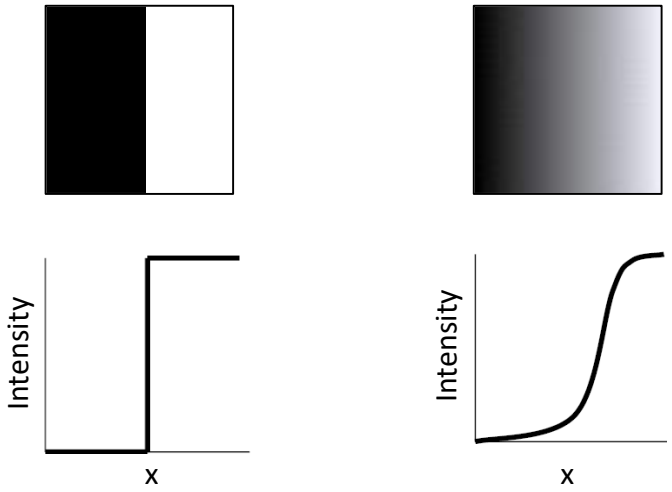
BLG 453E

Week 6
Edge Detection

Dr. Yusuf H. Sahin

Edges

- Observation: Boundaries of objects show up as intensity discontinuities

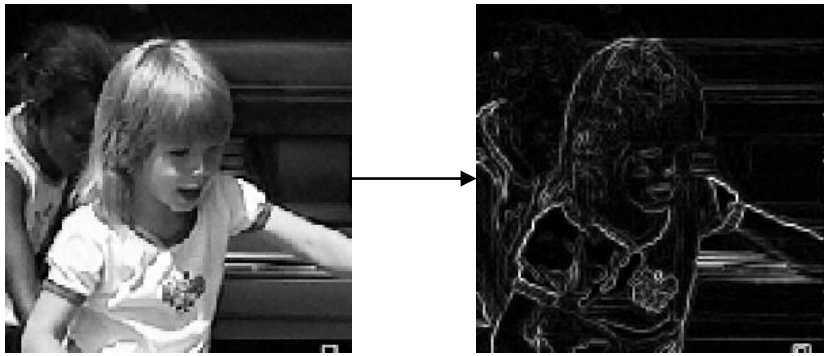


- Edges are those places in an image that correspond to object boundaries.
- Edges are pixels where image brightness changes abruptly

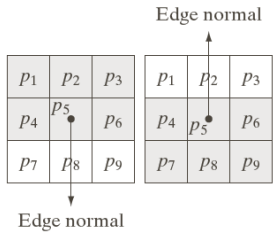
Extraction of Edge Information from Images

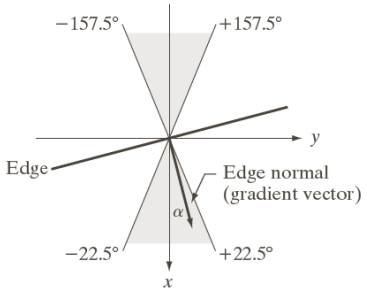
- An edge is a property attached to an individual pixel and is calculated from the image function behavior in a neighborhood of the pixel.

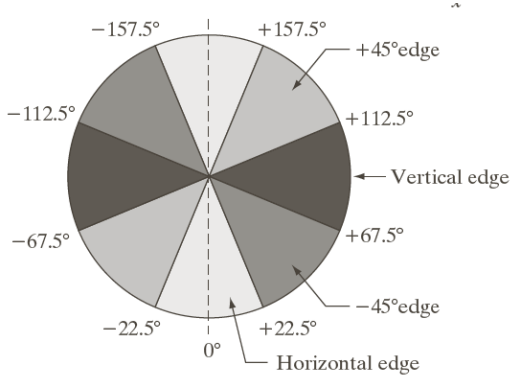
It is either a scalar variable (an edge map)



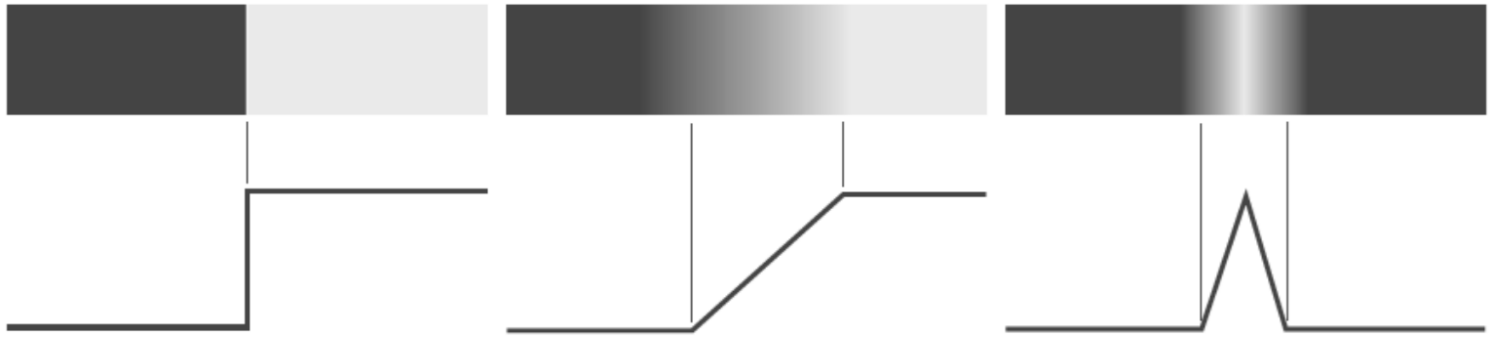
Or a vector variable (magnitude of the gradient, direction of an edge).
Typically: we extract Magnitude and Direction







Edge Models

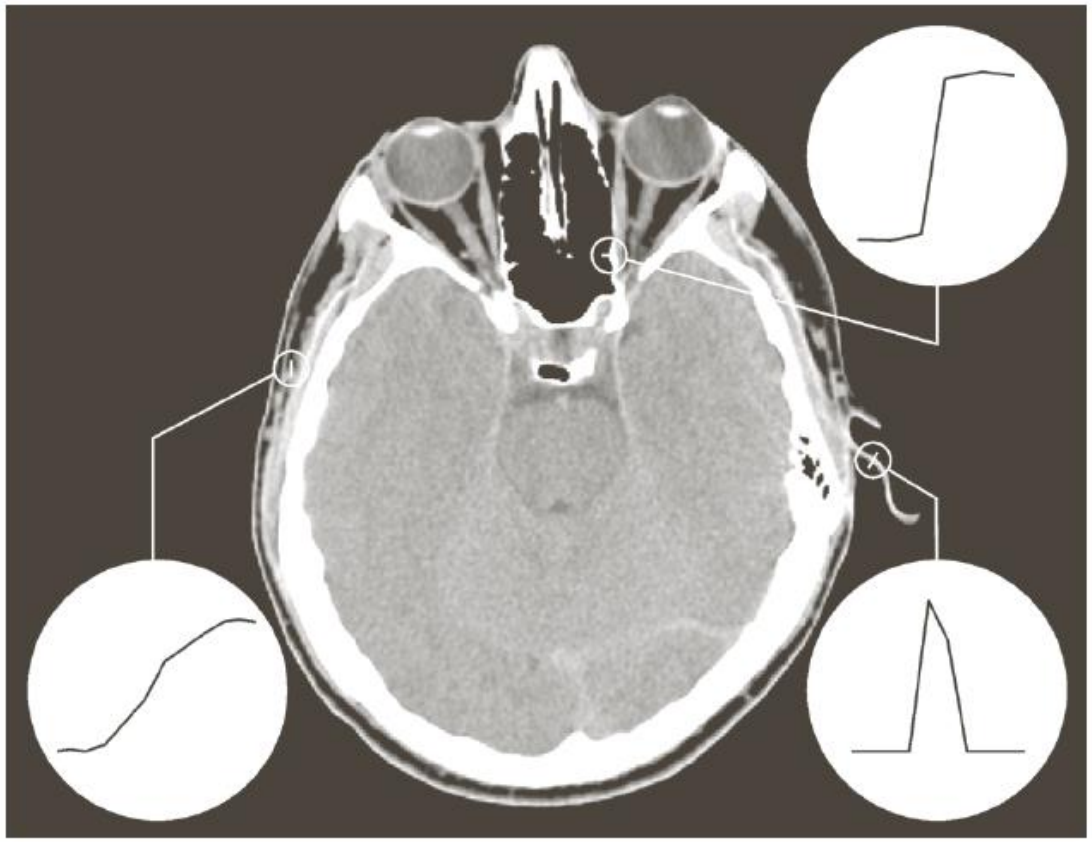


Step edge
Ideal case

Ramp edge
In pratice

Roof edge
Line

Edges & Pixel Neighborhood



- If a pixel's gray-level value is similar to those around it, there is probably not an edge at that point.
- If a pixel has neighbors with highly varying gray levels, it may present an edge point.

Edge Detection Operators

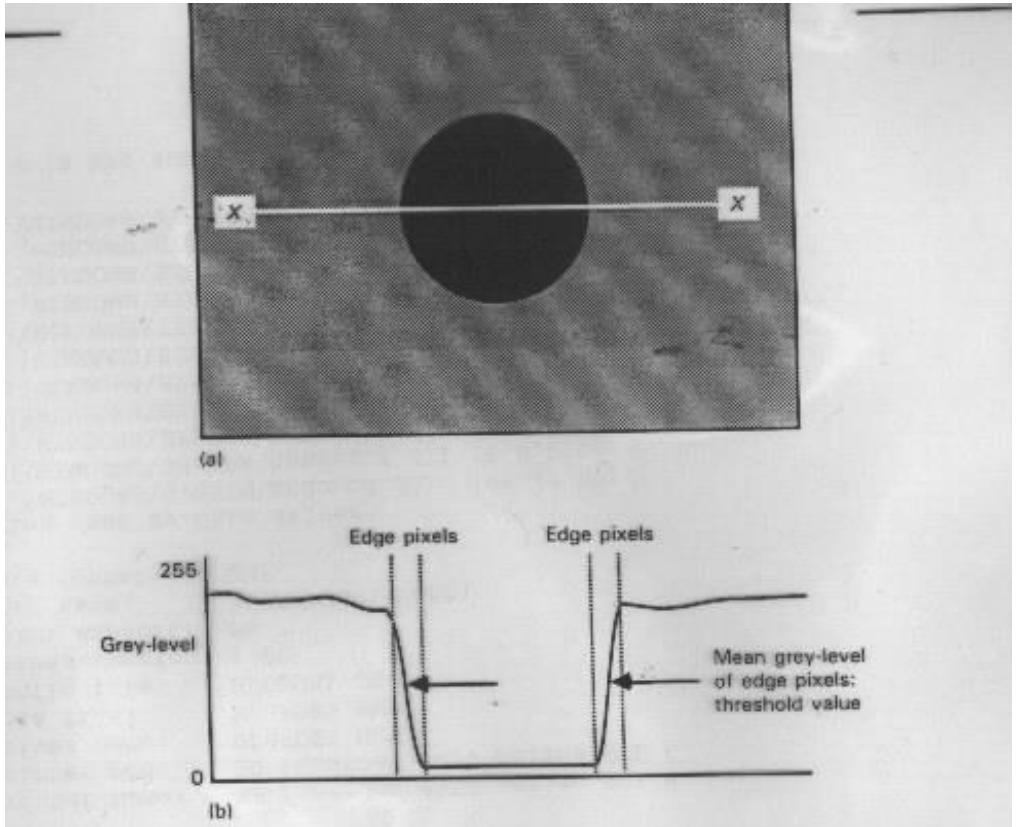
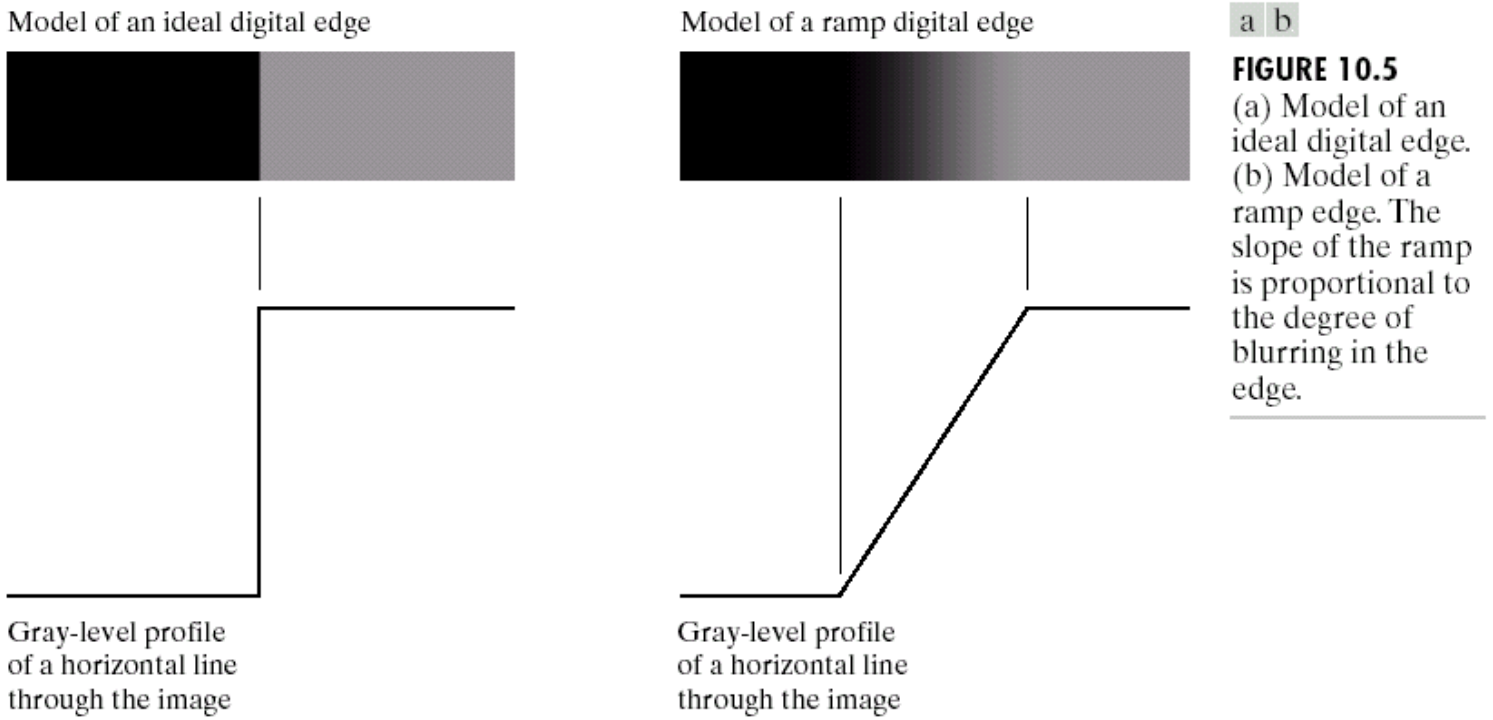


