基于边缘检测的分割

许 向 阳
xuxy@hust.edu.cn

什么是边缘？

An **edge** is said to occur at a point in the image if some image attribute changes in value discontinuously at that point.(here attribute is intensity in our discussion)

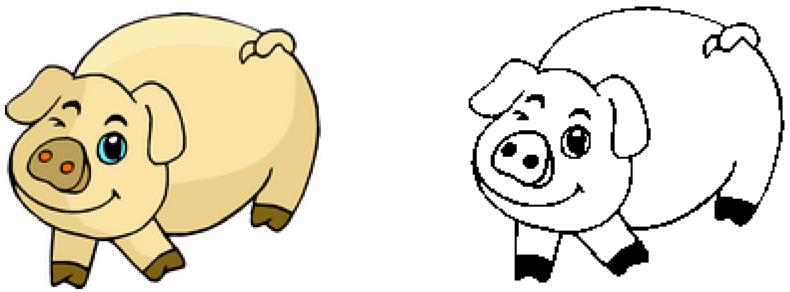
Edge detection techniques aim to find local discontinuities in some image attribute, such as intensity or color.

为什么检测边缘？

华中科技大学

Detection of object boundaries is an important part of the perception process.

Ex: for human beings to understand cartoons.



不影响理解图像，可有效压缩

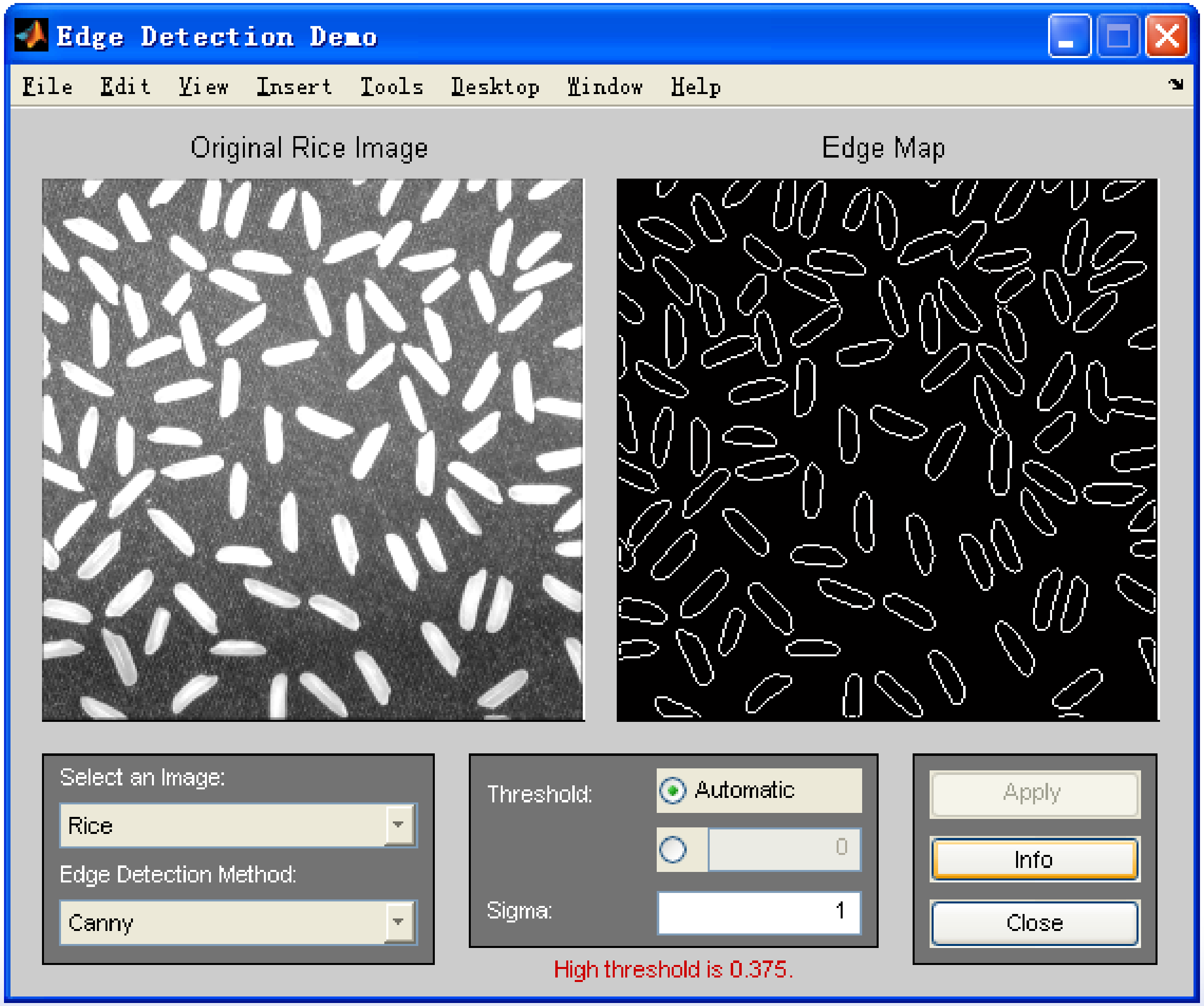
如何检测边缘?

华中科技大学

理想的边缘具有怎样的特性?

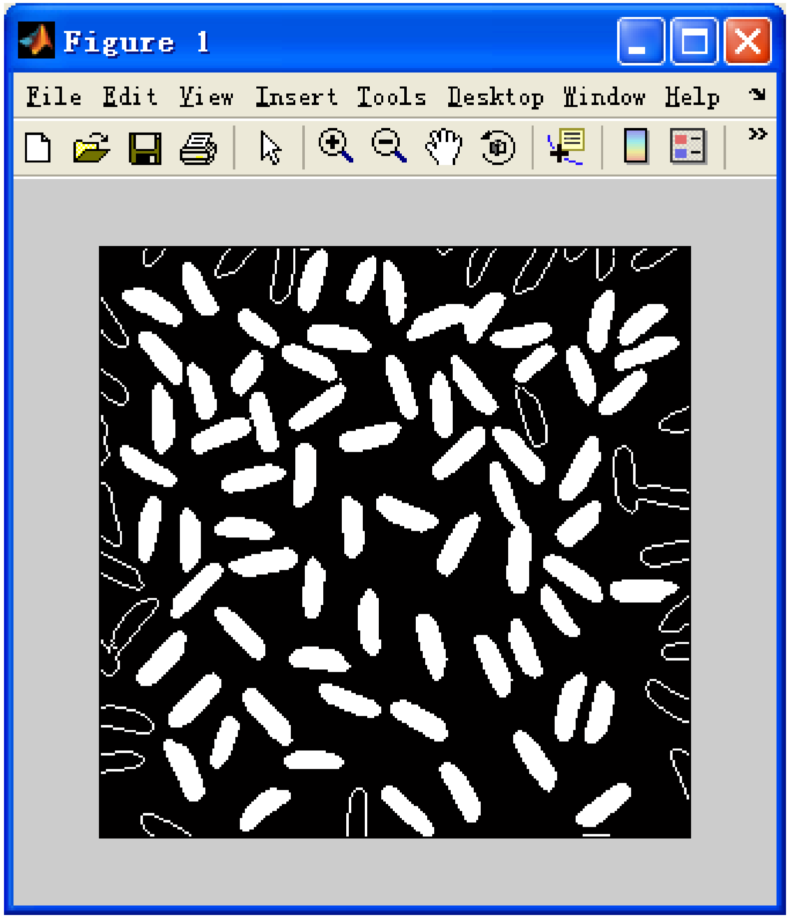
如何由边缘构成区域?

