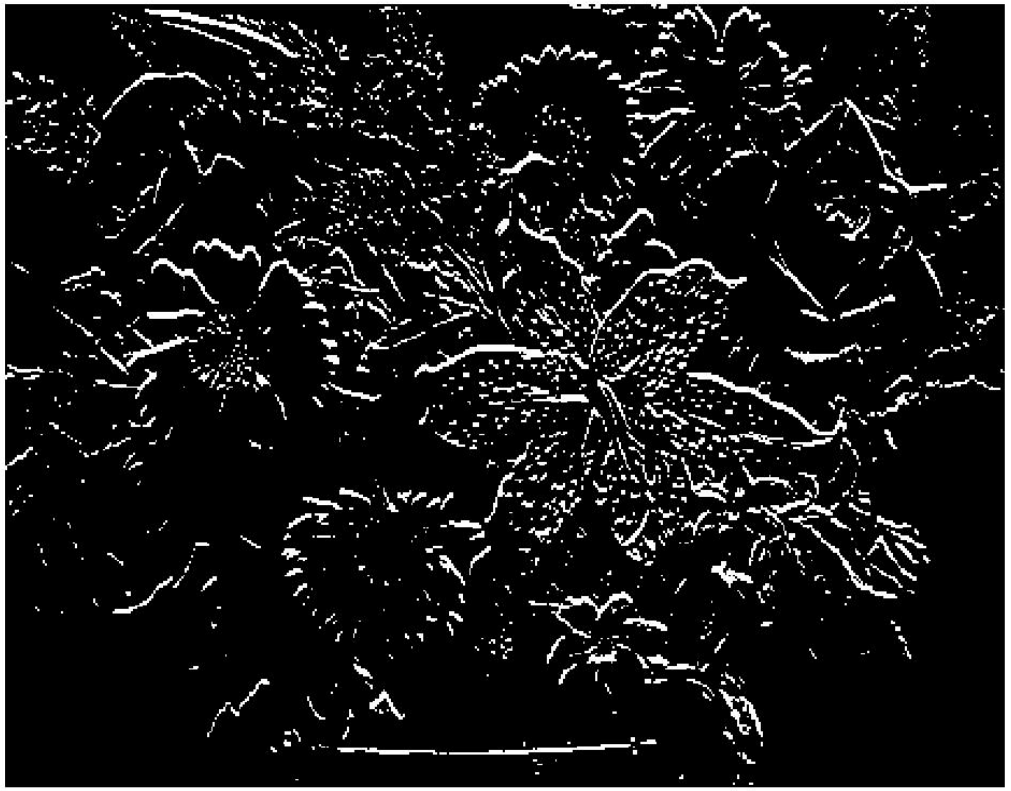
Edge detection

Edge detection

- Goal: identify objects in images
  - but also feature extraction, multiscale analysis, 3D reconstruction, motion recognition, image restoration, registration
- Classical definition of the edge detection problem: localization of *large local changes* in the grey level image → large graylevel *gradients*
  - This definition does not apply to apparent edges, which require a more complex definition
- Contours are very important perceptual cues!
  - They provide a first *saliency map* for the interpretation of image semantics

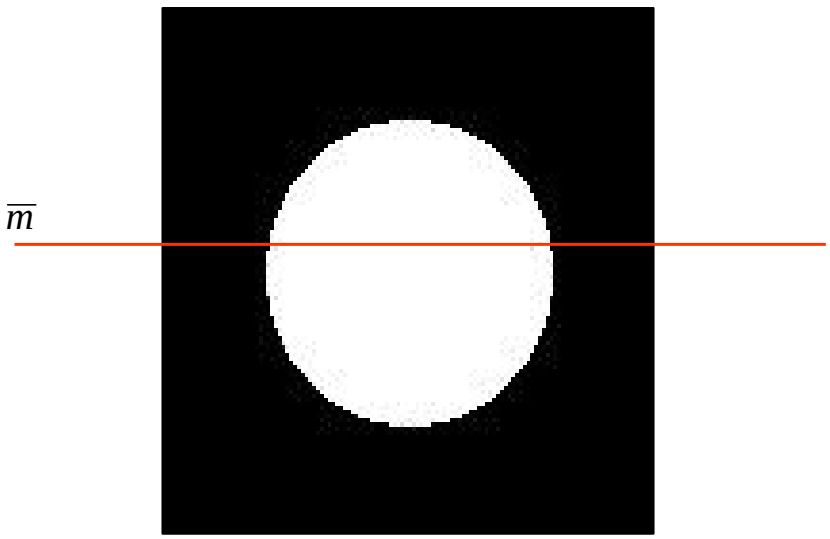
Contours as perceptual cues

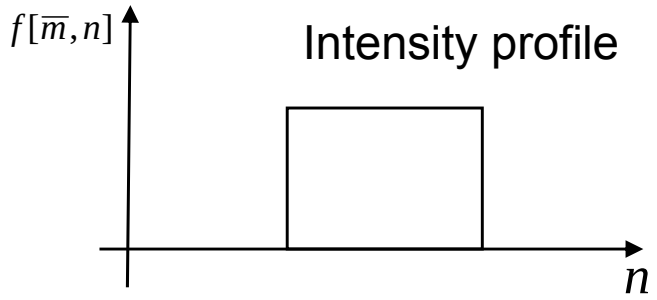


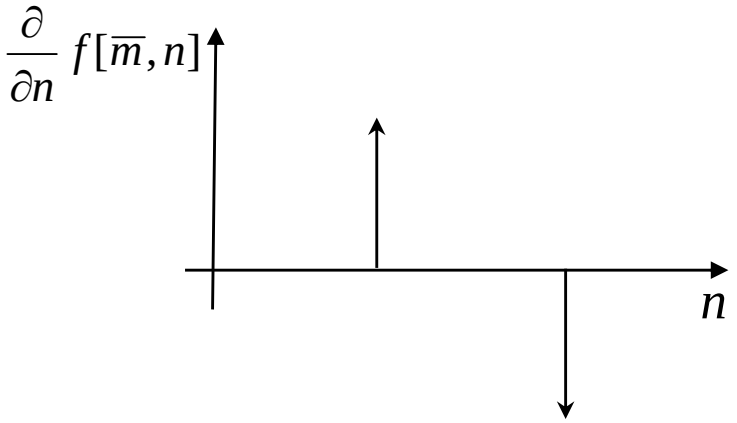
Contours as perceptual cues

Edge detection

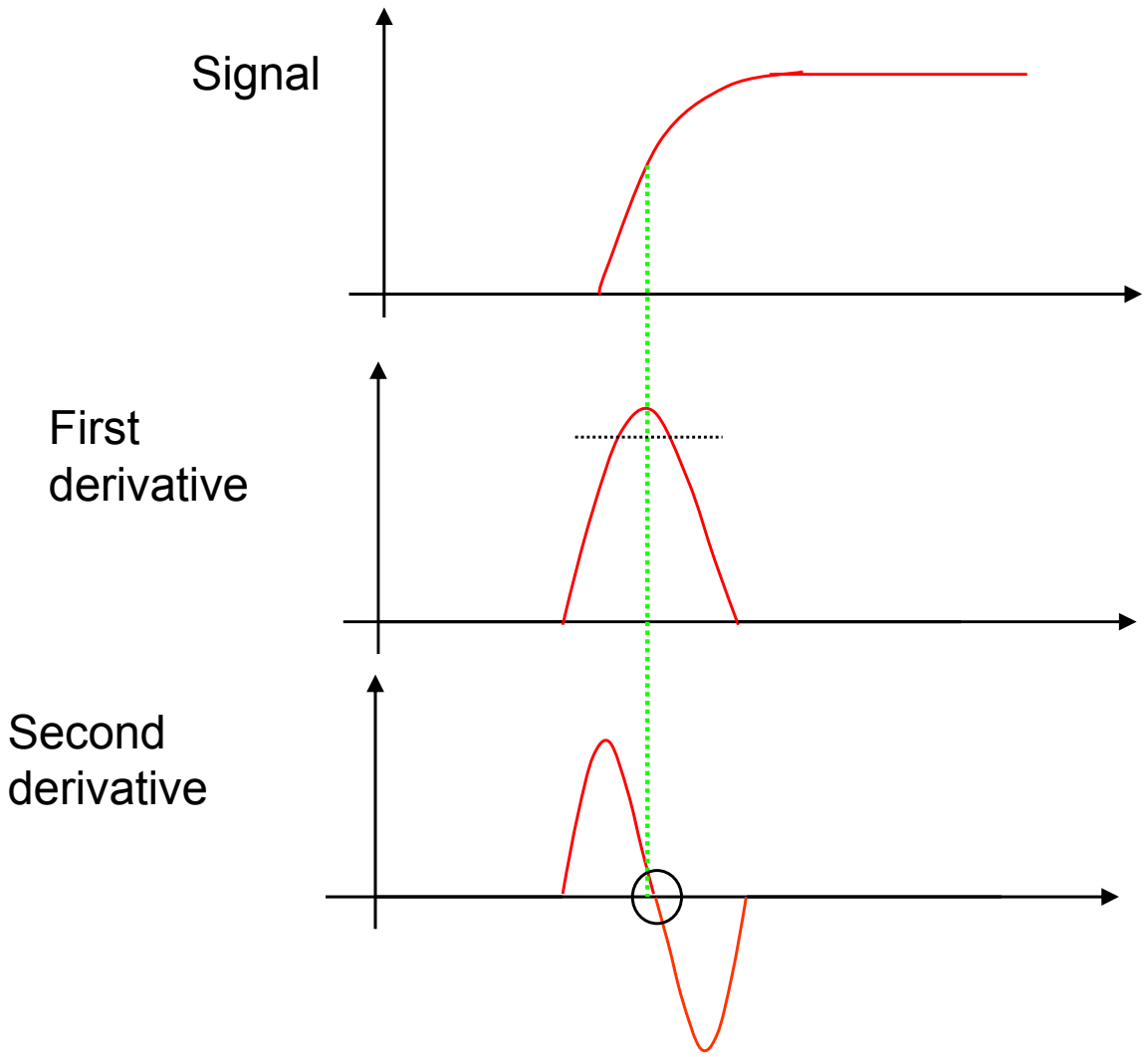
- Image locations with abrupt changes → *differentiation → high pass filtering*







Basic edge detection methods



Gradient thresholding

Zero-crossing of the second order derivative

Types of edge detectors

- *Unsupervised* or autonomous: only rely on local image features
  - No contextual information is accounted for
  - Simple to implement, flexible, suitable for generic applications
  - Not robust
- *Supervised or contextual*: exploit other sources of information
  - Some a-priori knowledge on the semantics of the scene
  - Output of other kinds of processing
  - Less flexible
  - More robust
- There is no golden rule: the choice of the edge detection strategy depends on the application

Edge detection: algorithm

1. Smoothing of the image
   - To reduce the impact of noise and the number of spurious (non meaningful) edges
   - To regularize the differentiation
2. Calculation of first and second order derivatives
   - Isolation of high spatial frequencies
   - Required features: invariance to rotations, linearity
   - Critical point: choice of the scale
3. Labeling
   - Plausibility measure for the detected point belonging to a contour (to get rid of false edges)

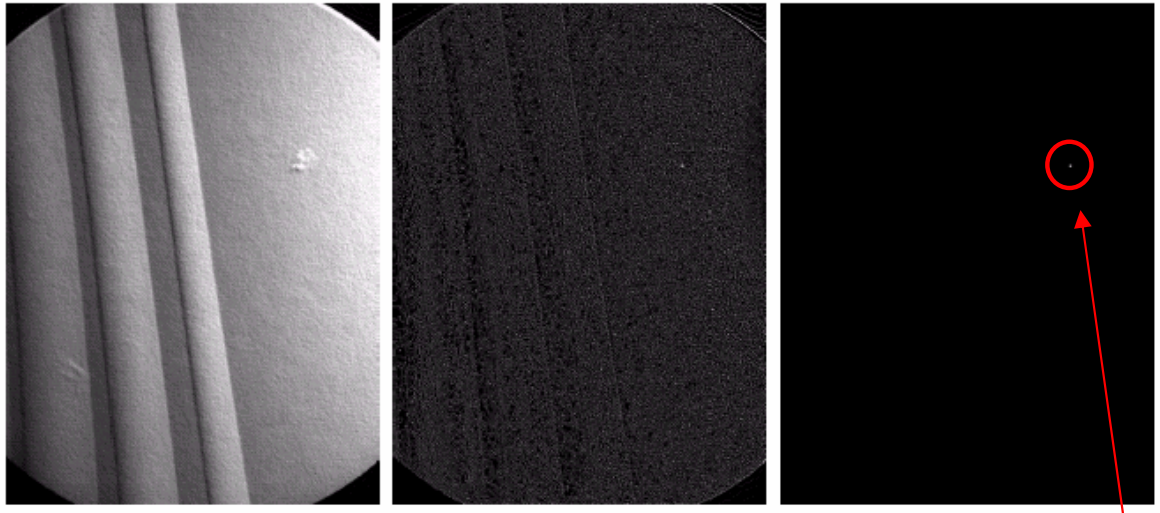
What do we detect?

- Depending on the impulse response of the filter, we can detect different types of graylevel discontinuities
  - Isolate points (pixels)
  - Lines with a predefined slope
  - Generic contours
- However, edge detection implies the evaluation of the local gradient and corresponds to a (directional) derivative

Detection of Discontinuities

- Point Detection

−1	−1	−1
−1	8	−1
−1	−1	−1



Detected point

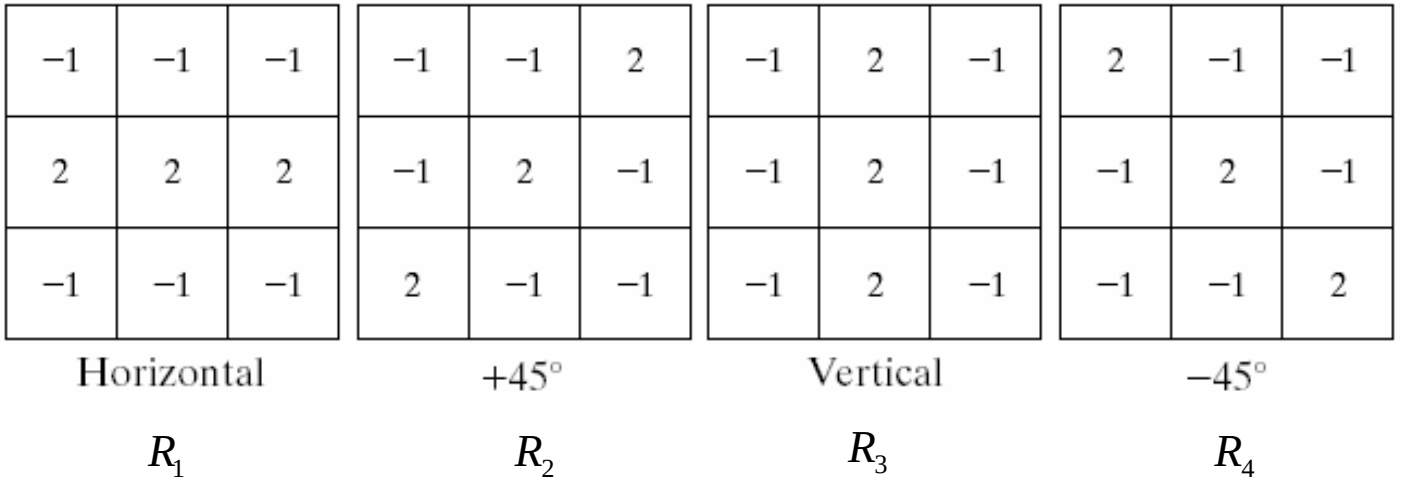
a
b c d

FIGURE 10.2
(a) Point detection mask. (b) X-ray image of a turbine blade with a porosity. (c) Result of point detection. (d) Result of using Eq. (10.1-2). (Original image courtesy of X-TEK Systems Ltd.)

Detection of Discontinuities

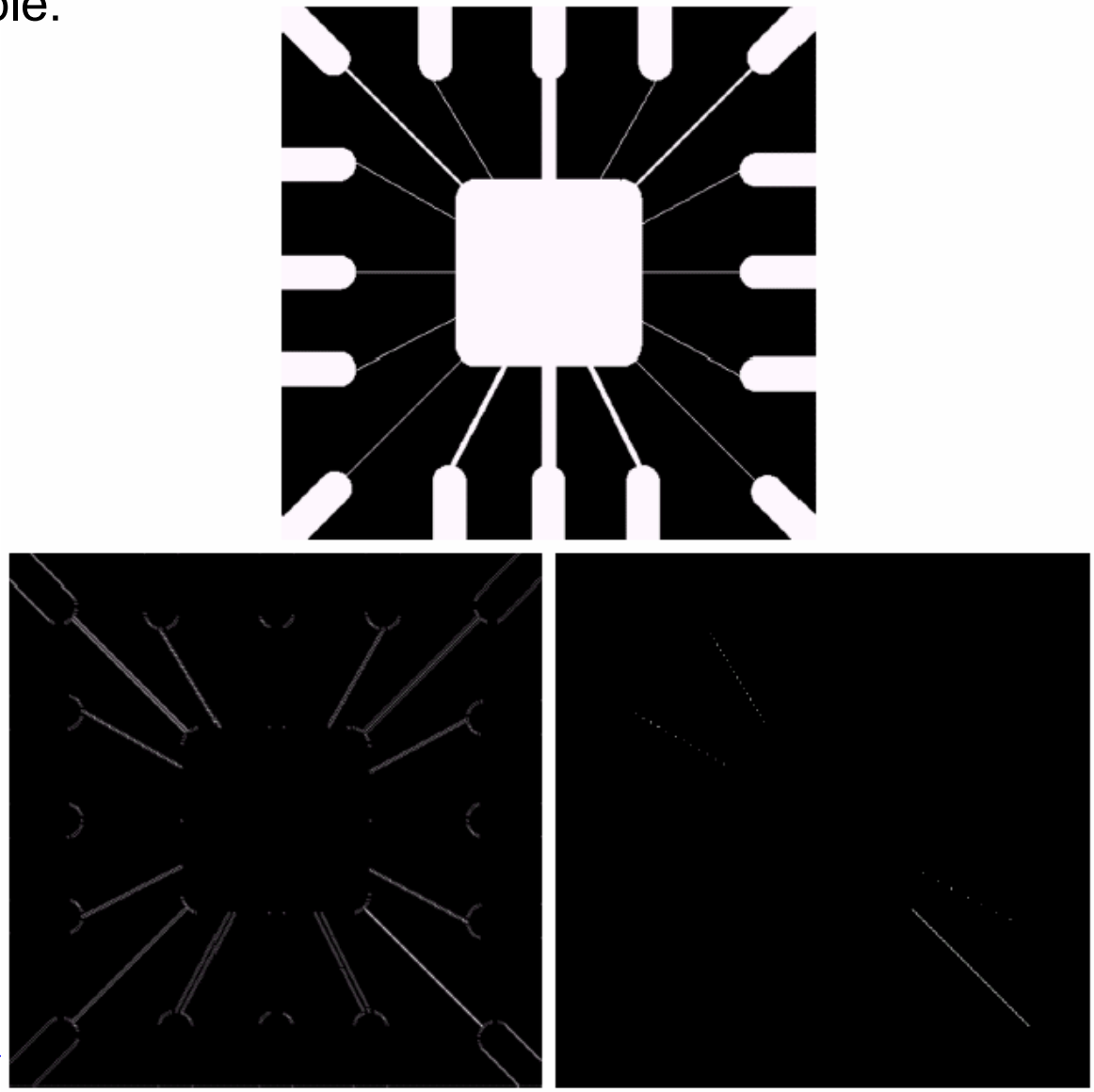
- Line detection

FIGURE 10.3 Line masks.



