**。
- 这种方法用于补偿光照和/或反射的不均匀性。选择的矩形要足够小，以便每个矩形的光照都近似是均匀的



a b c  
d e f

**FIGURE 10.46** (a) Noisy, shaded image and (b) its histogram. (c) Segmentation of (a) using the iterative global algorithm from Section 10.3.2. (d) Result obtained using Otsu's method. (e) Image subdivided into six subimages. (f) Result of applying Otsu's method to each subimage individually.

# 基于局部统计的可变阈值处理

## ■ 基于局部统计的可变阈值处理

- 更为一般的方法是在一幅图像中的每点 $(x, y)$ 计算阈值，该阈值以一个或多个在 $(x, y)$ 邻域计算的特性为基础
- 令 $\sigma_{xy}$ 和 $m_{xy}$ 表示一幅图像中以坐标 $(x, y)$ 为中心的邻域 $S_{xy}$ 所包含的像素集合的标准差和均值。下面是可变局部阈值的通用形式：

$$T_{xy} = a\sigma_{xy} + bm_{xy}$$

式中， $a$ 和 $b$ 是非负常数，且 $T_{xy} = a\sigma_{xy} + bm_{xy}$ 。分割后的图像计算如下：

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

该式对图像中的所有像素位置进行求值，并在每个点 $(x, y)$ 处使用邻域 $S_{xy}$ 中的像素计算不同的阈值

- 使用以 $(x, y)$ 处的邻域计算出的参数为基础的属性，有效的权值可显著增强局部阈值处理：

$$g(x, y) = \begin{cases} 1, & Q(\text{局部参数}) \text{为真} \\ 0, & Q(\text{局部参数}) \text{为假} \end{cases}$$

式中， $Q$ 是以邻域 $S_{xy}$ 中像素计算的参数为基础的一个属性。

# 基于局部统计的可变阈值处理

## ■ 基于局部统计的可变阈值处理

```
f = tofloat(imread('xxx.tif'));  
subplot(2,2,1),imshow(f);title('(a) 酵母细胞的图像');  
[TGlobal] = graythresh(f);  
gGlobal = im2bw(f, TGlobal);  
subplot(2,2,2),imshow(gGlobal);title('(b)用 Otsus 方法分割的图像');  
g = localthresh(f, ones(3), 30, 1.5, 'global');  
SIG = stdfilt(f, ones(3)); %局部标准差  
subplot(2,2,3), imshow(SIG, [ ]) ;title('(c) 局部标准差图像');  
subplot(2,2,4),imshow(g);title('(d) 用局部阈值处理分割的图像');
```