边缘检测示例1



Edge Detection Demo
File Edit View Insert Tools Desktop Window Help
Original Rice Image
Edge Map
Select an Image:
Rice
Edge Detection Method:
Canny
Threshold:
Automatic
0
Sigma:
1
Apply
Info
Close
High threshold is 0.375.

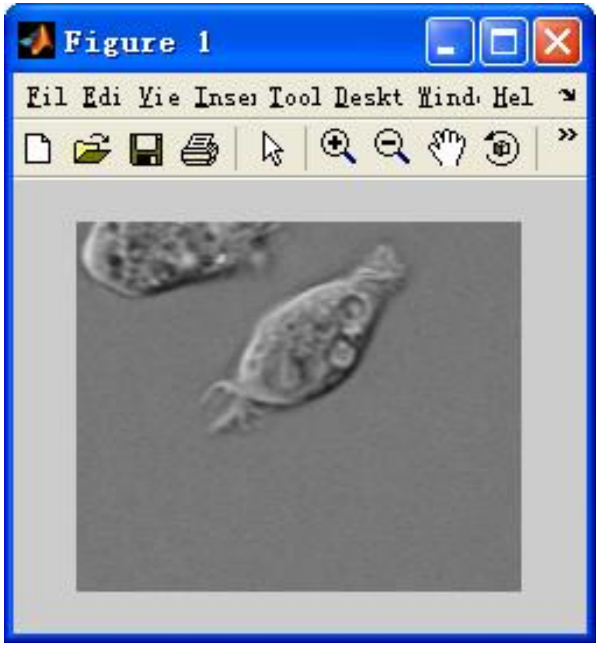
边缘检测示例1

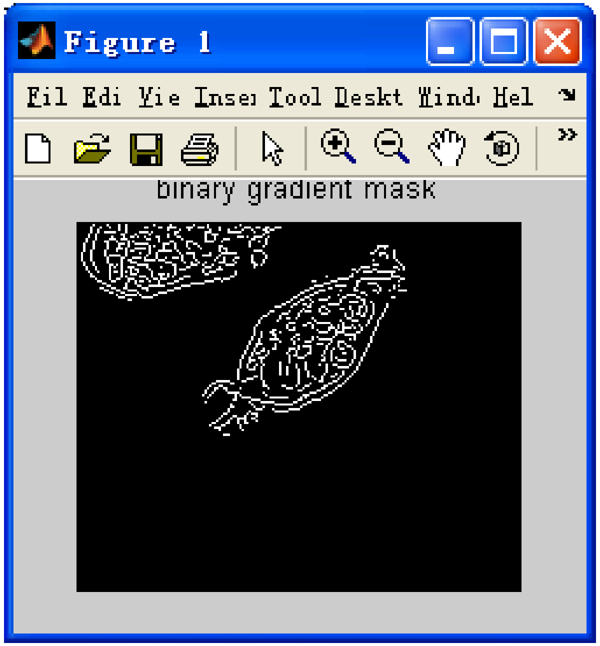



```
I=imread('rice.png');
J=edge(I,'canny');
K=imfill(J,'holes');
imshow(K)
```

边缘检测示例2

Cell.tif






```
BWs = edge(I, 'sobel', (graythresh(I) * .1));
figure, imshow(BWs), title('binary gradient mask');
```

Detecting a Cell Using Image Segmentation

See also: Image Processing Toolbox Demos
Morphological Segmentation

检测边缘的方法

华中科技大学

- 梯度算子
- Roberts梯度算子
- Prewitt算子
- Sobel算子
- Kirsch算子
- Laplace算子
- LOG算子
- DOG算子
- CANNY算子

模板产生的方法

h = fspecial(*type*)
h = fspecial(*type*, parameters)
type can have any of the following values:

Value	Description
'average'	Averaging filter
'disk'	Circular averaging filter (pillbox)
'gaussian'	Gaussian lowpass filter
'laplacian'	Approximates the two-dimensional Laplacian operator
'log'	Laplacian of Gaussian filter
'motion'	Approximates the linear motion of a camera
'prewitt'	Prewitt horizontal edge-emphasizing filter
'sobel'	Sobel horizontal edge-emphasizing filter

参见：**MATLAB\toolbox\images\images\fspecial.m**

高斯模板的产生方法

```
siz   = (p2-1)/2;          % p2 指明产生 ？ ×？ 的模板
std   = p3;                % p3 指明标准差
[x,y] = meshgrid(-siz(2):siz(2),-siz(1):siz(1));
arg   = -(x.*x + y.*y)/(2*std*std);
h     = exp(arg);
h(h<eps*max(h(:))) = 0;
sumh = sum(h(:))
if sumh ~= 0
     h  = h/sumh;
end;
```

$$h_g(n_1, n_2) = e^{\frac{-(n_1^2+n_2^2)}{2\sigma^2}}$$

$$h(n_1, n_2) = \frac{h_g(n_1, n_2)}{\sum_{n_1}\sum_{n_2} h_g}$$

Laplacian 模板的产生方法

h=fspecial('laplacian', alpha);

Alpha 的取值 为 0 ~ 1， 缺省值为 0.2

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

$$\nabla^2 = \frac{4}{(\alpha + 1)} \begin{bmatrix} \frac{\alpha}{4} & \frac{1-\alpha}{4} & \frac{\alpha}{4} \\ \frac{1-\alpha}{4} & -1 & \frac{1-\alpha}{4} \\ \frac{\alpha}{4} & \frac{1-\alpha}{4} & \frac{\alpha}{4} \end{bmatrix}$$

