Machine Vision

Lecture Set – 06

Edge Detection

Huei-Yung Lin

Robot Vision Lab

What is an Image Feature?

What is an Image Feature?

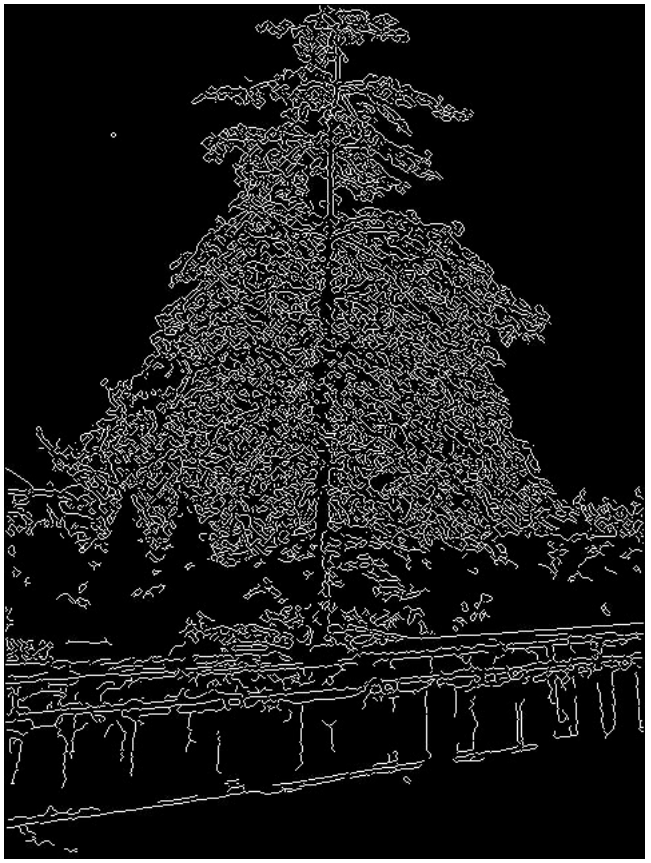


Image Features

- Image features in computer vision
  - Global feature:
    - A global property of an image
    - e.g. the average grey value, the area in pixel, etc.
  - Local feature:
    - A part of the image with some special properties
    - e.g. a circle, a line, a textured regions, etc.

Image Features

- Image features (a practical definition): local meaningful detectable parts of the image
  - Points
  - Edges: step edges, line edges
  - Contours: closed contours are boundaries
  - Regions: similar color, similar feature, etc.

Remarks

- Most vision systems begin by detecting and locating some features in the input images
- In 3-D vision, feature extraction is an *intermediate* step, not the goal
  - We do not extract lines just to obtain line maps
  - We extract lines to navigate robots in corridors or for camera calibration
- Feature extraction is for certain purpose of the system
- "Perfect" feature extraction is not necessary (and also not possible?)

Edge Detection

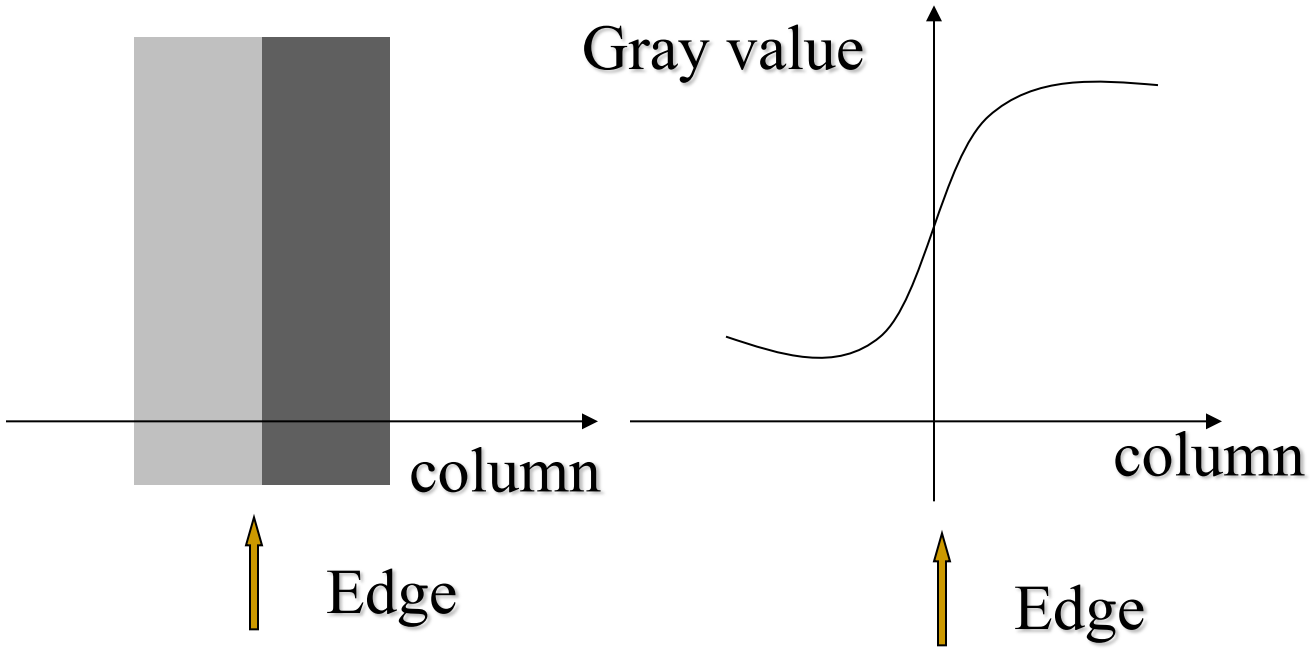
- Edges typically occur on the boundary between two different regions in an image
- Edge detection is frequently the first step in recovering information from images
- An edge is a significant local change in the intensity
  - Usually associated with a discontinuity in *either the image intensity or its first derivative*
  - The discontinuities can be step or line (ideally)
  - In reality, step becomes ramp and line becomes roof due to the smoothing of sharp edges in most images