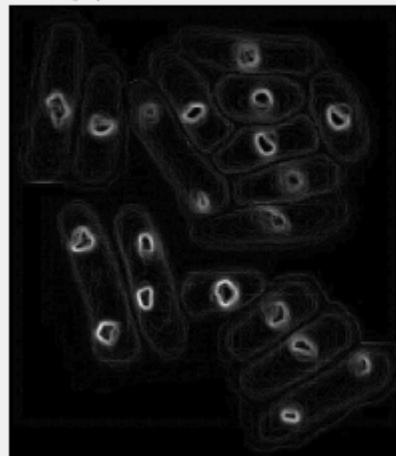
(a) 酵母细胞的图像



(b)用 Otsus 方法分割的图像



(c) 局部标准差图像



(d) 用局部阈值处理分割的图像



- 图像分割的概念
- 点、线、边缘检测
- 霍夫变换
- 阈值处理
- 基于区域的分割
- 使用分水岭变换的分割

# 基于区域的分割

- 分割的目的是将一幅图像划分为多个区域。
- 基于灰度级的不连续性尝试寻找区域间的边界
- 以像素特性分布为基础的阈值处理来完成

本节讨论以**直接寻找区域为基础**的分割技术。

- 区域生长
- 区域分裂与聚合

## 基本表达式

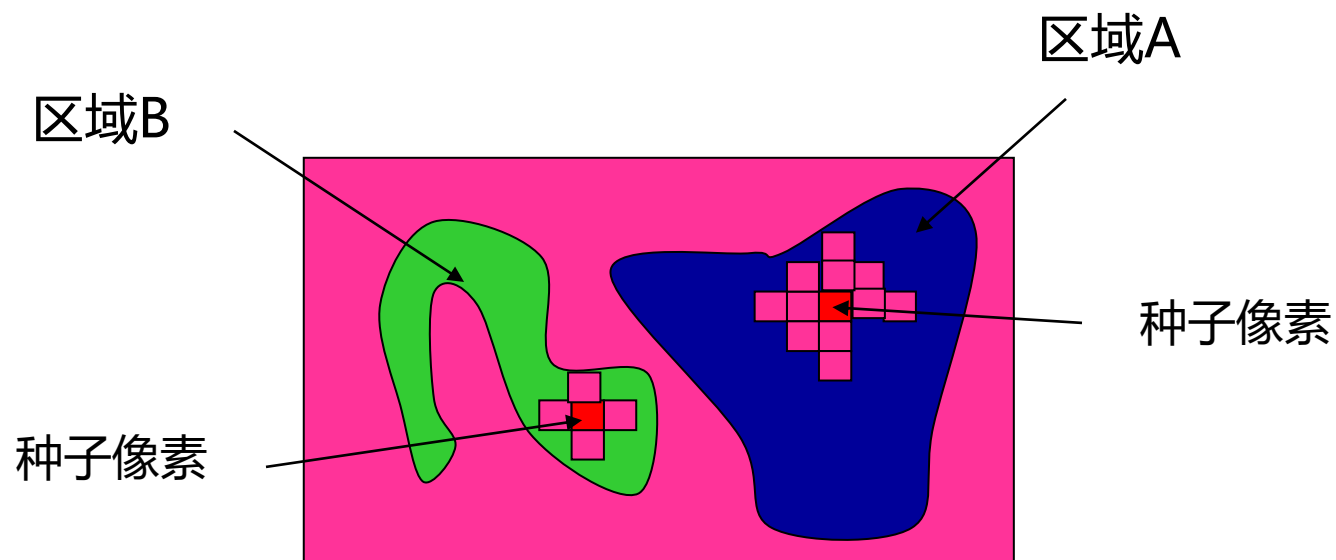
令集合 $R$ 代表整个图像区域，对 $R$ 的分割可看作将 $R$ 分成 $N$ 个满足以下五个条件的非空子集（子区域） $R_1, R_2, \dots, R_N$

- ①**完备性**:  $R_1 + R_2 + \dots + R_N = R$ ;
- ②**独立性**: 对所有的 $i$ 和 $j$ ,  $i \neq j$ , 有 $R_i \cap R_j = \varnothing$ ;
- ③**单一性**: 每个区域内的灰度级相等。对 $i = 1, 2, \dots, N$ , 有 $P(R_i) = \text{TRUE}$ ;
- ④**互斥性**: 任两个区域的灰度级不等, 对 $i \neq j$ , 有 $P(R_i \cup R_j) = \text{FALSE}$ ;
- ⑤**连通性**: 对 $i = 1, 2, \dots, N$ ,  $R_i$ 是连通的区域。

其中 $P(R_i)$ 是对所有在集合 $R_i$ 中元素的逻辑谓词, $\varnothing$ 代表空集。

条件①指出在对一幅图像的分割应将图像中的每个像素都分进某个子区域中；条件②指出在分割结果中各个子区域是互不重叠的；条件③指出在分割结果中每个子区域都有独特的特性；条件④指出在分割结果中，各个子区域具有不同的特性，没有共同元素；条件⑤指出分割结果中同一个子区域内的像素应该是连通的。

- **区域生长**：根据预先定义的生长准则，将像素或子区域组合为**更大区域**的过程。
- **基本方法**：从一组“**种子**”点开始，将与种子预先定义的**性质相似**的那些领域像素添加到每个种子上，来形成这些**生长区域**。