Laplacian 模板的产生方法

華中科技大学

hd=fspecial('laplacian');

0.1667	0.6667	0.1667
0.6667	-3.3333	0.6667
0.1667	0.6667	0.1667

h0=fspecial('laplacian',0);

0	1	0
1	-4	1
0	1	0

h1=fspecial('laplacian',1);

0.5000	0	0.5000
0	-2.0000	0
0.5000	0	0.5000

hh=fspecial('laplacian',0.5);

0.3333	0.3333	0.3333
0.3333	-2.6667	0.3333
0.3333	0.3333	0.3333

Laplacian 模板的产生方法

```
I=imread('cameraman.tif’);
ID=im2double(I);
h=fspecial('laplacian’);
J=imfilter(ID,h);
figure,imshow(J,[])
figure,imshow(abs(J),[])
```

Laplacian 算子的特性

华中科技大学

- 对噪声非常敏感
- 幅值产生双边缘
- 不能检测边缘的方向
- 二阶导数的符号可用于判断一个边缘像素是在边缘亮的一边，还是在边缘暗的一边。
- 一条连接二阶导数正极值和负极值的直线将在边缘中点附近穿过零点。
- 零交叉点的近似求法，将正值保留，负值变成0。

Laplacian of Gaussian

- Marr, D., and Hildreth, E., Theory of edge detection, *Proceedings of the Royal Society of London*, Series B, 1980,**207**:pp.187-217

Laplacian of Gaussian

- The Laplacian is seldom used on its own for edge detection because of its sensitivity to noise.
- The Laplacian-of-Gaussian (LoG) uses a Gaussian filter to blur the image and a Laplacian to enhance edges.
- Also known as Marr & Hildreth edge detector
- Edge localisation is done by finding zerocrossings.

Laplacian of Gaussian

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$\nabla^2 G(x,y) = [\frac{x^2+y^2-2\sigma^2}{\sigma^4}]e^{-\frac{x^2+y^2}{2\sigma^2}}$$

h = fspecial('log', hsize, sigma);

The default value for hsize is [5 5] and 0.5 for sigma.

Matlab 的实现

$$h_g(n_1,n_2) = e^{\frac{-(n_1^2+n_2^2)}{2\sigma^2}}$$

$$h(n_1,n_2) = \frac{(n_1^2+n_2^2-2\sigma^2)h_g(n_1,n_2)}{2\pi\sigma^6\sum_{n_1}\sum_{n_2}h_g}$$

Laplacian of Gaussian

h = fspecial('log');

0.0448	0.0468	0.0564	0.0468	0.0448
0.0468	0.3167	0.7146	0.3167	0.0468
0.0564	0.7146	-4.9048	0.7146	0.0564
0.0468	0.3167	0.7146	0.3167	0.0468
0.0448	0.0468	0.0564	0.0468	0.0448

h = fspecial('log', [7 7], 0.8);

0.0000	0.0003	0.0022	0.0042	0.0022	0.0003	0.0000
0.0003	0.0079	0.0454	0.0726	0.0454	0.0079	0.0003
0.0022	0.0454	0.0916	-0.0778	0.0916	0.0454	0.0022
0.0042	0.0726	-0.0778	-0.7771	-0.0778	0.0726	0.0042
0.0022	0.0454	0.0916	-0.0778	0.0916	0.0454	0.0022
0.0003	0.0079	0.0454	0.0726	0.0454	0.0079	0.0003
0.0000	0.0003	0.0022	0.0042	0.0022	0.0003	0.0000

Laplacian of Gaussian

华中科技大学

5×5 Laplacian of Gaussian kernel

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

2	4	4	4	2
4	0	-8	0	4
4	-8	-24	-8	4
4	0	-8	0	4
2	4	4	4	2

LOG模板的产生方法

```
siz   = (p2-1)/2;          % p2 指明产生 ？ ×？ 的模板
std   = p3;                % p3 指明标准差
[x,y] = meshgrid(-siz(2):siz(2),-siz(1):siz(1));
arg   = -(x.*x + y.*y)/(2*std*std);
h     = exp(arg);
h(h<eps*max(h(:))) = 0;
sumh = sum(h(:))
if sumh ~= 0
      h  = h/sumh;
end;
h1 = h.*(x.*x + y.*y - 2*std2)/(std2^2);
h   = h1 - sum(h1(:))/prod(p2);
```

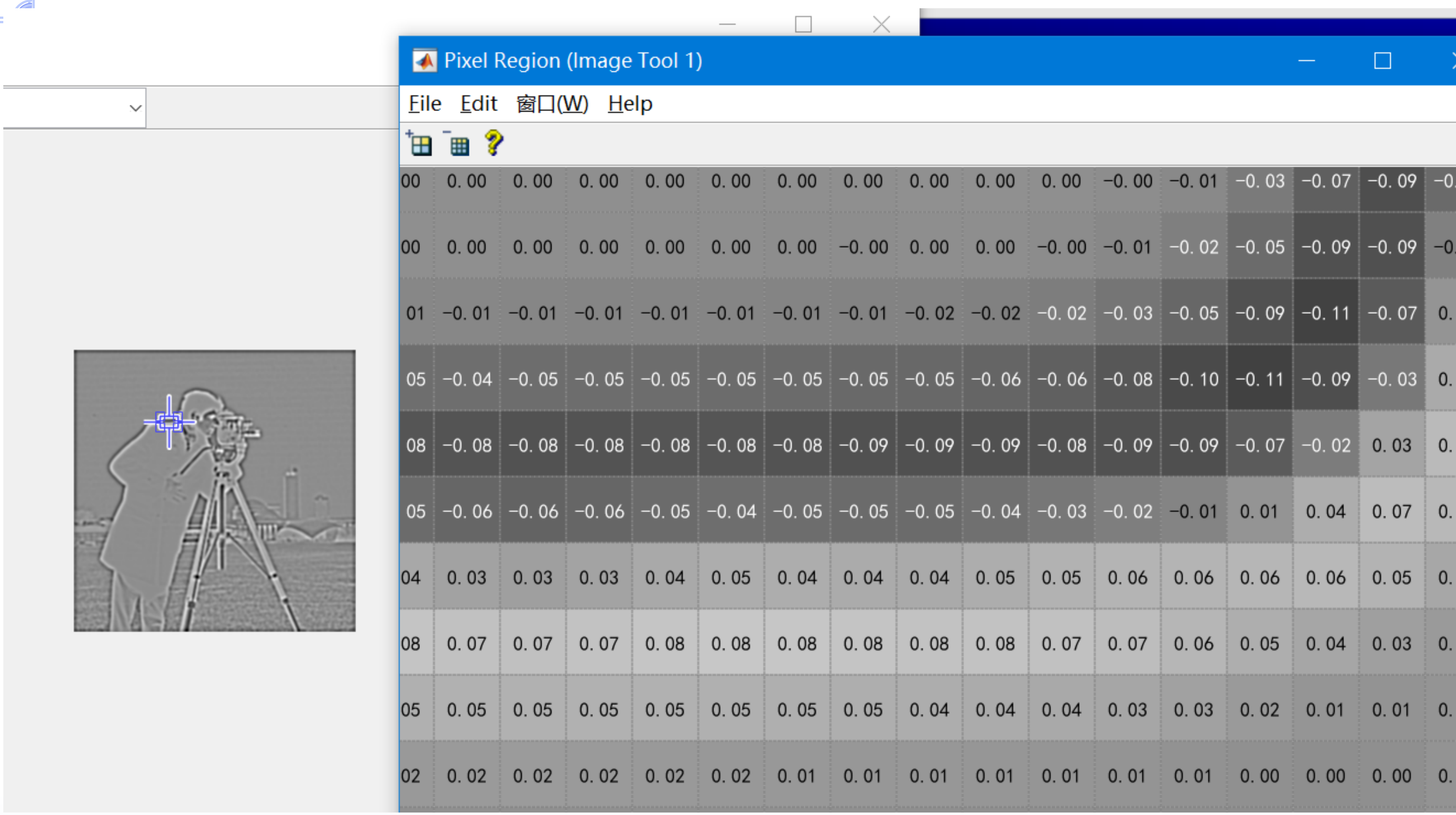
LOG 边缘检测

```
I=imread('cameraman.tif');
ID=im2double(I);
h=fspecial('log',[13 13], 1.414);
J=imfilter(I,h);
figure,imshow(J,[])
JD=imfilter(ID,h);
figure,imshow(JD,[])
figure,imshow(abs(JD),[])
```