Figure 5.4 Using edge pixels to select a threshold: (a) image of dark, round object on a light background with section $X-X$ shown; (b) profile

- Gradient and difference based operators
- Detect and measure local discontinuities in intensity or its gradient

Edge Detection Operators

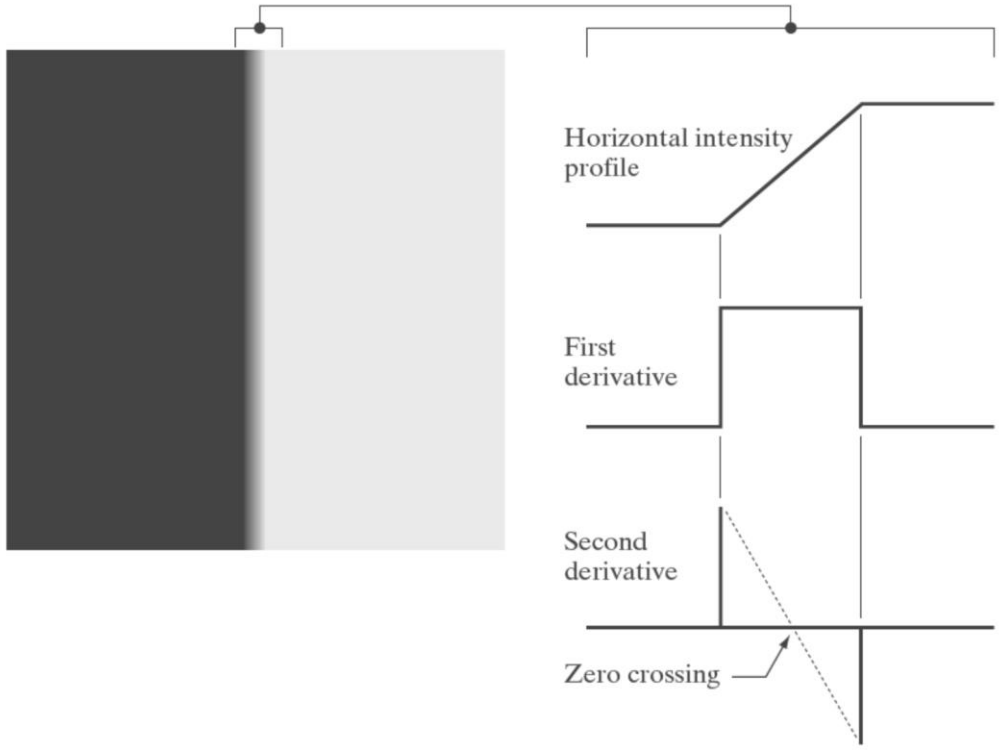


a b

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.

- If we define a local edge in an image to be a transition between two regions of significantly different intensities then the magnitude of the gradient (derivatives) of the image, which measure the rate of change in image brightness will have large values in these transitional boundary areas.

Edge Detection



- First derivative: We can check its MAGNITUDE
- Second derivative: We can check its ZERO-CROSSINGS
- Note that 2nd derivative produces double response (2 values) for an edge.
- Same idea carries from 1D profile to 2D (on a profile perpendicular to the edge)

Edge Detection Methods

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

- Many are implemented with convolution mask and based on discrete approximations to differential operators.
- Kernels are typically determined by discretized image derivatives.

Discretization: First derivative Approximation

Expand $f(x)$ in a Taylor series about x_0.

$$f(x) = f(x_0) + (x - x_0)f'(x_0) + O(h)$$

Then evaluate at $x = x_0 + h$.

$$f(x_0 + h) = f(x_0) + hf'(x_0) + O(h)$$

First order forward difference:

$$f'(x_0) \approx \frac{f(x_0 + h) - f(x_0)}{h}$$

Discretization: First derivative Approximation

Replace h with $-h$ in the previous derivation

$$f(x_0 - h) = f(x_0) - hf'(x_0) + O(h)$$

First order backward difference:

$$f'(x_0) \approx \frac{f(x_0) - f(x_0 - h)}{h}$$

Discretization: First derivative Approximation

Subtract the first order expansions:

$$f(x_0 + h) = f(x_0) + hf'(x_0) + \ldots$$

$-$

$$f(x_0 - h) = f(x_0) - hf'(x_0) + \ldots$$

$=$

$$f(x_0 + h) - f(x_0 - h) = 2hf'(x_0)$$

Divide by $2h$ to get the centered difference:

$$f'(x_0) \approx \frac{f(x_0 + h) - f(x_0 - h)}{2h}$$

Discretization: Second derivative Approximation

The second order terms in the expansion cancel, so the remainder is $O(h^2)$

$$f(x_0 + h) = f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(x_0) + O(h^2)$$

$$f(x_0 - h) = f(x_0) - hf'(x_0) + \frac{h^2}{2}f''(x_0) + O(h^2)$$

For images : Use h = 1 pixel.

We can approximate the **second derivative** by adding two second order expansions:

$$f(x_0 + h) = f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(x_0) + O(h^2)$$

+

$$f(x_0 - h) = f(x_0) - hf'(x_0) + \frac{h^2}{2}f''(x_0) + O(h^2)$$

=

$$f(x_0 + h) + f(x_0 - h) = 2f(x_0) + h^2 f''(x_0) + O(h^2)$$

Discretization: Second derivative Approximation

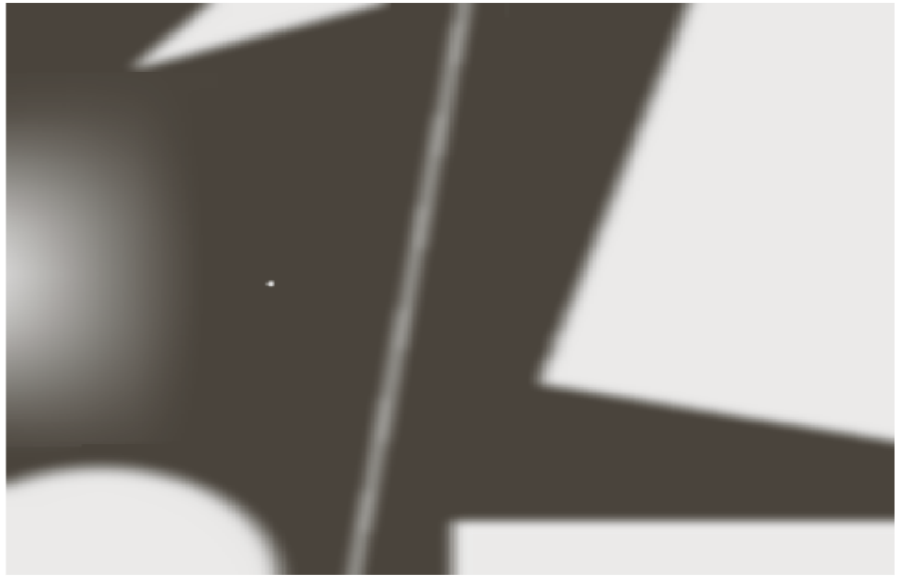
Rearrange

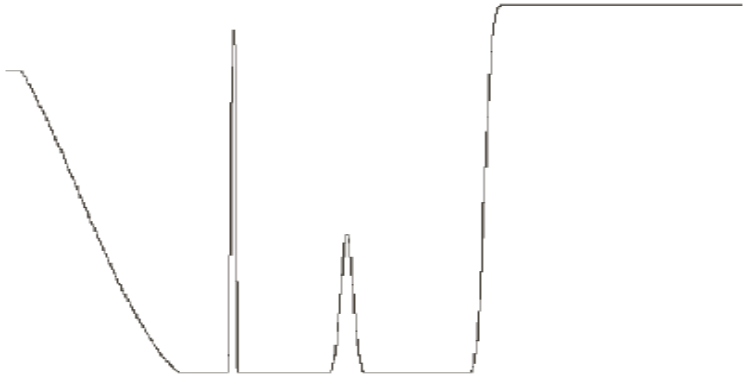
$$f(x_0 + h) + f(x_0 - h) = 2f(x_0) + h^2 f''(x_0) + O(h^2)$$

