数字图像处理

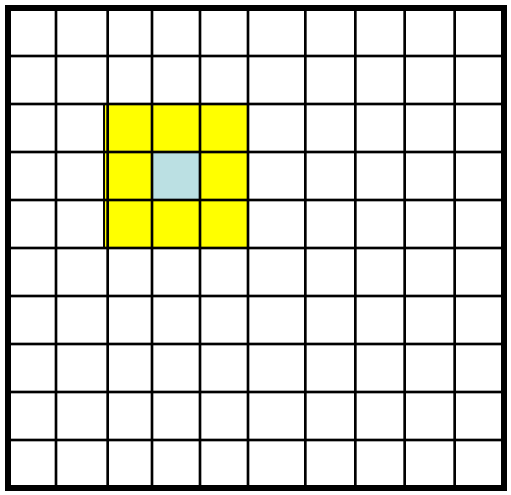
指导教师： 胡晓雁

电子邮件： huxy@bnu.edu.cn

北京师范大学信息科学与技术学院

局部平滑法

- 对图像采用3×3的邻域平均法，对于像素(m,n)，其邻域像素如下：



(m-1,n-1)	(m-1,n)	(m-1,n+1)
(m,n-1)	(m,n)	(m,n+1)
(m+1,n-1)	(m+1,n)	(m+1,n+1)

则有：$g(m,n)=\frac{1}{9}\sum_{i?\{-1,0,1\}}\sum_{j?\{-1,0,1\}} I(m+i,n+j)$

图像的积分与微分

- 图像积分使得图像更平滑、边缘模糊

$$g(m,n)=\frac{1}{9}\sum_{i\in\{-1,0,1\}}\sum_{j\in\{-1,0,1\}} I(m+i,n+j)$$

- 图像微分可使得图像边缘突出、清晰

$$\frac{\partial I(x,y)}{\partial x}=\lim_{\Delta x\to 0}\frac{I(x+\Delta x,y)-I(x,y)}{\Delta x}$$

微积分知识回顾

- 对于一元函数f(x)，一阶导数是：

$$f'(x) = \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

微积分知识回顾

- 对于一元函数f(x)，一阶导数是：

$$f'(x) \approx \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

微积分知识回顾

- 图像中的像素间隔为1，因此取 $\Delta x = 1$

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

微积分知识回顾

- 二阶导数是一阶导数的导数

$$f''(x) = \lim_{\Delta x \to \infty} \frac{f'(x+\Delta x) - f'(x)}{\Delta x}$$

微积分知识回顾

- 二阶导数是一阶导数的导数

$$f''(x) \approx \frac{f'(x+\Delta x) - f'(x)}{\Delta x}$$

$$\approx \frac{f(x+2\Delta x) - 2f(x+\Delta x) + f(x)}{\Delta x^2}$$

微积分知识回顾

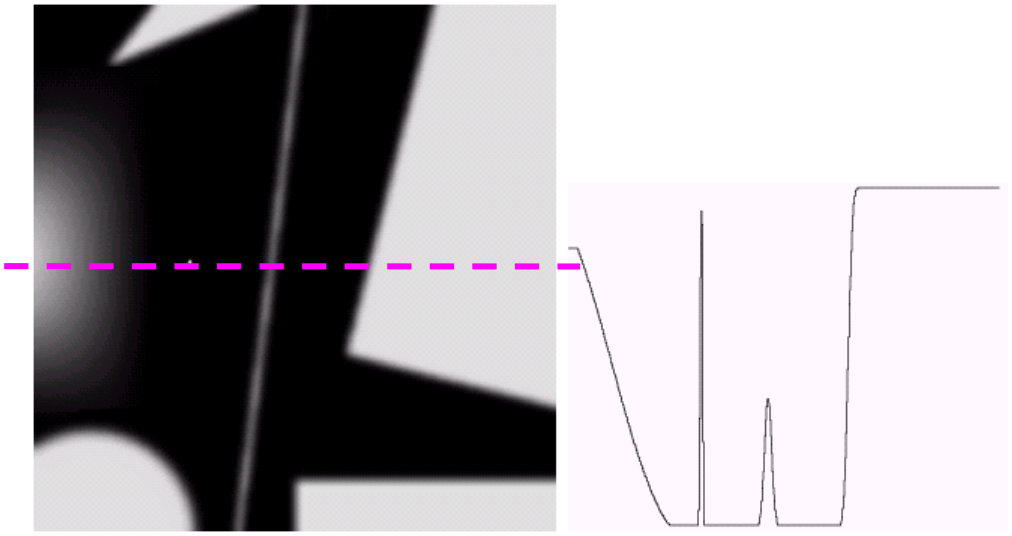
- 图像中的像素间隔为1，因此取 $\Delta x = 1$

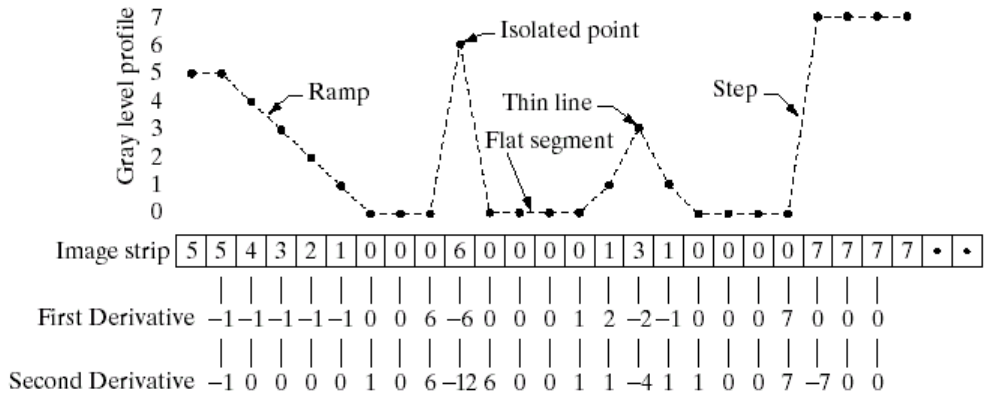
$$f''(x) \approx \frac{f'(x+\Delta x) - f'(x)}{\Delta x} = f(x+1) + f(x-1) - 2f(x)$$

图像中的微分

a b
c

FIGURE 3.38
(a) A simple image. (b) 1-D horizontal gray-level profile along the center of the image and including the isolated noise point.
(c) Simplified profile (the points are joined by dashed lines to simplify interpretation).





图像中的微分

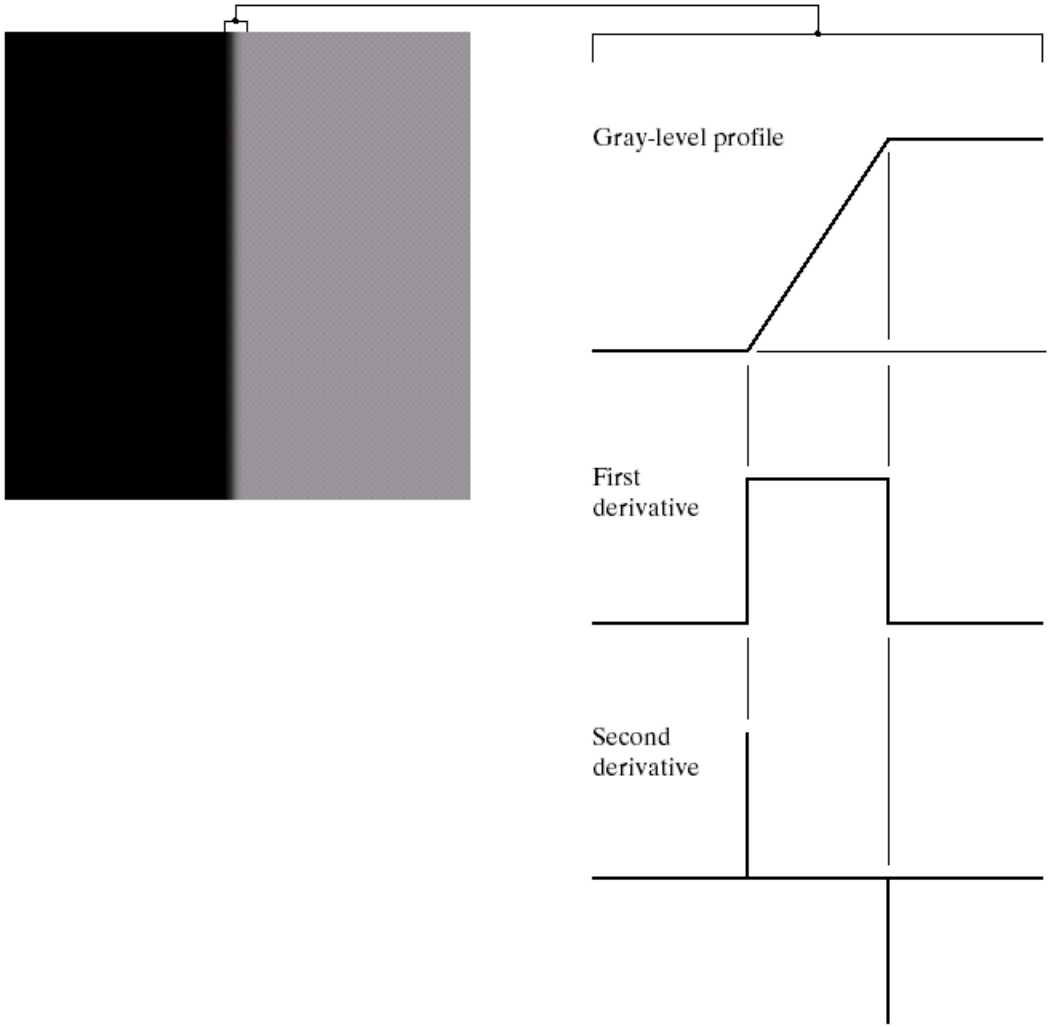
- 根据上图知：
  - 一阶微分只有在平坦的区域为0或接近0
  - 一阶微分处理一般对灰度阶梯有较强响应
  - 一阶微分处理通常产生较宽边缘

  - 二阶微分在平坦处以及连续均匀斜坡处为0，在斜坡的起始和终点处非零
  - 二阶微分处理对灰度阶梯变化产生双响应
  - 二阶微分处理对细节有较强的响应如细线条和孤立点
  - 二阶微分对孤立噪声点及其周边点的响应比一阶微分强的多