Step Ramp Line Roof

5/2/2023

Edges

- They happen at places where the image values exhibit sharp variation



Definitions

- An edge point is a point with coordinate $[i, j]$ at the location of a significant local intensity change
- An edge fragment corresponds to the i and j coordinates of an edge and the edge orientation θ, which may be the gradient angle
- An edge detector is an algorithm that produces a set of edges (edge points/edge fragments) from an image
- A contour is a list of edges or the mathematical curve that models the list of edges
- Edge linking is the process of forming an ordered list of edges from an *unordered list*
- Edge following is the process of searching the image to determine contours

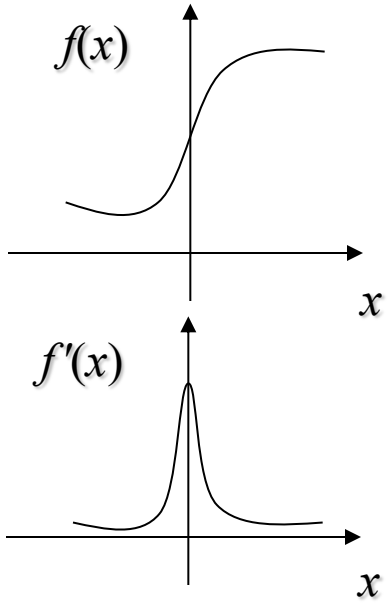
Edge Detection

- The term edge is commonly used for either edge points or edge fragments
  - An edge point may be the integer row and column indices of the pixel where the edge was detected or "at subpixel resolution"
  - An edge fragment may be conceptualized as a small line segment about the size of a pixel, or as a point with an orientation attribute

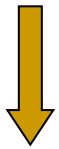
- There are three different kind of edges
  - Correct edges
  - False edges
  - Missing edges (true but not detected)

Edge Detection Scheme

- The basic approach to edge detection is to compute "spatial derivatives" of the intensity image
- The act of taking spatial derivatives is usually approximated by convolution



Edge = sharp variation



Large first derivative

Gradient

- The gradient is a measure of change in a function
- Significant changes in gray values can be detected by using a "discrete approximation" to the gradient
- The gradient is a 2-D equivalent of the first derivate and is defined as a vector

$$\mathbf{G}[f(x,y)] = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \partial f / \partial x \\ \partial f / \partial y \end{bmatrix}$$

  - The vector $\mathbf{G}[f(x,y)]$ points in the direction of the maximum rate of increase of the function $f(x,y)$
  - The magnitude of the gradient given by $G[f(x,y)] = \sqrt{G_x^2 + G_y^2}$ equals the maximum rate of increase of $f(x,y)$ per unit distance

Approximation of Gradient

- The absolute values are commonly used to approximate the gradient magnitude

$$G[f(x,y)] \approx |G_x| + |G_y| \quad \text{or} \quad G[f(x,y)] \approx \max(|G_x|, |G_y|)$$

- The direction of gradient is define as

$$\alpha(x,y) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

- The magnitude of the gradient is "independent" of the direction of the edge
- Such operators are called isotropic operators

Digital Approximation

- For digital images, the gradient approximation can be

$$G_x \approx f[i, j+1] - f[i, j] \quad \text{and} \quad G_y \approx f[i, j] - f[i+1, j]$$

- Simple convolution masks:
  - 2×1 and 1×2

-1	1

1
-1

  - 2×2 and 2×2 (why?)