不同的数据类型，
运算结果有区别；

UINT8 中无负数

LOG 边缘检测



00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.00	-0.01	-0.03	-0.07	-0.09	-0.0
00	0.00	0.00	0.00	0.00	0.00	0.00	-0.00	0.00	0.00	-0.00	-0.01	-0.02	-0.05	-0.09	-0.09	-0.0
01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.02	-0.02	-0.02	-0.03	-0.05	-0.09	-0.11	-0.07	0.0
05	-0.04	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.05	-0.06	-0.06	-0.08	-0.10	-0.11	-0.09	-0.03	0.0
08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.08	-0.09	-0.09	-0.09	-0.08	-0.09	-0.09	-0.07	-0.02	0.03	0.0
05	-0.06	-0.06	-0.06	-0.05	-0.04	-0.05	-0.05	-0.05	-0.04	-0.03	-0.02	-0.01	0.01	0.04	0.07	0.0
04	0.03	0.03	0.03	0.04	0.05	0.04	0.04	0.04	0.05	0.05	0.06	0.06	0.06	0.06	0.05	0.0
08	0.07	0.07	0.07	0.08	0.08	0.08	0.08	0.08	0.08	0.07	0.07	0.06	0.05	0.04	0.03	0.0
05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.04	0.04	0.04	0.03	0.03	0.02	0.01	0.01	0.0
02	0.02	0.02	0.02	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.00	0.00	0.00	0.0

- 幅值产生双边缘，具有 Laplacian的特性
- 具有高斯平滑特性，噪声减少

边缘检测方法存在的问题及对策

一般对目标轮廓边界的期望：

- 连，连续无间断
- 准，各边界点在正确的位置上。
- 细，轮廓为单象素宽；

边缘检测方法存在的问题及对策

华中科技大学

- 断边现象：

　　应连接的但相距较远的同一目标边界上的边缘点连不起来。

- 伪边现象：

　　不应连接的但相距较近的不同目标边界上的边缘点被连接起来。

- 边界粗：

　　把一个微分算于应用于梯度强度变化范围较宽的区域，会检出若干行并列的边缘象素。

Canny 算子

E=edge(I, ’canny’, thresh, sigma);

E=edge(I, ’canny’, thresh);   // sigma为默认值 1

E=edge(I, ’canny’);   // thresh通过算法生成

sigma 高斯模板的标准差，控制对图像的平滑程度

Thresh ： 空，   程序自动生成
一个值，high_thresh
两个值，[low_thresh high_thresh]

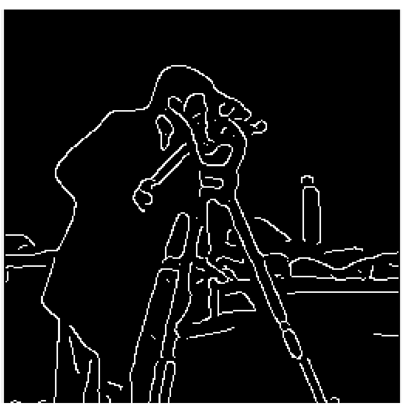
梯度幅值 > high_thresh 强边缘点
梯度幅值 < low_thresh   非边缘点
low_thresh < 梯度幅值 < high_thresh 弱边缘点

Canny 算子

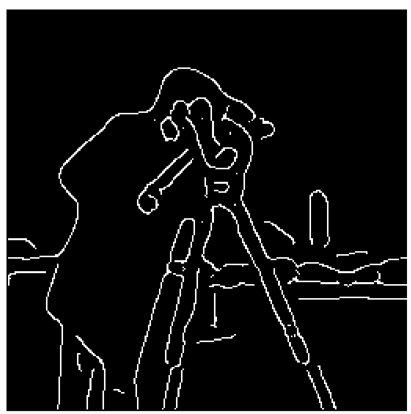
I

J1

J2



J3



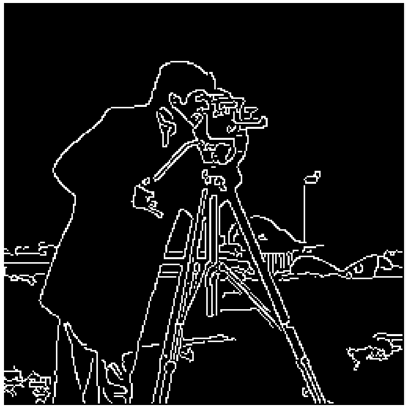
J4

```
I=imread('cameraman.tif’);
J1 =edge(I,'canny',[], 1);
J2 =edge(I,'canny',[], 2);
J3 =edge(I,'canny',[], 3);
J4 =edge(I,'canny',[], 4);
```