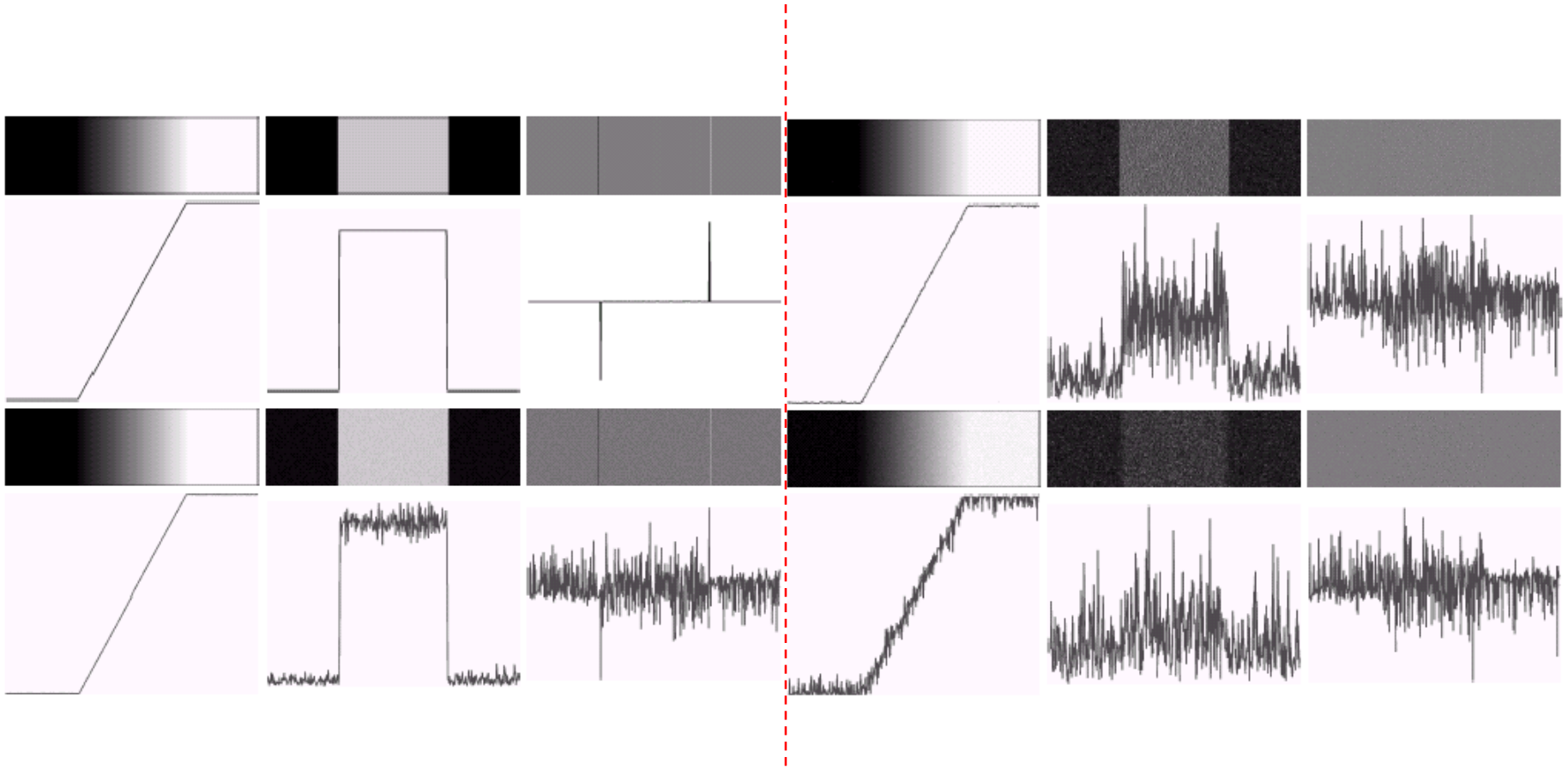
图像中的微分

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.



图像中的微分



图像的梯度

- 对于图像f(x，y)，在(x，y)处的梯度定义为

$$grad(x,y)=\begin{bmatrix} f_x' \\ f_y' \end{bmatrix}=\begin{bmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{bmatrix}$$

- 图像梯度是一个矢量，其大小和方向为

$$\left|grad(x,y)\right|=\sqrt{f_x'^2+f_y'^2}=\sqrt{(\frac{\partial f(x,y)}{\partial x})^2+(\frac{\partial f(x,y)}{\partial y})^2}$$

$$\theta=tg^{-1}(f_y'/f_x')=tg^{-1}(\frac{\partial f(x,y)}{\partial y}/\frac{\partial f(x,y)}{\partial x})$$

图像的梯度

- 对于图像f(x，y)，在(x，y)处的梯度定义为

$$\nabla I(x,y)=\left(\frac{\partial I(x,y)}{\partial x},\frac{\partial I(x,y)}{\partial y}\right)$$

- 图像梯度是一个矢量，其大小和方向为

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

$$\theta=tg^{-1}(\frac{\partial I}{\partial y}/\frac{\partial I}{\partial x})$$

图像的梯度

- 对于离散图像处理而言，常用到梯度的大小，因此把梯度的大小习惯称为“梯度”。并且一阶偏导数采用一阶差分近似表示，即

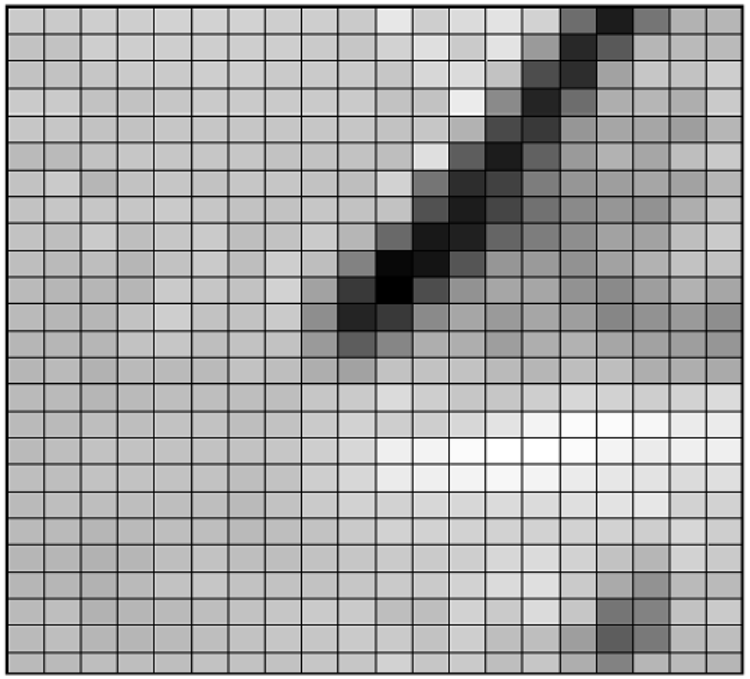
$$f_x' = f(x+1, y) - f(x, y)$$
$$f_y' = f(x, y+1) - f(x, y)$$