## ■ 区域生长的算法实现：

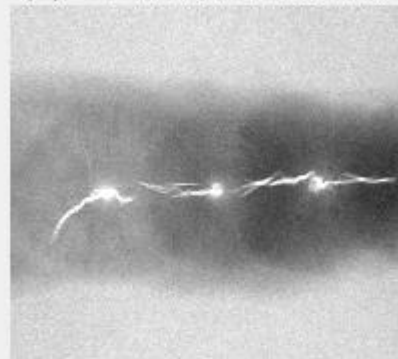
- 1) 根据图像的不同应用选择**一个或一组种子**，它或者是最亮或最暗的点，或者是位于点簇中心的点。
- 2) 选择一个描述符（条件）。
- 3) 从该种子开始向外扩张，首先把种子像素加入结果集合，**然后不断将与集合中各个像素连通、且满足描述符的像素加入集合。**
- 4) 上一过程进行到不再有满足条件的新结点加入集合为止。



# 区域生长

```
f = imread( 'xxx.tif');  
subplot(2,2,1),imshow(f);  
title('(a)显示有焊接缺陷的图像');  
%函数regiongrow返回的NR为是不同区域的数目，参数SI是  
一副含有种子点的图像  
%TI是包含在经过连通前通过阈值测试的像素  
[g,NR,SI,TI]=regiongrow(f,1,0.26);%种子的像素值为255， 65  
为阈值  
subplot(2,2,2),imshow(SI);  
title('(b)种子点');  
subplot(2,2,3),imshow(TI);  
title('(c)通过了阈值测试的像素的二值图像(白色)');  
subplot(2,2,4),imshow(g);  
title('(d)对种子点进行8连通分析后的结果');
```

(a)显示有焊接缺陷的图像



(b)种子点



(c)通过了阈值测试的像素的二值图像(白色)

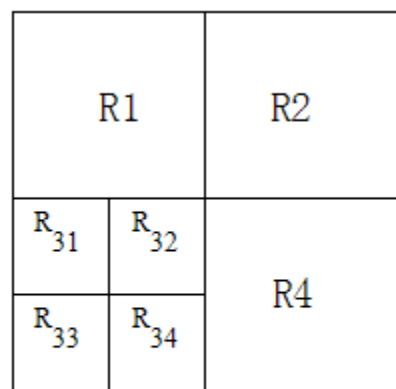


(d)对种子点进行8连通分析后的结果

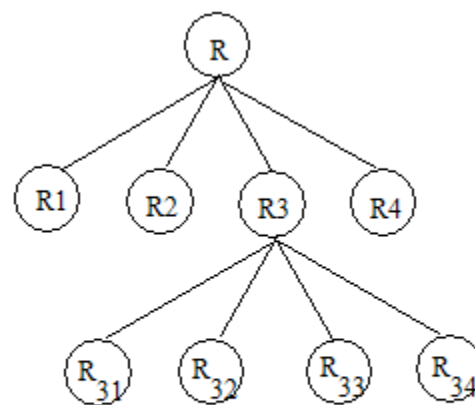


<https://blog.csdn.net/Duijing2019>

- **区域分裂合并法**无需预先指定种子点，它按某种**一致性准则**分裂或者合并区域。
- 可以先进行**分裂运算**，然后再进行**合并运算**；也可以**分裂和合并运算同时进行**，经过连续的分裂和合并，最后得到图像的精确分割效果。
- 分裂合并法对分割复杂的场景图像比较有效。
- 具体实现时，**分裂合并算法**是基于四叉树数据表示方式进行的。



(a) 分裂图像



(b) 相应的四叉树结构

## (1) 初始分割

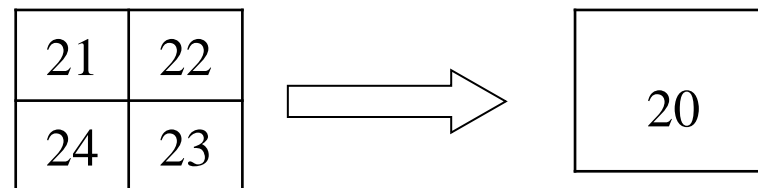
把一幅图像分裂到第二层，子块数 $n = 16$ ，子块标号如下图。

## (2) 合并处理

按预先给定的合并原则，对第二层的每4个子块分别进行检查。假定子块21、22、23、24符合合并原则，合并后标记为20。

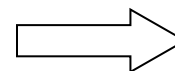
11	12	21	22
14	13	24	23
41	42	31	32
44	43	34	33