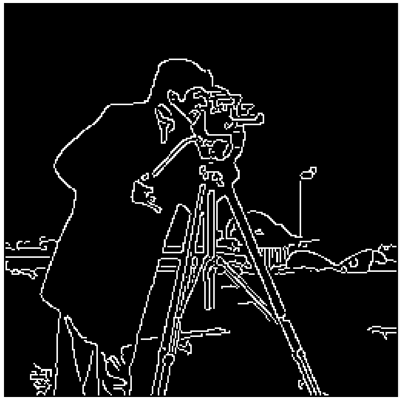
Q：调整 sigma 对结果有何影响？为什么？

Canny 算子

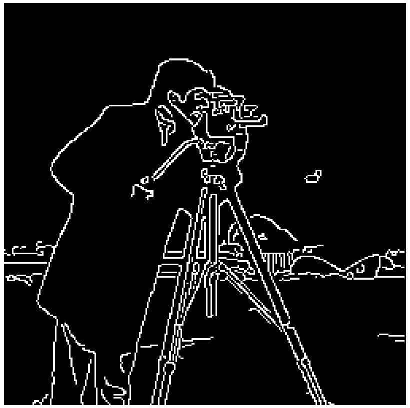
I



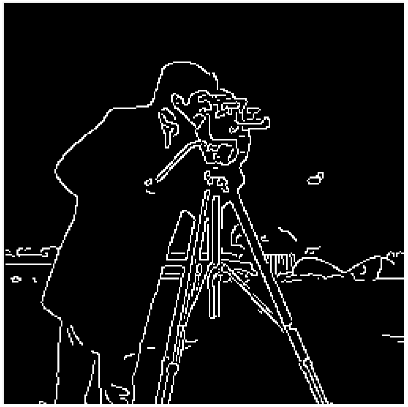
J1



J2



J3



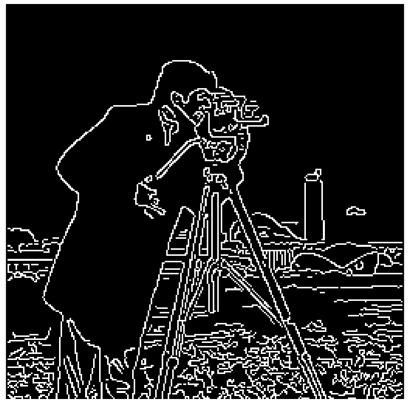
J4

```
I=imread('cameraman.tif');
J1=edge(I,'canny',[0.01 0.2], 1);
J2=edge(I,'canny',[0.05 0.2], 1);
J3=edge(I,'canny',[0.1 0.2], 1);
J4=edge(I,'canny',[0.15 0.2], 1);
```

Q：调整 low_thresh 对结果有何影响？为什么？

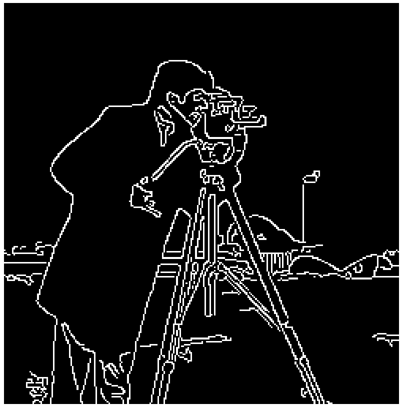
Canny 算子

I

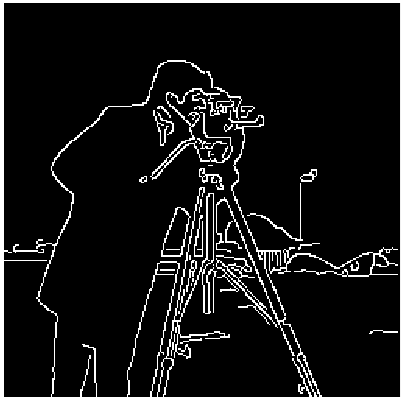


J1

J2



J3



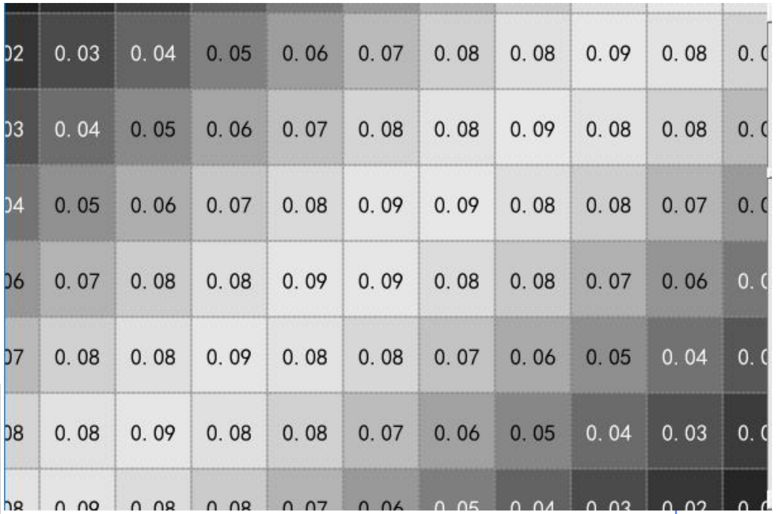
J4

```
I=imread('cameraman.tif');
J1=edge(I,'canny',[0.05 0.1], 1);
J2=edge(I,'canny',[0.05 0.15], 1);
J3=edge(I,'canny',[0.05 0.2], 1);
J4=edge(I,'canny',[0.05 0.25], 1);
```