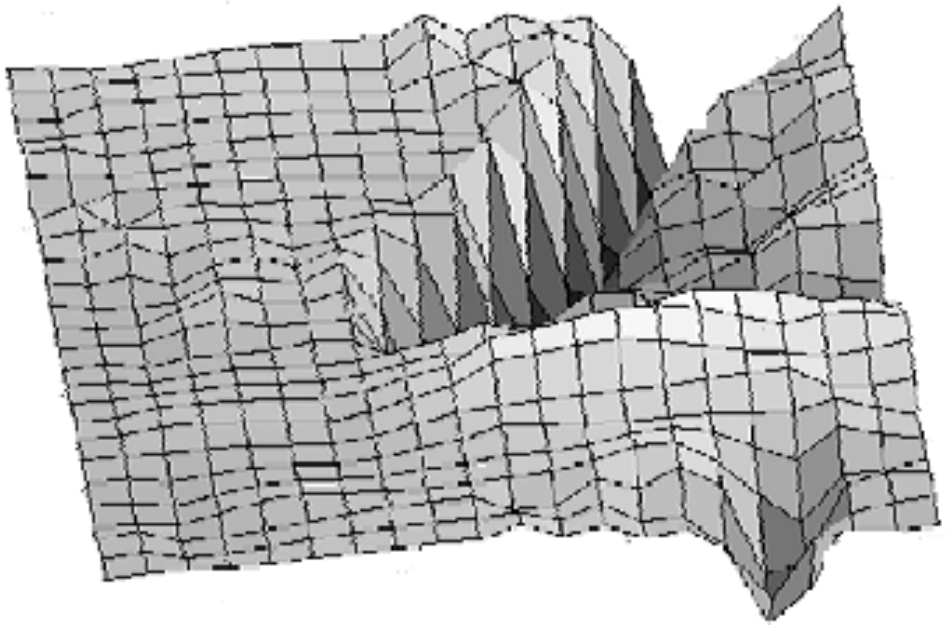
为简化梯度的计算，经常使用以下两种近似计算值

$$\nabla f = \max(|f_x'|, |f_y'|)$$

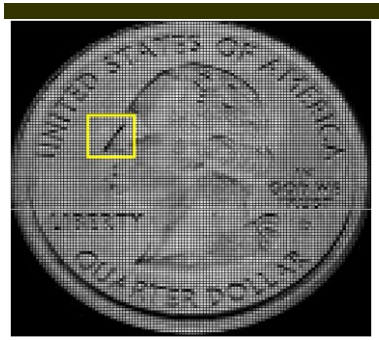
$$\nabla f = |f_x'| + |f_y'|$$

图像的梯度



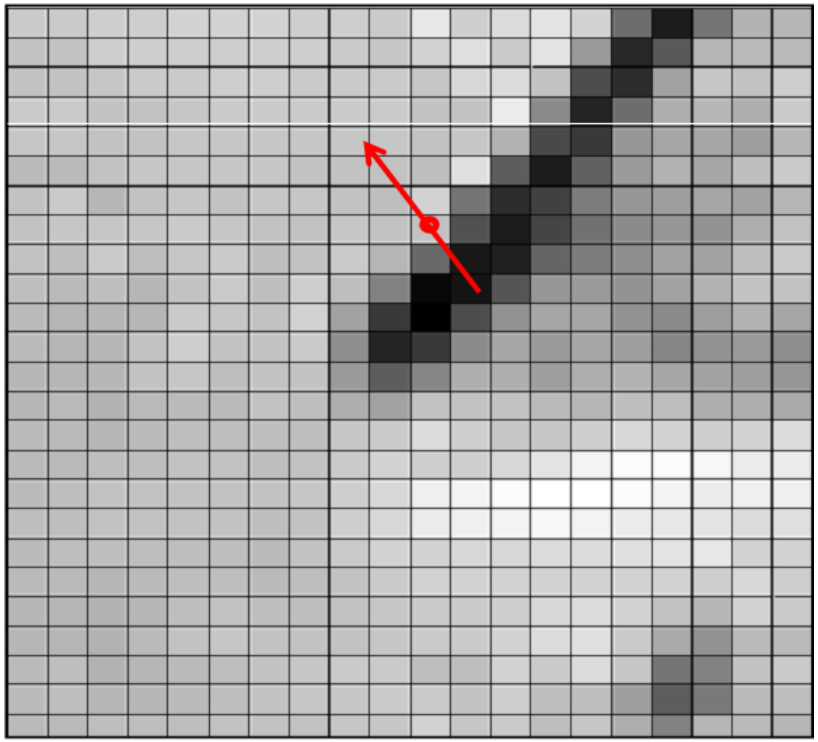


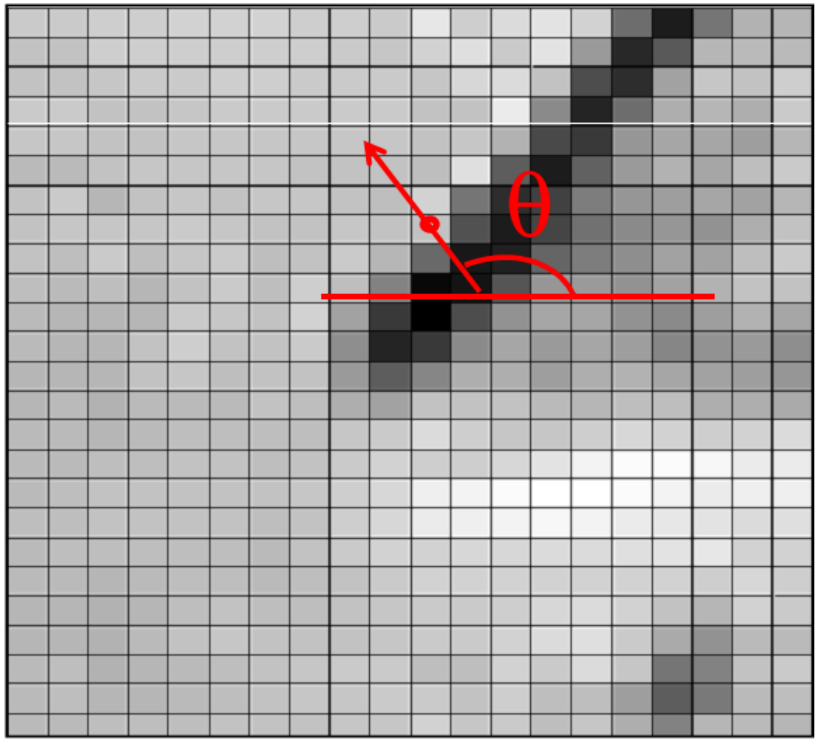
图像的梯度



$$\left|grad(x,y)\right| = \sqrt{f_x'^2 + f_y'^2} = \sqrt{(\frac{\partial f(x,y)}{\partial x})^2 + (\frac{\partial f(x,y)}{\partial y})^2}$$

$$\theta = tg^{-1}(f_y' / f_x') = tg^{-1}(\frac{\partial f(x,y)}{\partial y} / \frac{\partial f(x,y)}{\partial x})$$





梯度算子及模板

$$f_x' = f(x+1,y) - f(x,y) = z_6 - z_5$$
$$f_y' = f(x,y+1) - f(x,y) = z_8 - z_5$$

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

-1	1

-1
1

梯度算子及模板

a b
c d

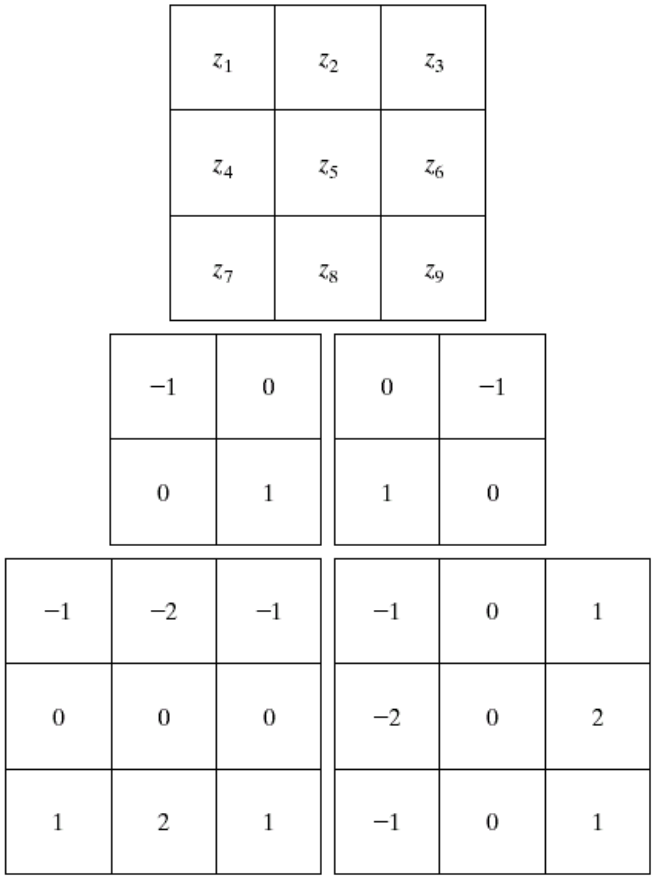
FIGURE 10.10
(a) Original image. (b) $|G_x|$, component of the gradient in the x-direction. (c) $|G_y|$, component in the y-direction. (d) Gradient image, $|G_x| + |G_y|$.

梯度算子及模板

- 除梯度算子以外，还可采用以下算子计算梯度
  - Roberts
  - Prewitt
  - Sobel

- 平滑算子模板对比

$$H_1 = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad H_2 = \frac{1}{10}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad H_3 = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$



图像边缘检测 Roberts算子

- Roberts[1963]使用交叉差分算法:

$$f_x' = f(x+1, y+1) - f(x, y) = z_9 - z_5$$
$$f_y' = f(x, y+1) - f(x+1, y) = z_8 - z_6$$

−1	0
0	1

0	−1
1	0

According to Roberts, an edge detector should have the following properties:
the produced edges should be well-defined,
the background should contribute as little noise as possible,
and the intensity of edges should correspond as close as possible to what a human would perceive.

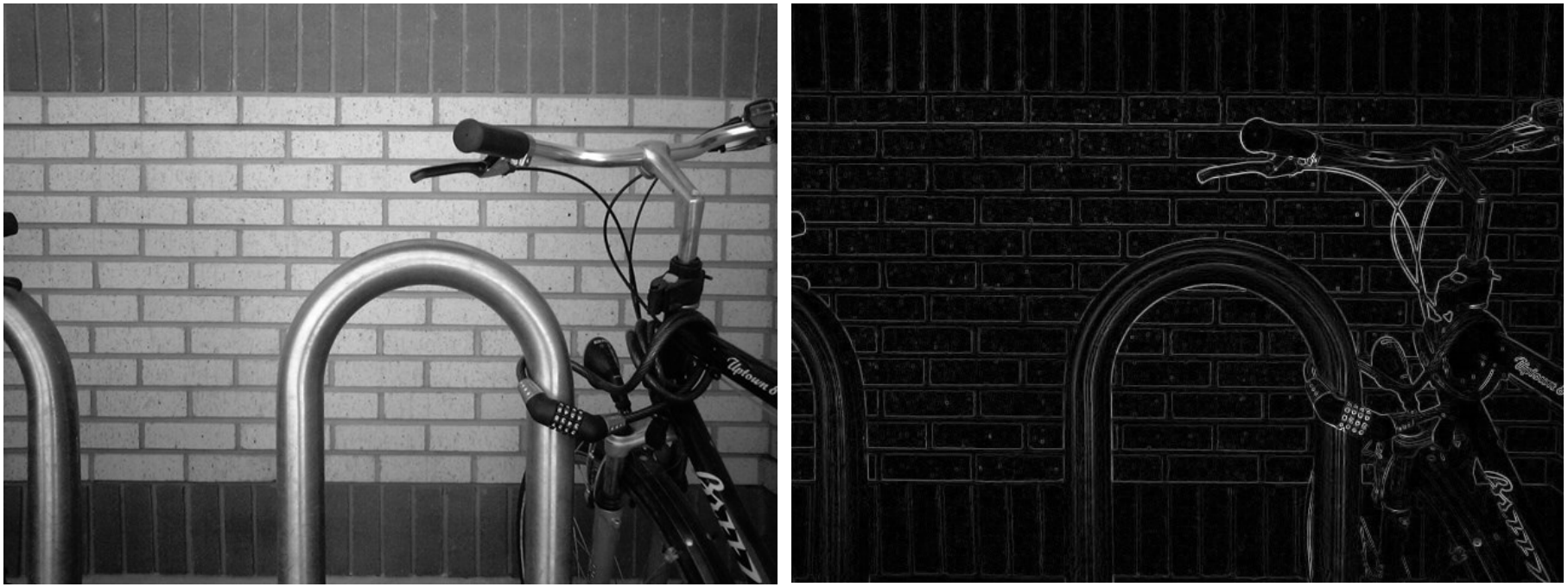
$$y_{i,j} = \sqrt{x_{i,j}}$$

$$z_{i,j} = \sqrt{(y_{i,j} - y_{i+1,j+1})^2 + (y_{i+1,j} - y_{i,j+1})^2}$$

图像边缘检测 Roberts算子

- Roberts[1963]
  - Background noise seems to be reduced by using operators symmetric in x and y.
  - In order to make equally apparent edges have equal derivatives, the intensity values of the picture can be subjected to **a gamma change so as to make intensity differences proportional to a human's ability to perceive them**.
  - According to psychophysical theory, the square root of the intensities should be used in order to achieve the desired effect

图像边缘检测 Roberts算子



By Davidwkennedy - http://en.wikipedia.org/wiki/File:Bikesgray.jpg, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=8904801

图像边缘检测 Prewitt算子

- 为在锐化边缘的同时减少噪声的影响，Prewitt从加大边缘增强算子的模板大小出发，由2x2扩大到3x3来计算差分，如图(a)所示。

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

Prewitt 算子

图像边缘检测 Prewitt算子

By Davidwkennedy - http://en.wikipedia.org/wiki/File:Bikesgray.jpg, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=8904801