Detection of Discontinuities

- Line Detection Example:



a
b c

FIGURE 10.4
Illustration of line detection.
(a) Binary wire-bond mask.
(b) Absolute value of result after processing with −45° line detector.
(c) Result of thresholding image (b).

Profiles of image intensity edges

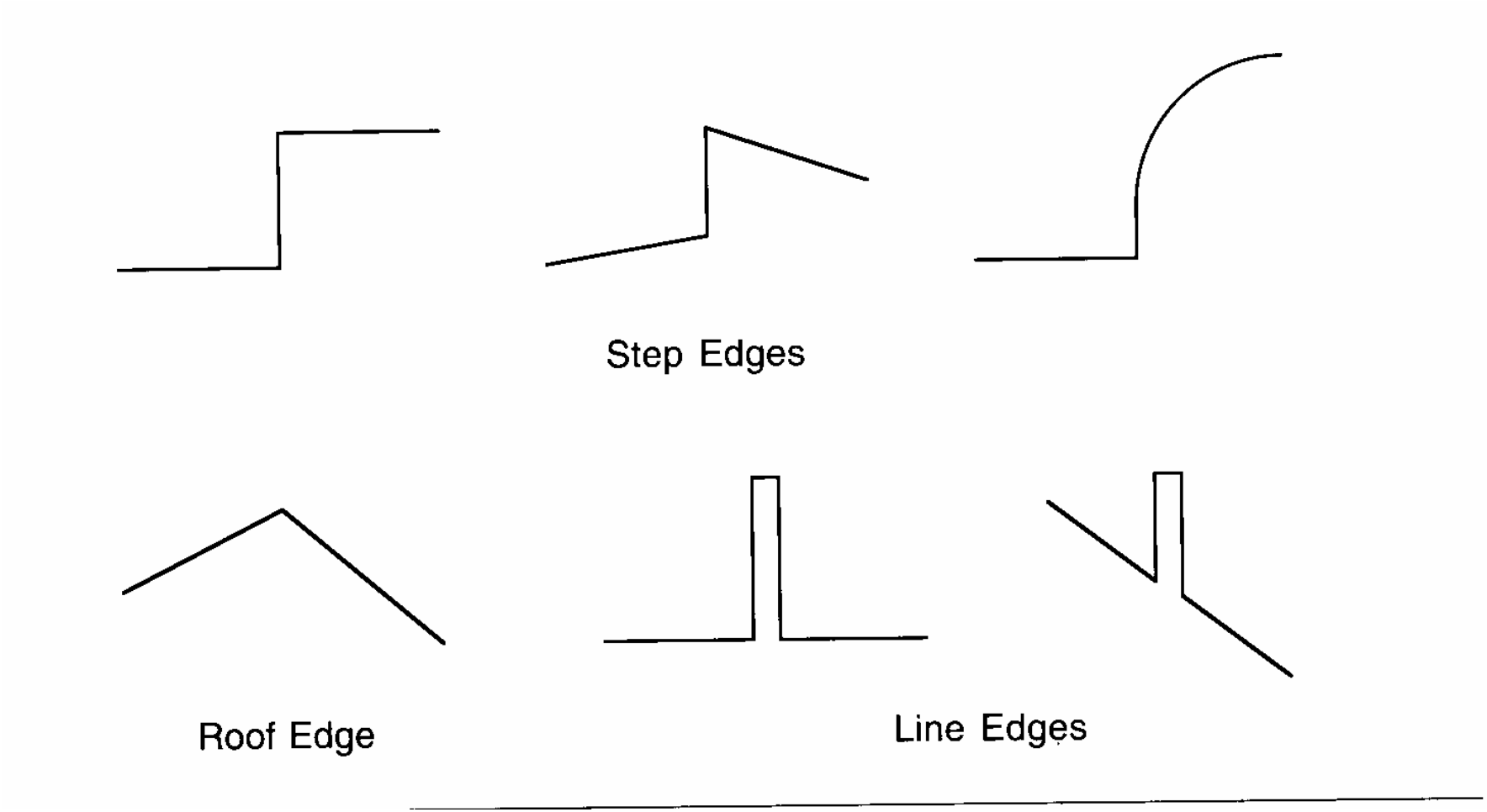


Image gradient

- The *gradient* of an image

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

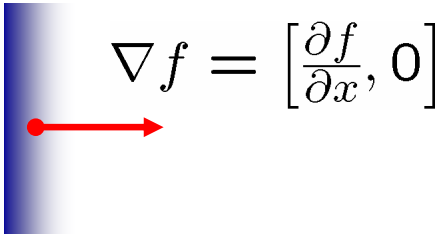
- The gradient *points in the direction of most rapid change in intensity*
- The gradient *direction* is given by

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

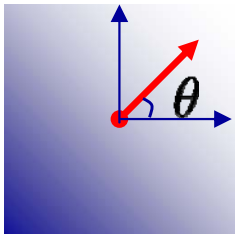
- The *edge strength* is given by the gradient magnitude

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

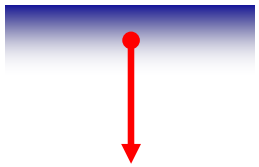
Gradient vector



$$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$$



$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$



$$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$$

Gradient vector

$$G(x,y) = \frac{\partial f}{\partial x}\cos\theta + \frac{\partial f}{\partial y}\sin\theta$$

gradient along the line normal to the edge slope

$$G[j,k] = \left(\left|G_{row}[j,k]\right|^2 + \left|G_{col}[j,k]\right|^2\right)^{1/2}$$

$$G[j,k] = \left|G_{row}[j,k]\right| + \left|G_{col}[j,k]\right|$$

discrete approximation

$$\vartheta[j,k] = \arctan\left\{\frac{G_{col}[j,k]}{G_{row}[j,k]}\right\}$$

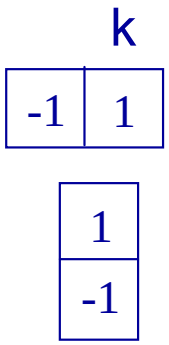
further approximation for computational efficiency

orientation of the spatial gradient with respect to the row axis

Simplest row/col gradient approximations

$$G_{row}[j,k] \cong f[j,k] - f[j,k-1]$$

$$G_{col}[j,k] \cong f[j,k] - f[j+1,k]$$



k

↓

vertical step edge model: a a a a b b b b b

0 0 0 0 h 0 0 0 0

vertical ramp edge model: a a a a c b b b b

0 0 0 0 h/2 h/2 0 0 0

c=(a+b)/2

the edge is not located at the midpoint of the ramp

$$G_{row}[j,k] \cong f[j,k+1] - f[j,k-1]$$

$$G_{col}[j,k] \cong f[j-1,k] - f[j+1,k]$$

0 0 h/2 h h/2 0 0

The discrete gradient

- How can we differentiate a digital image f[x,y]?
  - Option 1: reconstruct a continuous image, then take gradient
  - Option 2: take *discrete derivative* (finite difference)

$$\frac{\partial f[x,y]}{\partial x} = f[x+1,y] - f[x,y]$$

$$\frac{\partial f[x,y]}{\partial y} = f[x,y+1] - f[x,y]$$

  - Discrete approximation

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

Diagonal gradients

- Robert's cross-difference operator

$$G[j,k] = \left(\left| G_1[j,k] \right|^2 + \left| G_2[j,k] \right|^2 \right)^{1/2}$$

$$G[j,k] = \left| G_1[j,k] \right| + \left| G_2[j,k] \right|$$

$$\vartheta[j,k] = \frac{\pi}{4} \arctan \left\{ \frac{G_2[j,k]}{G_1[j,k]} \right\}$$

$$G_1[j,k] = f[j,k] - f[j+1,k+1]$$

$$G_2[j,k] = f[j,k+1] - f[j+1,k]$$

Gradients by convolutions

- The gradient calculation is a neighborhood operation, so it can be put in matrix notations

$$G_{row}[j,k] = f[j,k] * H_{row}[j,k]$$
$$G_{col}[j,k] = f[j,k] * H_{col}[j,k]$$

  - $H_{row/col}$: row and column impulse response arrays

Derivative of convolution

- The derivative of a convolution is equal to the convolution of either of the functions with the derivative of the other

$$h(x) = f(x) * g(x)$$

$$\frac{dh}{dx} = \frac{df}{dx} * g = f * \frac{dg}{dx}$$

- Iterating

$$h(x) = f(x) * g(x)$$

$$\frac{d^2h}{dx^2} = \frac{d}{dx}\left(\frac{df}{dx} * g\right) = \frac{d^2f}{dx^2} * g$$

Intuition

- Intuition (OP)

$$c(t) = f(t) * g(t) = f * g(t) = \int_{-\infty}^{+\infty} f(\tau) g(t-\tau) d\tau$$

$$c'(t) = \frac{dc(t)}{dt} = \frac{d}{dt}\left(f(t) * g(t) \right)$$

$$C(\omega) = \Im\{c(t)\} = \Im\{ f(t) * g(t)\} = F(\omega) G(\omega)$$

$$\Im\{c'(t)\} = j\omega \Im\{c(t)\} = j\omega F(\omega) G(\omega) = \begin{cases} [j\omega F(\omega)] G(\omega) \rightarrow f'(t) * g(t) \\ F(\omega)[j\omega G(\omega)] \rightarrow f(t) * g'(t) \end{cases}$$

Remark

- The *order* in which differentiation and smoothing are performed depends on their properties.
  - Such operations are interchangeable as long as they are linear. Thus, if both smoothing and differentiation are performed by linear operators they are interchangeable
  - In this case they can be performed at the same time by filtering the image with the *differentiation of the smoothing filter*
    - *Laplacian of Gaussian*