Q：调整 high_thresh 对结果有何影响？为什么？

Canny 算子

- 图像高斯平滑
- 求平滑图像各点梯度
- 梯度幅值的非极大值抑制
- 双阈值计算边缘



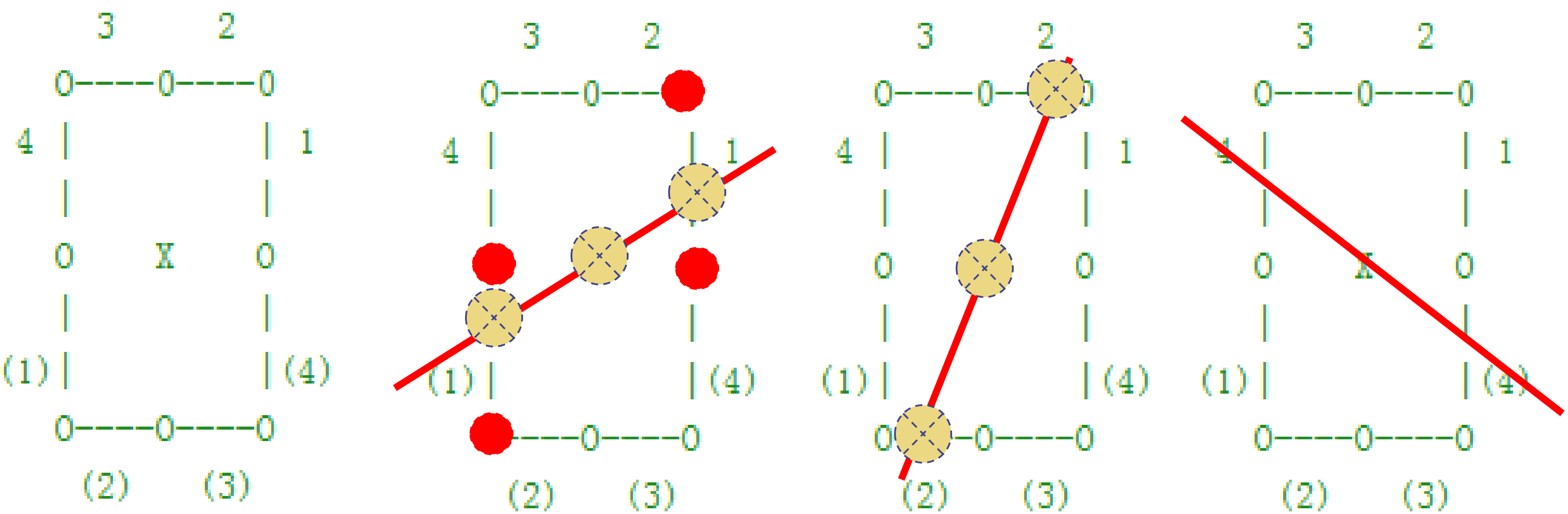
02	0. 03	0. 04	0. 05	0. 06	0. 07	0. 08	0. 08	0. 09	0. 08	0. (
03	0. 04	0. 05	0. 06	0. 07	0. 08	0. 08	0. 09	0. 08	0. 08	0. (
04	0. 05	0. 06	0. 07	0. 08	0. 09	0. 09	0. 08	0. 08	0. 07	0. (
06	0. 07	0. 08	0. 08	0. 09	0. 09	0. 08	0. 08	0. 07	0. 06	0. (
07	0. 08	0. 08	0. 09	0. 08	0. 08	0. 07	0. 06	0. 05	0. 04	0. (
08	0. 08	0. 09	0. 08	0. 08	0. 07	0. 06	0. 05	0. 04	0. 03	0. (
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

```
I=imread('cameraman.tif');
h=fspecial('gaussian',[21 21],3);
B=imfilter(I,h);
[gx gy]=gradient(B);
mag=sqrt(gx.*gx+gy.*gy);
imtool(mag,[])
```

Canny 算子

华中科技大学

- **梯度幅值的非极大值抑制**



Canny 算子

华中科技大学

- 对图像进行**高斯平滑**，
  以消除噪声，减少可能出现的弱小边缘
- 求平滑图像各点 x方向、y方向的**梯度**
- 沿着梯度方向，对**非极大**的梯度幅值进行**抑制**
  边缘变细
  非极大值变成 0，极大值保持不变
- 双阈值计算边缘，
  梯度幅值高于大阈值的点为边缘点，梯度幅值低于小阈值的点为非边缘点；
  弱边缘点若与强边缘点连通，则被“磁化”为
  边缘点，否则就被当成背景点。