图像边缘检测 Sobel算子

- Sobel在Prewitt算子的基础上，对4-邻域采用带权的方法计算差分，对应的模板如图

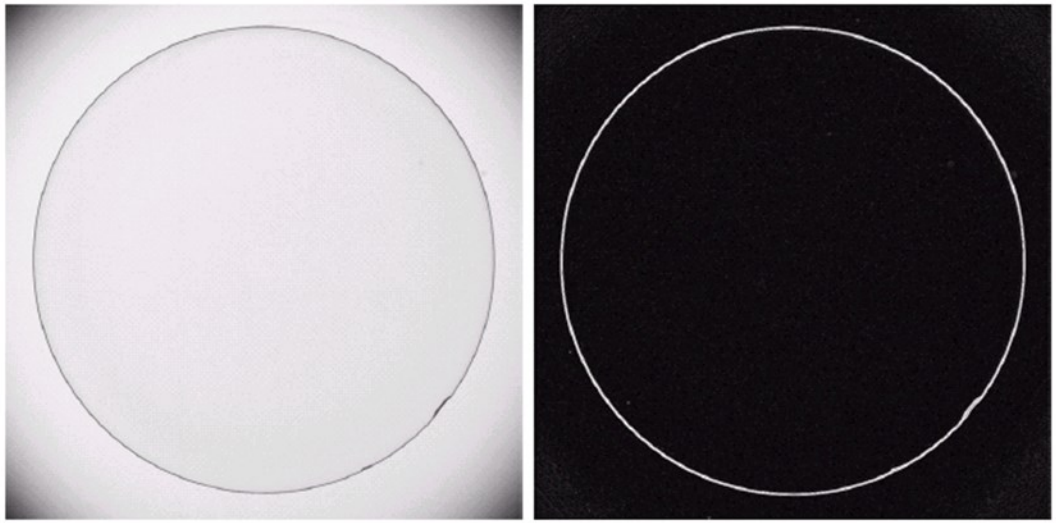
−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

图像边缘检测 Sobel算子

By Davidwkennedy - http://en.wikipedia.org/wiki/File:Bikesgray.jpg, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=8904801

例：隐形眼镜检测



a b

FIGURE 3.45 Optical image of contact lens (note defects on the boundary at 4 and 5 o'clock). (b) Sobel gradient. (Original image courtesy of Mr. Pete Sites, Perceptics Corporation.)

4点、5点钟处两个边缘缺陷，去除了灰度平坦区域，梯度处理突出了小斑点

图像梯度的显示

根据梯度计算式就可以计算Roberts、Prewitt和Sobel梯度。一旦梯度算出后，就可根据不同的需要生成不同的梯度增强图像

Roberts

Prewitt

Sobel

图像梯度的显示

- 第一种输出形式

$$g(x,y) = grad(x,y)$$

- 仅显示灰度变化比较徒的边缘轮廓，而灰度变化比较平缓或均匀的区域则呈黑色

图像梯度的显示

- 第二种输出形式

$$g(x,y)=\begin{cases} grad(x,y), grad(x,y)\geq T \\ f(x,y), 其它 \end{cases}$$

图像梯度的显示

- 第三种输出形式

$$g(x,y)=\begin{cases} L_G & , \quad grad(x,y)^3 \ T \\ f(x,y), & \text{其他} \end{cases}$$

- 它将明显边缘用一固定的灰度级L_G来表现

图像梯度的显示

- 第四种输出形式

$$g(x,y)=\begin{cases} grad(x,y) & ,grad(x,y)\geq T \\ L_B, & 其他 \end{cases}$$

- 此方法将背景用一个固定的灰度级 L_B 来表现，便于研究边缘灰度的变化

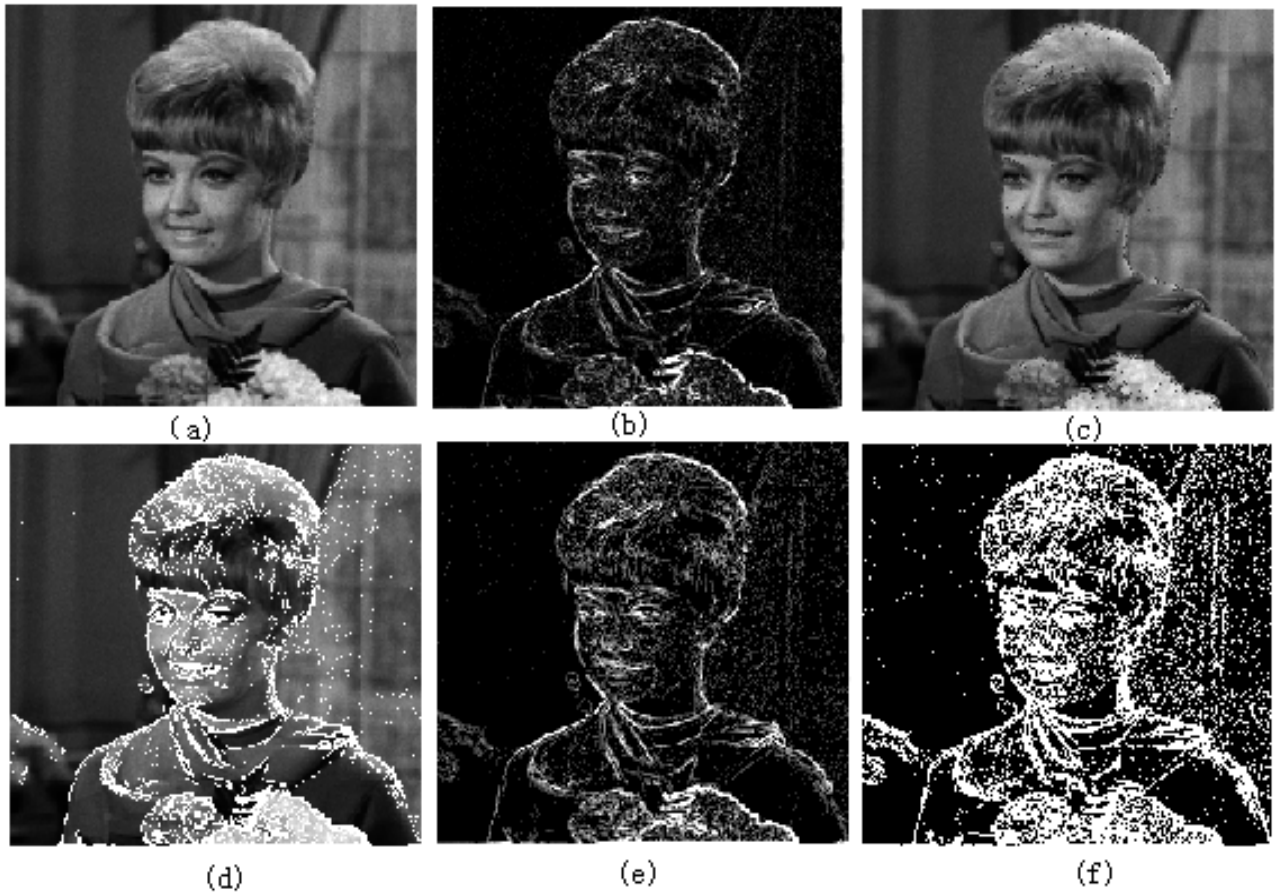
图像梯度的显示

- 第五种输出形式

$$g(x,y)=\begin{cases} L_G & ,grad(x,y)\geq T \\ L_B, & \text{其他} \end{cases}$$

- 这种方法将明显边缘和背景分别用灰度级L_G和L_B表示，生成二值图像，便于研究边缘所在位置

图像梯度的显示



Giving Photos a "Painted" Look

Case study: From P. Litwinowicz's SIGGRAPH'97 paper
"Processing Images and Videos for an Impressionist Effect"

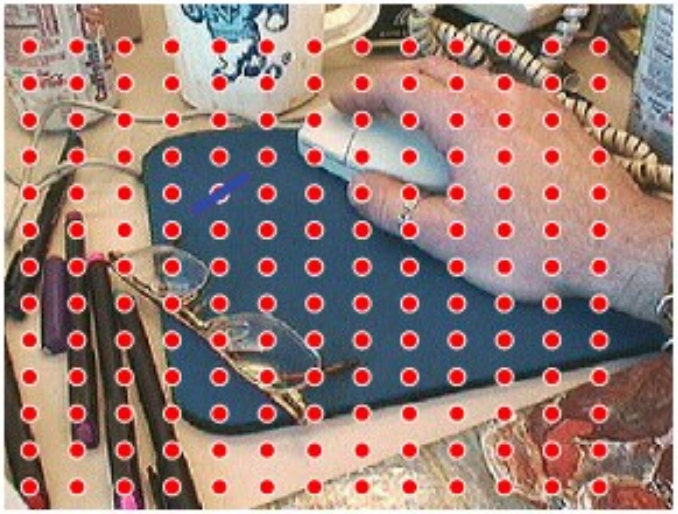
Giving Photos a “Painted” Look

How would you do it?

Original photo

Step 1: Stroke Scan-Conversion

Original photo

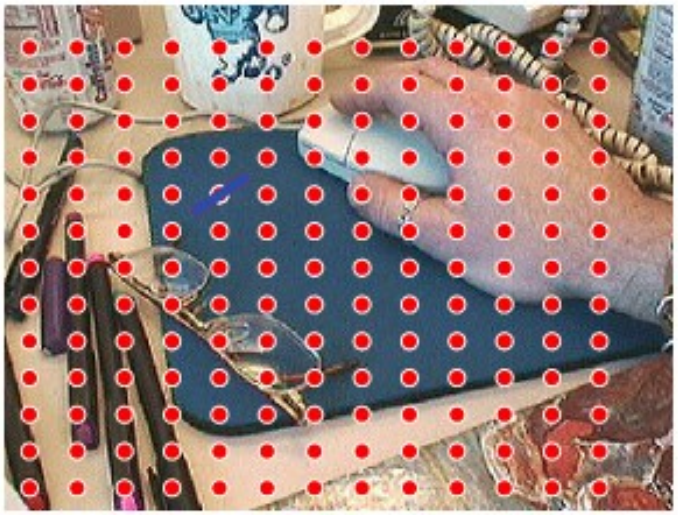


Stroke photo

- Stroke: A short line drawn over the photo
- Strokes are drawn every k pixels
- Strokes drawn at a fixed angle (45 deg.)
- Strokes take color of their origin pixel
- Stroke length is chosen at random

Step 1: Stroke Scan-Conversion

Original photo



Stroke photo

Cool, but jagged edges not cool

Step 2: Edge Detection

Original photo

Edge image

Edge detection step: For every pixel in original photo

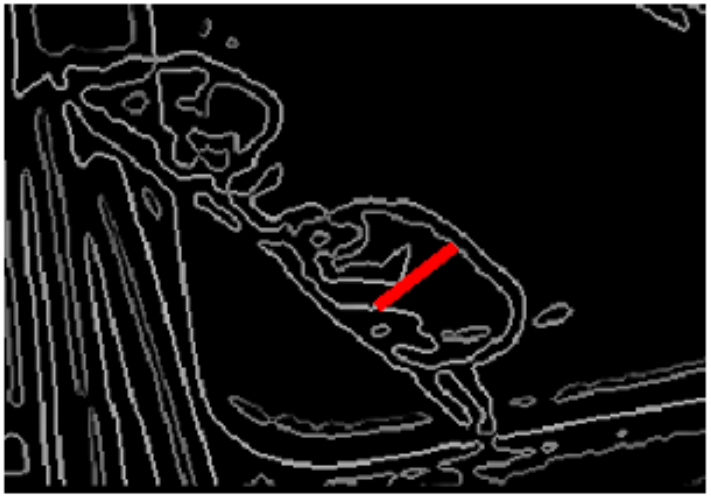
- Compute image gradient at the pixel
- Compute gradient magnitude (in the range 0-255)
- If magnitude > threshold, label pixel as an "edge pixel"
- Compute gradient orientation
- Compute the vector v perpendicular to pixel's gradient

Step 3: Stroke Clipping

Motivation: To avoid "spill-over" artifacts, strokes are clipped at edges detected in the image (i.e., a stroke should not cross an edge pixel)

Original stroke

Clipped stroke



Step 3: Stroke Clipping Results

Original Stroke Photo

Clipped Stroke Photo

Cooler, but still not van Gogh!