(a)  $n=16$



## (3) 分裂处理

当第二层中的某一子块内像素不满足特性均匀性条件时，将它们分裂成4个子块。如对13子块分裂。



131	132
134	133

经对图像分裂与合并处理，共分成28块。

11	12	21	22
14	13	24	23
41	42	31	32
44	43	34	33

(a)  $n=16$



110		120		20			
140		131	132				
		134	133				
411	412	421	422	311	312	320	
414	413	424	423	314	313		
440		431	432	340		330	
		434	433				

(b)  $n=28$

## (4) 组合处理

以每块为中心，检查与其相邻各块，凡符合特征均匀性的，再次合并。

110		120		20			
140		131	132				
		134	133				
411	412	421	422	311	312	320	
414	413	424	423	314	313		
440		431	432	340		330	
		434	433				

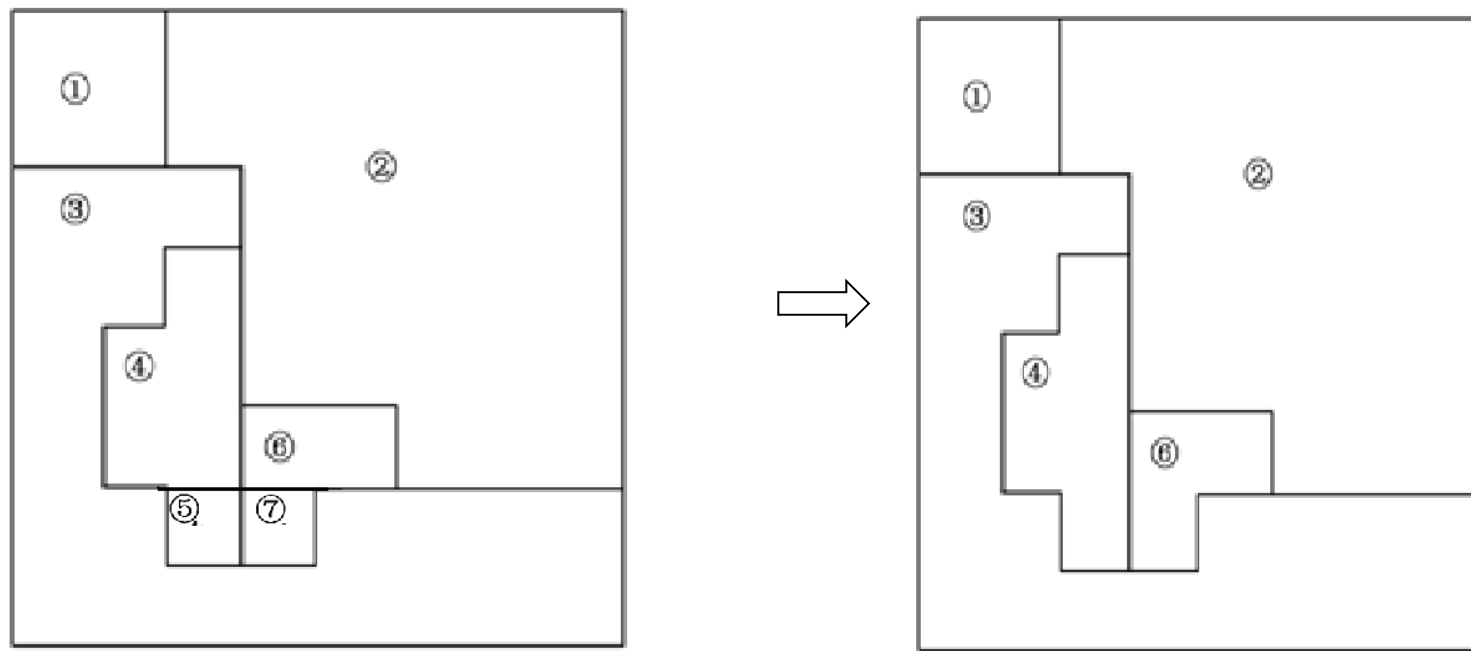
(b)  $n=28$



① 110	120		②				
③ 140		131	132	20			
		134	133				
411	412	421	422	311	312	320	
414	413	424	④ 423	314	313		
440		⑤ 431	⑦ 432	340		330	
		434	433				