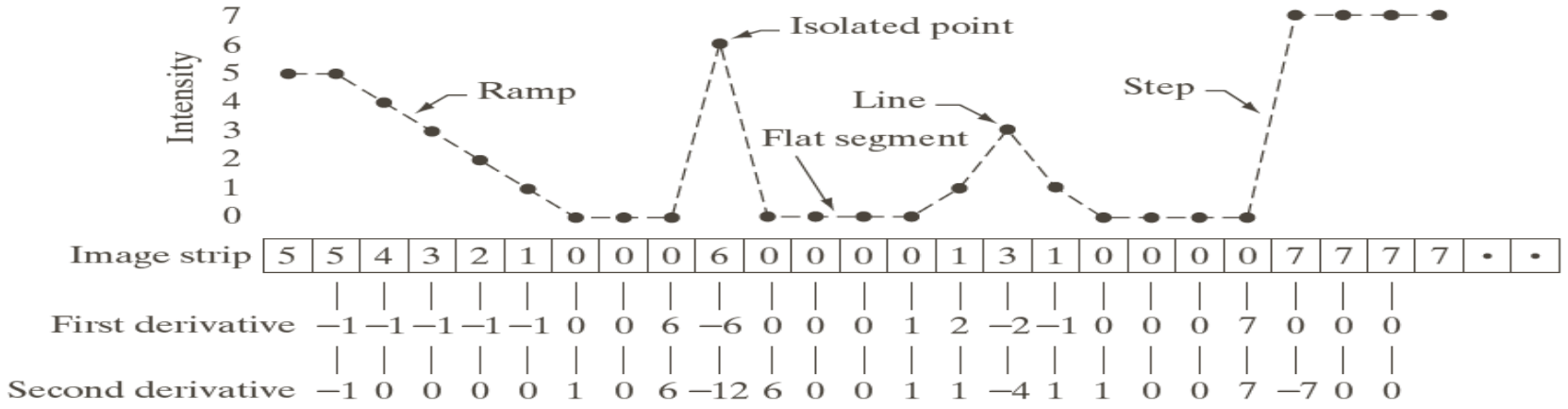
to get the (second order) second centered difference

$$f''(x_0) = \frac{f(x_0 + h) - 2f(x_0) + f(x_0 - h)}{h^2}$$

Derivative Computations

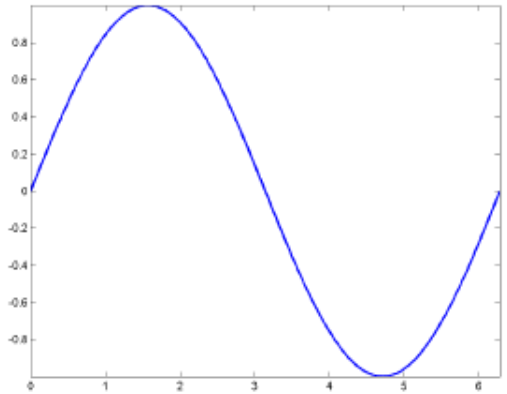




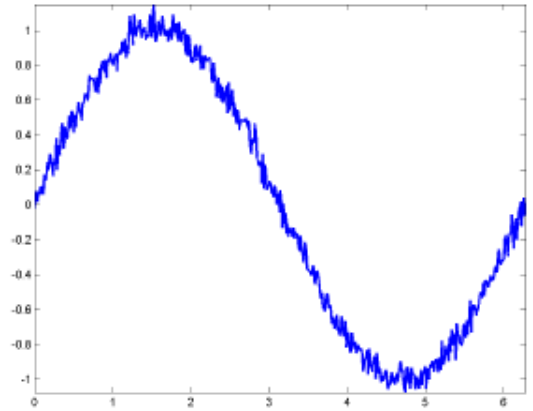


Derivative Computations

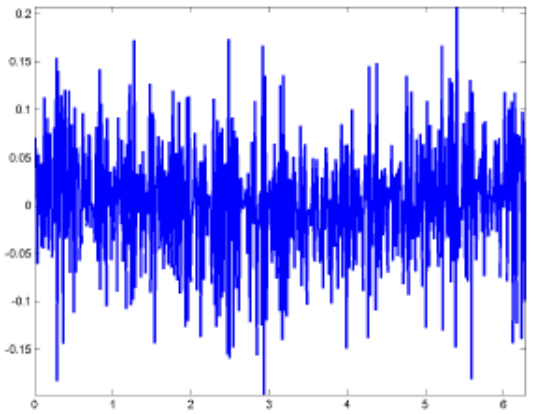
Derivatives are noise sensitive!



Original



Noisy



Derivative

Derivative Computations

Derivatives are noise sensitive!

- Second derivative, as expected, is even more sensitive to noise!
- Therefore, image smoothing before computing derivatives is required!

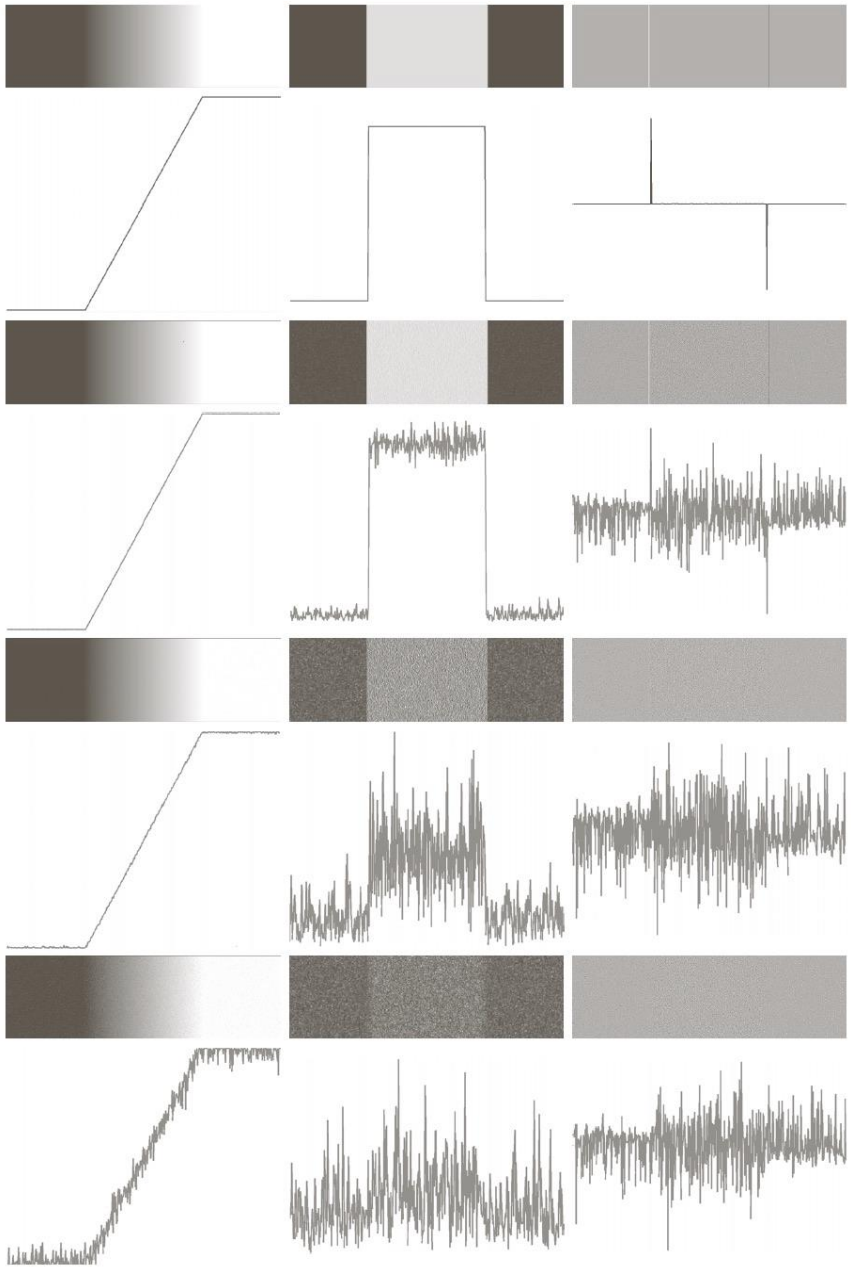
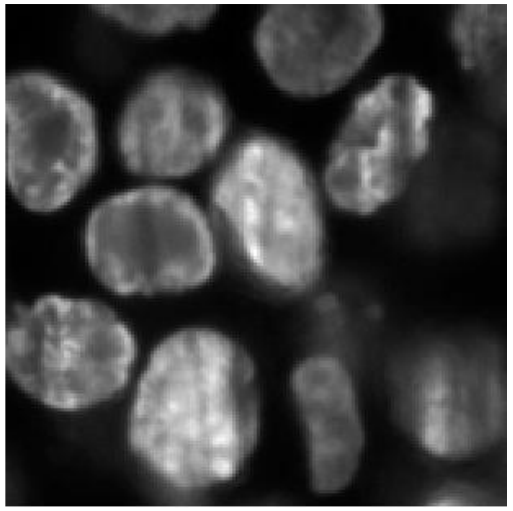


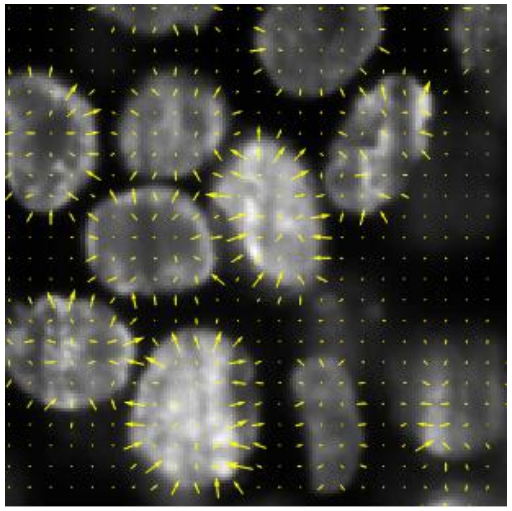
Image Gradient

We said some (many) edge operators only return information about the existence of an edge at each point.

Some edge operators return magnitude and orientation information: image gradient.

$$\nabla I(x,y) = \begin{pmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{pmatrix}$$



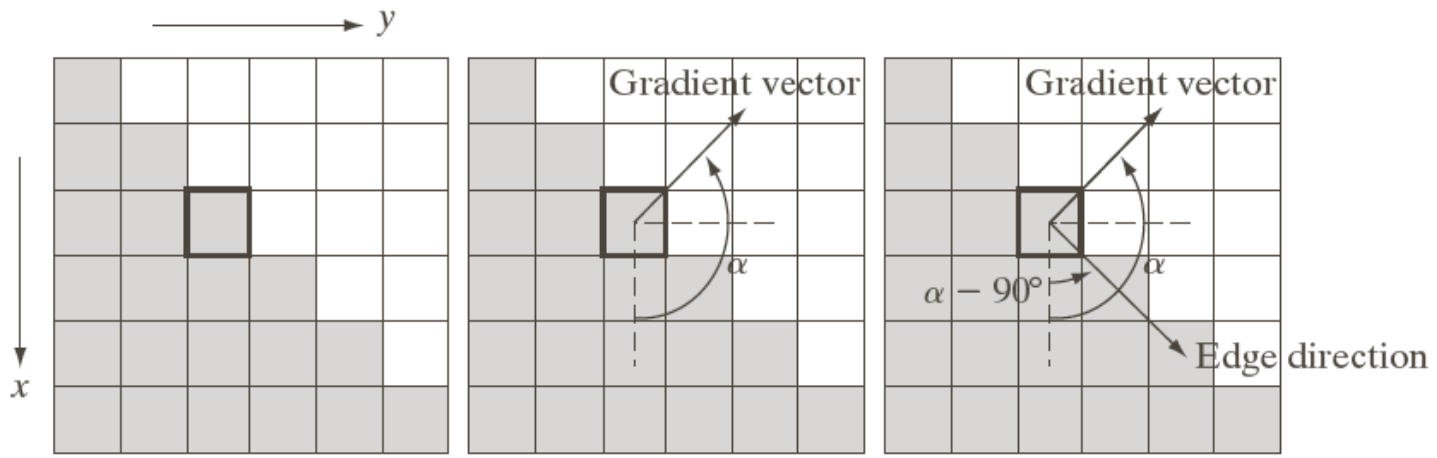


Basic Edge Detection

Compute the image gradient vector!

$$I_x = \frac{\partial I}{\partial x} = \frac{I(x+1,y) - I(x-1,y)}{2} \qquad I_y = \frac{\partial I}{\partial y} = \frac{I(x,y+1) - I(x,y-1)}{2}$$

$$\nabla I(x,y) = \begin{pmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{pmatrix}$$



Basic Edge Detection

Spatial Derivatives can be computed with Spatial Masks/Filters.
For edge detection, filter weights typically sum to zero.

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

$$R(x,y) = \sum_{k=1}^{9} w_k I_k$$

Horizontal filtering: $\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$

Roberts Operator

- Diagonal differences are investigated.

−1	0
0	1

0	−1
1	0

- First form of Roberts Operator:

$$\sqrt{[I(r,c)-I(r-1,c-1)]^2+[I(r,c-1)-I(r-1,c)]^2}$$

- Second form of Roberts Operator

$$|I(r,c)-I(r-1,c-1)|+|I(r,c-1)-I(r-1,c)|$$

Roberts Operator

- Diagonal differences are investigated.

−1	0
0	1

0	−1
1	0

- First form of Roberts Operator:

$$\sqrt{[I(r,c)-I(r-1,c-1)]^2+[I(r,c-1)-I(r-1,c)]^2}$$

- Second form of Roberts Operator

$$|I(r,c)-I(r-1,c-1)|+|I(r,c-1)-I(r-1,c)|$$

- Not symmetric around the center point!