-1	1
-1	1

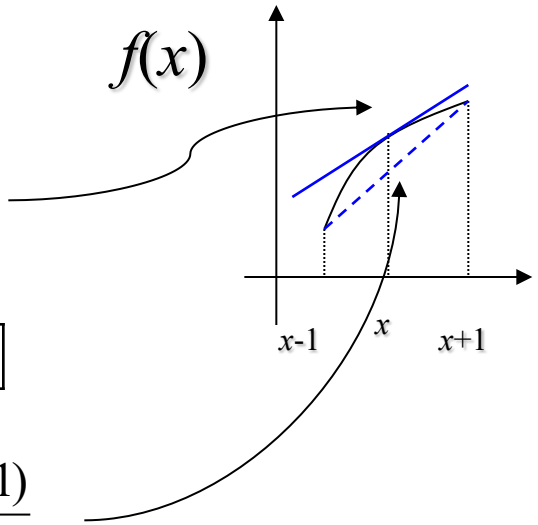
1	1
-1	-1

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

Convolve with:

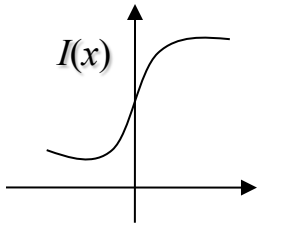
-1	0	1

$$\frac{df(x)}{dx} \cong \frac{f(x+1) - f(x-1)}{2}$$



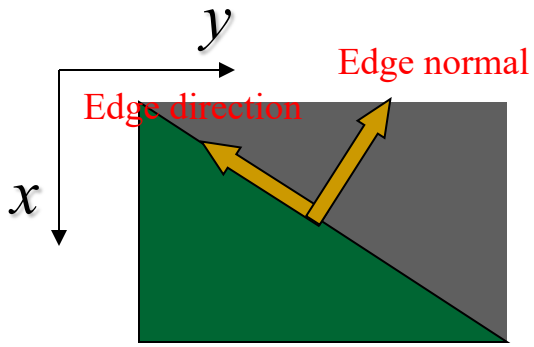
Edge Detection

1D



$I(x)$

2D



$I(x,y)$

$$\frac{dI(x)}{dx}$$

$$\left|\frac{dI(x)}{dx}\right| > threshold$$

$$\begin{aligned}\nabla I(x,y) &= [\partial I(x,y)/\partial x \quad \partial I(x,y)/\partial y]^T \\ &= [I_x(x,y) \quad I_y(x,y)]^T\end{aligned}$$

$$\nabla I(x,y) = \sqrt{I_x(x,y)^2 + I_y(x,y)^2} > threshold$$

$$\tan\theta = \frac{I_x(x,y)}{I_y(x,y)}$$

Edge Detection

- Vertical edges:
  - Convolve with:

-1	0	1

- Horizontal edges:
  - Convolve with:

-1
0
1

The Essential Edge Descriptor

- Edge position or center
  - The image position at which the edge is located
  - Usually saved in a binary image (1 : edge, 0 : no edge)
- Edge normal
  - The direction (unit vector) of the maximum intensity variation at the edge point
  - This identifies the direction perpendicular to the edge
- Edge direction
  - The direction perpendicular to the edge normal
  - This identifies the direction tangent to the edge
- Edge strength
  - A measure of the local image contrast; i.e., how marked the intensity variation is across the edge along the normal