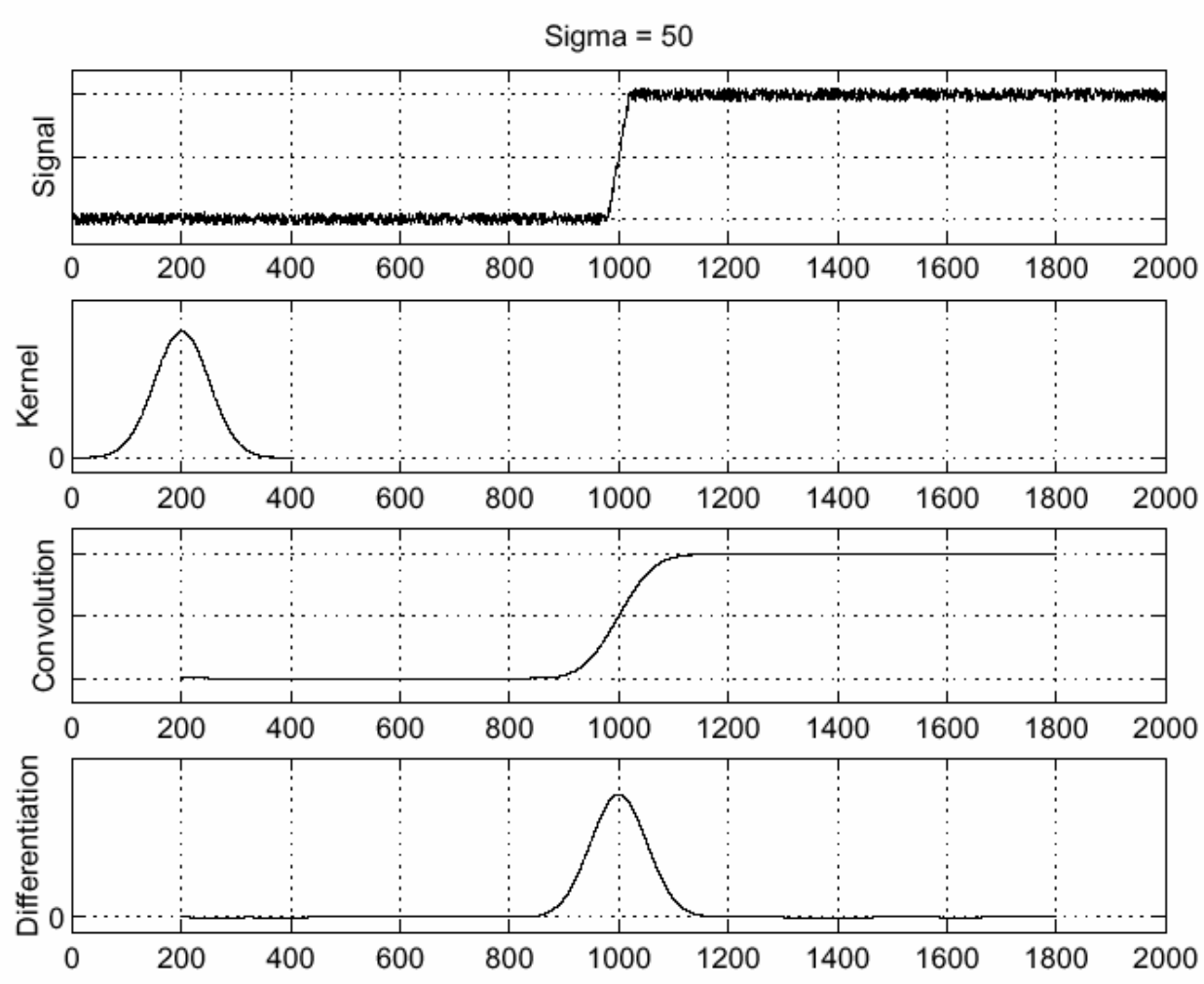
Smoothing+Differentiation

f

h

$h \star f$

$\frac{\partial}{\partial x}(h \star f)$



Look for peaks in $\frac{\partial}{\partial x}(h \star f)$

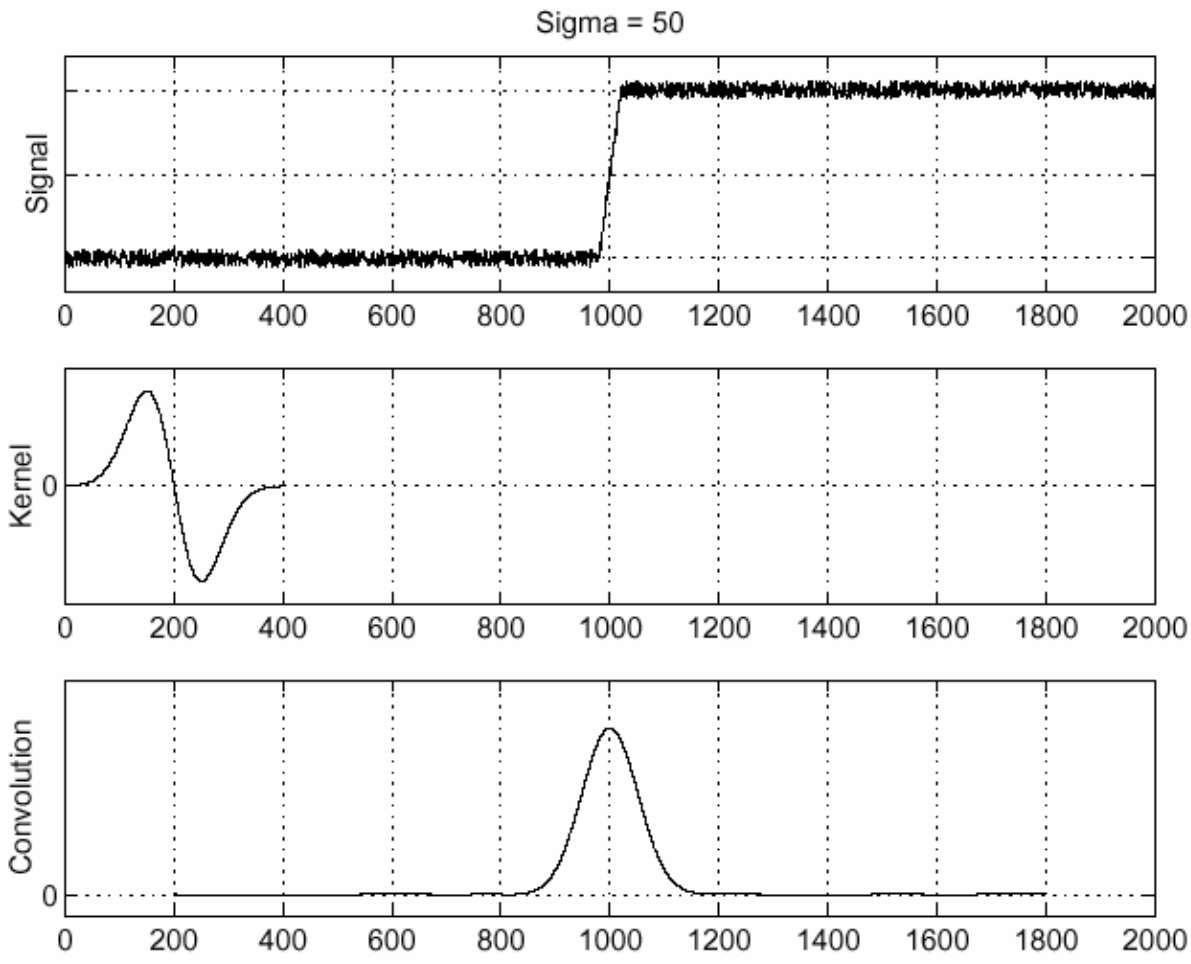
Differentiation of the smoothing filter

$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$

f

$\frac{\partial}{\partial x}h$

$(\frac{\partial}{\partial x}h) \star f$



Gradient filters

- Pixel differences

1	-1

- Symmetric differences

1	0	-1

- Roberts

0	-1
1	0

- Prewitt

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

- Sobel

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

$h_y = h_x^T$

The filter along the y direction is obtained by transposition

Example: Sobel

$S_x = (1/4)$

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

$S_y = (1/4)$

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

where

a_0	a_1	a_2
a_7	(i,j)	a_3
a_6	a_5	a_4

$G_{row}[i,j] = (a0 + c\ a7 + a6) - (a2 + c\ a3 + a4)$

$G_{col} = (a6 + c\ a5 + a4) - (a0 + c\ a1 + a2)$

$c = 2$

$$G = \sqrt{G_{row}^2 + S_{col}^2}$$

Sobel extentions

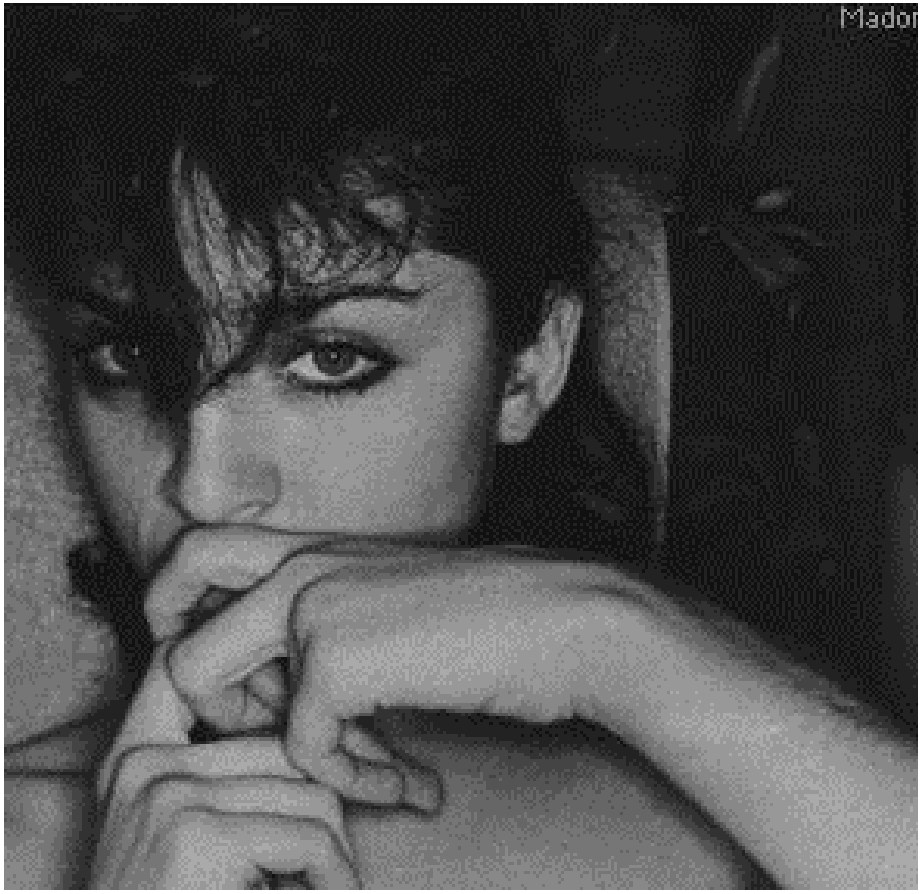
Sobel 7x7,
truncated pyramid

$$S_x = k \begin{bmatrix} 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 2 & 2 & 0 & -2 & -2 & -1 \\ 1 & 2 & 3 & 0 & -3 & -2 & -1 \\ 1 & 2 & 3 & 0 & -3 & -2 & -1 \\ 1 & 2 & 3 & 0 & -3 & -2 & -1 \\ 1 & 2 & 2 & 0 & -2 & -2 & -1 \\ 1 & 1 & 1 & 0 & -1 & -1 & -1 \end{bmatrix}$$

Sobel 7x7

$$S_y = \begin{bmatrix} -1 & -1 & -1 & -2 & -1 & -1 & -1 \\ -1 & -1 & -1 & -2 & -1 & -1 & -1 \\ -1 & -1 & -1 & -2 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 2 & 1 & 1 & 1 \\ 1 & 1 & 1 & 2 & 1 & 1 & 1 \\ 1 & 1 & 1 & 2 & 1 & 1 & 1 \end{bmatrix}$$

Example



Original

Sobel filtered

Prewitt

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad S_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \qquad c = 1$$

- Kirsch operator
  - 8 directional masks, each selecting one specific direction
  - “winner takes all” paradigm for the absolute value of the gradient and direction selected by the index of the corresponding mask
- Robinson operator
  - 8 directional masks, similar to Kirsh

Directional masks

$S_1 =$

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

$S_2 =$

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

$S_3 =$

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

$S_4 =$

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

$S_5 =$

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

$S_6 =$

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

$S_7 =$

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

$S_8 =$

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

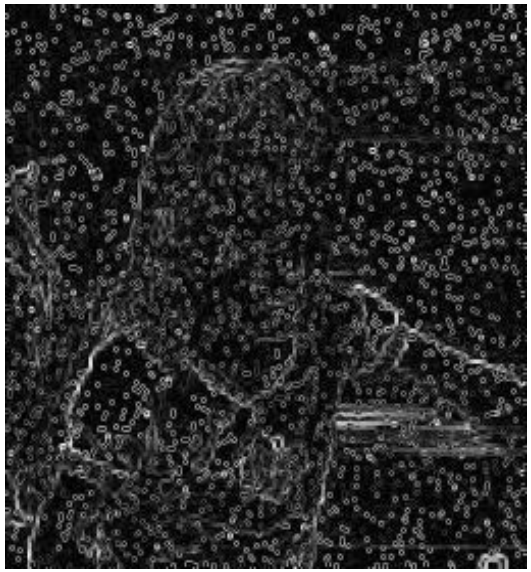
Sobel

Prewitt

Roberts

Kirsch

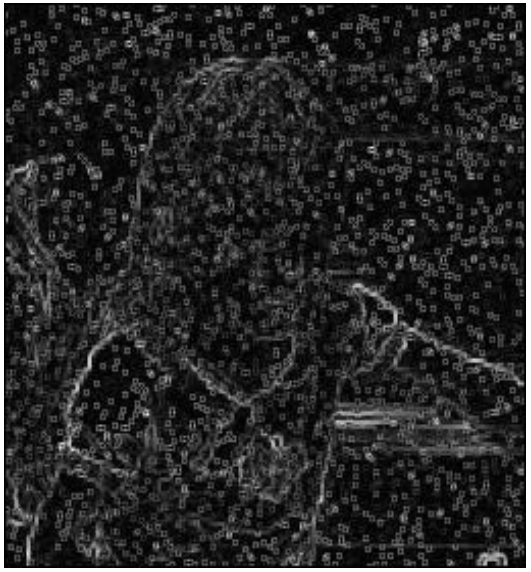
Robinson



Sobel 3x3

Sobel 7x7

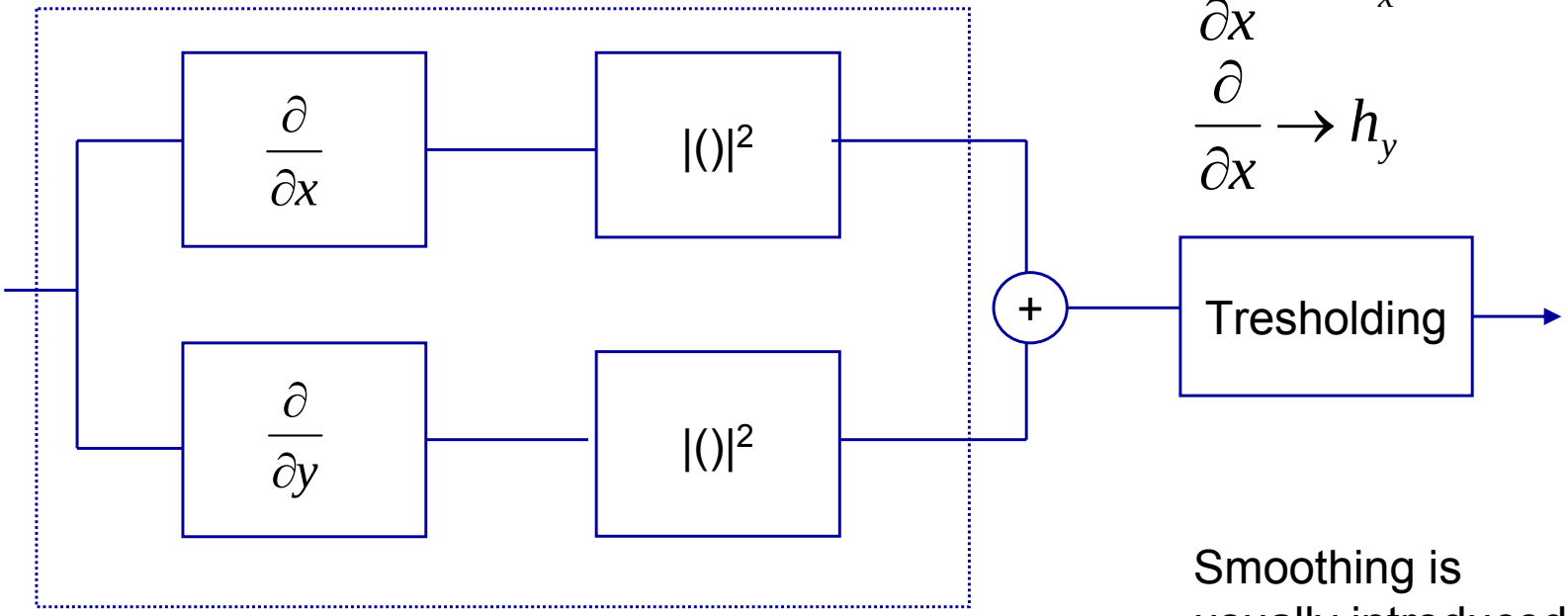
Prewitt 3x3



Prewitt 7x7

Gradient thresholding

Modulus of the gradient thresholding

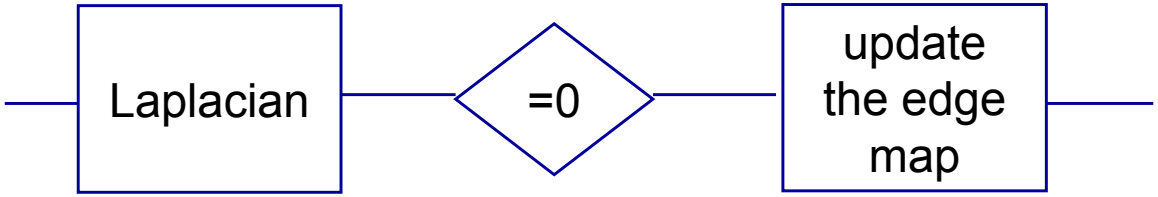


$\frac{\partial}{\partial x} \rightarrow h_x$

$\frac{\partial}{\partial x} \rightarrow h_y$

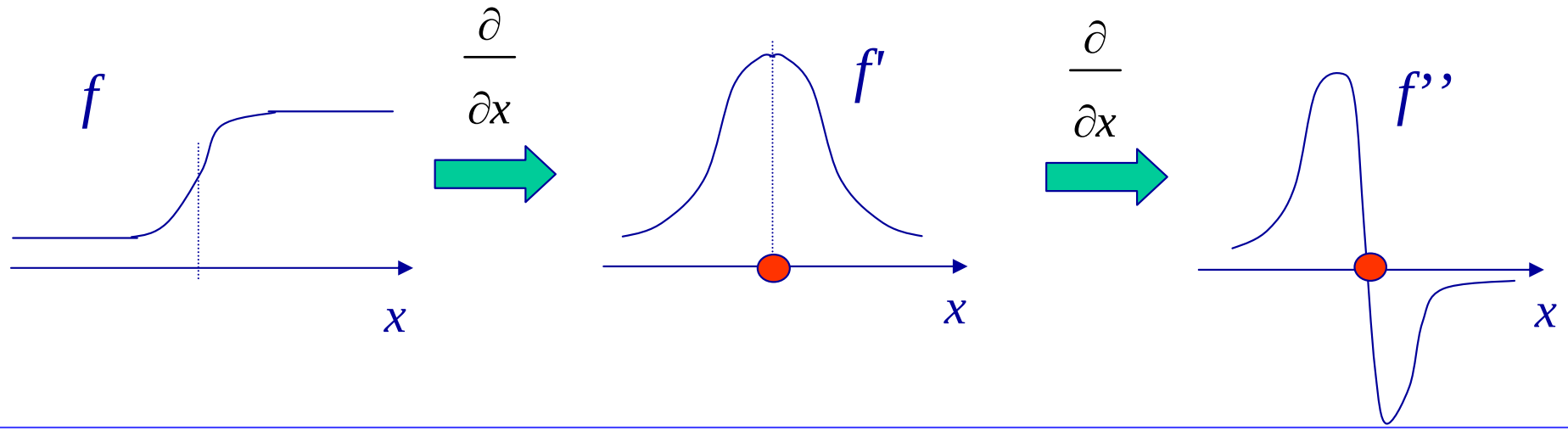
Smoothing is usually introduced either before or after the filtering

Laplacian zero-crossing



Second order derivative

- Edge detectors based on first order derivative are not robust
  - High sensitivity to noise, need a threshold
- Second order derivative operators detect the edge at the zero-crossing of the second derivative → more robust, more precise
  - Less sensitive to noise, don't need a threshold



Laplace operator

- Second order differentiation operator

$$\Delta f = \nabla^2 f = \nabla \cdot \nabla f$$

$$\nabla^2 f = \sum_{i=1}^{N} \frac{\partial^2 f}{\partial x_i^{\,2}}$$

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

- Directional derivative

$$D_{\vec{v}} f(\vec{x}) = \sum_{i=1}^{N} v_i \frac{\partial f}{\partial x_i}$$

$$v_i = \left\langle \vec{v}, \vec{i} \right\rangle$$

Laplace operator

- Second order derivative in the continuous domain

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

- Discrete approximation

$$\frac{\partial^2 f}{\partial x^2} = \frac{\partial G_x}{\partial x} = \frac{\partial}{\partial x}\left[f(i, j+1) - f(i, j)\right] =$$

$$= \frac{\partial f(i, j+1)}{\partial x} - \frac{\partial f(i, j)}{\partial x} =$$

$$= \left[f(i, j+2) - f(i, j+1)\right] - \left[f(i, j+1) - f(i, j)\right] =$$

$$= f(i, j+2) - 2f(i, j+1) + f(i, j)$$

Discrete approximation: proof

- Centering the estimation on (i,j), the simplest approximation is to compute the difference of slopes along each axis

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

$$G_{row}[j,k] = \left(f[j,k] - f[j,k-1]\right) - \left(f[j,k+1] - f[j,k]\right) = 2f[j,k] - f[j,k-1] - f[j,k+1]$$

$$G_{col}[j,k] = \left(f[j,k] - f[j+1,k]\right) - \left(f[j-1,k] - f[j,k]\right) = 2f[j,k] - f[j+1,k] - f[j-1,k]$$

- This can be put in operator and matrix form as

$$G[j,k] = f[j,k] * H[j,k]$$

$$H = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Discrete approximation

- The 4-neighbors Laplacian is often normalized to provide unit gain averages of the positive and negative weighted pixels in the 3x3 neighborhood
- Gain normalized 4-neighbors Laplacian

$$H = \frac{1}{4}\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

- The weights of the pixels in the neighborhood, and thus the normalization coefficient, can be changed to emphasize the edges. Ex. Prewitt modified Laplacian

$$H = \frac{1}{8}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Discrete approximation

– Gain normalized separable 8 neighbors Laplacian

$$H = \frac{1}{8}\begin{bmatrix} -2 & 1 & -2 \\ 1 & 4 & 1 \\ -2 & 1 & -2 \end{bmatrix}$$

$$a \quad a \quad a \quad a \quad b \quad b \quad b$$

$$0 \quad 0 \quad 0 \quad -\frac{3}{8}h \quad \frac{3}{8}h \quad 0 \quad 0$$

$$a \quad a \quad a \quad c \quad b \quad b \quad b$$

$$0 \quad 0 \quad -\frac{3}{16}h \quad 0 \quad \frac{3}{16}h \quad 0 \quad 0$$

Other formulation

- Without sign change after the evaluation of the Laplacian
  - However, the sign is meaningless if we evaluate the modulus of the gradient

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

$$\frac{\partial^2 f}{\partial y^2} = f(i+1, j) - 2f(i, j) + f(i-1, j)$$

- Different possible Laplacian matrices

$$\nabla^2 = \begin{array}{|c|c|c|}\hline 0 & 1 & 0 \\ \hline 1 & -4 & 1 \\ \hline 0 & 1 & 0 \\ \hline \end{array} \qquad \begin{array}{|c|c|c|}\hline 1 & 4 & 1 \\ \hline 4 & -20 & 4 \\ \hline 1 & 4 & 1 \\ \hline \end{array} \qquad \nabla^2 = \begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\ \hline -1 & 4 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array} \qquad \begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline -1 & 8 & -1 \\ \hline -1 & -1 & -1 \\ \hline \end{array}$$

Remarks

- Quite often the zero crossing does not happen at a pixel location
  - See the example of the step edge
- It is common choice to locate the edge at a pixel with a positive response having a neighbor with a negative response
- Laplacian of Gaussian: Marr&Hildrith have proposed an operator in which Gaussian shaped smoothing is performed prior to the application of the Laplacian

➢ Continuous LoG gradient

$$LOG(x,y) = -\nabla^2 \{ f(x,y) * H_S(x,y) \}$$

$$H_S(x,y) = g(x,s)g(y,s)$$

$$g(x,s) = \frac{1}{\sqrt{2\pi s^2}} \exp\left\{ -\frac{1}{2}\left(\frac{x}{s}\right)^2 \right\}$$

impulse response of the Gaussian smoothing kernel

LoG operator

- As a result of the linearity of the second derivative operator and of the convolution

$$LOG[j,k] = f[j,k] * H[j,k] \qquad (1)$$

$$H(x,y) = -\nabla^2 \{g(x,s)g(y,s)\} =$$

$$= \frac{1}{\pi s^4}\left(1 - \frac{x^2+y^2}{2s^2}\right)\exp\left\{-\frac{x^2+y^2}{2s^2}\right\}$$

- It can be shown that
  - The convolution (1) can be performed separately along rows and cols
  - It is possible to approximate the LOG impulse response closely by a difference of Gaussians (DOG) operator

$$H(x,y) = g(x,s_1)g(y,s_1) - g(x,s_2)g(y,s_2), \qquad s_1 < s_2$$

The LoG operator

$$g(x,y) = \frac{1}{2\pi\sigma^2}\exp\left[-\frac{x^2+y^2}{2\sigma^2}\right]$$

$$h(x,y) = \nabla^2\left[g(x,y) * f(x,y)\right] = \left[\nabla^2 g(x,y)\right] * f(x,y)$$

where

$$\nabla^2 g(x,y) = \frac{x^2+y^2-2\sigma^2}{2\pi\sigma^4}\exp\left[-\frac{x^2+y^2}{2\sigma^2}\right] \quad \text{mexican hat}$$

- How to choose σ?
  - Large values: pronounced smoothing → better denoising BUT smear out sharp boundaries reducing the precision in edge localization
  - Small values: soft smoothing → lower noise reduction BUT better boundary preservation
  - A good solution could be to follow a multiscale approach (σ is the scale)