Strokes are all oriented: boring

Step 4: Incorporating Edge Orientation

Toss the 45-degree angle strokes

Draw strokes in the direction normal to the gradient!

Clipped Stroke Photo

Oriented Stroke Photo

v.G. would be proud!

Laplacian算子

- The Laplace operator is a scalar operator defined as the dot product (inner product) of two gradient vector operators:

$$\triangle = \nabla \cdot \nabla = \nabla^2 = \left[\frac{\partial}{\partial x_1}, \cdots, \frac{\partial}{\partial x_N}\right] \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \vdots \\ \frac{\partial}{\partial x_N} \end{bmatrix} = \sum_{n=1}^{N} \frac{\partial^2}{\partial x_n^2}$$

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

Laplacian算子

- Laplacian 算子是线性二阶微分算子。即

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

- 对离散的数字图像而言，二阶偏导数可用二阶差分近似，可推导出Laplacian算子表达式为

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

Laplacian算子模板

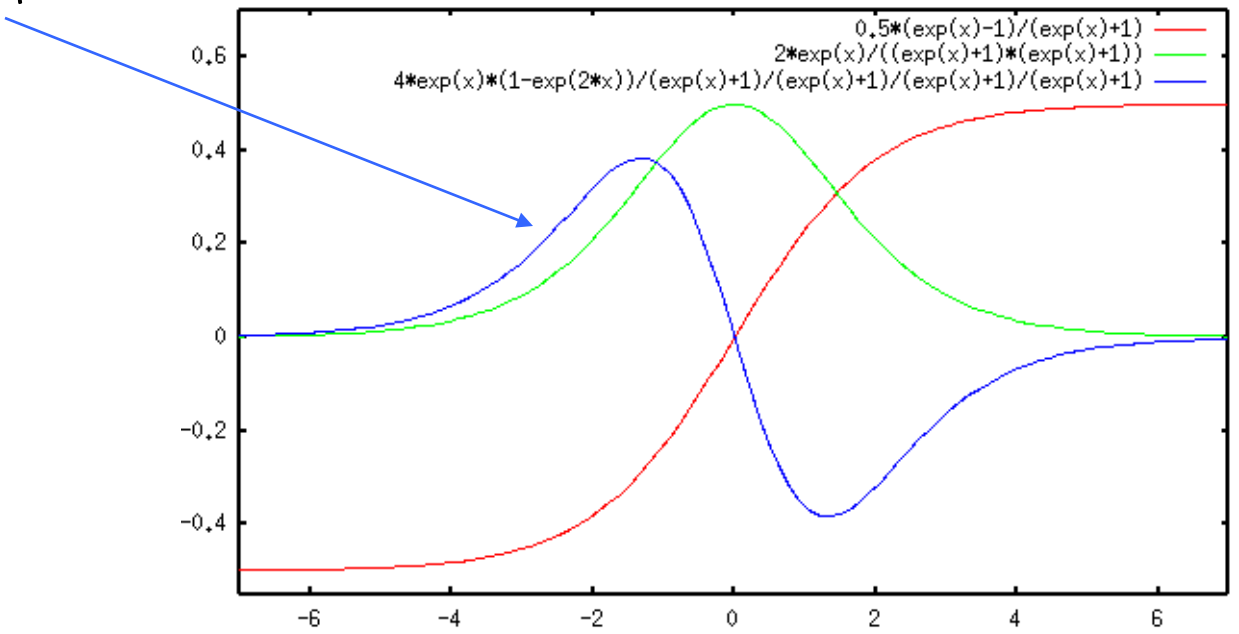
0	1	0	1	1	1
1	−4	1	1	−8	1
0	1	0	1	1	1
0	−1	0	−1	−1	−1
−1	4	−1	−1	8	−1
0	−1	0	−1	−1	−1

a b
c d

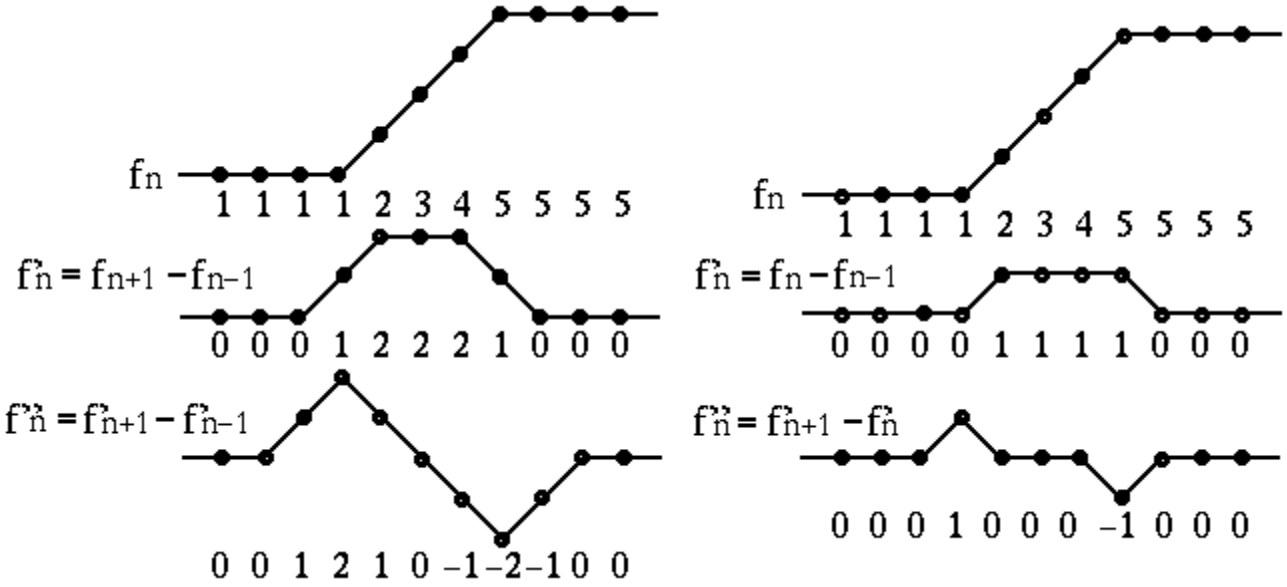
FIGURE 3.39
(a) Filter mask used to implement the digital Laplacian, as defined in Eq. (3.7-4). (b) Mask used to implement an extension of this equation that includes the diagonal neighbors. (c) and (d) Two other implementations of the Laplacian.

Laplacian算子过零点检测

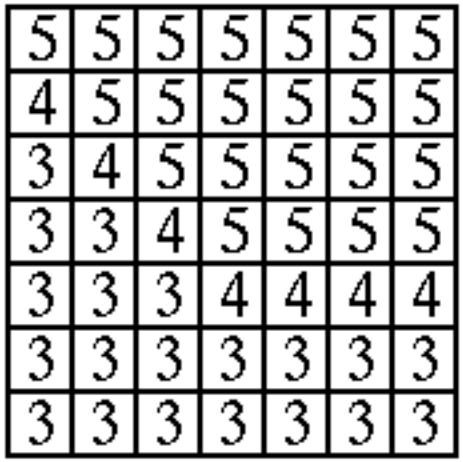
- 梯度算子对于图像亮度的急速变化有很好的响应，但是对于缓慢变化区域则产生较宽的边缘
- Laplacian算子



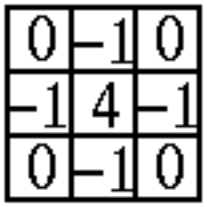
Laplacian算子过零点检测



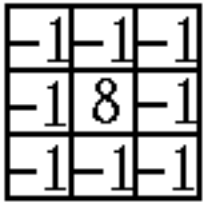
Laplacian算子过零点检测



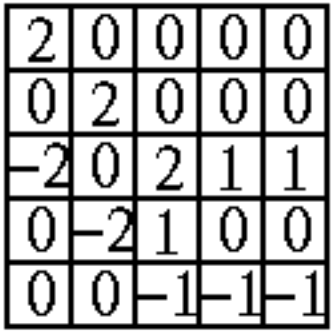
5	5	5	5	5	5	5
4	5	5	5	5	5	5
3	4	5	5	5	5	5
3	3	4	5	5	5	5
3	3	3	4	4	4	4
3	3	3	3	3	3	3
3	3	3	3	3	3	3