拟合曲面求导法

一次、二次曲面拟合求梯度

试证明: 对3×3子域采用二次曲面拟合，结果为**Prewitt**算子（相差**1/3**系数）

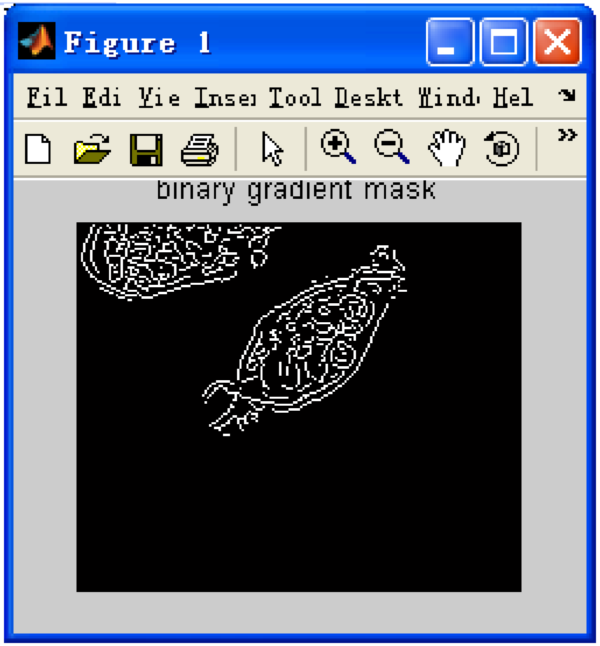
三次曲面拟合求梯度方向二阶导数过**0**点

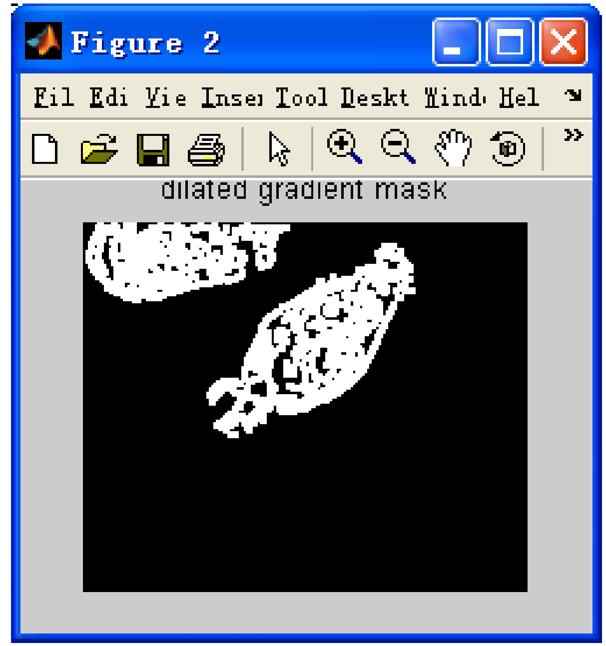
估计拟合曲面阶跃幅度

断与连

形态学方法
图搜索方法
梯度方向和幅度相似法

用形态学方法——连边



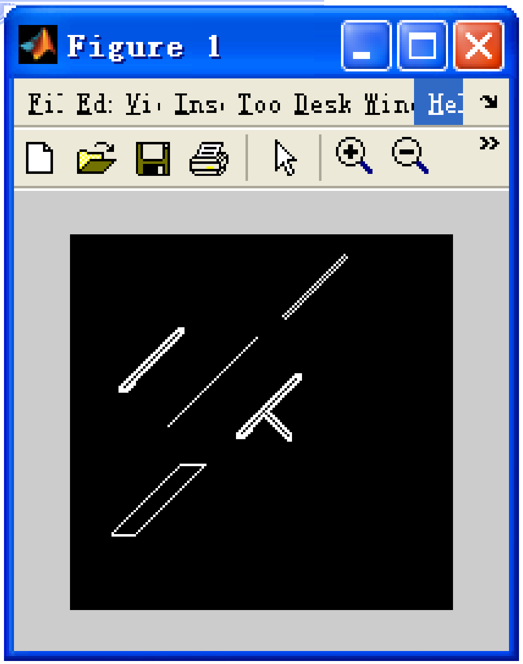



```
se90 = strel('line', 3, 90);
se0 = strel('line', 3, 0);
BWsdil = imdilate(BWs, [se90 se0]);
figure, imshow(BWsdil),
title('dilated gradient mask');
```

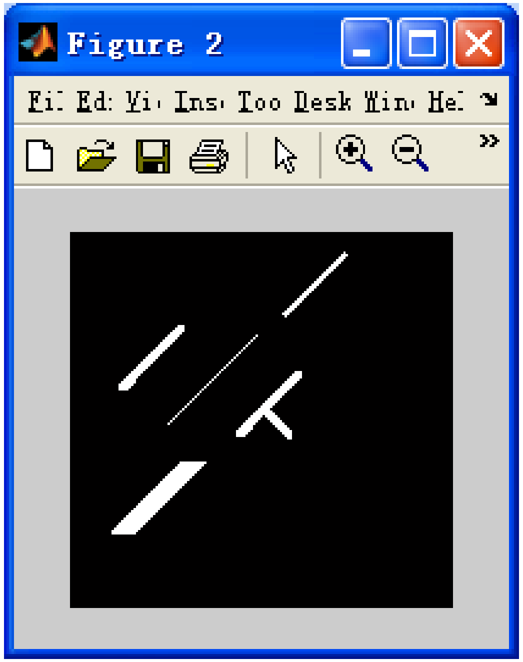
Detecting a Cell Using Image Segmentation

思考题:
孔洞填充的算法思想是什么？

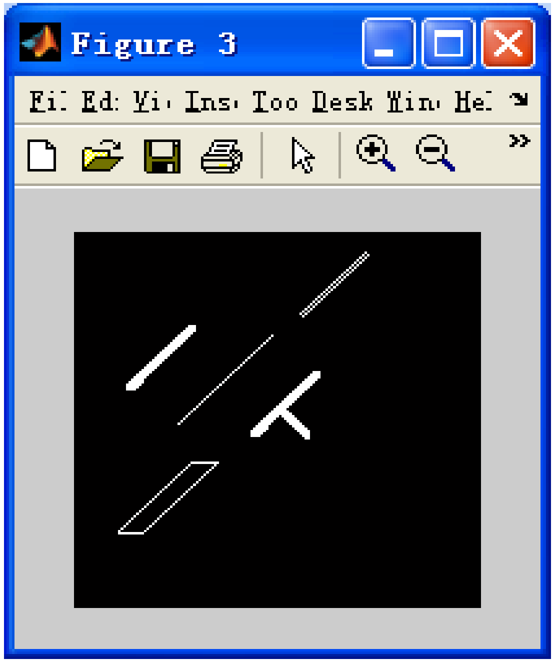
```
BWdfill = imfill(BWsdil, 'holes');
figure, imshow(BWdfill);
title('binary image with filled holes');
```



J



P



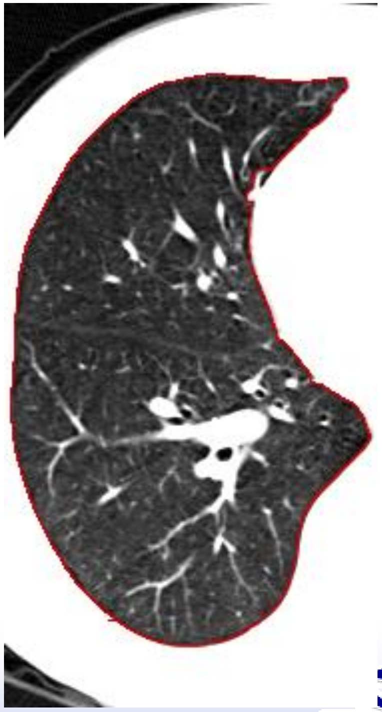
Q

使用孔洞填充应注意的问题

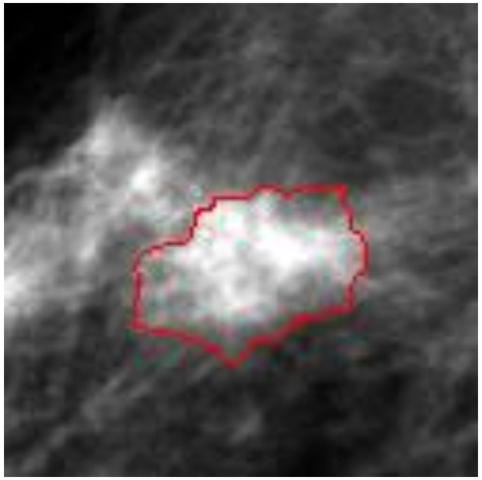
```
I=imread('hole_fill_test.bmp');
J=im2bw(I);
P=imfill(J,4,'holes');
Q=imfill(J,8,'holes');
```

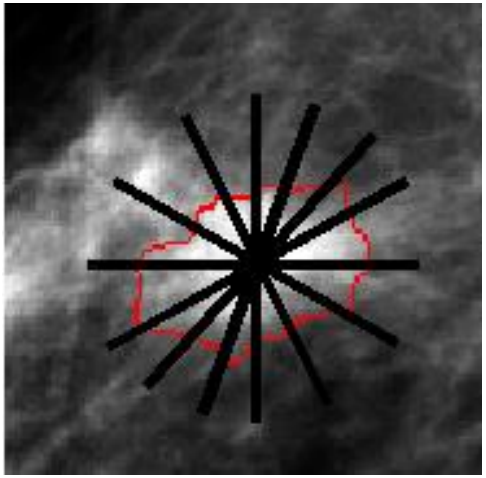
用图的方法——连边

Alexandre X.Falcao, Jayaram K.Udupa, etc.
User-Steered Image Segmentation Paradigms：Live Wire and Live Lane.
Graphic models and Image Processing, 1998