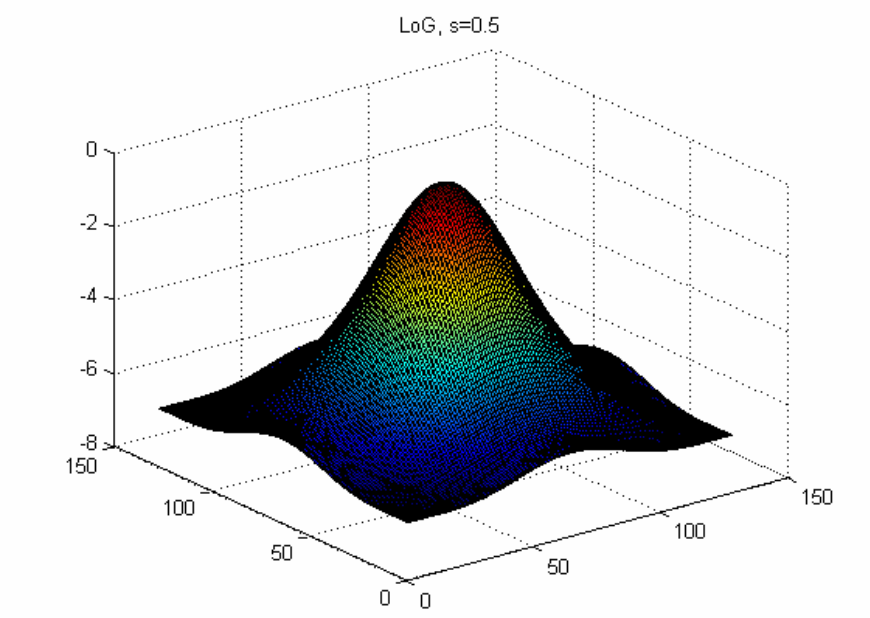
LoG filtering

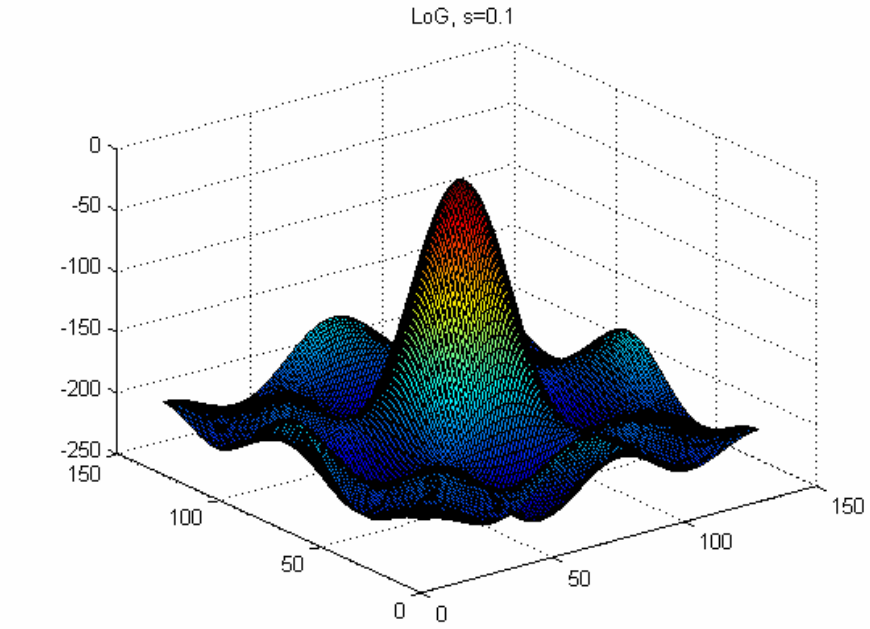
- Gaussian smoothing (low-pass filter)
  - Noise reduction (the larger the filter, the higher the smoothing)
  - BUT
    - Smears out edges
    - Blurs the image (defocusing)
- Laplacian detection (high-pass filter)
- Edge location by interpolation
  - The zero-crossing does not happen in a pixel site

LoG filtering = Gaussian smoothing + Laplacian detection

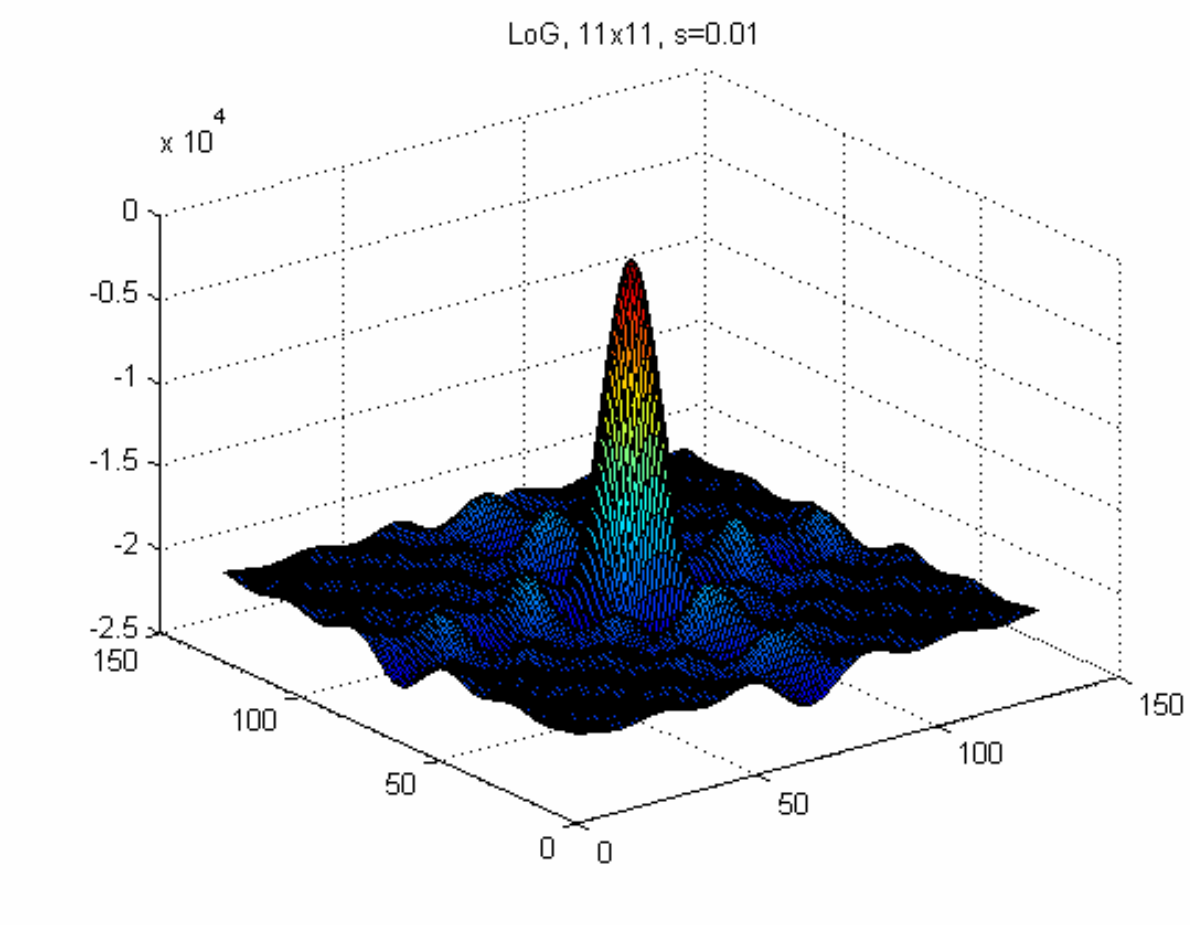
5x5 LoG

$$H[j,k] = \begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}$$





11x11 LoG

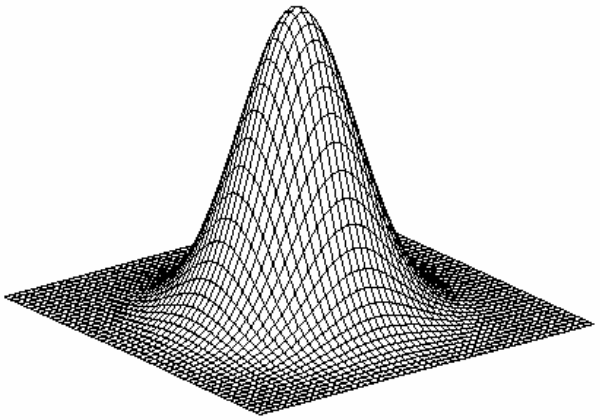


LoG

- Independent variables
  - s value: larger values allow larger denoising but smear out details and made contour extraction not quite precise
- Solutions
  - Trade off
  - Multiscale

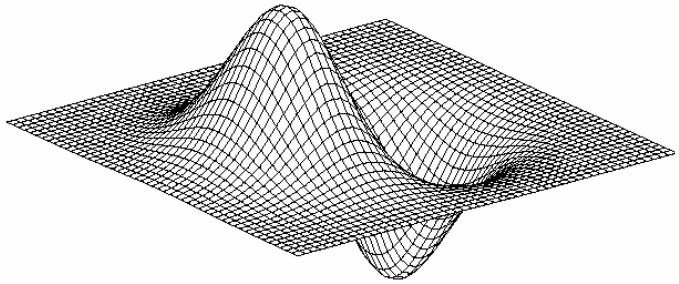
2D edge detection filters

Gaussian



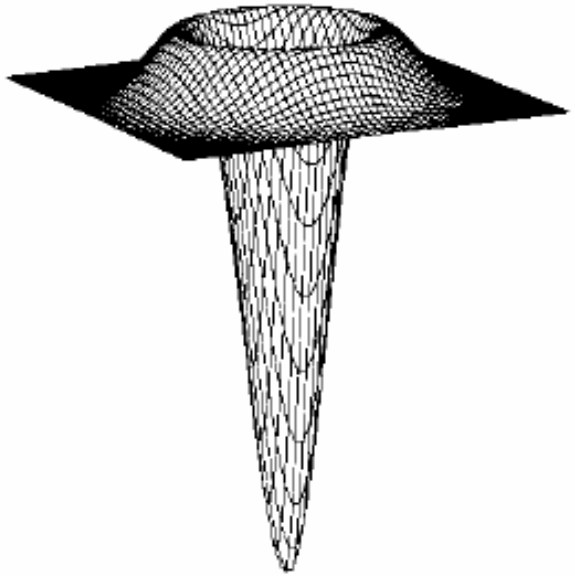
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$

derivative of Gaussian



$$\frac{\partial}{\partial x} h_\sigma(u,v)$$

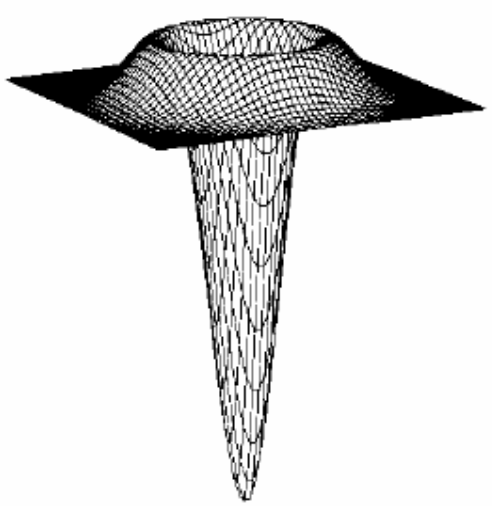
Laplacian of Gaussian



$$\nabla^2 h_\sigma(u,v)$$

LoG: example

- The Laplacian of a Gaussian filter



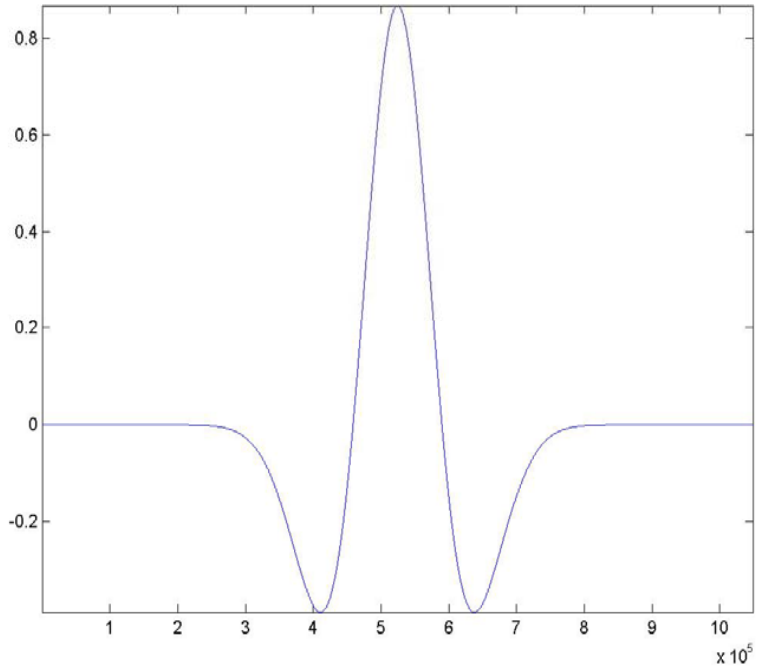
A digital approximation:

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

Second derivative

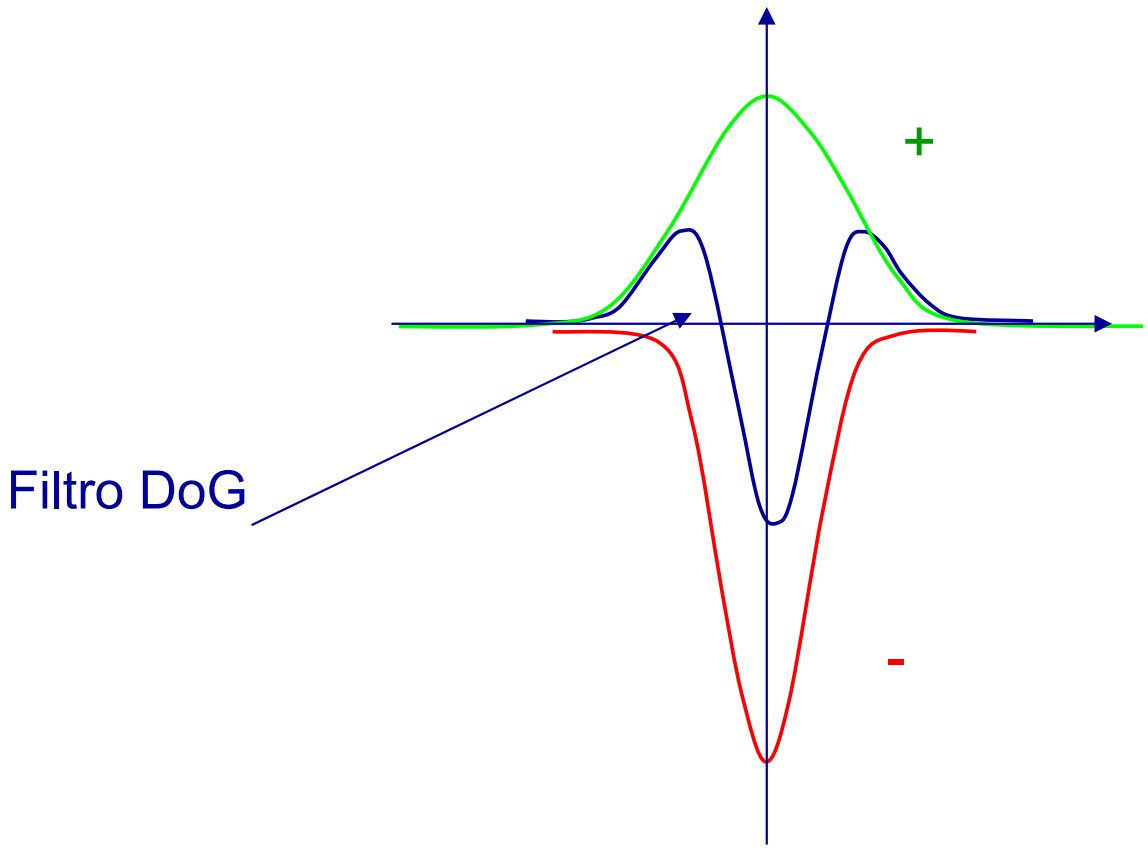
- Laplacian of Gaussian: (LoG) – Mexican Hat

0	1	0
1	-4	1
0	1	0



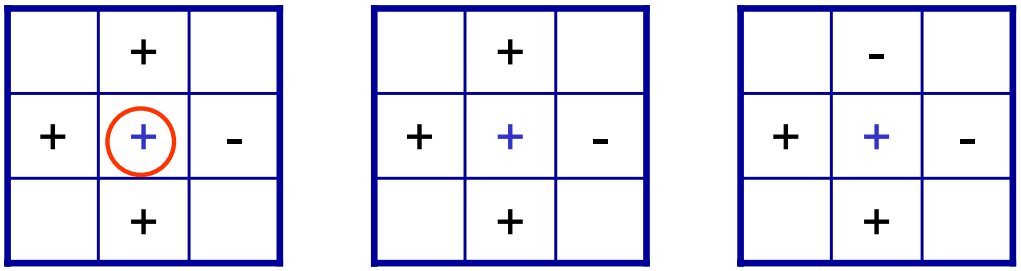
- Laplacian of Gaussian: Link to early vision: the 2D Mexican Hat closely resembles the receptive field of simple cells in the retina → edge detection is one of the first steps in vision

DoG



Laplacian zero-crossing detection

- Zero-valued Laplacian response pixels are unlikely in real images
- Practical solution: form the maximum of all positive Laplacian responses and the minimum of all Laplacian responses in a 3x3 window. If the *difference* between the two exceeds a threshold an edge is assumed to be present.
- Laplacian zero-crossing patterns