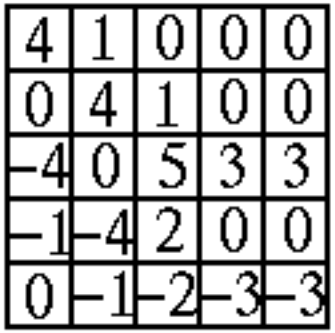
0	−1	0
−1	4	−1
0	−1	0



−1	−1	−1
−1	8	−1
−1	−1	−1

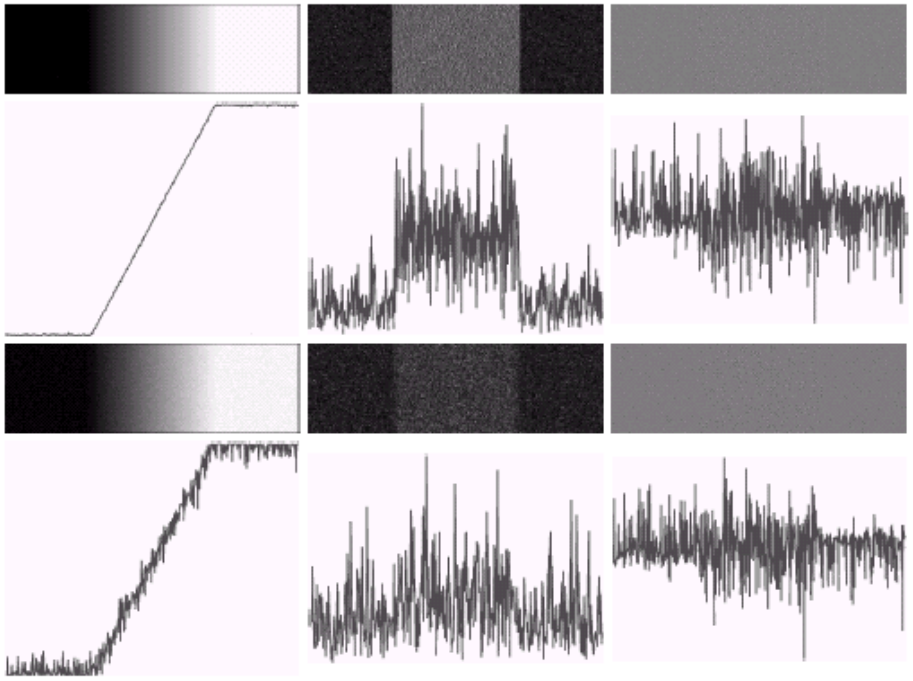


2	0	0	0	0
0	2	0	0	0
−2	0	2	1	1
0	−2	1	0	0
0	0	−1	−1	−1



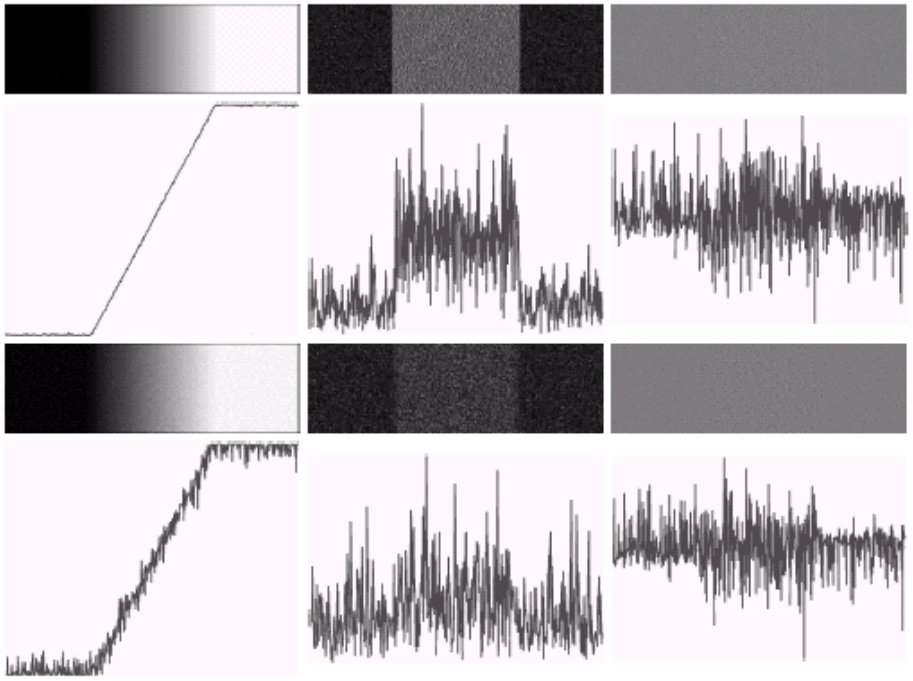
4	1	0	0	0
0	4	1	0	0
−4	0	5	3	3
−1	−4	2	0	0
0	−1	−2	−3	−3

Laplacian of Gaussian (LoG)



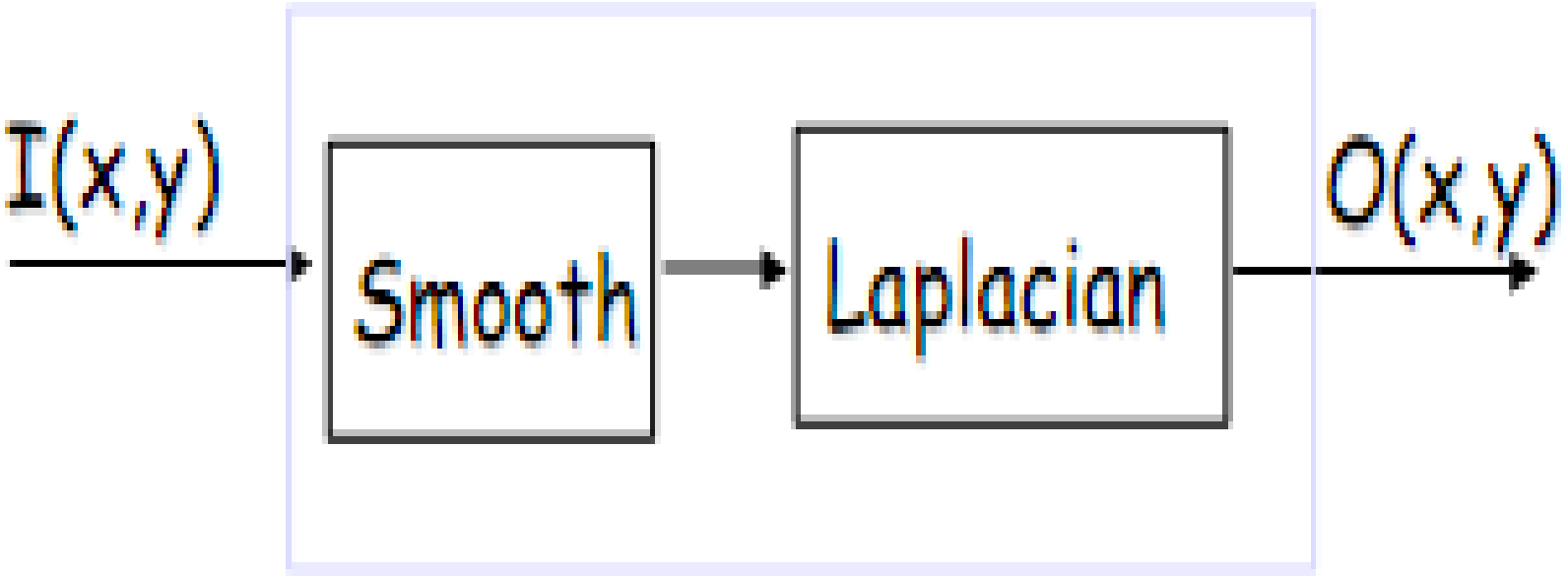
- 图像二阶微分很容易受到噪声的干扰

Laplacian of Gaussian (LoG)

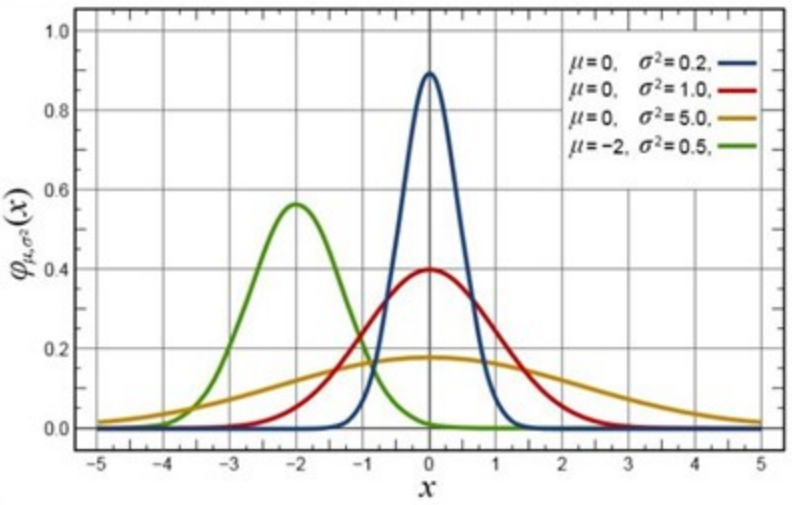


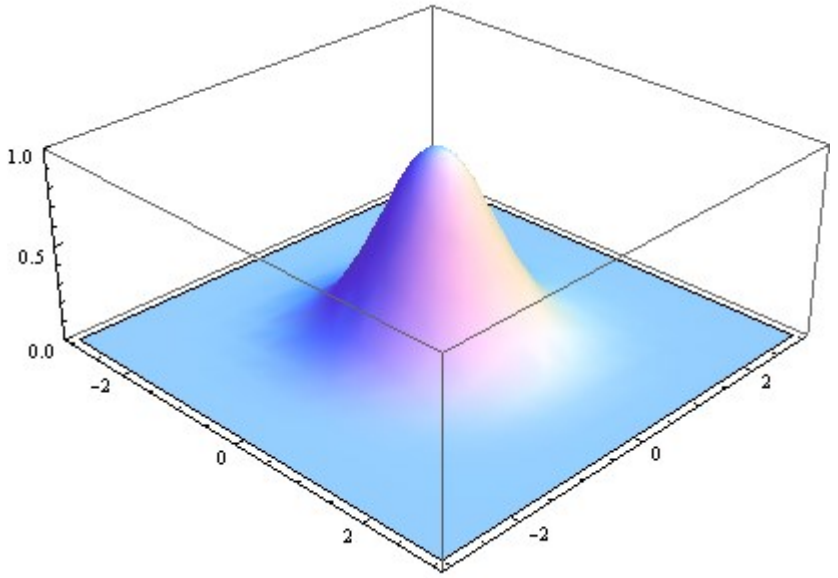
- 图像二阶微分很容易受到噪声的干扰
- 先去除噪声，再进行边缘检测

Laplacian of Gaussian (LoG)



高斯滤波器





$$\frac{1}{2\pi\sigma^2}.e^{\frac{-(x^2+y^2)}{2\sigma^2}}$$

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

$$\begin{bmatrix} 0.0275 & 0.1102 & 0.0275 \\ 0.1102 & 0.4421 & 0.1102 \\ 0.0275 & 0.1102 & 0.0275 \end{bmatrix}$$

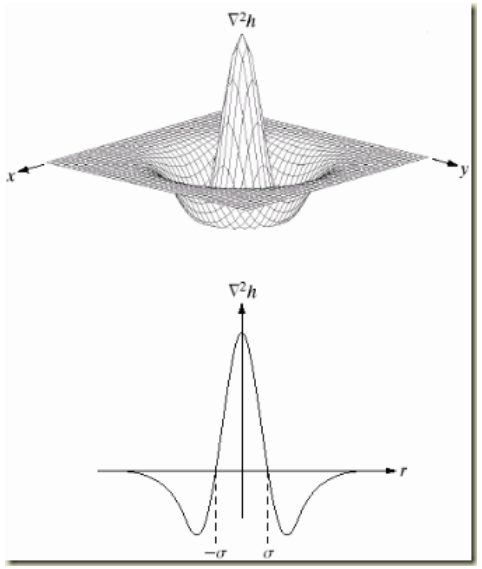
高斯滤波器

Noisy input: PSNR = 39.1 dB　　Gaussian filtered: PSNR = 67.9 dB

Laplacian of Gaussian (LoG)