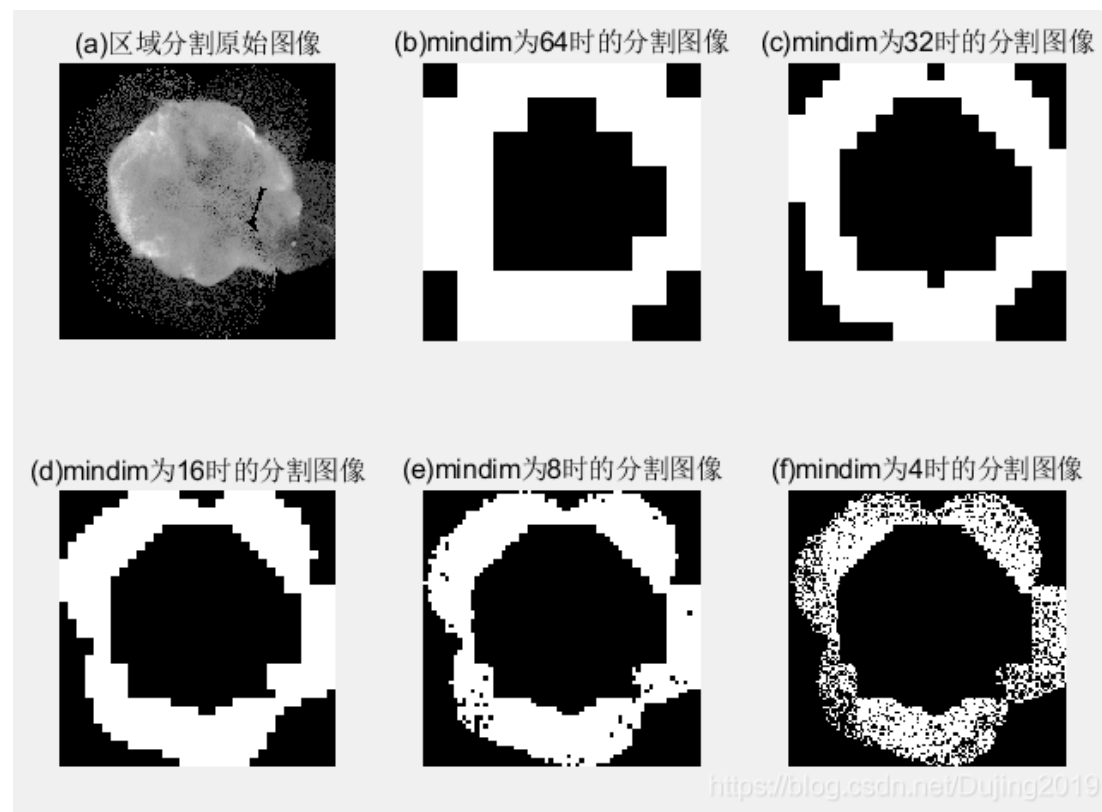
## (5) 消失小区

标号为⑤和⑦的两个小区，与相邻大块比较，按它们对邻近大块的均匀性程度分别划到区号为④和⑥中去，经这样处理后的输出结果为五个区域。



# 区域分离聚合法

```
f = imread('xxx.tif');  
subplot(2,3,1),imshow(f);  
title('(a)区域分割原始图像');  
g64=splitmerge(f,64,@predicate);%64代表分割中允许最小的块  
subplot(2,3,2),imshow(g64);  
title('(b)mindim为64时的分割图像');  
g32=splitmerge(f,32,@predicate);%32代表分割中允许最小的块  
subplot(2,3,3),imshow(g32);  
title('(c)mindim为32时的分割图像');  
g16=splitmerge(f,16,@predicate);%16代表分割中允许最小的块  
subplot(2,3,4),imshow(g16);  
title('(d)mindim为16时的分割图像');  
g8=splitmerge(f,8,@predicate);%8代表分割中允许最小的块  
subplot(2,3,5),imshow(g8);  
title('(e)mindim为8时的分割图像');  
g4=splitmerge(f,4,@predicate);%4代表分割中允许最小的块  
subplot(2,3,6),imshow(g4);  
title('(f)mindim为4时的分割图像');
```