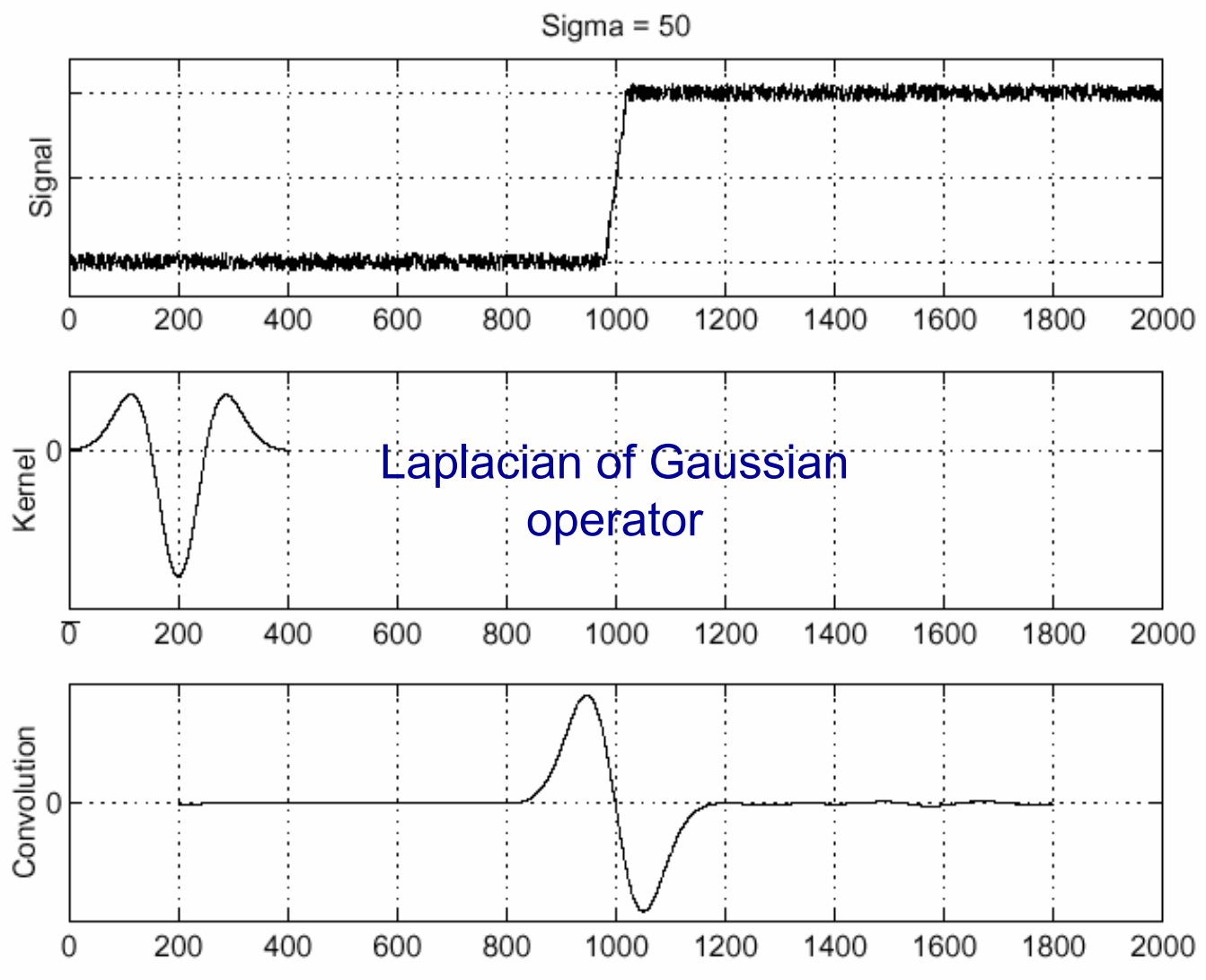
zero or positive

Laplacian of Gaussian (LoG)

f

$\frac{\partial^2}{\partial x^2} h$

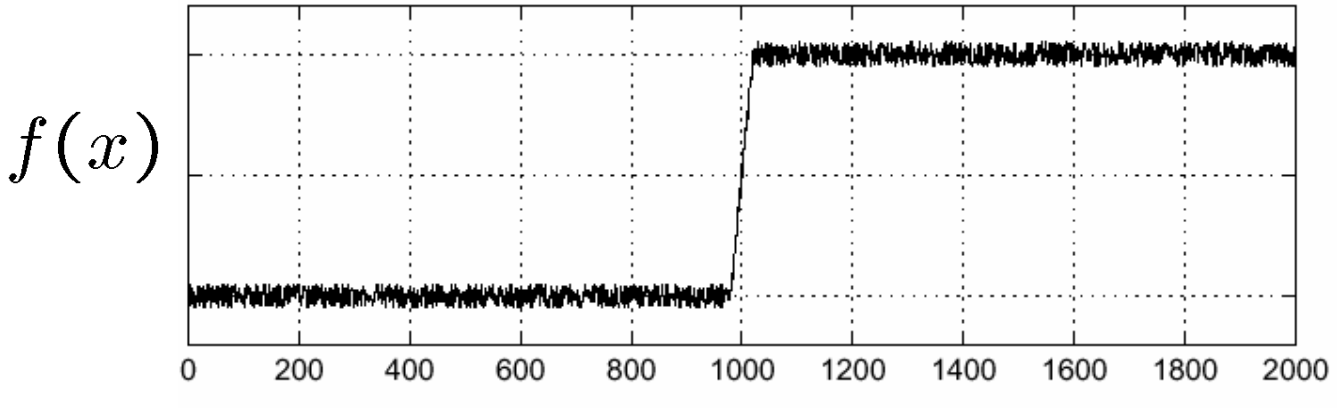
$(\frac{\partial^2}{\partial x^2} h) \star f$

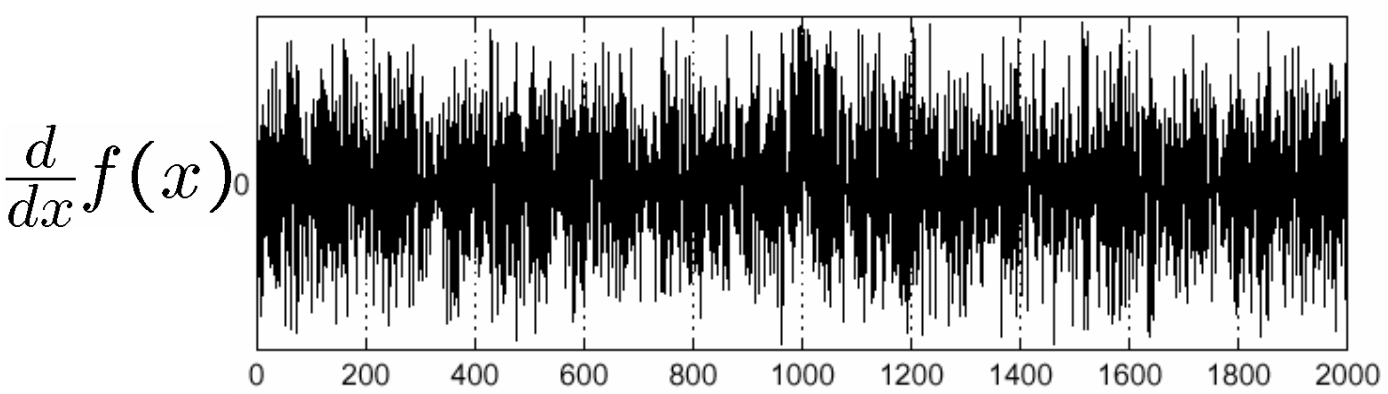


Zero-crossings of bottom graph

Effects of noise

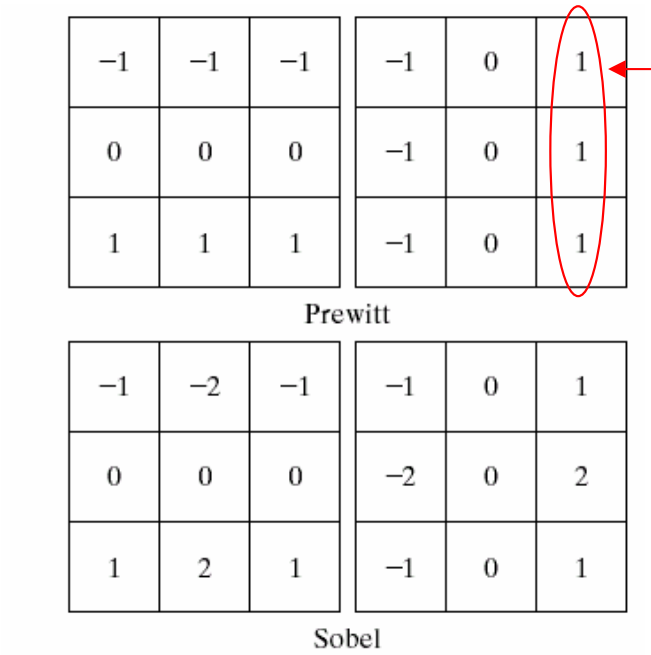
- Consider a single row or column of the image
  - Plotting intensity as a function of position gives a signal





Revisiting Line detection

- Possible filters to find gradients along vertical and horizontal directions



−1	−1	−1	−1	0	1
0	0	0	−1	0	1
1	1	1	−1	0	1

Prewitt

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

Sobel

Averaging provides noise suppression

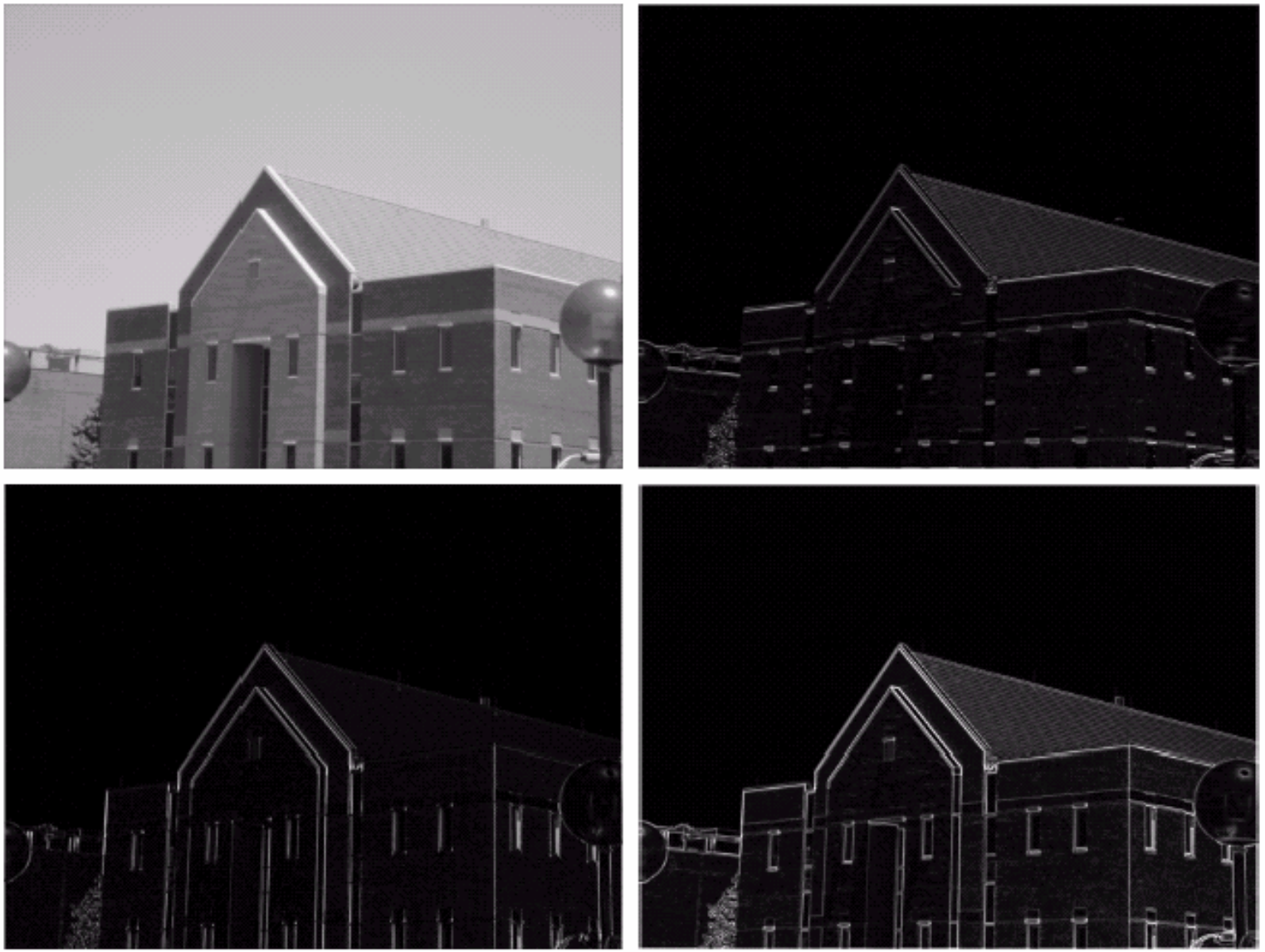
This gives more importance to the center point.

Edge Detection

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, $|G_x| + |G_y|$.

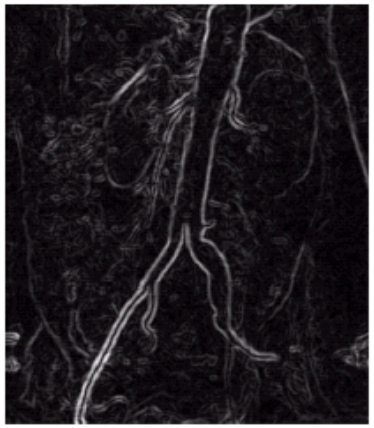
Edge Detection



a b
c d

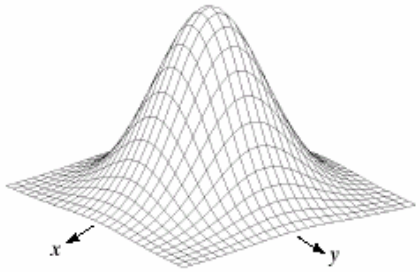
FIGURE 10.11
Same sequence as in Fig. 10.10, but with the original image smoothed with a 5×5 averaging filter.

Edge Detection



a b
c d
e f g

FIGURE 10.15 (a) Original image. (b) Sobel gradient (shown for comparison). (c) Spatial Gaussian smoothing function. (d) Laplacian mask. (e) LoG. (f) Thresholded LoG. (g) Zero crossings. (Original image courtesy of Dr. David R. Pickens, Department of Radiology and Radiological Sciences, Vanderbilt University Medical Center.)



−1	−1	−1
−1	8	−1
−1	−1	−1

One simple method to find zero-crossings is black/white thresholding:
1. Set all positive values to white
2. Set all negative values to black
3. Determine the black/white transitions.

Compare (b) and (g):
•Edges in the zero-crossings image is thinner than the gradient edges.
•Edges determined by zero-crossings have formed many closed loops.

Edge detection: Gradient thresholding

Prewitt filter: decreasing the threshold

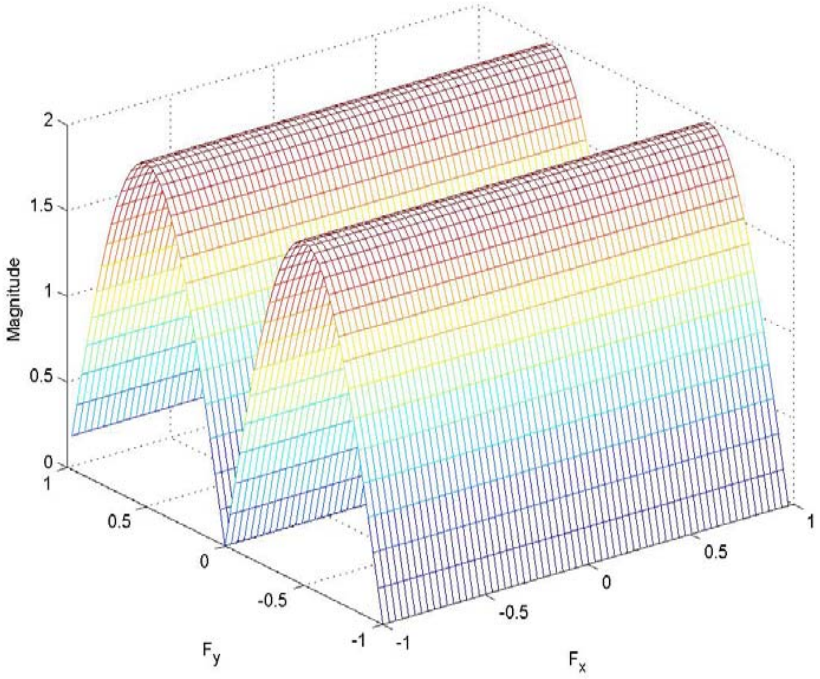
Edge detection: Gradient thresholding

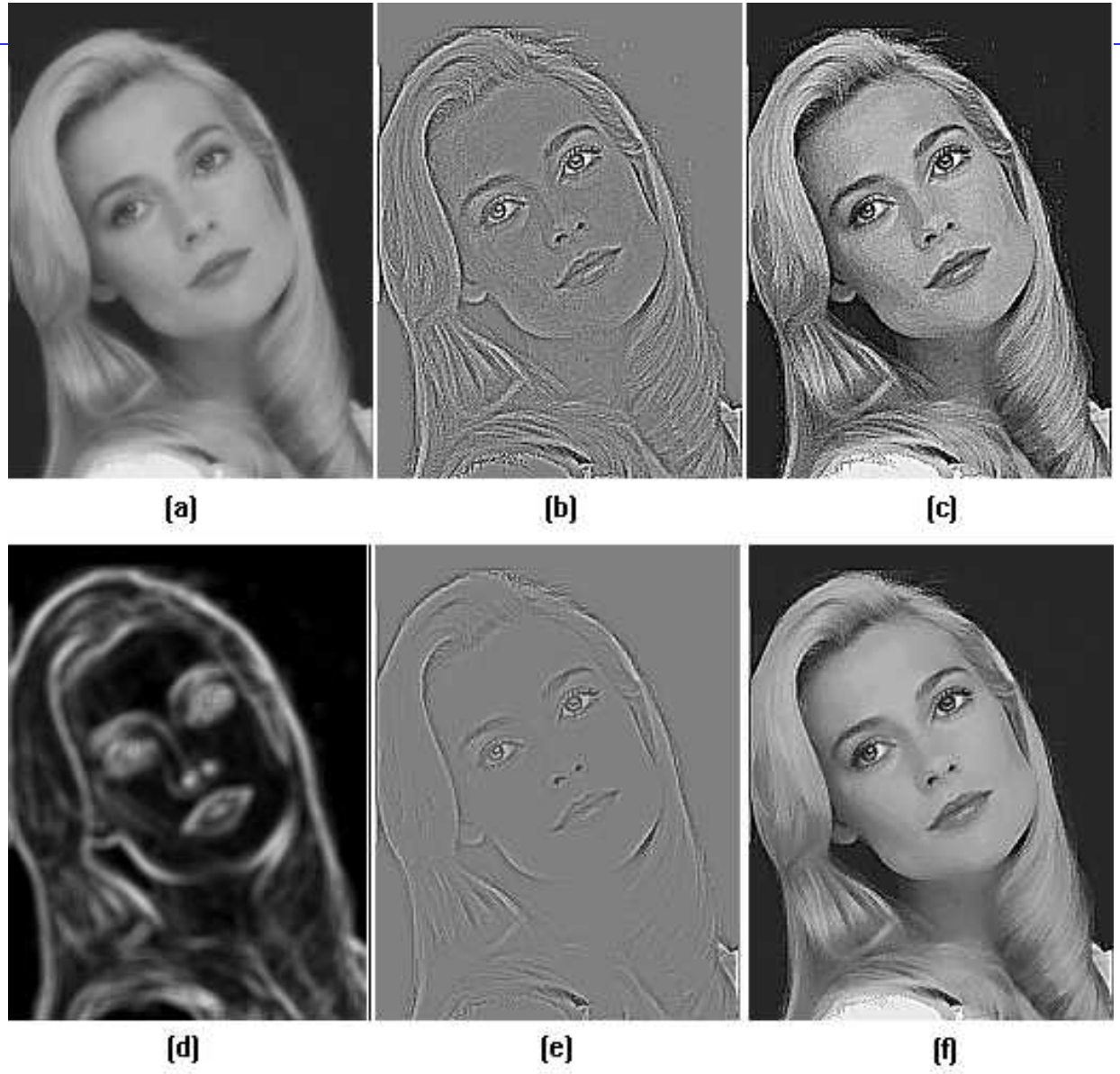
Prewitt filter: decreasing the threshold

Edge detection

Using only the vertical high frequencies

$$h_{highpass} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix}$$





(a) Input image; **(b)** Laplacian of (a); **(c)** Spatially invariant high-pass filtering [sum of (a) and (b)]; **(d)** Mask image [Sobel gradient of (a) smoothed by a 5x5 box filter]; **(e)** Product of (b) and (d); **(f)** Space-variant enhancement [sum of (a) and (e)].