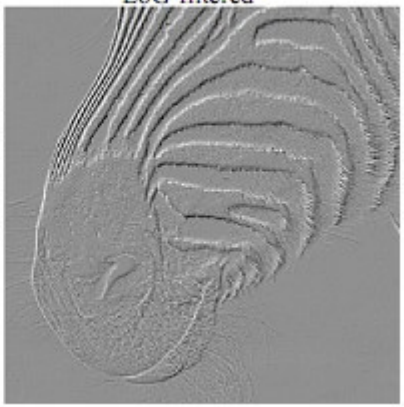
$$\triangle[G_\sigma(x,y) * f(x,y)] = [\triangle G_\sigma(x,y)] * f(x,y) = LoG * f(x,y)$$

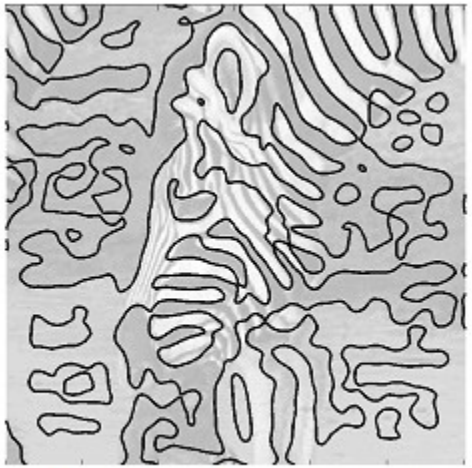
$$LoG \triangleq \triangle G_\sigma(x,y) = \frac{\partial^2}{\partial x^2}G_\sigma(x,y) + \frac{\partial^2}{\partial y^2}G_\sigma(x,y) = \frac{x^2+y^2-2\sigma^2}{\sigma^4}e^{-(x^2+y^2)/2\sigma^2}$$



Original

LoG-filtered

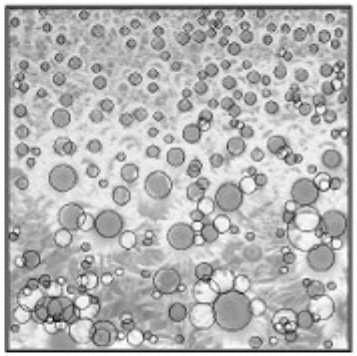




LoG sigma = 8, zero-crossing

Robert Collins
CSE486

Other uses of LoG: Blob Detection



Lindeberg: ``Feature detection with automatic scale selection''. International Journal of Computer Vision, vol 30, number 2, pp. 77--116, 1998.

Robert Collins
CSE486

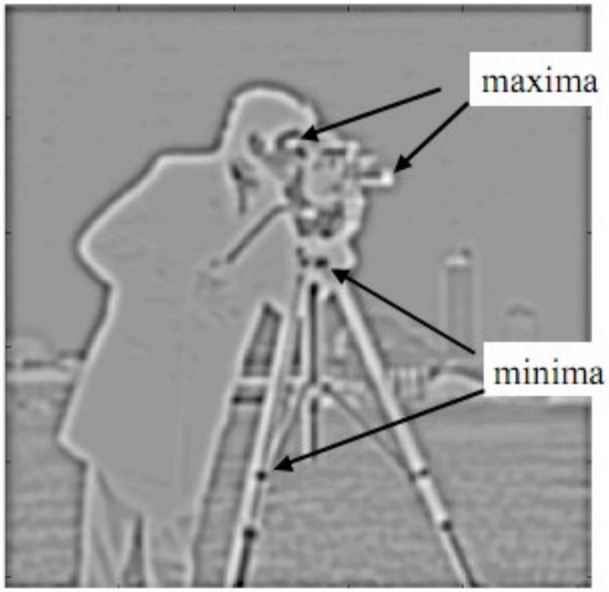
Pause to Think for a Moment:

How can an edge finder also be used to find blobs in an image?

Robert Collins
CSE486

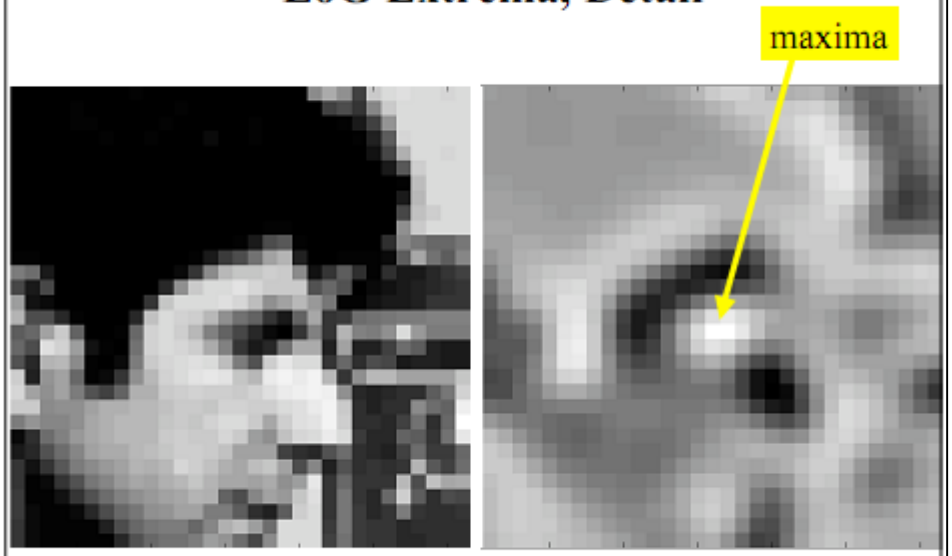
Example: LoG Extrema

LoG
sigma = 2



Robert Collins
CSE486

LoG Extrema, Detail



Robert Collins
CSE486

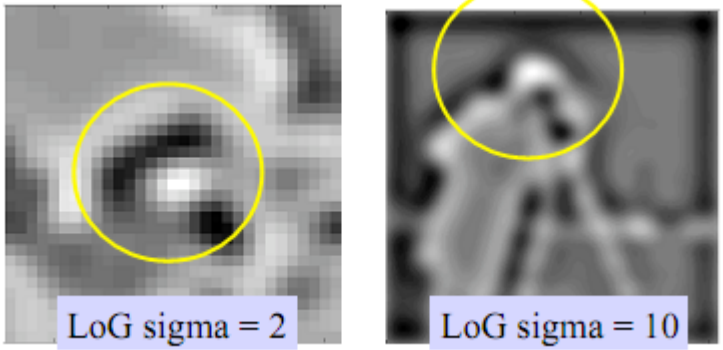
LoG Blob Finding

LoG filter extrema locates "blobs"

maxima = dark blobs on light background

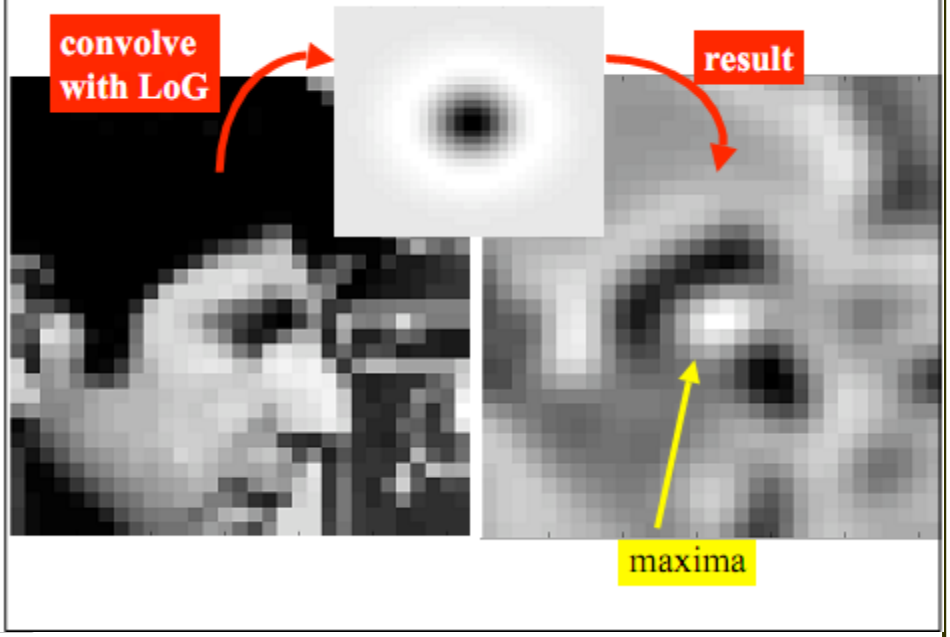
minima = light blobs on dark background

Scale of blob (size ; radius in pixels) is determined by the sigma parameter of the LoG filter.



Robert Collins
CSE486

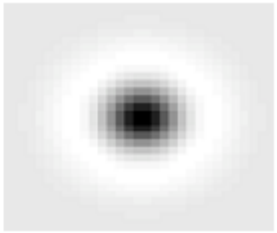
Observe and Generalize



Robert Collins
CSE486

Observe and Generalize

LoG looks a bit like an eye.

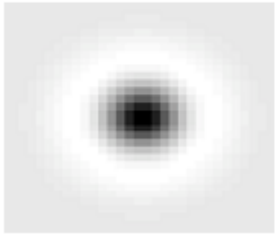


and it responds maximally in the eye region!

Robert Collins
CSE486

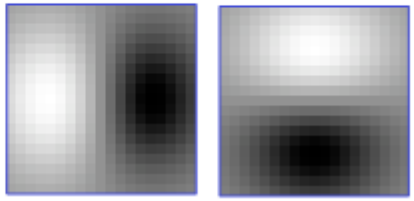
Observe and Generalize

LoG



Looks like dark blob on light background

Derivative of Gaussian



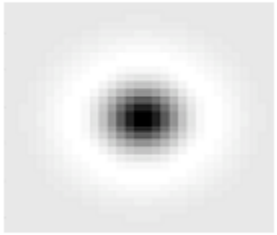
Looks like vertical and horizontal step edges

Recall: Convolution (and cross correlation) with a filter can be viewed as comparing a little "picture" of what you want to find against all local regions in the mage.

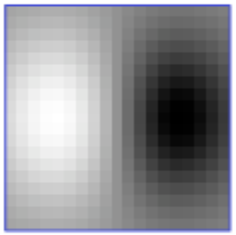
Robert Collins
CSE486

Observe and Generalize

Key idea: Cross correlation with a filter can be viewed as comparing a little "picture" of what you want to find against all local regions in the image.