- Few pixels are used to approximate the gradient.
  High sensitivity to noise!

Prewitt Operator

R_y			R_x		
−1	−1	−1	−1	0	1
0	0	0	−1	0	1
1	1	1	−1	0	1

- Looks for edges in both horizontal and vertical directions, then combine the information into a single metric.

- The filter responses R_x and R_y could be used to calculate Edge magnitude

$$\sqrt{R_x^{\,2} + R_y^{\,2}}$$

and edge direction.

$$\tan^{-1}\left(\frac{R_y}{R_x}\right)$$

Sobel Operator

R_y

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

R_x

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

- Similar to the Prewitt, with different mask coefficients

- The filter responses R_x and R_y could be used to calculate

Edge magnitude

$$\sqrt{R_x^{\,2} + R_y^{\,2}}$$

and edge direction.

$$\tan^{-1}\left(\frac{R_y}{R_x}\right)$$

Kirsch Compass Masks

- Detects responses to 8 major compass orientations: N, NW, W, SW, S, SE, E, and NE.
- Edge magnitude and direction are directly defined from the response values.

$$N=\begin{bmatrix}-3 & -3 & -3\\ -3 & 0 & -3\\ 5 & 5 & 5\end{bmatrix}\quad W=\begin{bmatrix}-3 & -3 & 5\\ -3 & 0 & 5\\ -3 & -3 & 5\end{bmatrix}\quad S=\begin{bmatrix}5 & 5 & 5\\ -3 & 0 & -3\\ -3 & -3 & -3\end{bmatrix}\quad E=\begin{bmatrix}5 & -3 & -3\\ 5 & 0 & -3\\ 5 & -3 & -3\end{bmatrix}$$

$$NW=\begin{bmatrix}-3 & -3 & -3\\ -3 & 0 & 5\\ -3 & 5 & 5\end{bmatrix}\quad SW=\begin{bmatrix}-3 & 5 & 5\\ -3 & 0 & 5\\ -3 & -3 & -3\end{bmatrix}\quad SE=\begin{bmatrix}5 & 5 & -3\\ 5 & 0 & -3\\ -3 & -3 & -3\end{bmatrix}\quad NE=\begin{bmatrix}-3 & -3 & -3\\ 5 & 0 & -3\\ 5 & 5 & -3\end{bmatrix}$$

Robinson Compass Masks

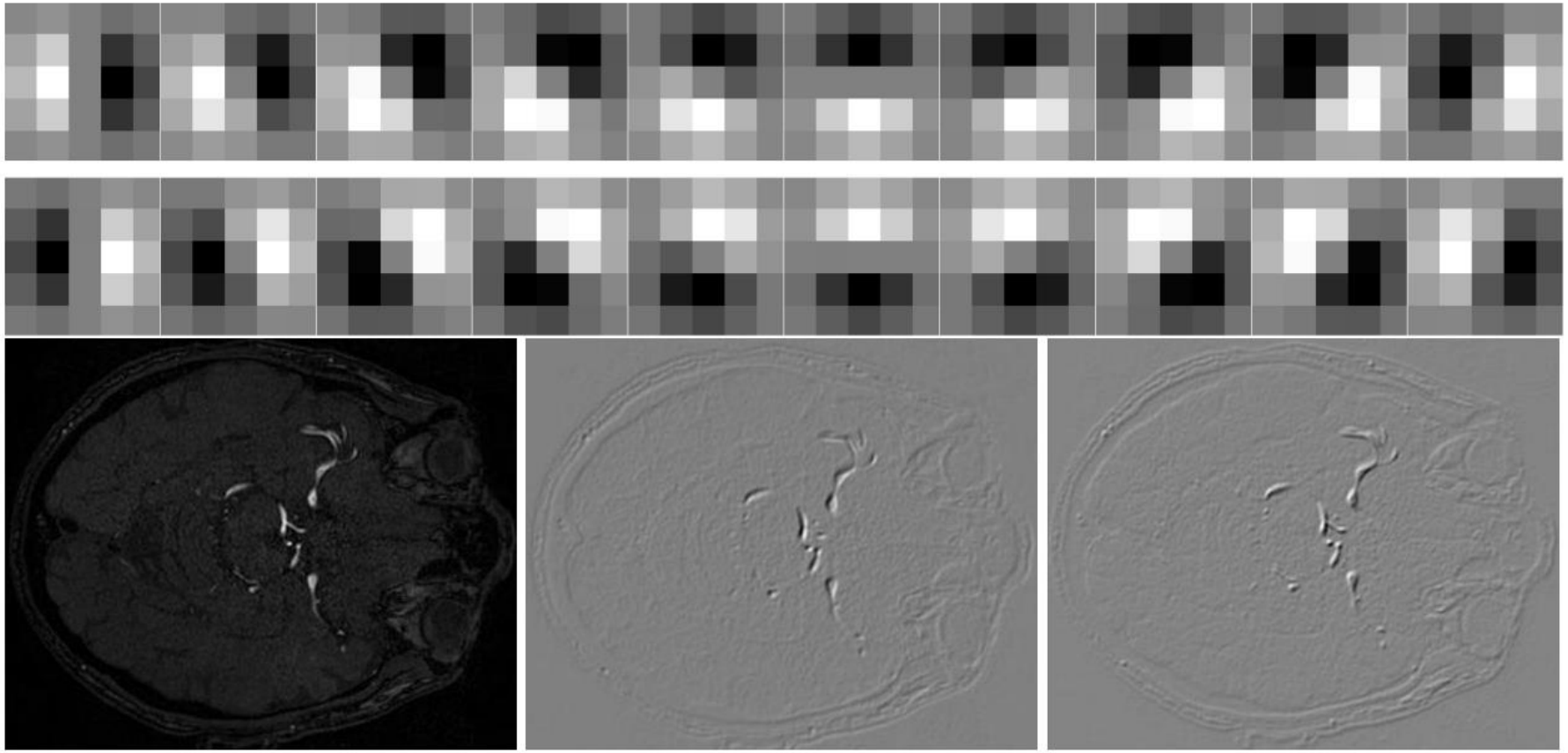
- Similar to the Kirsch masks, with mask coefficients of 0, 1, and 2:

$$W = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad N = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad S = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad E = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

$$SE = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{bmatrix} \quad SW = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix} \quad NE = \begin{bmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{bmatrix} \quad NW = \begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

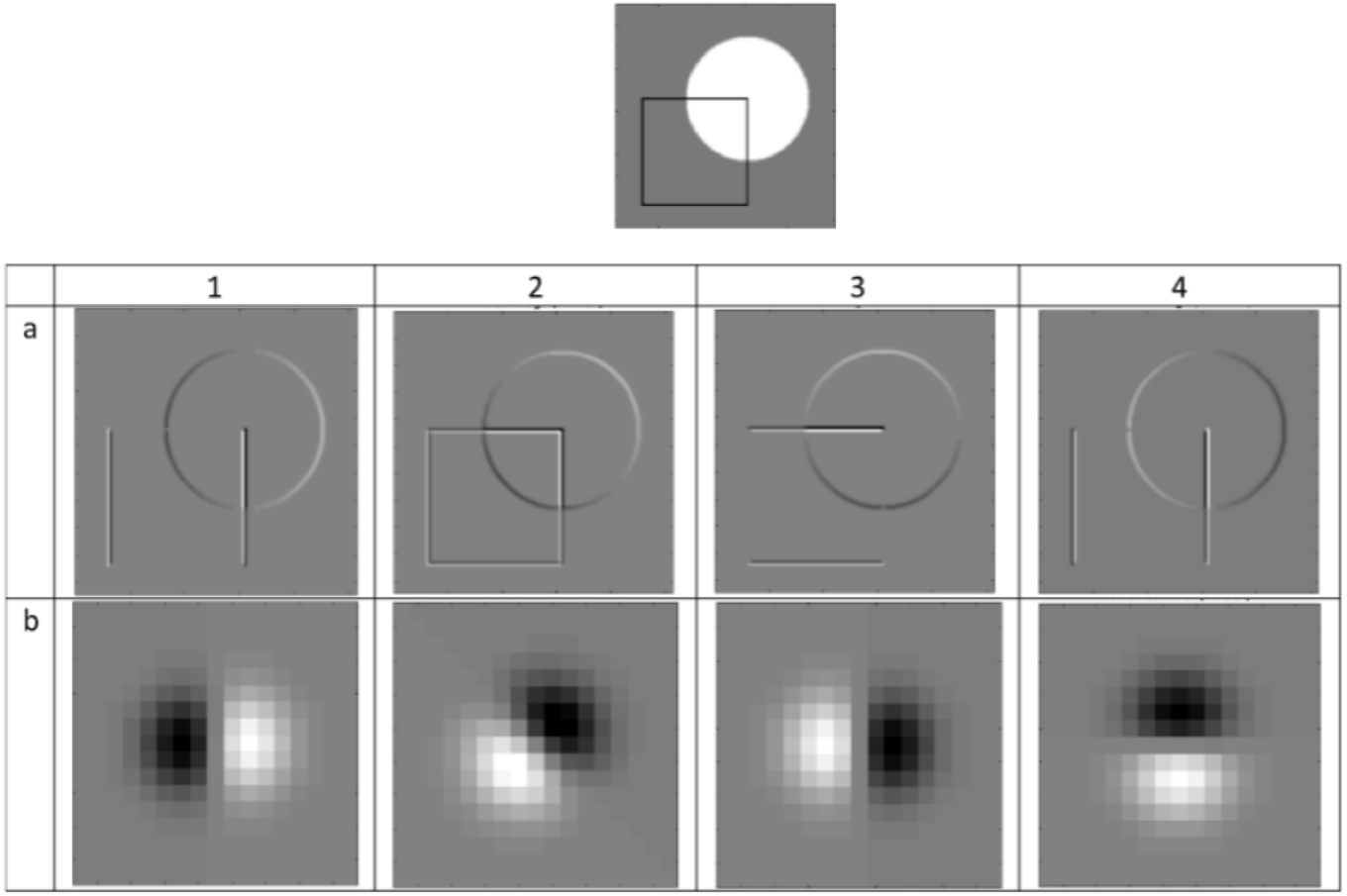
Steerable Filters

- Divides the directional space to more than 8 components.