Multiscale edge detection

- The information obtained by filtering the image at different scales is combined to determine the edge map
  - scale ↔ width (sigma parameter) of the filter
- Different possibilities
  - Adapting the filter bandwidth to the local characteristics of the image (Wiener)
  - Combining edge maps obtained at different scales
- Canny algorithm
  - Smoothing (allows for different scales)
  - Gradient maxima
  - Two thresholds to detect both *weak* and *strong* edges. Weak edges are retained if they are connected to strong ones (labeling)
  - Less sensible to noise

Canny algorithm

- Based on a 1D continuous model of a step edge of amplitude h_E plus additive Gaussian noise of amplitude σ_n
- The impulse response of the filter h(x) is assumed to be FIR and *antisymmetric*
- First order derivative: the edge is located at the local maxima of

$$f(x) * h(x)$$

- A threshold has to be chosen
- Criterion: the Canny operator impulse response h(x) is chosen to satisfy three criteria
  - Good *detection*
  - Good *localization*
  - *Single response*

Step edge model

- Parameters
  - Edge direction (tangent to the curve)
  - Normal direction (vector orthogonal to the contour at edge location)
  - Local contrast (edge strength)
  - Edge location (along the normal direction)

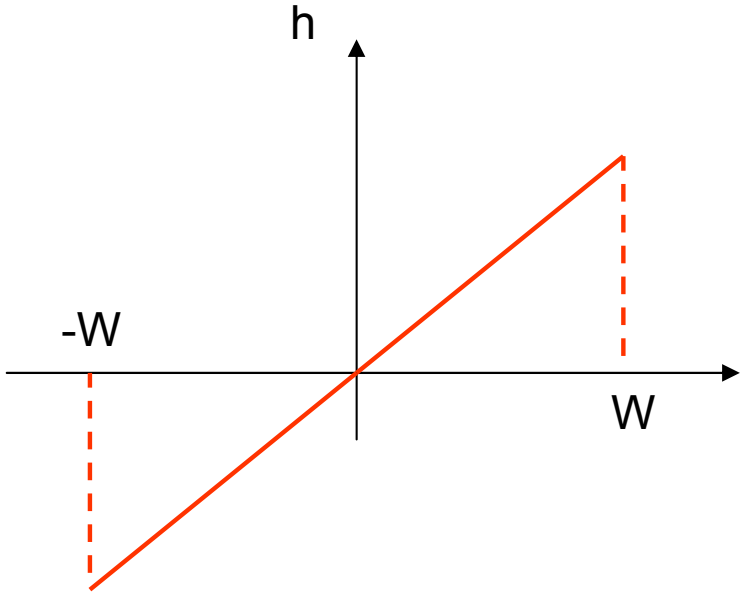
$$G(x) = \begin{cases} 0, & x < 0 \\ A, & x \geq 0 \end{cases}$$

Detection

- Criterion: The amplitude of the Signal to Noise Ratio (SNR) of the gradient is maximized for good detection
  - to obtain low probability of failure to mark edge points (*false negative rate*) and low probability to mark non-edge points (*false positive rate*)

$$SNR = \frac{h_E S(h)}{\sigma_n}$$

$$S(h) = \frac{\int_{-W}^{0} h(x)dx}{\int_{-W}^{W} [h(x)]^2 dx}$$



Localization

- Criterion: Edge points marked by the ed operator must be as close as possible to the center of the edge
- Localization factor

$$LOC = \frac{h_E L(h)}{\sigma_n}$$

$$L(h) = \frac{h'(0)}{\int_{-W}^{W} [h'(x)]^2 \, dx}$$

$$h'(x) = \frac{dh(x)}{dx}$$

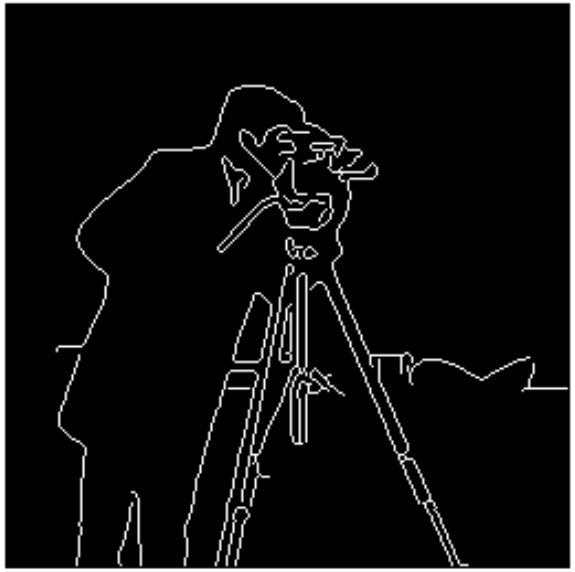
Single response (qui)

- Criterion: There should be only a single response to a true edge
  - The distance between peaks of the gradient when only noise is present is set to

$$x_m = kW \qquad (2)$$

- **Global criterion**: maximization of the product $S(h)L(h)$ subject to (2)
  - Constrained maximization
  - Note: a large filter (W) improves detection (better denoising) BUT reduces the precision in localization
  - No close form solution, numerical ones are adopted
  - For low x_m, h(x) resembles the boxcar, while for larger x_m it is closely approximated by a FDOG (first derivative of Gaussian)

Example



Example

threshold = 0.5

Performance assessment

- Possible errors
  - False negatives (an edge point is present but it is not detected)
  - False positives (a non-edge point is detected)
  - Error in the estimation of the orientation
  - Error in the localization of the edge

- Paradigms
  - Use of synthetic images + noise with known parameters
  - Tests on sets of real images

Performance evaluation

Objective

- The ground truth is assumed to be available and represented by the actual contour (*full reference metric*)
- Concerns low level features
  - Measure to which extent the estimated contour represents the actual contour
- Metric: MSE among the estimated ($f[j,k]$) and the real ($s[j,k]$) edges

Subjective

- The ground truth is not necessarily given (*reduced or no-reference metric*)
- Concerns high-level features
  - Measures to which extent the estimated contour allows to identify the corresponding object in the image
  - Focus on semantics or image content
- Metric: subjective scores given to the different algorithms
- Lead to perception-based models and metrics

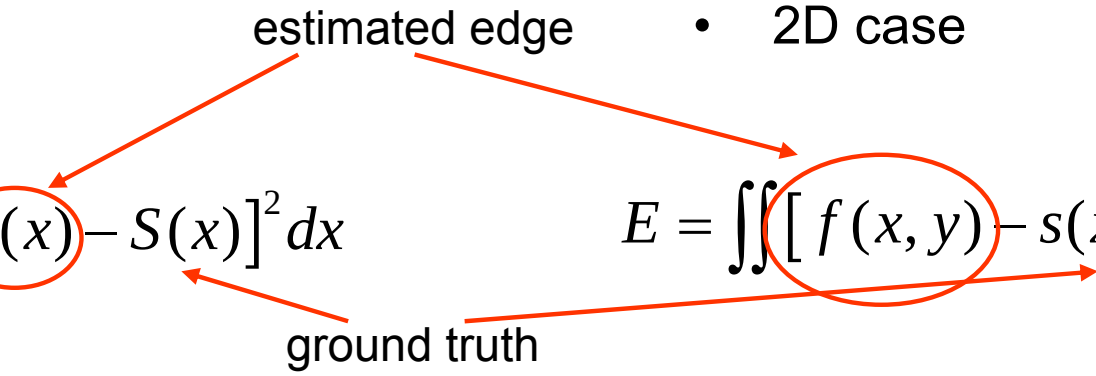
Objective assessment

- 1D case

$$E = \int_{x_0 - L}^{x_0 + L} [f(x) - S(x)]^2 dx$$

- 2D case

$$E = \iint [f(x,y) - s(x,y)]^2 dxdy \quad (3)$$



A common strategy in signal detection theory is to establish a bound on the probability of false detection resulting from noise and then try to maximize the probability of true signal detection

- When applied to edge detection, this translates in setting a the minimum value of the threshold such that the *FP rate does not exceed the predefined bound*. Then the probability of true edge detection can be calculated by a coincidence comparison of the edge maps of the ideal versus the real edge detectors

Performance assessment: Figure of Merit

- Types of errors
- Detection
  - Missing valid edge points (False Negative)
  - Failure to *localize* edge points
  - Classification of noise fluctuations as edge points (False Positives)
- Localization
  - Error in estimating the edge angle;
    - Mean square distance of the edge estimate from the true edge
- Accuracy
  - Algorithm's tolerance to distorted edges and other features such as corners and junctions

Performance assessment: Figure of Merit

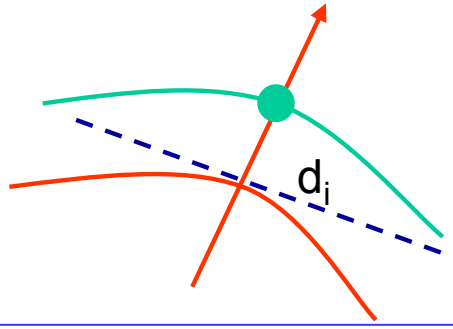
ensures a penalty for smeared or fragmented edges

$$F_M = \frac{1}{\max(I_A, I_I)} \sum_{i=1}^{I_A} \frac{1}{1 + d_i \alpha^2}$$

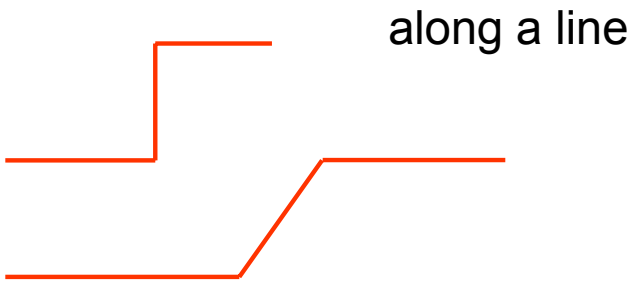
to penalized edges that are localized by offset from the true position

$$F_M = 1: \text{ perfectly detected edge}$$

I_A, I_I: number of ideal and detected edge points, respectively
d_i: distance among the ideal and the detected edge point along the normal to a line of ideal edge points (evaluated according to (3))
α: scaling constant

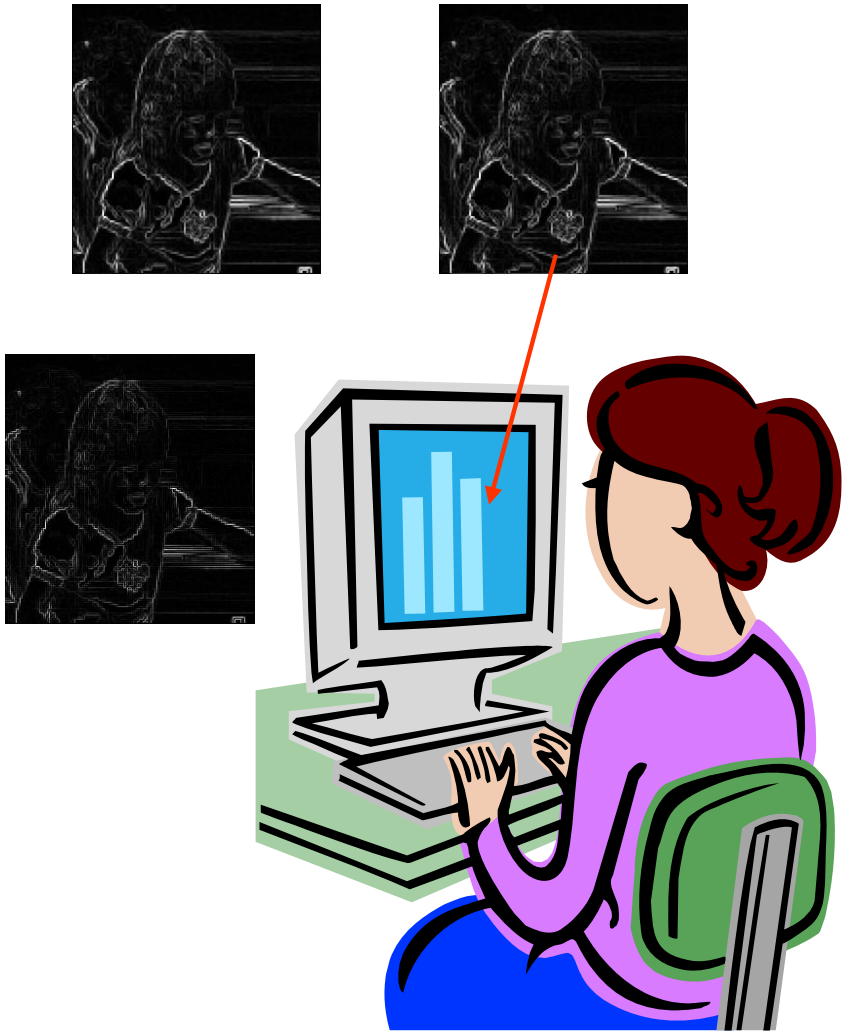


Filters competition



- A possible classification strategy
  - Synthetic image
    - 64x64 pixels
    - vertical oriented edge with variable slope and contrast
    - added Gaussian noise of variance σ_n
    - Control parameter SNR=(h/ σ_n), h being the normalize edge value (0<h<=1)
  - Filter threshold: maximize the FM constrained to maximum bound for false detection rate
    - False detection=false positives
    - Probability to detect an edge when no edge is present
- See figure 15.5-11 and 12, Pratt

Subjective evaluation



- Task: “Give a score to the detected edges”
- Many trials
  - The experiment is repeated at least two times for each subject
- Many subjects
  - A sufficiently high number of subjects must participate in the experiment to make data analysis significant from the statistical point of view
- Output: {scores}
- Data analysis

Color images

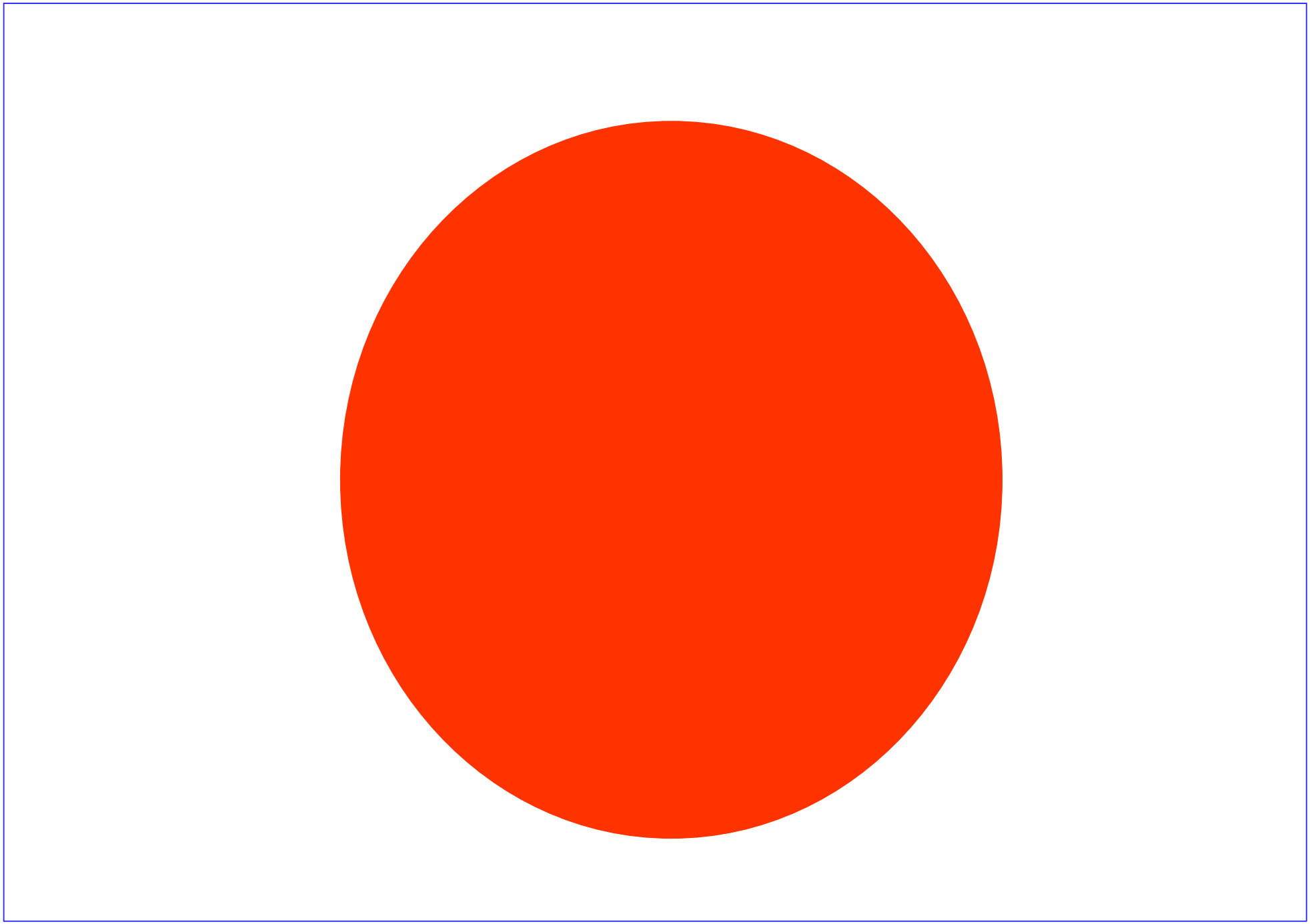
- Different approaches
  - An edge is present iif there is a gradient in the luminance
  - An edge exists if there is a gradient in any of the tristimulus components
  - "Total gradient" above a predefined threshold

$$G(j,k) = G_1(j,k) + G_2(j,k) + G_3(j,k)$$

  - "Vector sum gradient" above a predefined threshold

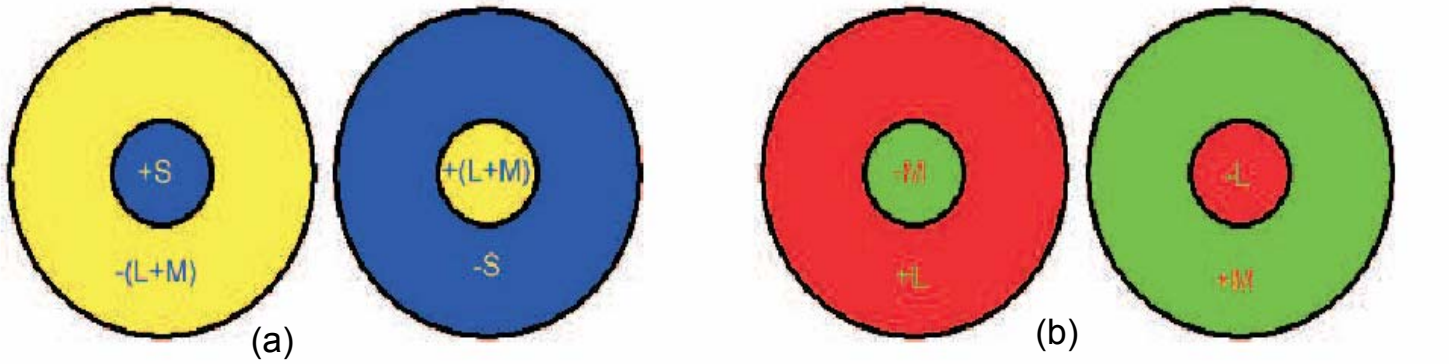
$$G(j,k) = \left\{ \left|G_1(j,k)\right|^2 + \left|G_2(j,k)\right|^2 + \left|G_3(j,k)\right|^2 \right\}^{1/2}$$