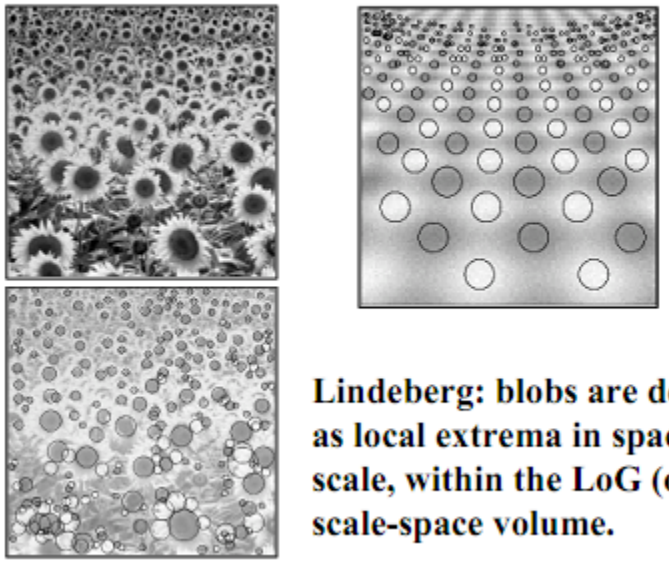
Maximum response:
dark blob on light background
Minimum response:
light blob on dark background



Maximum response:
vertical edge; lighter on left
Minimum response:
vertical edge; lighter on right

Robert Collins
CSE486

Back to Blob Detection

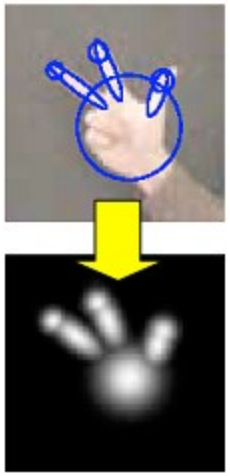


Lindeberg: blobs are detected as local extrema in space and scale, within the LoG (or DoG) scale-space volume.

Robert Collins
CSE486

Other uses of LoG: Blob Detection

- Bretzner, Laptev, Lindeberg "Hand-gesture recognition using multi-scale colour features, hierarchical features and particle filtering", *Proc. Face and Gesture 2002*, Washington DC, pages 423--428. IEEE Computer Society Press. (PDF 159 kb)



Gesture recognition for the ultimate couch potato

Laplacian增强算子

- Laplacian算子作为二阶微分算子，强调的是图像中灰度的突变，并且淡化图像中灰度变化缓慢的区域

- 因此拉普拉斯算子产生的图像是以浅色显示源图像中的边缘和突变点，而以黑色或暗灰色显示原图中的其他背景区域

- 将拉普拉斯算子产生的图像叠加到原图像中可以使得图像中的边界和突变细节得到锐化，同时也保留了原图像中的信息

Laplacian增强算子

- Laplacian增强算子：将拉普拉斯算子产生的图像叠加到原图像

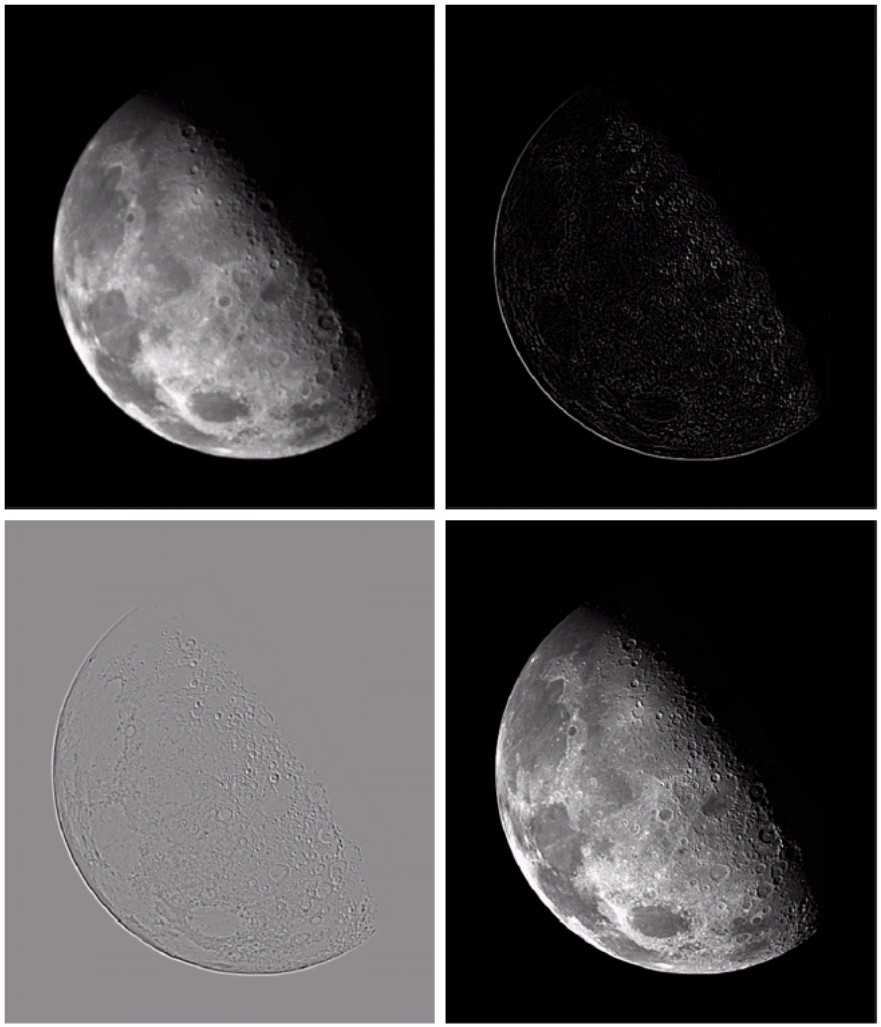
$$g(x,y)=\begin{cases} f(x,y)-\nabla^2 f(x,y) & \text{Laplace掩膜中心系数为负} \\ f(x,y)+\nabla^2 f(x,y) & \text{Laplace掩膜中心系数为正} \end{cases}$$

拉普拉斯增强算子模板

0	-1	0
-1	5	-1
0	-1	0

$$g(x,y)=$$
$$5f(x,y)-[f(x+1,y)+f(x-1,y)+f(x,y+1)+f(x,y-1)]$$

拉普拉斯增强



a b
c d

FIGURE 3.40
(a) Image of the North Pole of the moon.
(b) Laplacian-filtered image.
(c) Laplacian image scaled for display purposes.
(d) Image enhanced by using Eq. (3.7-5).
(Original image courtesy of NASA.)

a)月球北极图像

b) Laplace算子产生的图像可能产生负数

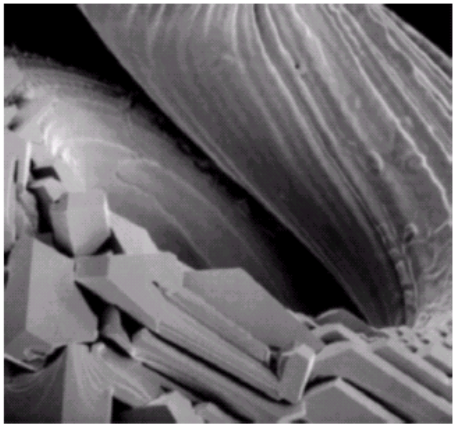
c)图像重新标定后的图像

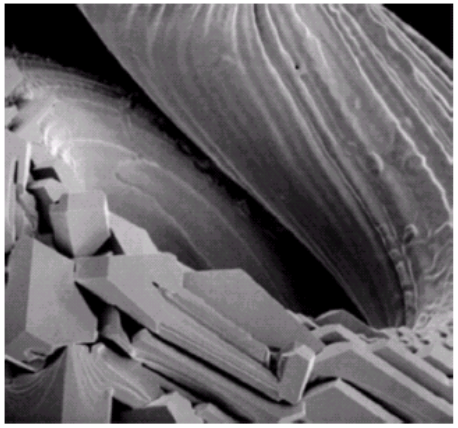
d)增强后的图像

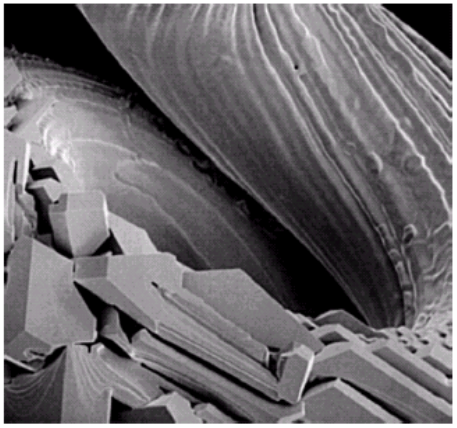
合成拉普拉斯掩膜

0	−1	0
−1	5	−1
0	−1	0

−1	−1	−1
−1	9	−1
−1	−1	−1







a b c
d e

FIGURE 3.41 (a) Composite Laplacian mask. (b) A second composite mask. (c) Scanning electron microscope image. (d) and (e) Results of filtering with the masks in (a) and (b), respectively. Note how much sharper (e) is than (d). (Original image courtesy of Mr. Michael Shaffer, Department of Geological Sciences, University of Oregon, Eugene.)

高频提升滤波

0	−1	0	−1	−1	−1
−1	$A+4$	−1	−1	$A+8$	−1
0	−1	0	−1	−1	−1

a b

FIGURE 3.42 The high-boost filtering technique can be implemented with either one of these masks, with $A \geq 1$.

设锐化图像：$f_s(x,y) = f(x,y) - \overline{f}(x,y) \qquad A \geq 1$

$$\begin{aligned} f_{hb}(x,y) &= Af(x,y) - \overline{f}(x,y) \\ &= (A-1)f(x,y) + f(x,y) - \overline{f}(x,y) \\ &= (A-1)f(x,y) + f_s(x,y) \end{aligned}$$

高频提升滤波

0	−1	0	−1	−1	−1
−1	$A + 4$	−1	−1	$A + 8$	−1
0	−1	0	−1	−1	−1

a b

FIGURE 3.42 The high-boost filtering technique can be implemented with either one of these masks, with $A \geq 1$.

选择拉普拉斯增强选择作为锐化图像：

$$f_s(x,y) = f(x,y) - \nabla^2 f(x,y)$$

$$f_{hb}(x,y) = Af(x,y) - \nabla^2 f(x,y)$$

提升滤波应用之一是处理较暗的输入图像，
通过设置不同的A，起到增亮原图像且锐化细节

高频提升滤波

a b
c d

FIGURE 3.43
(a) Same as Fig. 3.41(c), but darker.
(a) Laplacian of (a) computed with the mask in Fig. 3.42(b) using $A = 0$.
(c) Laplacian enhanced image using the mask in Fig. 3.42(b) with $A = 1$. (d) Same as (c), but using $A = 1.7$.