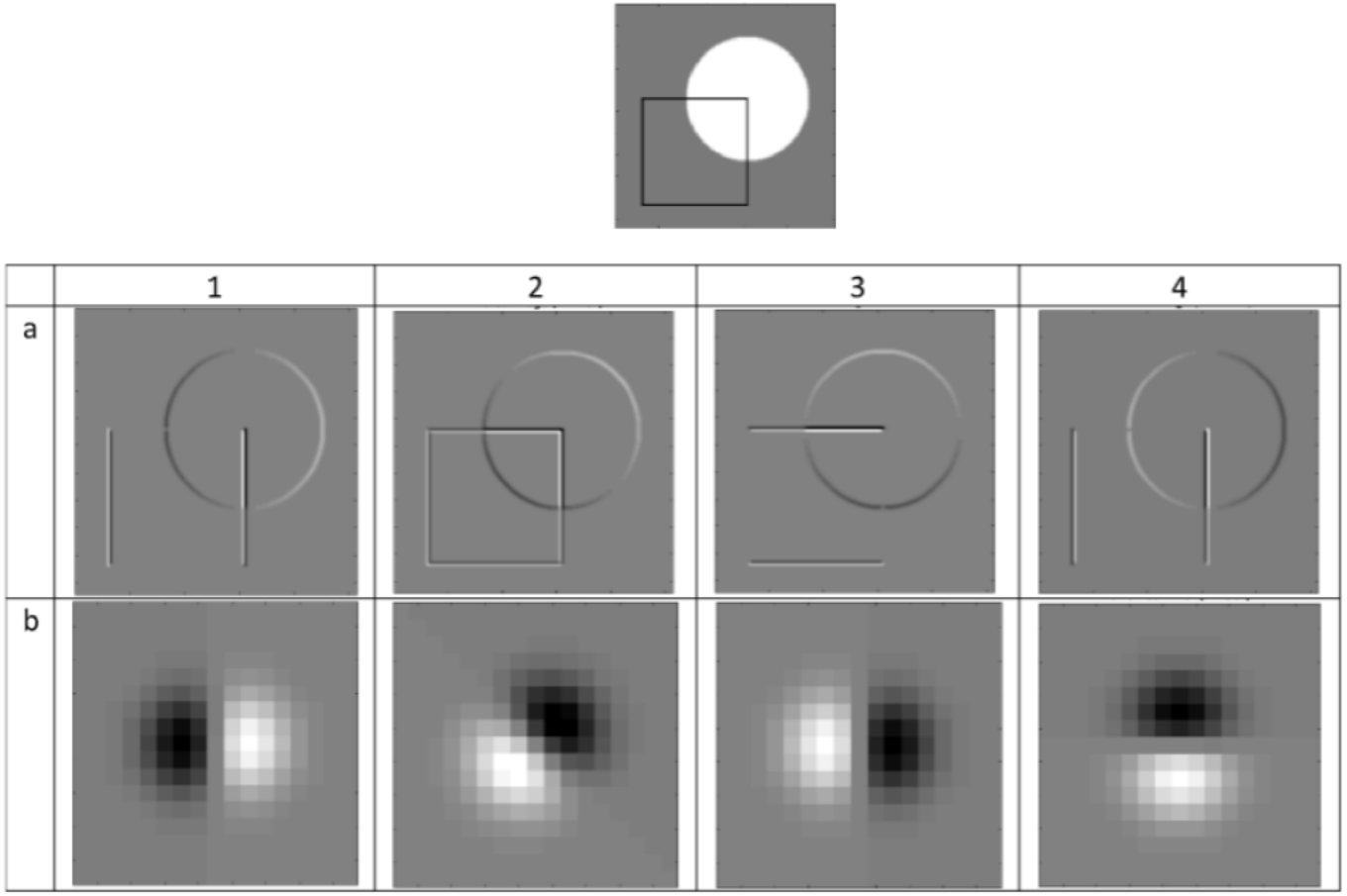
Q:

- For the example image, some image filters and their responses (cross correlation) are given below in changed order. Match each filter with its response.



A:

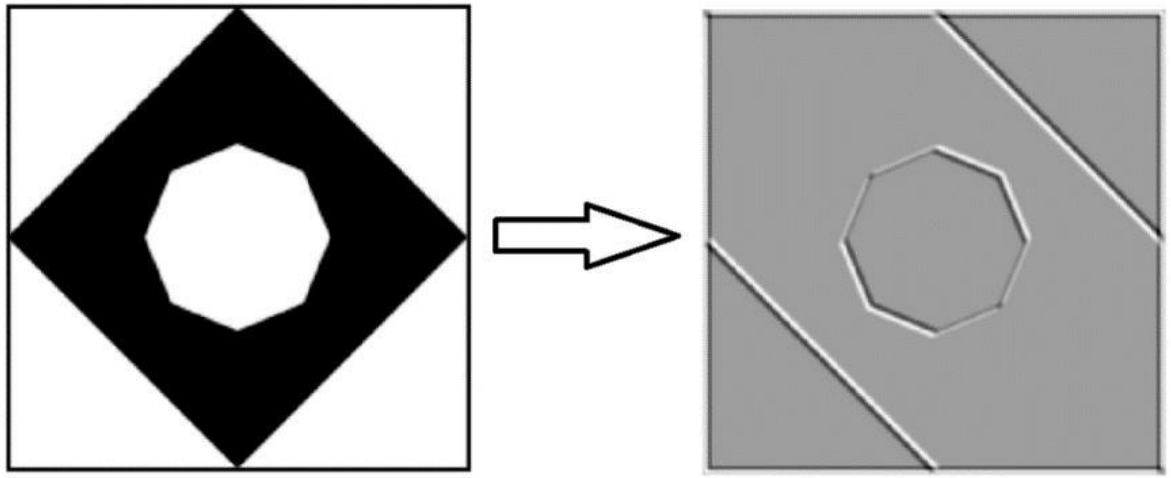
- For the example image, some image filters and their responses (cross correlation) are given below in changed order. Match each filter with its response.



a1-b3, a2-b2, a3-b4, a4-b1

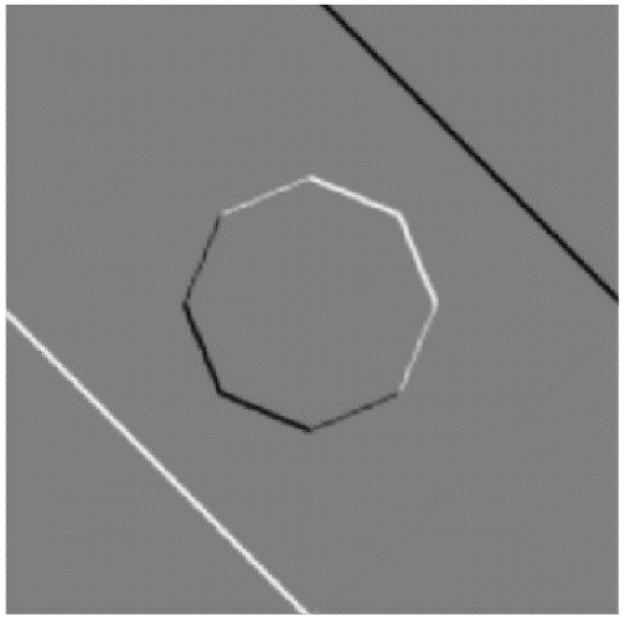
Q:

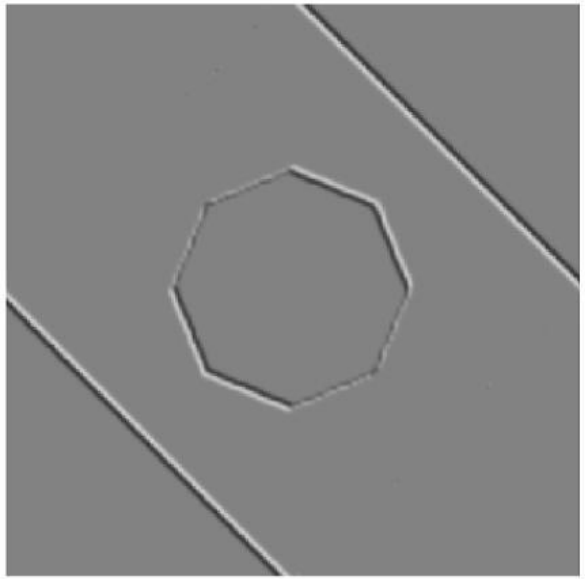
- Using the Robinson Compass masks, how could we obtain the second picture from the first picture?



A:

- Use SW mask twice!





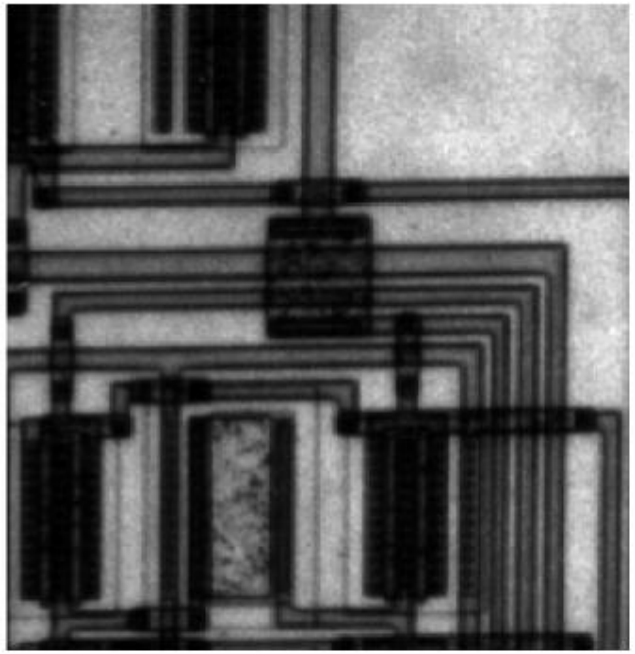
Line Detection Masks/Filters

- The maximum responses will occur at those desired locations with horizontal line features.

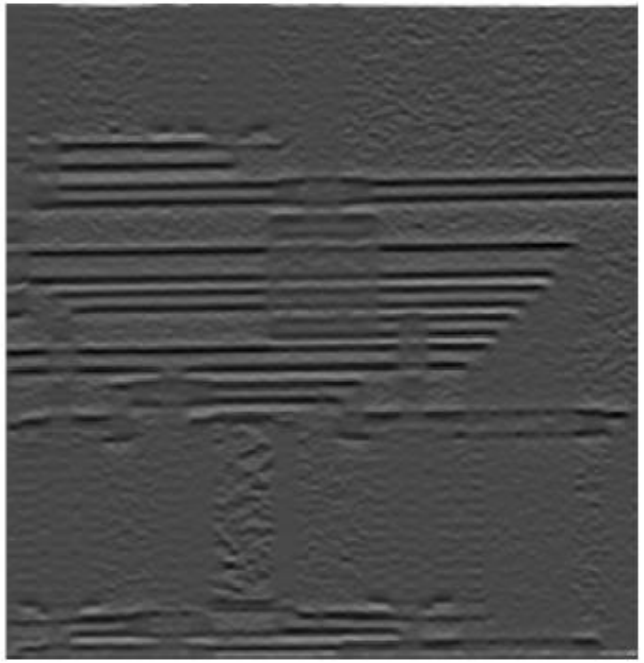
−1	−1	−1	2	−1	−1	−1	2	−1	−1	−1	2
2	2	2	−1	2	−1	−1	2	−1	−1	2	−1
−1	−1	−1	−1	−1	2	−1	2	−1	2	−1	−1
Horizontal			+45°			Vertical			−45°		

Question: Which filter mask above will give the maximum response and help you detect the lines in this image?

Line Detection: Line Filter



Original.



Horizontal Line Filter.

Edges via Gradient Magnitude

$$\left|\nabla I\right| = \sqrt{I_x^2 + I_y^2}$$

$$\nabla I(x,y) = \begin{pmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{pmatrix} = \begin{pmatrix} I_x \\ I_y \end{pmatrix}$$

Original

With Noise

Gradient magnitude original.

Gradient magnitude noisy.

Edges via Operators

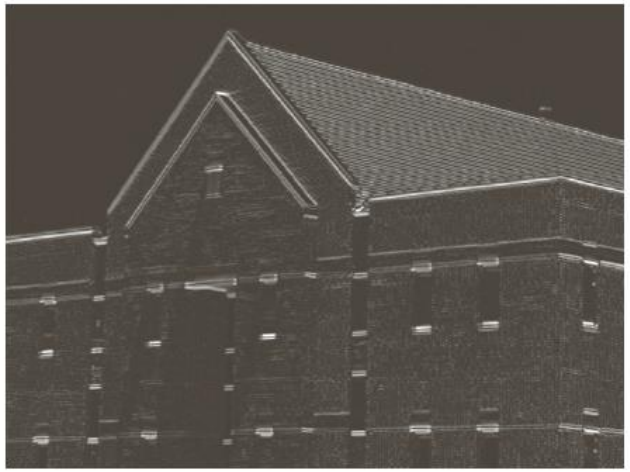
Prewitt, noisy.

Sobel, noisy.