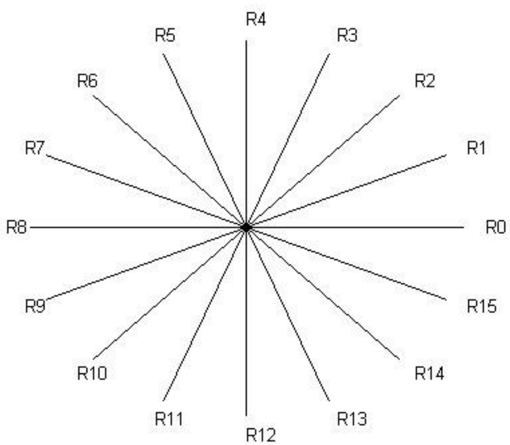


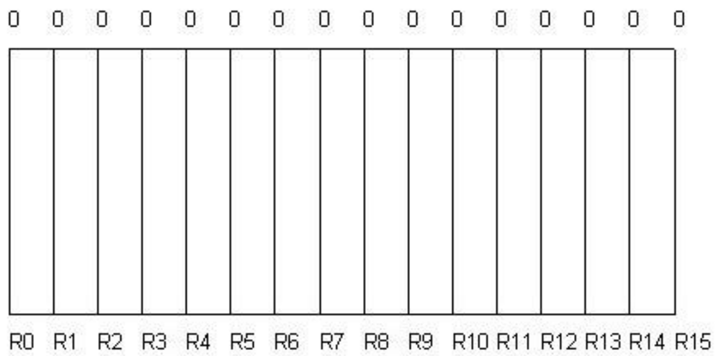
用动态规划的方法——连边

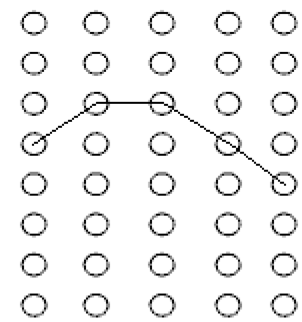


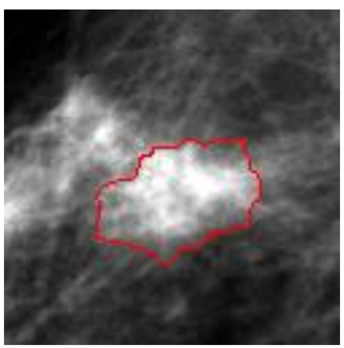


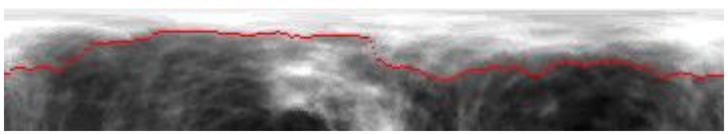
用动态规划的方法——连边


R4
R5
R3
R6
R2
R7
R1
R8
R0
R9
R15
R10
R14
R11
R13
R12


0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
R0 R1 R2 R3 R4 R5 R6 R7 R8 R9 R10 R11 R12 R13 R14 R15



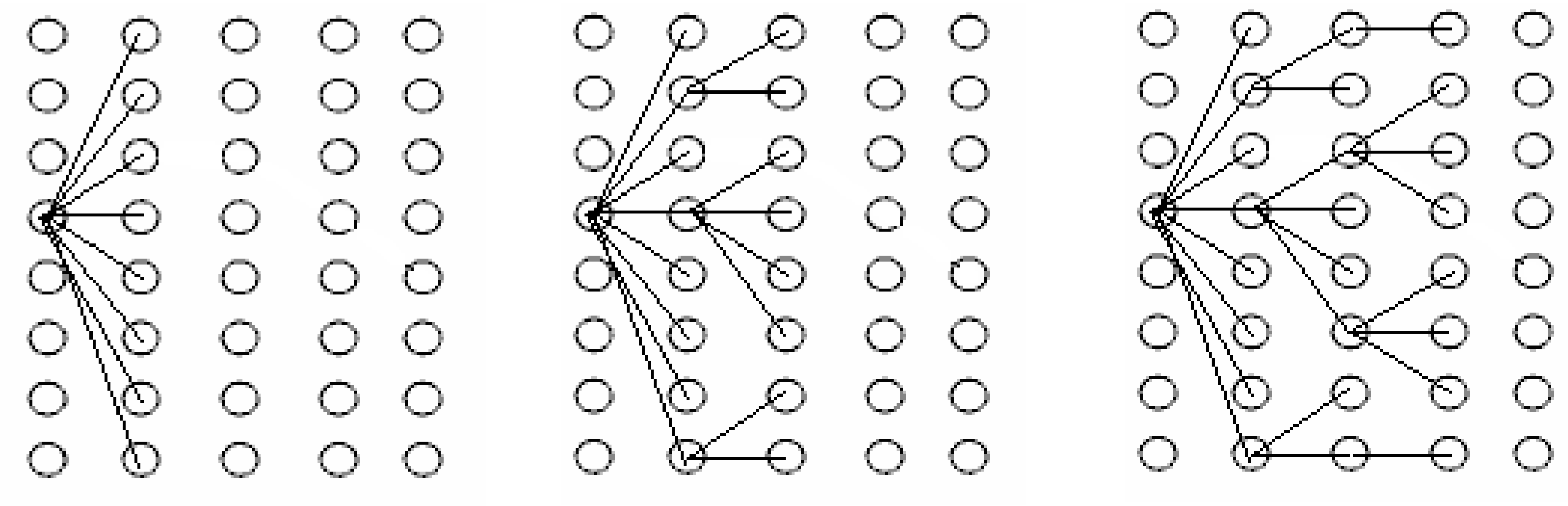




用动态规划的方法——连边

动态规划分割的基本思想

——多段图中找出最短路径



用动态规划的方法——连边

华中科技大学

动态规划分割方法的关键：

路径代价由哪些部分组成？

各部分的权重如何确定？

- **梯度代价** **平面拟合、径向梯度、归一化**
- **灰度代价** **边界灰度估计**
- **结点间距离代价**

基于图割的图像分割方法

华中科技大学

图割 (Graph Cuts)

基于图论的图像分割方法

基于最大流最小割的能量优化方法

总结

常规的边缘检测算子的产生方法

常规的边缘检测方法的特性

Canny 算子