$G_i(j,k)$: i-th linear or non-linear tristimulus value



Opponent Color Model

- Perception is mediated by *opponent color channels*
  - *Evidences*
    - Afterimages
    - Certain colors cannot be perceived simultaneously (i.e. no *reddish-green* or *bluish-yellow*)

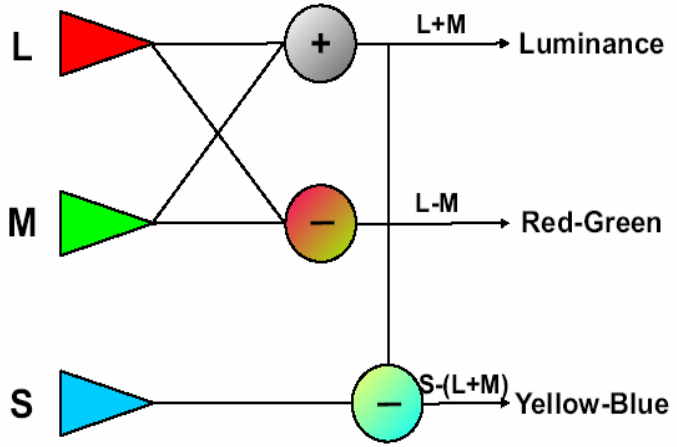


Example of typical center-surround antagonistic receptive fields: (a) on-center yellow-blue receptive fields; (b) on-center red-green receptive fields.

Because of the fact that the L, M and S cones have different spectral sensitivities, are in different numbers and have different spatial distributions across the retina, the respective receptive fields have quite different properties.

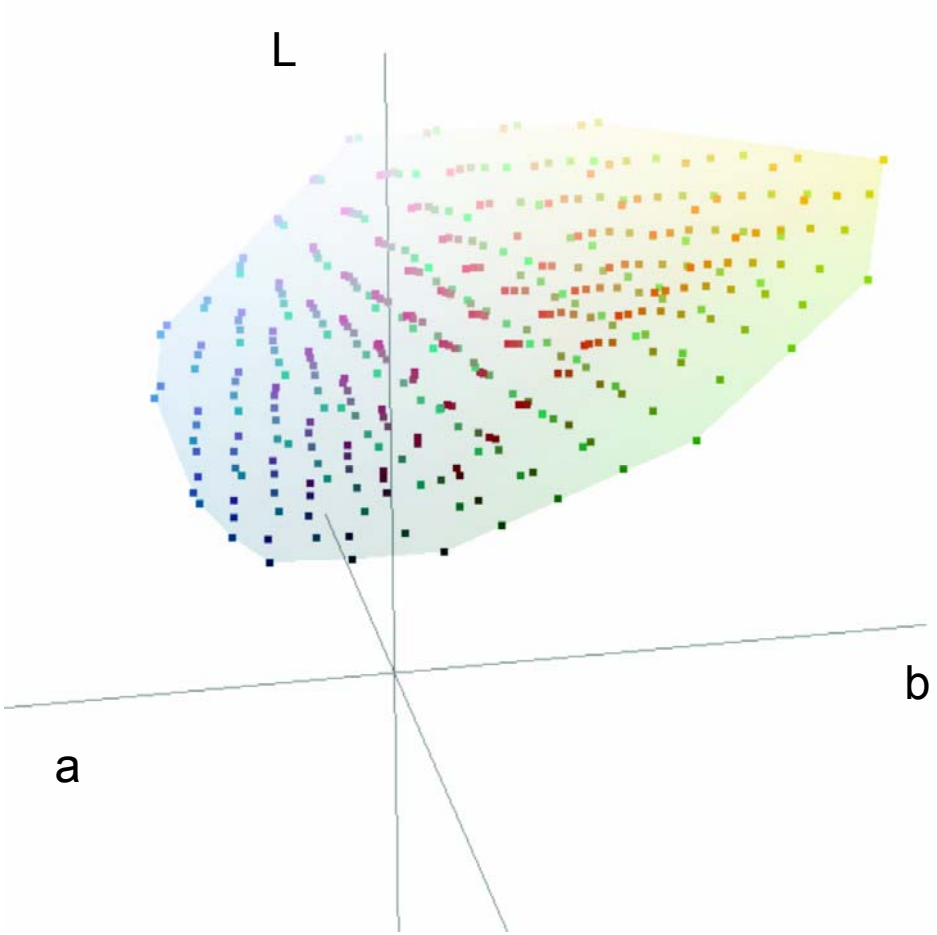
Opponent color channels

Cone interconnections in the retina leading to opponent color channels

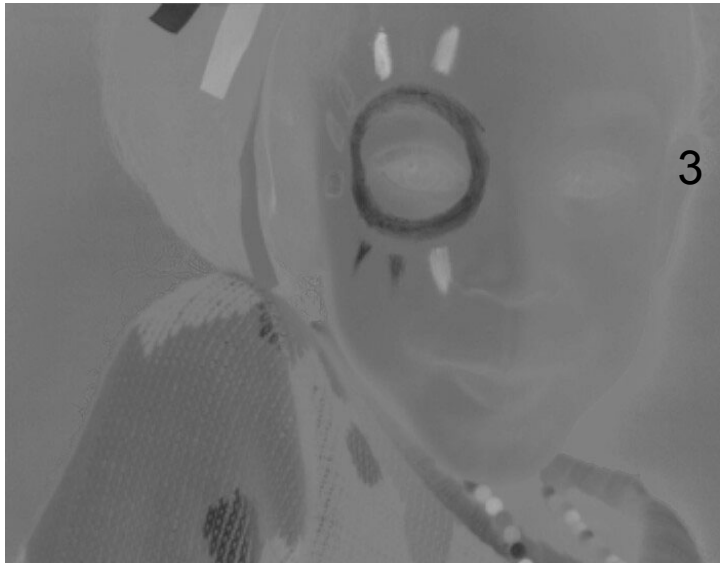


- As a convenient simplification, the existence of three types of color receptive fields is assumed, which are called *opponent channels*.
- The black-white or *achromatic* channel results from the sum of the signals coming from L and M cones (L+M). It has the highest spatial resolution.
- The *red-green* channel is mainly the result of the M cones signals being subtracted from those of the L cones (L-M). Its spatial resolution is slightly lower than that of the achromatic channel (L+M).
- Finally the *yellow-blue* channel results from the addition of L and M and subtraction of S cone signals. It has the lowest spatial resolution.

Color representation in Lab



Opponent Colors

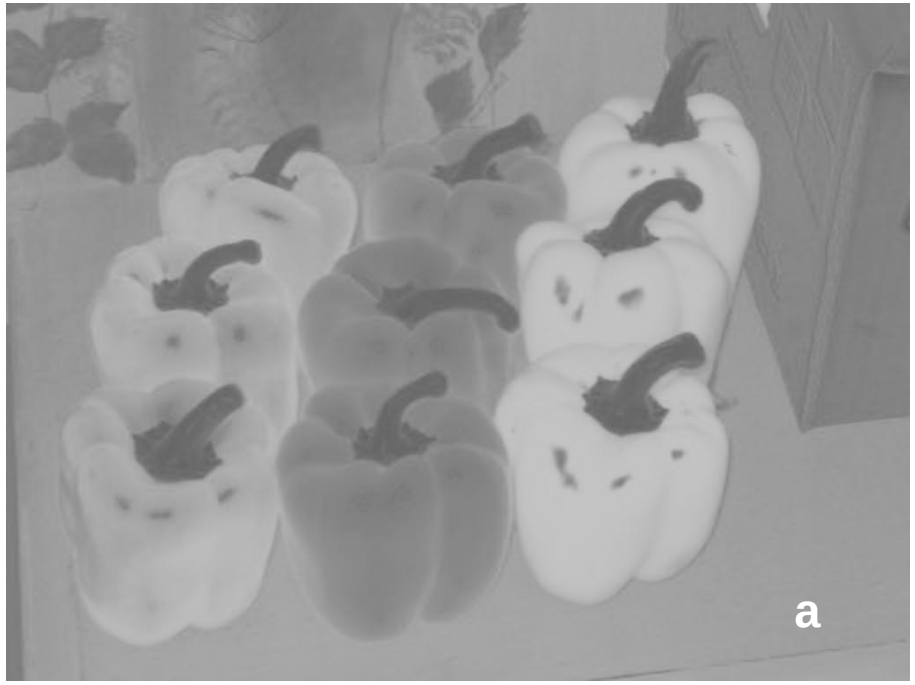



R


G


B

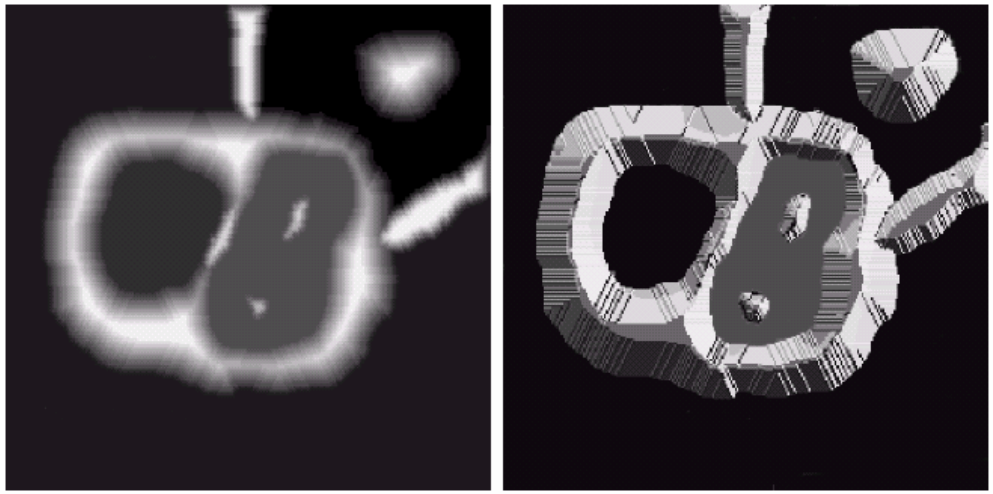

L

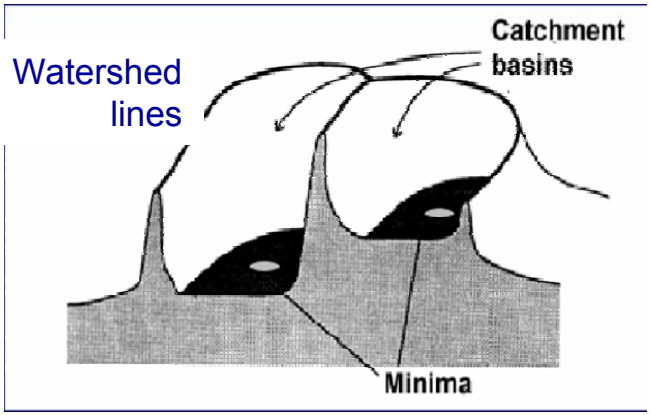

a


b

Watershed Segmentation Algorithm

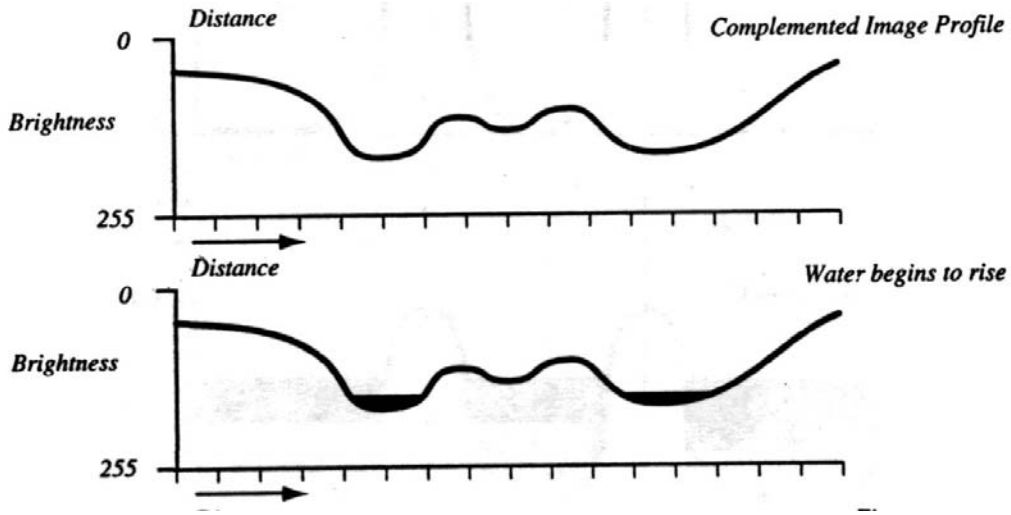
- Visualize an image in 3D: spatial coordinates and gray levels.
- In such a topographic interpretation, there are 3 types of points:
  - Points belonging to a regional minimum
  - Points at which a drop of water would fall to a single minimum. (→The *catchment basin* or *watershed* of that minimum.)
  - Points at which a drop of water would be equally likely to fall to more than one minimum. (→The *divide lines* or *watershed lines*.)



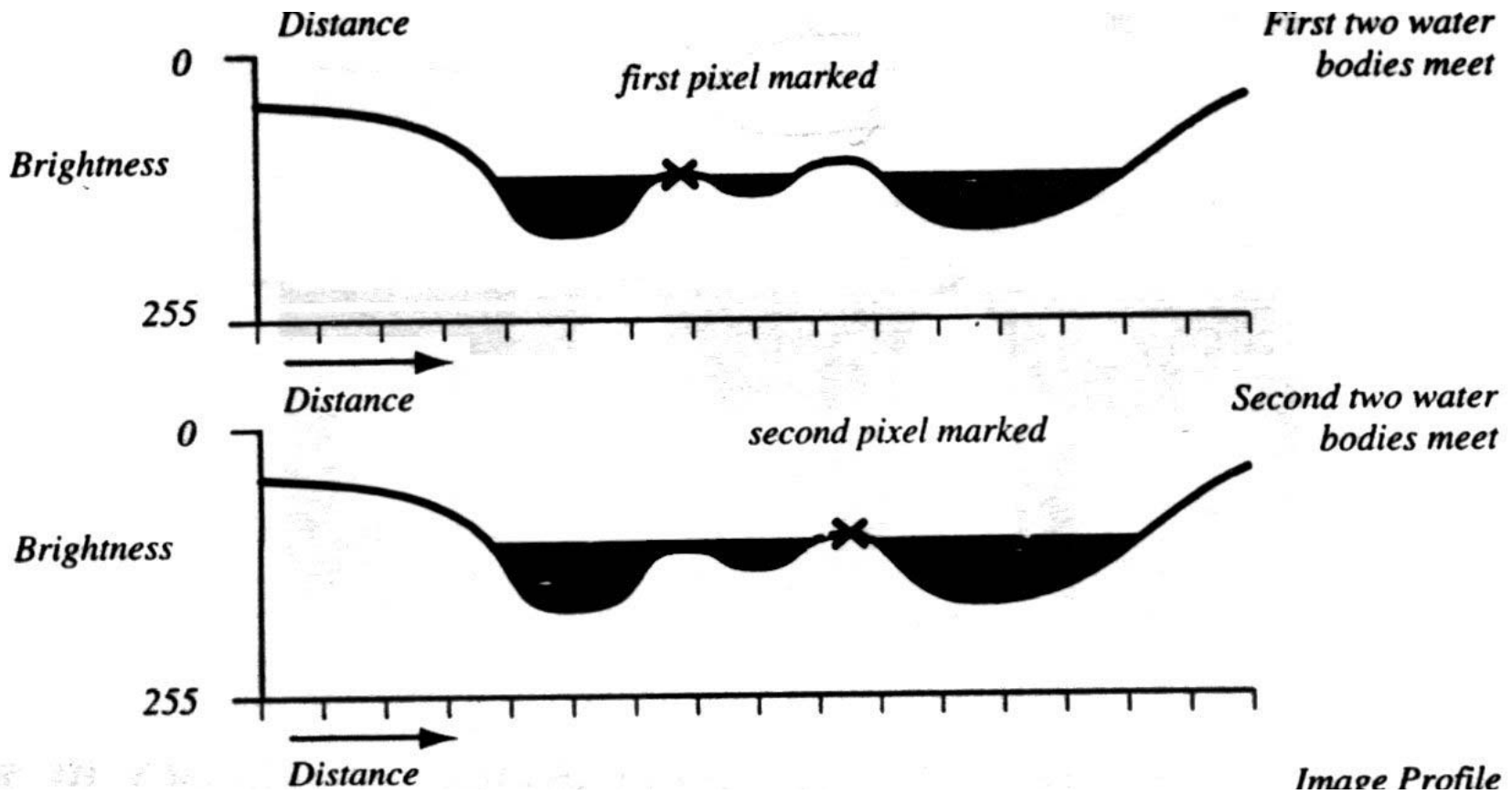


Watershed Segmentation Algorithm

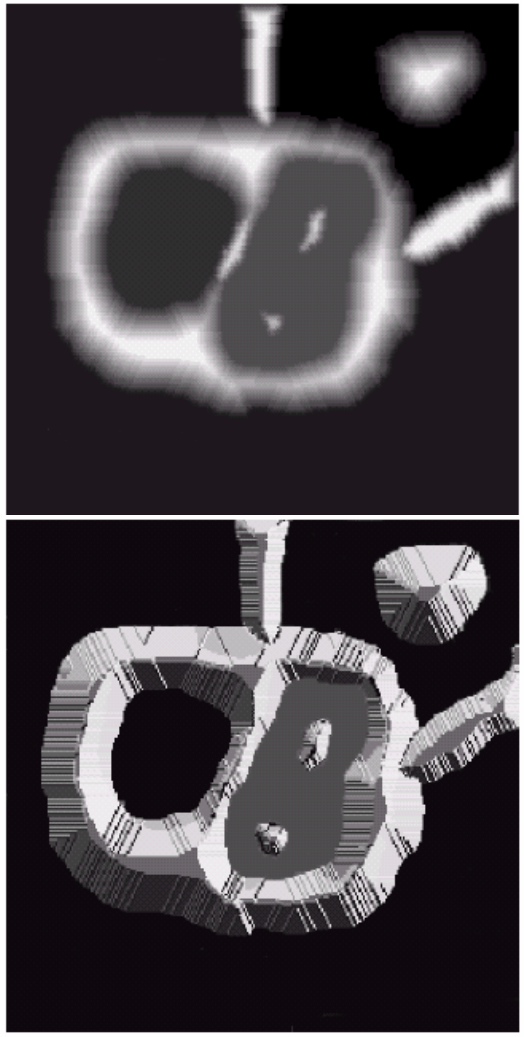
- *The objective is to find watershed lines.*
- The idea is simple:
  - Suppose that a hole is punched in each regional minimum and that the entire topography is flooded from below by letting water rise through the holes at a uniform rate.
  - When rising water in distinct catchment basins is about the merge, a dam (*diga*) is built to prevent merging.
  - Dam boundaries correspond to the watershed lines.