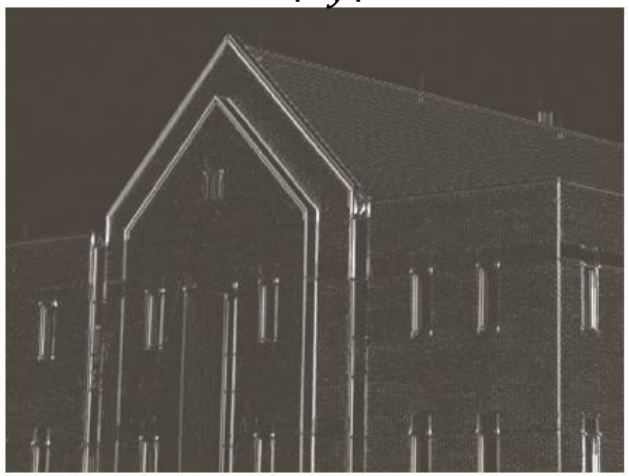
∇_x, ∇_y

Original

$|\nabla_x|$



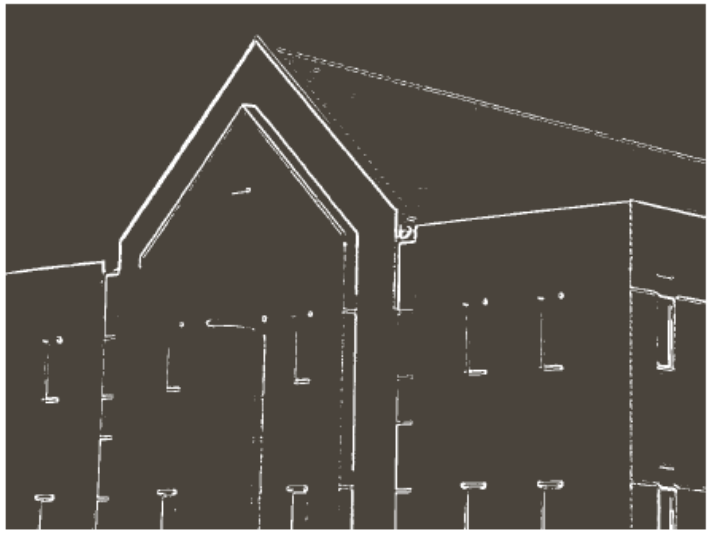
$|\nabla_y|$



$|\nabla_x| + |\nabla_y|$

Binary Edge Map

- Thresholding the binary image may help cleaning the noise.



Second-derivative based operator

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

$$L(x,y) = I(x,y) - \frac{1}{4}\Big(I(x,y+1) + I(x,y-1) + I(x+1,y) + I(x-1,y)\Big)$$

0	−1	0
−1	4	−1
0	−1	0

- This is called the Laplacian operator:

$$\Delta I(x,y) = \nabla^2 I(x,y) = \frac{\delta^2 I(x,y)}{\delta x^2} + \frac{\delta^2 I(x,y)}{\delta y^2} = I_{xx} + I_{yy}$$

- For a derivative mask the coefficients in the mask sum to 0.

$$\sum_{k=1}^{9} w_k = 0$$

Point Detection

- Based on the 2nd derivatives, i.e. wero crossings using a 4-pt neighborhood.

0	1	0
1	−4	1
0	1	0

- Can include diagonal terms in the mask, using an 8-pt neighborhood.

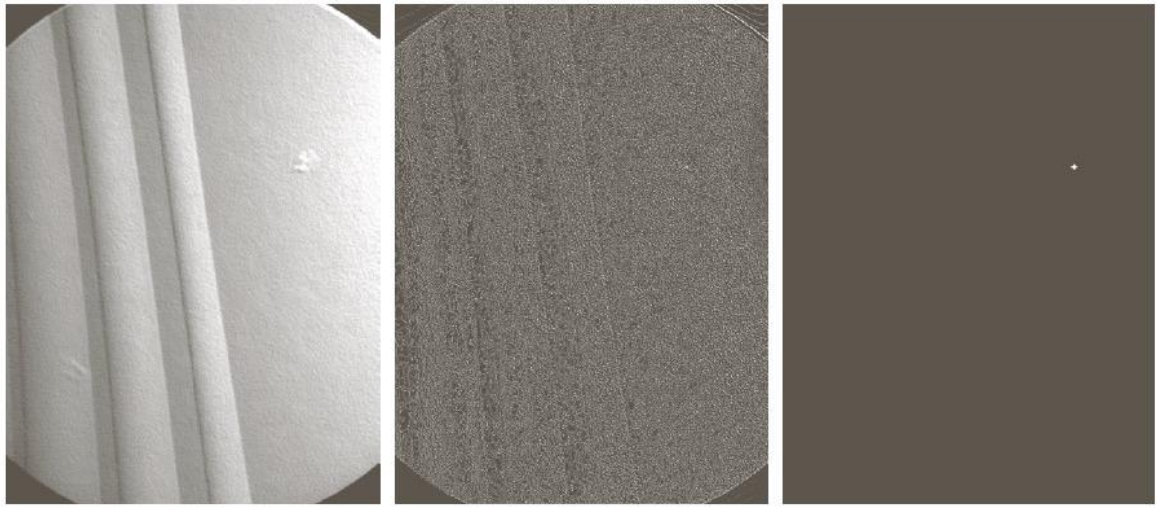
1	1	1
1	−8	1
1	1	1

Point Detection

- At each pixel coordinate: Compare the absolute value of the mask response with a threshold

- $P = \begin{cases} 1, & |R| > T \\ 0, & otherwise \end{cases}$

1	1	1
1	−8	1
1	1	1



Laplacian Operators (*)

- • Laplacian Masks for 4 and 8 neighborhoods

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

- Mask with stressed significance of the central pixel or its neighborhood

$$\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix} \qquad \begin{bmatrix} -2 & 1 & -2 \\ 1 & 4 & 1 \\ -2 & 1 & -2 \end{bmatrix}$$

Notes on the Laplacian operator

- It is an isotropic filter.
  - Invariant to rotation.

- As a second-order derivative, the Laplacian typically is unacceptably sensitive to noise.
- Laplacian produces double edges.

- The Laplacian is unable to detect edge direction.

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

More advanced techniques for Edge Detection

- Earlier operators we discussed have fixed kernel size (e.g. 3x3 or 5x5 or 2x2)

- To take into account the scale of the features, size of the operators should change!
  - Small operators detect sharply focused fine edges.
  - Large operators detect blurry edges.

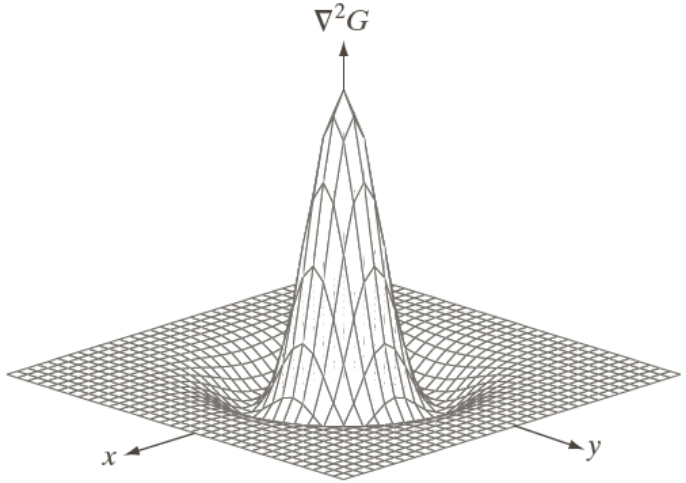
- Typically we UTILIZE GAUSSIAN function to pre-filter.

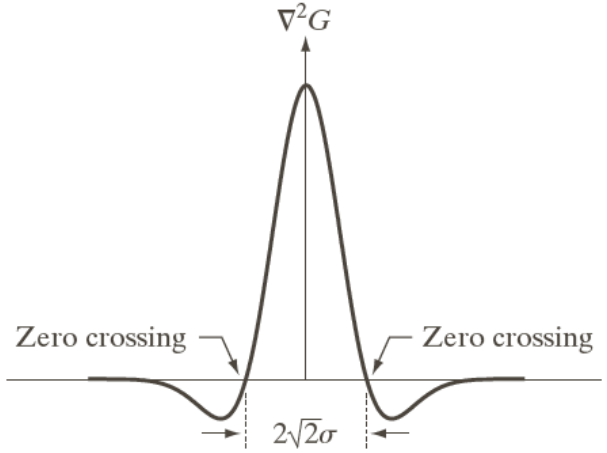
Laplacian of Gaussian (LoG)

- Second derivative of the Gaussian function could be used as an edge detector.

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

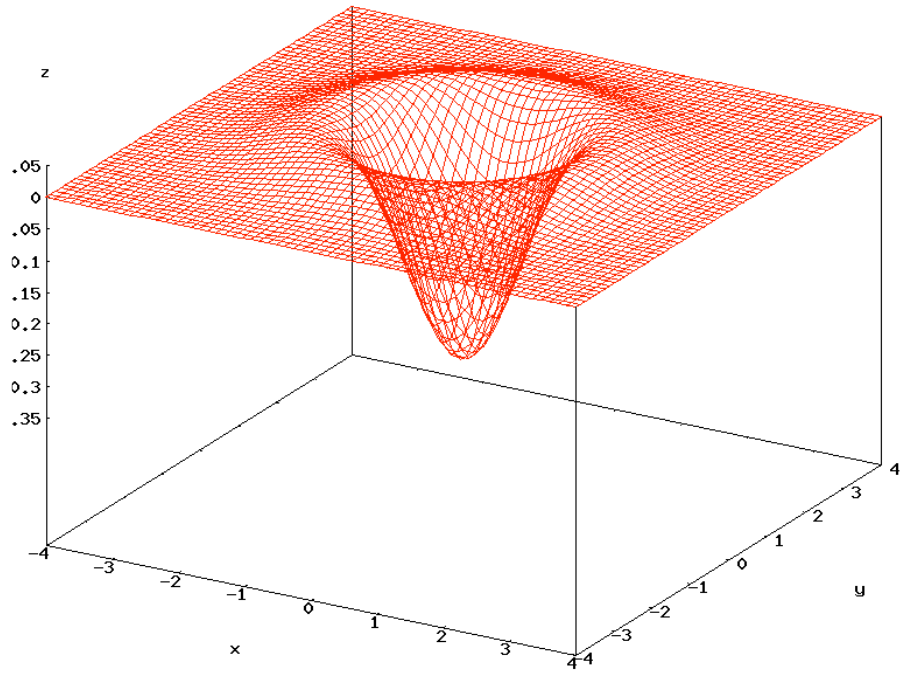
$$\nabla^2 G(x,y) = \left[\frac{x^2+y^2-2\sigma^2}{\sigma^4}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}$$





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

Laplacian of Gaussian (LoG)



- Masks of arbitrary size can be generated by
  - sampling Gaussian function to the desired nxn size
  - Convolving it with a Laplacian mask
- Gaussian part blurs the image, reducing noise.
- Convolution is a linear operator:

$$J(x,y) = \left[\nabla^2 G(x,y)\right] * I(x,y) \qquad J(x,y) = \nabla^2\left[G(x,y) * I(x,y)\right]$$

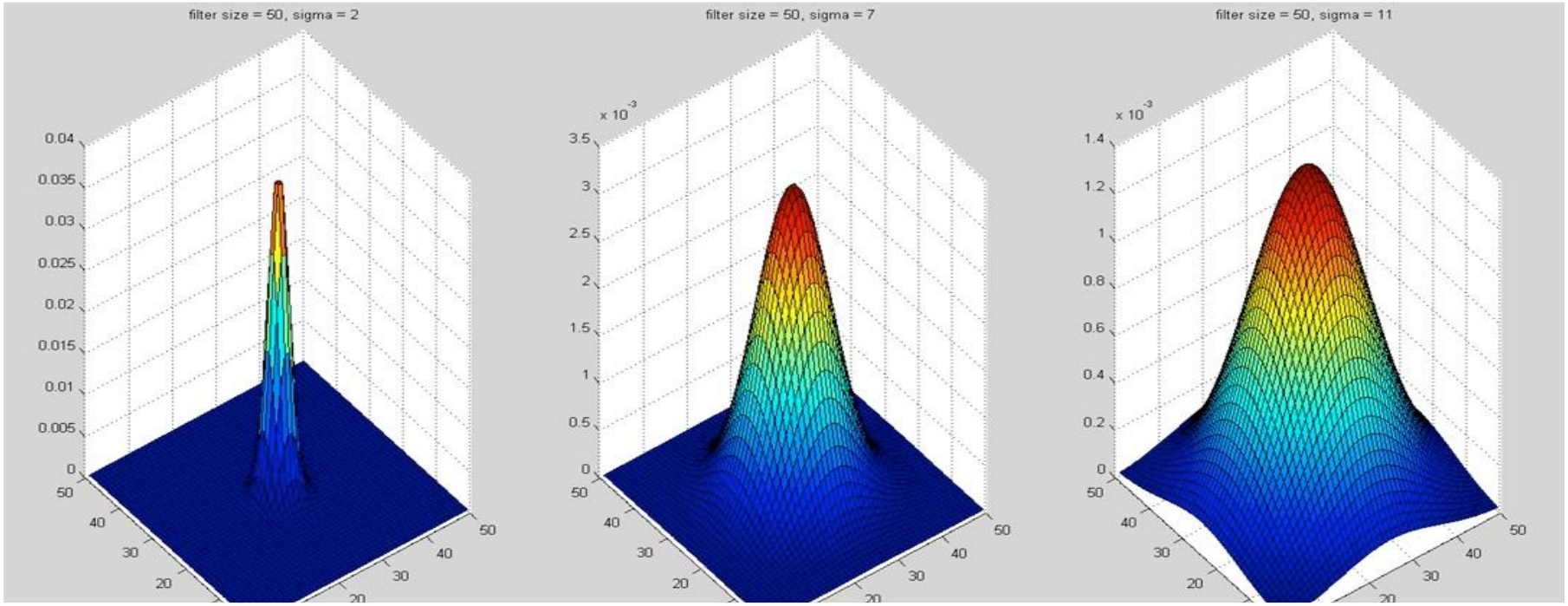
- It is equivalent: we smooth the image with a Gaussian filter then compute the Laplacian of the result

Laplacian of Gaussian (LoG)

- Summary of LoG Edge detection algorithm:
  - Filter the input image with an nxn Gaussian lowpass filter
  - Compute the Laplacian of the resulting image from using the 3x3 Laplacian mask
  - Find the zero-crossings of the image

Gaussian Function:
How to choose sigma (the right scale)?