- 图像分割的概念
- 点、线、边缘检测
- 霍夫变换
- 阈值处理
- 基于区域的分割
- 使用分水岭变换的分割



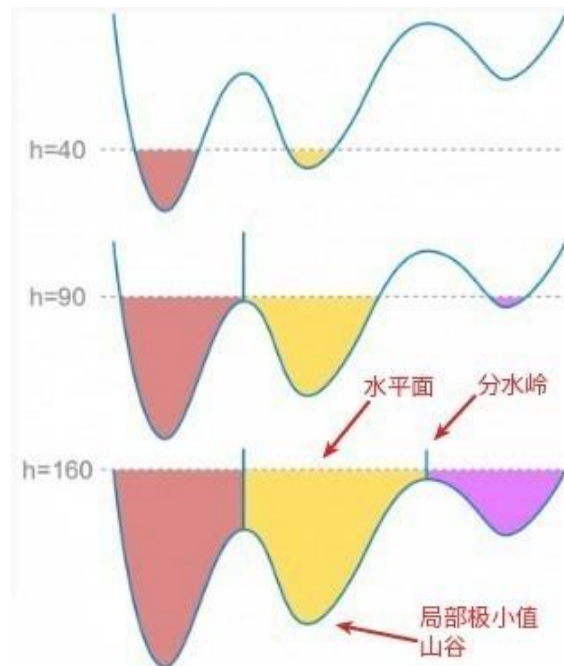
# 基于分水岭变换的分割

■ **分水岭**也称分水线/水线，将图像看成三维地形表示。



## ■ 基本思想:

- 图像的灰度空间很像地球表面的整个地理结构，**每个像素的灰度值代表高度**。其中的灰度值较大的像素连成的线可以看做**山脊**，也就是**分水岭**。二值化阈值可以理解为水平面，比水平面低的区域会被淹没，刚开始用水填充每个孤立的山谷(局部最小值)。
- 当水平面上升到一定高度时，水就会溢出当前山谷，可以通过在**分水岭上修大坝**，从而避免两个山谷的水汇集，这样图像就被分成**2个像素集**，一个是被水淹没的山谷像素集，一个是分水岭线像素集。最终这些大坝形成的线就对整个图像进行了分区，实现对图像的分割。



## ■ 分水岭算法过程：

1. 把梯度图像中的所有像素按照灰度值进行分类，并设定一个**测地距离阈值**。
2. 找到灰度值最小的像素点（默认标记为**灰度值最低点**），让阈值从最小值开始增长。
3. 水平面在增长的过程中，会碰到周围的邻域像素，测量这些像素到起始点（灰度值最低点）的**测地距离**，如果小于设定阈值，则将这些像素**淹没**，否则在这些像素上**设置大坝**，这样就对这些邻域像素进行了分类。
4. 随着水平面越来越高，会设置更多更高的大坝，直到灰度值的最大值，所有区域都在分水岭线上相遇，这些大坝就对整个图像像素的进行了分区。



- 图像分割的概念
- 点、线、边缘检测
- 霍夫变换
- 阈值处理
- 基于区域的分割
- 使用分水岭变换的分割