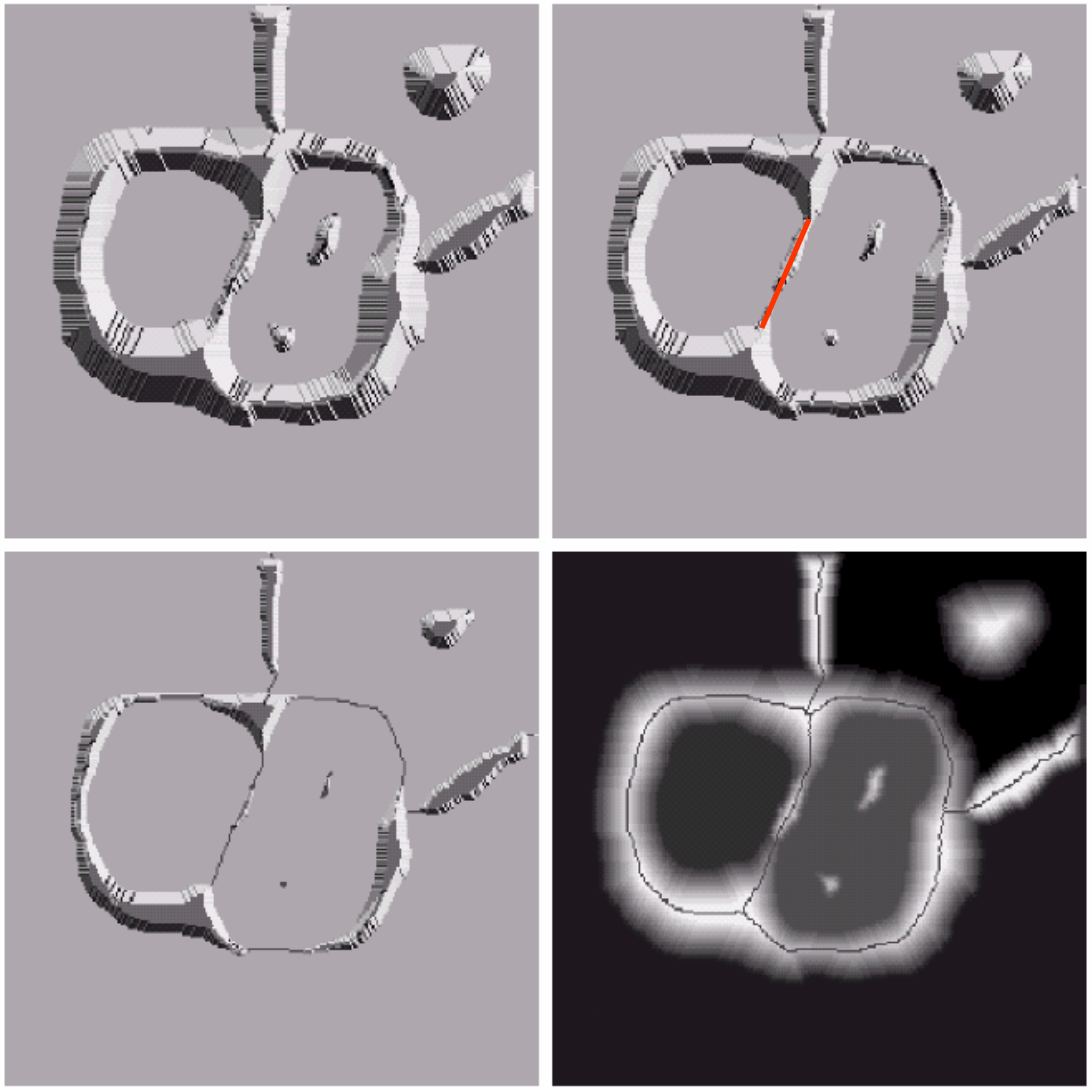


Watershed Segmentation Algorithm



Watershed Segmentation Algorithm





e f
g h

FIGURE 10.44
(Continued)
(e) Result of further flooding. (f) Beginning of merging of water from two catchment basins (a short dam was built between them). (g) Longer dams. (h) Final watershed (segmentation) lines. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

Watershed Segmentation Algorithm

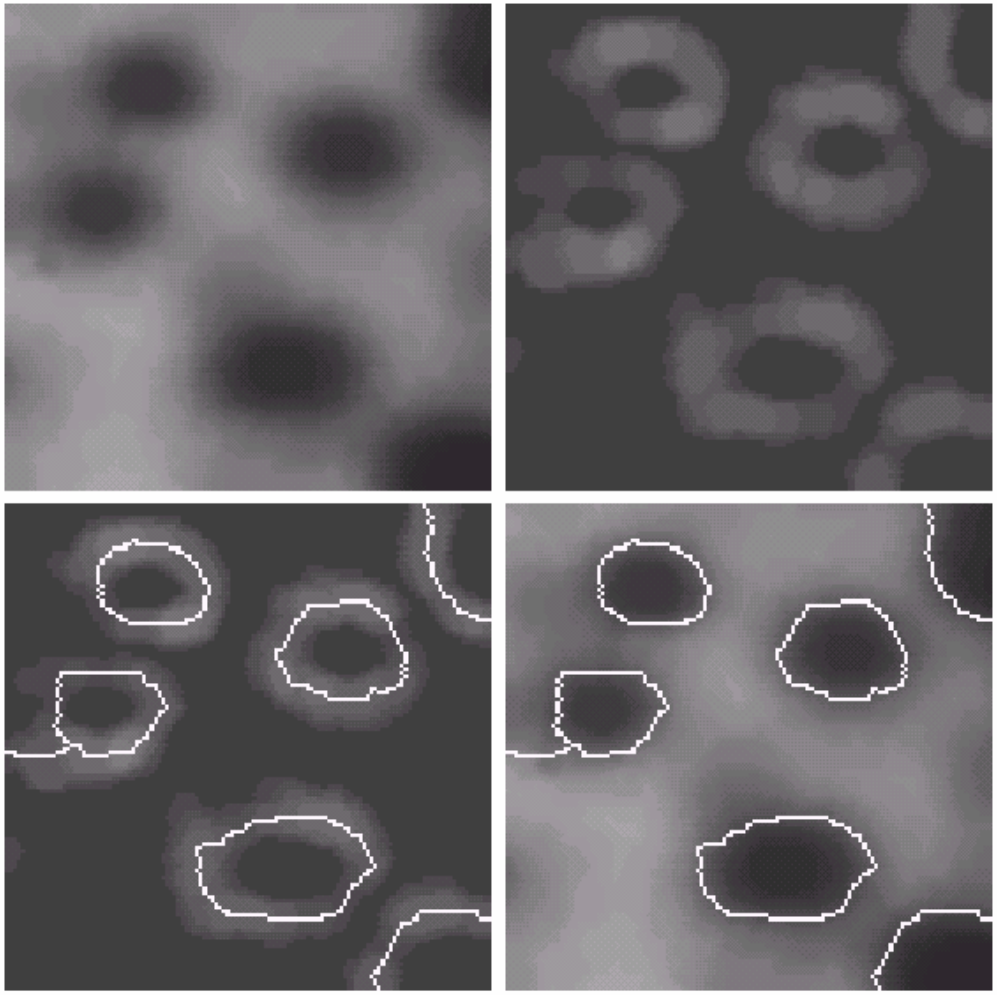
- Start with all pixels with the lowest possible value.
  - These form the basis for initial watersheds
- *For each intensity level k*:
  - For each group of pixels of intensity k
    - If adjacent to exactly one existing region, add these pixels to that region
    - Else if adjacent to more than one existing regions, mark as boundary
    - Else start a new region

Watershed Segmentation Algorithm

Watershed algorithm might be used on the gradient image instead of the original image.

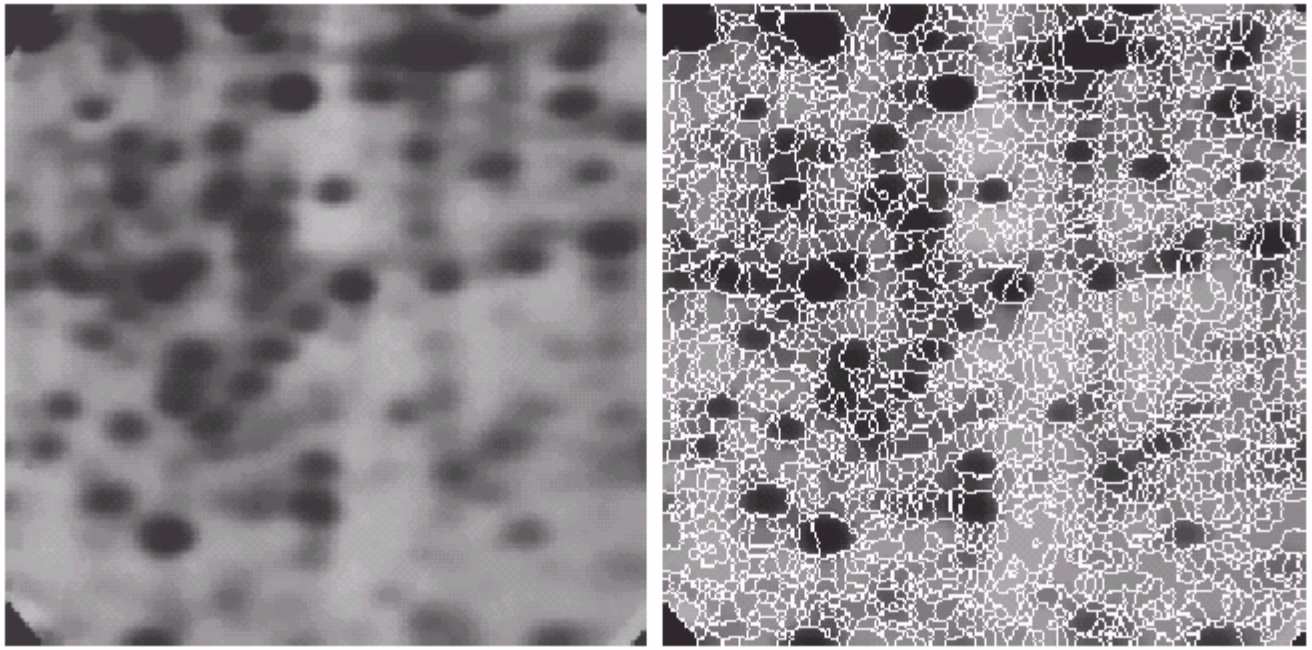
a b
c d

FIGURE 10.46
(a) Image of blobs. (b) Image gradient. (c) Watershed lines. (d) Watershed lines superimposed on original image. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)



Watershed Segmentation Algorithm

Due to noise and other local irregularities of the gradient, over-segmentation might occur.

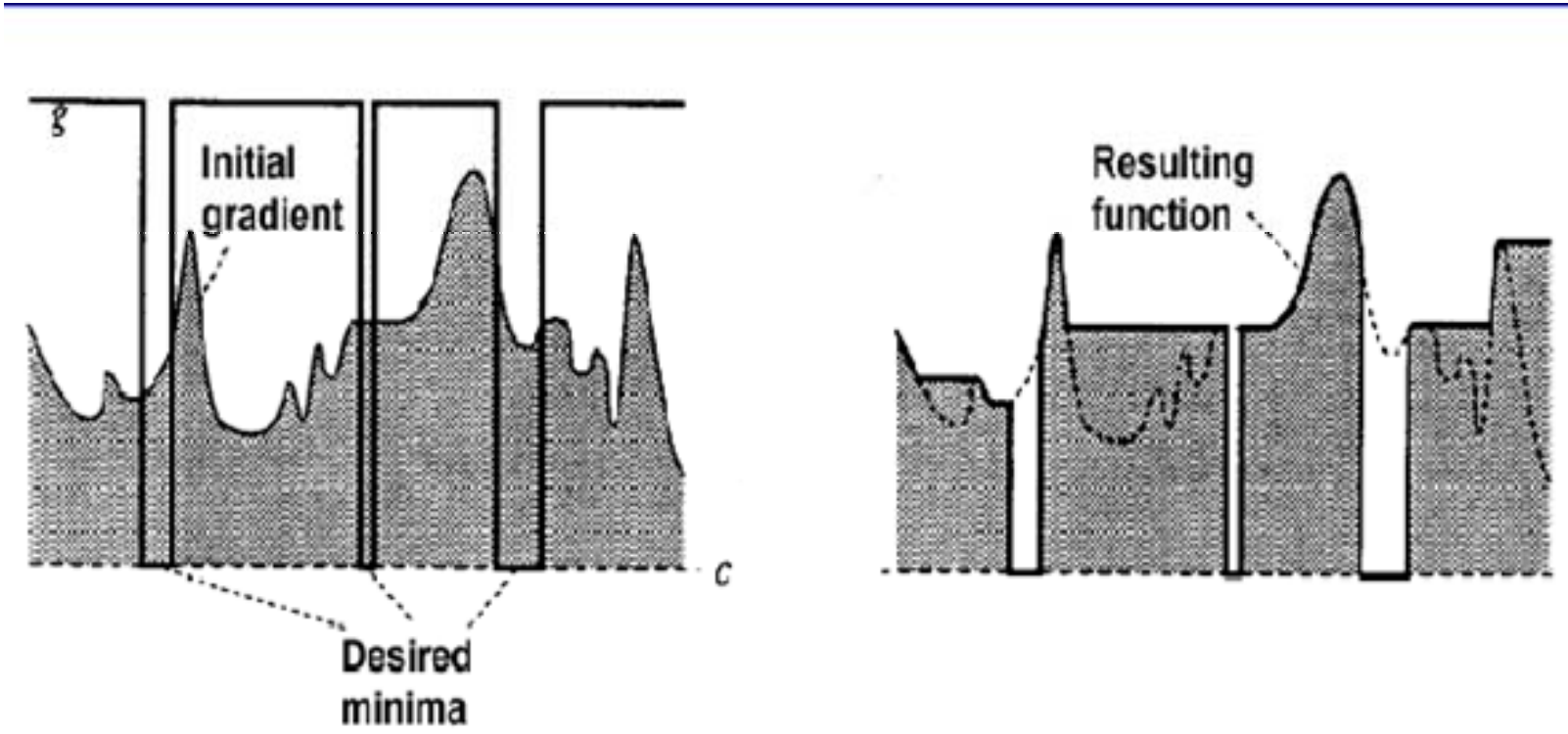


a b

FIGURE 10.47
(a) Electrophoresis image. (b) Result of applying the watershed segmentation algorithm to the gradient image. Oversegmentation is evident. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

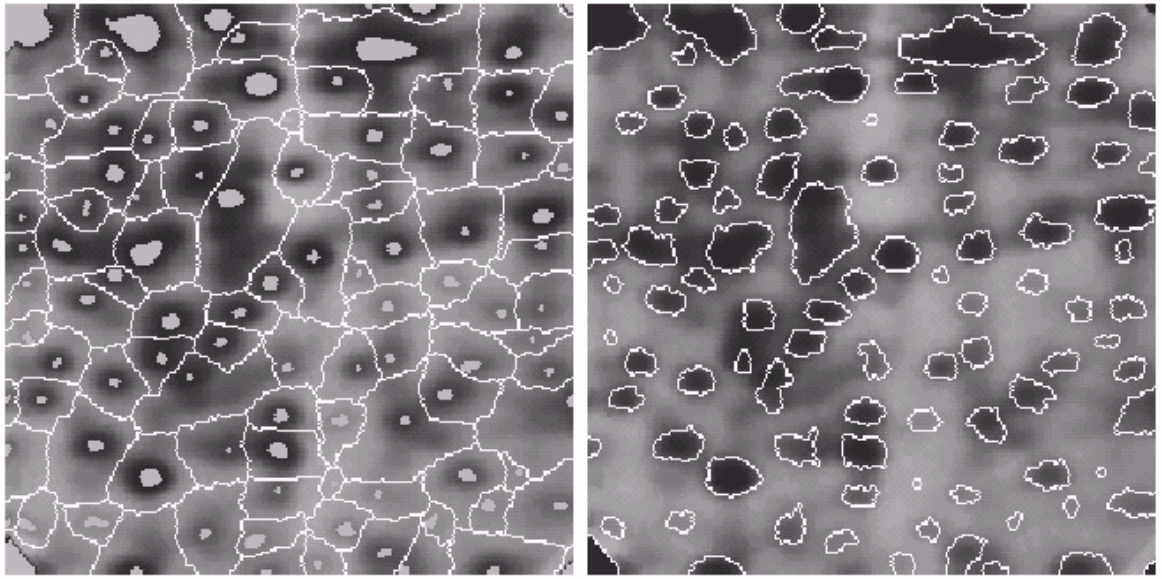
Supervised Watershed Segmentation

A solution is to limit the number of regional minima. Use markers to specify the only allowed regional minima.



Watershed Segmentation Algorithm

A solution is to limit the number of regional minima. Use markers to specify the only allowed regional minima. (For example, gray-level values might be used as a marker.)



a b

FIGURE 10.48
(a) Image showing internal markers (light gray regions) and external markers (watershed lines). (b) Result of segmentation. Note the improvement over Fig. 10.47(b). (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)