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



- Filter image with various σ.
- Analyze the image at different scales (i.e. Sigma) e.g. Apply the edge filters: Combine the resulting edge maps by keeping only edges that are common to all maps

Scale Space Idea

- Many approaches are developed which apply filtering masks of multiple sizes and then analyze the behaviour of edges at these different *scales* of filtering.
- The basic idea is to exploit the fact that at higher scales, larger filtering masks result in robust but displaced edges. The location of these edges can be determined at smaller scales.

σ increasing

Multiscale Processing of Images

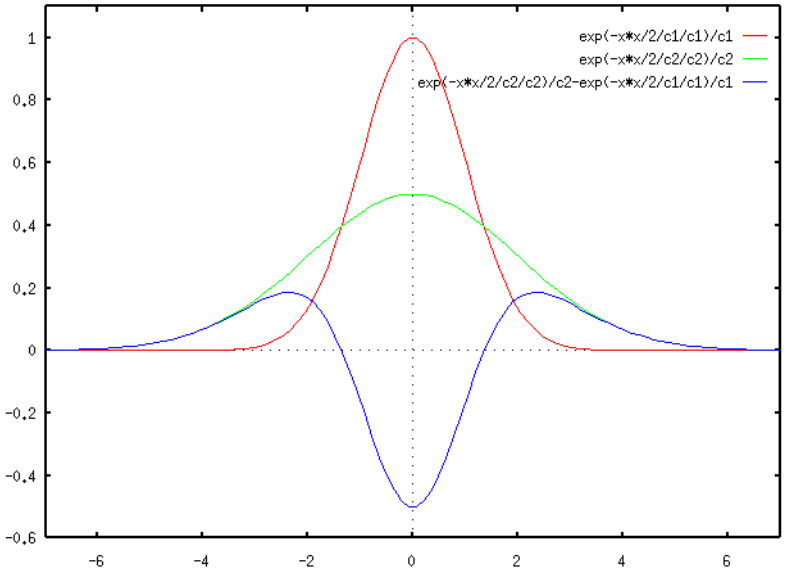
- It is reasonable to think of sigma as setting the *scale* at which we preserve information in the convolved image.
- Structure on a scale small compared with sigma will be removed, while structure on a larger scale is retained.
- **Trade-off:** If a small filter is used, there is likely to be more noise due to insufficient averaging. For large filters, edges which are close to each other may get merged by smoothing and may be detected as only a single edge.

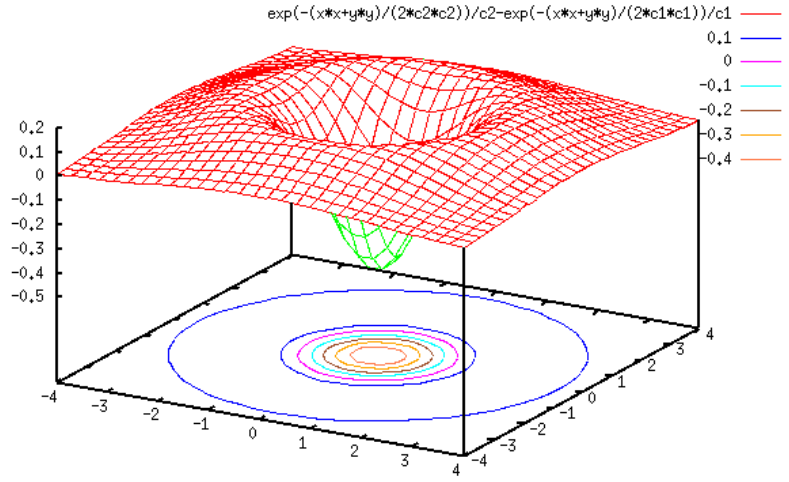
Difference of Gaussians (DoGs)

- DoG is a good approximation to the Laplacian of the Gaussian.

- We can look at the form of the 1-D DoG by generating two 1-D masks and subtracting one from the other. It turns out that the best approximation to the Laplacian occurs if the larger mask has a sigma about 1.6 times that of the smaller.

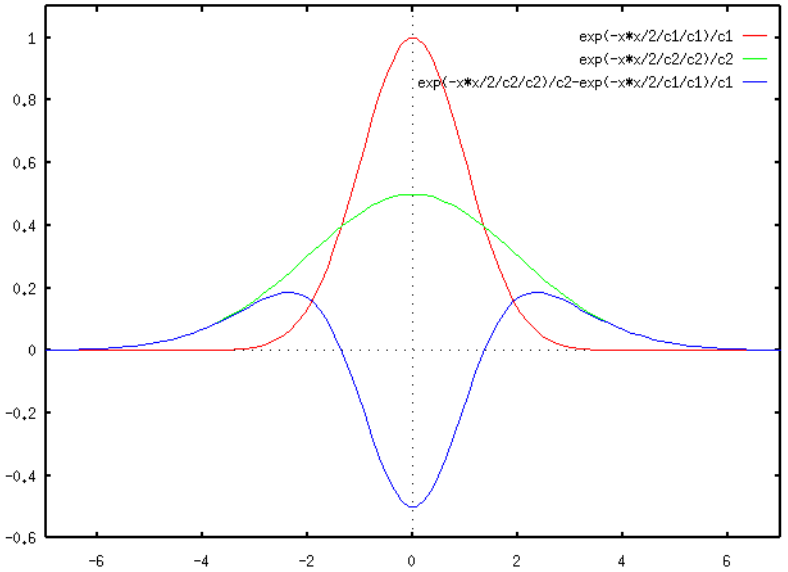
$$DoG(x,y) = \frac{1}{2\pi\sigma_1^2} e^{-\frac{x^2+y^2}{2\sigma_1^2}} - \frac{1}{2\pi\sigma_2^2} e^{-\frac{x^2+y^2}{2\sigma_2^2}}$$





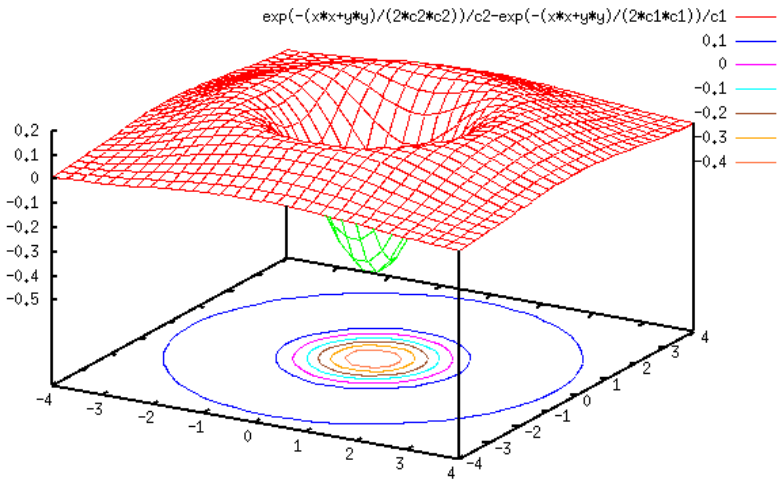
Difference of Gaussians (DoG)

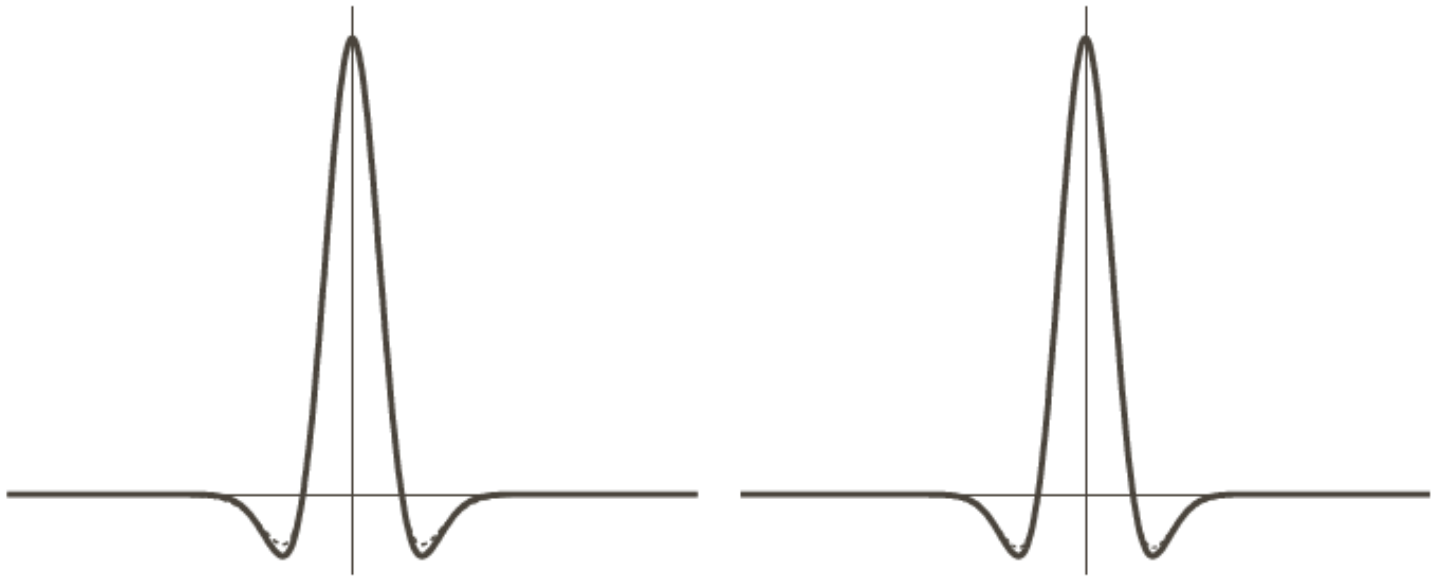
$$DoG(x,y) = \frac{1}{2\pi\sigma_1^2} e^{-\frac{x^2+y^2}{2\sigma_1^2}} - \frac{1}{2\pi\sigma_2^2} e^{-\frac{x^2+y^2}{2\sigma_2^2}}$$



- If $\sigma = \frac{\sigma_1^2\sigma_2^2}{\sigma_1^2-\sigma_2^2}\ln[\frac{\sigma_1^2}{\sigma_2^2}]$ and amplitudes are approximately = 1.75:1),

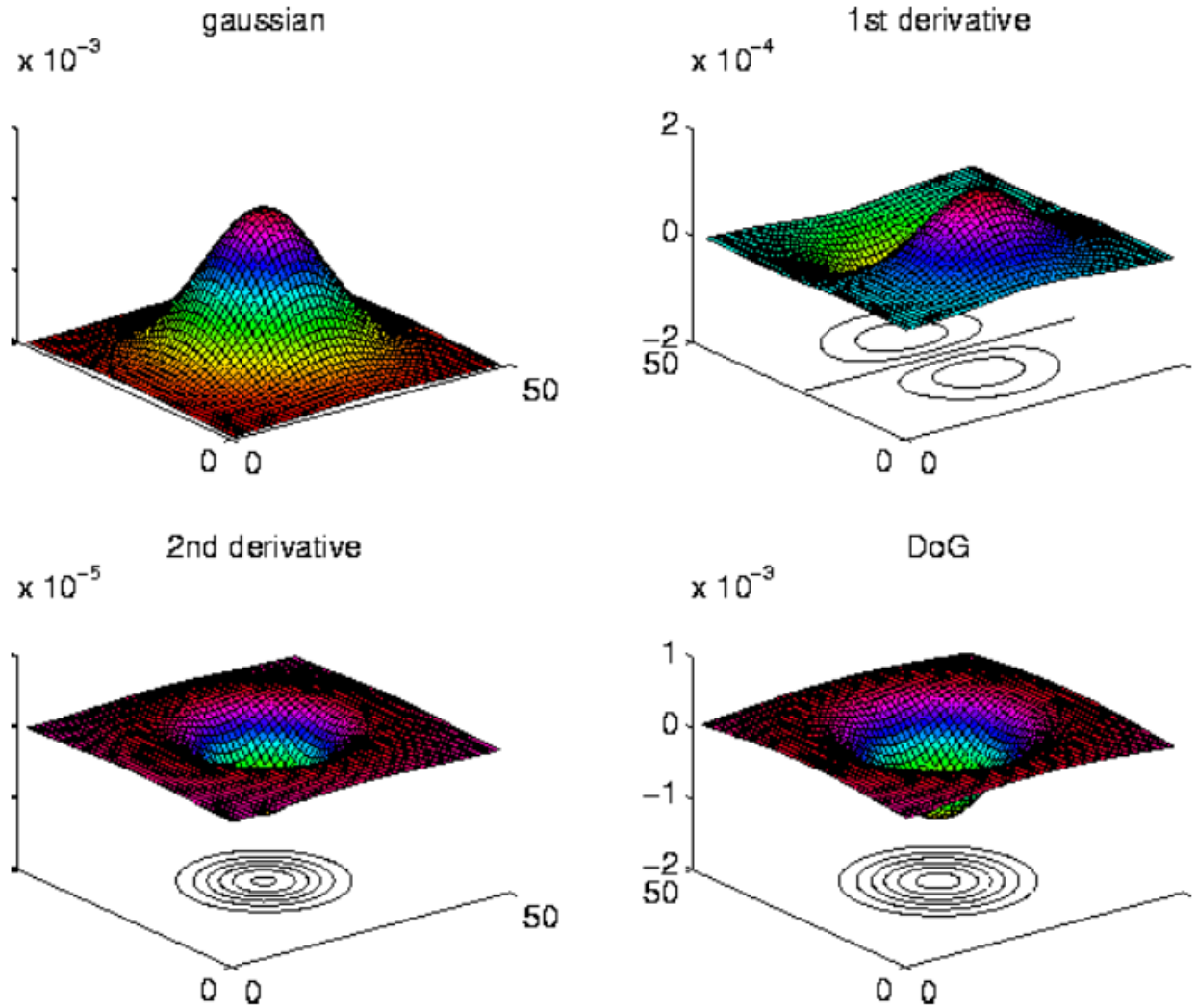
LoG and DoG have similar profiles.





a b

FIGURE 10.23
(a) Negatives of the LoG (solid) and DoG (dotted) profiles using a standard deviation ratio of 1.75:1. (b) Profiles obtained using a ratio of 1.6:1.


gaussian
x 10^-3
50
0 0
1st derivative
x 10^-4
2
0
−2
50
0 0
2nd derivative
x 10^-5
50
0 0
DoG
x 10^-3
1
0
−1
−2
50
0 0

Continue with zero-crossing detection (LoG Edge detection)

- To detect zero-crossings of the resulting image J

- Use a 3x3 neighborhood centered around each pixel (x,y)

- A zero-crossing implies that :
  - signs of at least two of its opposing neighboring pixels must differ:
    - e.g. Up/Down, Left/Right, Two diagonals
  - Also use a threshold to compare these, i.e. not only must the signs differ, but absolute value of their numerical difference must exceed a threshold

Laplacian of Gaussian (LoG)

Original.

Noisy.

- Edges are at zero crossings of the LoG output
- Need to be thresholded based on edge strength to remove spurious edges

Laplacian of Gaussian

Small variance.

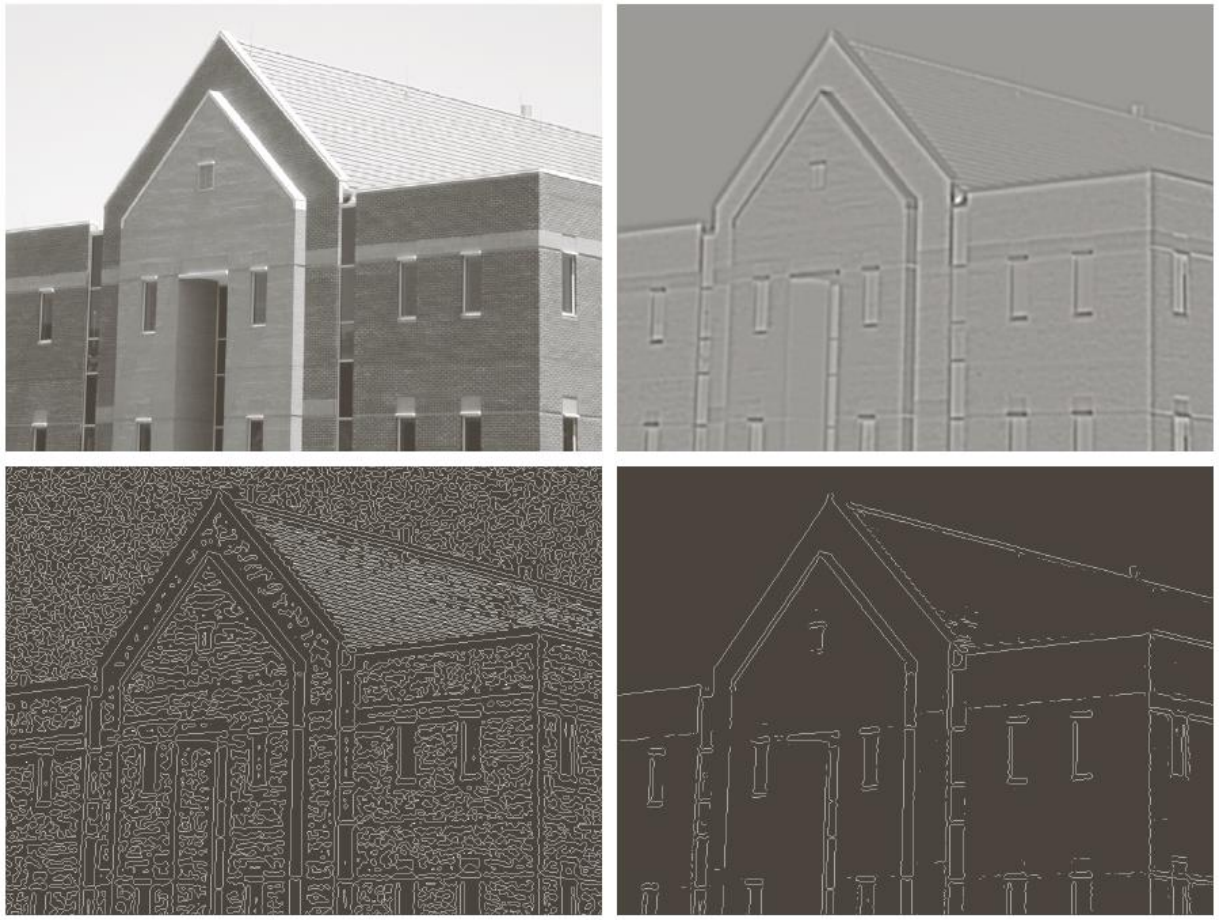
Large variance.

Laplacian of Gaussian

Low threshold.

High threshold.

Laplacian of Gaussian



a b
c d

FIGURE 10.22
(a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1]. (b) Results of Steps 1 and 2 of the Marr-Hildreth algorithm using $\sigma = 4$ and $n = 25$. (c) Zero crossings of (b) using a threshold of 0 (note the closed-loop edges). (d) Zero crossings found using a threshold equal to 4% of the maximum value of the image in (b). Note the thin edges.

The Canny edge detector

- Smoothing with Gaussian
- Gradient computation
- Non-maxima suppression
- Hysteresis thresholding and connectivity analysis
- Edge thinning

Using Deep Learning

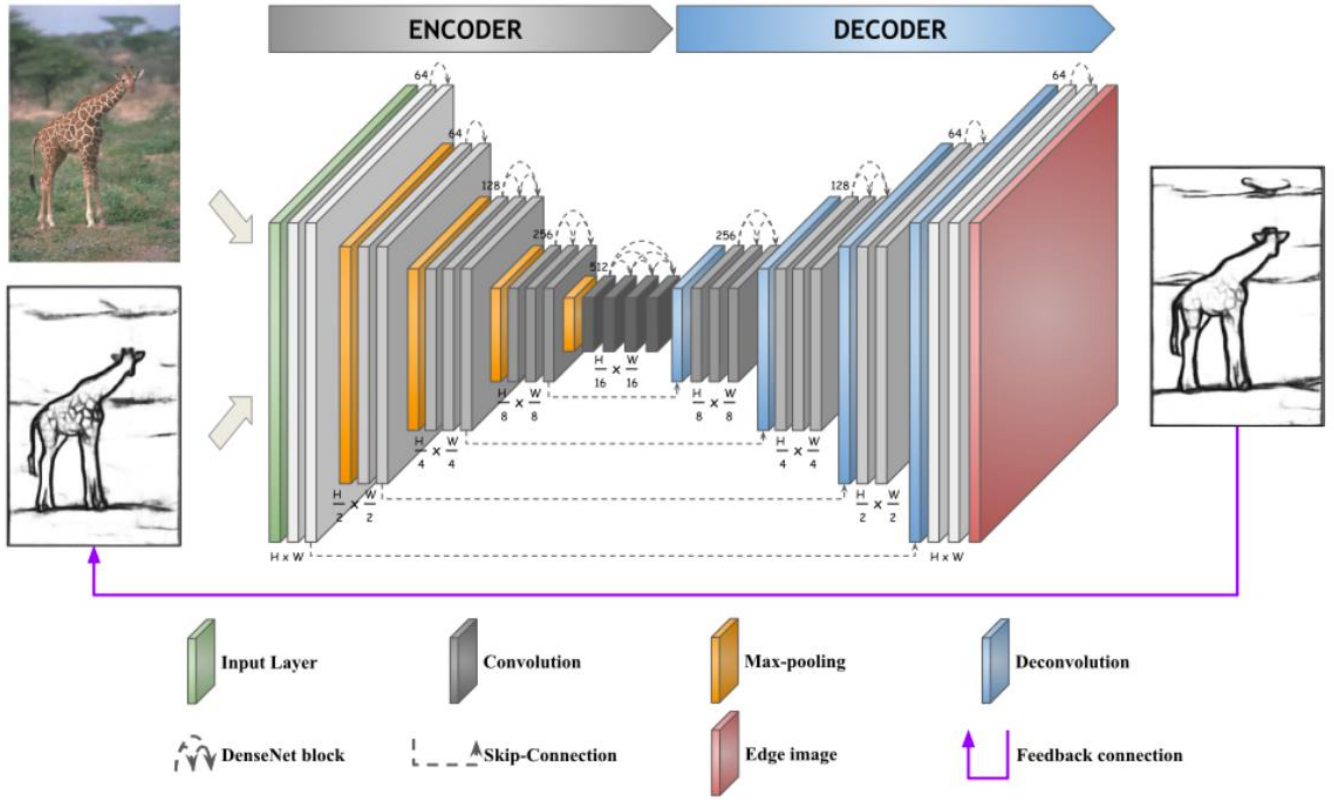


Figure 1: The architecture of our Recursive Encoder-Decoder Network with Skip Connections (RED-NET). Encoder-decoder network is at the heart of our design which consists of DenseNet blocks. There are four skip-connections that connects one layer of the encoder to a corresponding layer of the decoder. The feedback connection enables a deeper network with no extra parameters.