Edge Detection Steps

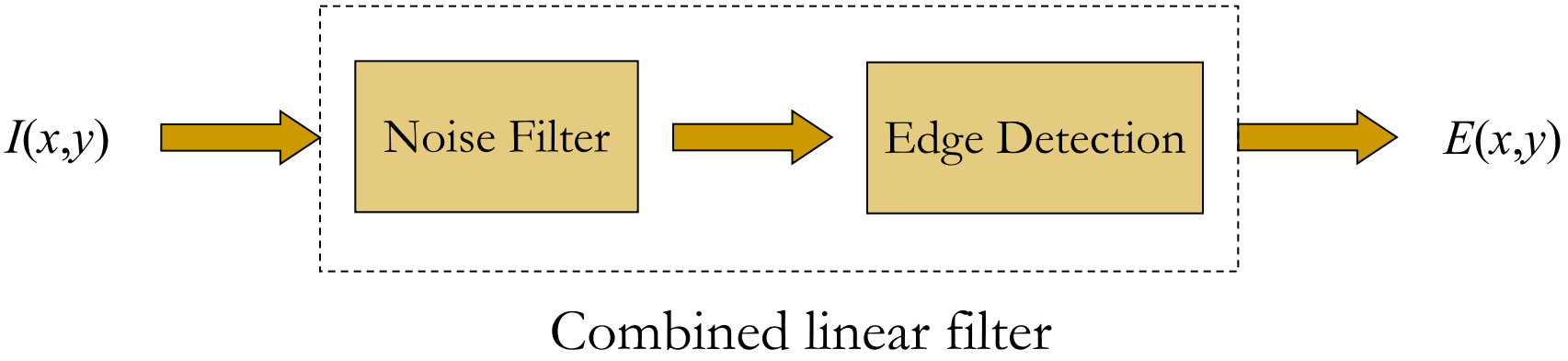
- Noise smoothing (filtering)
  - Suppress noise without destroying the true edges
- Edge enhancement
  - Design a filter responding to edges
  - Usually performed by computing gradient magnitude
- Edge localization
  - Thinning (non-maximum suppression)
  - Thresholding (used to decide whether the output is an edge point or not)

Some Assumptions

- The edge enhancement filter is linear
- The filter must be optimal for noisy step edge
- The image noise is additive, white and Gaussian
  - White noise: noise having a frequency spectrum that is continuous and uniform over a specified frequency band
  - White noise has equal power per hertz over the specified frequency band

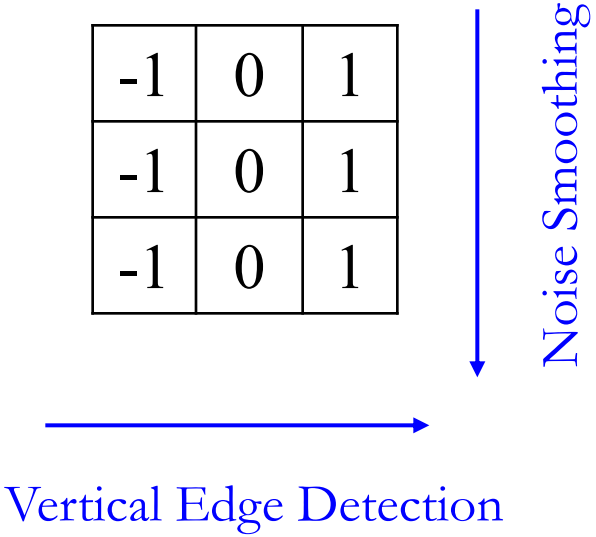
Noise Suppression

- The differential kernels act as "high pass filters" which tend to amplify noise
- This is why edge detection is usually preceded by a noise reduction or filtering operation



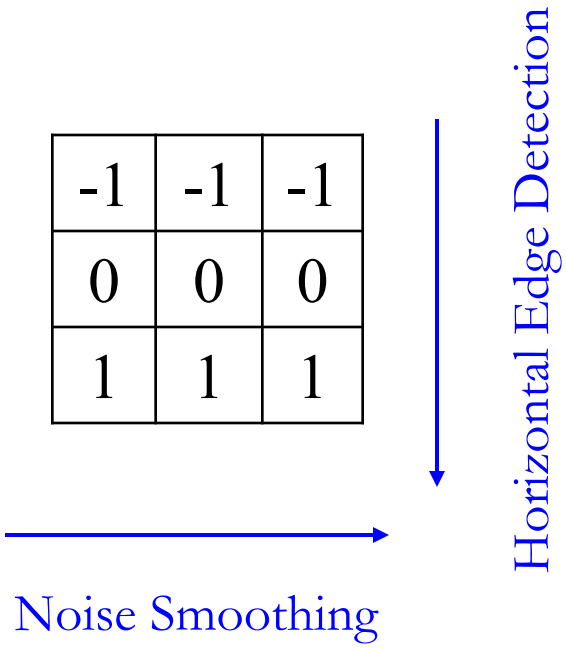
Noise Smoothing & Edge Detection

- Prewitt Edge Detector (vertical)
  - Convolve with:



Noise Smoothing & Edge Detection

- Prewitt Edge Detector (horizontal)
  - Convolve with:



Roberts Detector

+1	0
0	-1

G_x

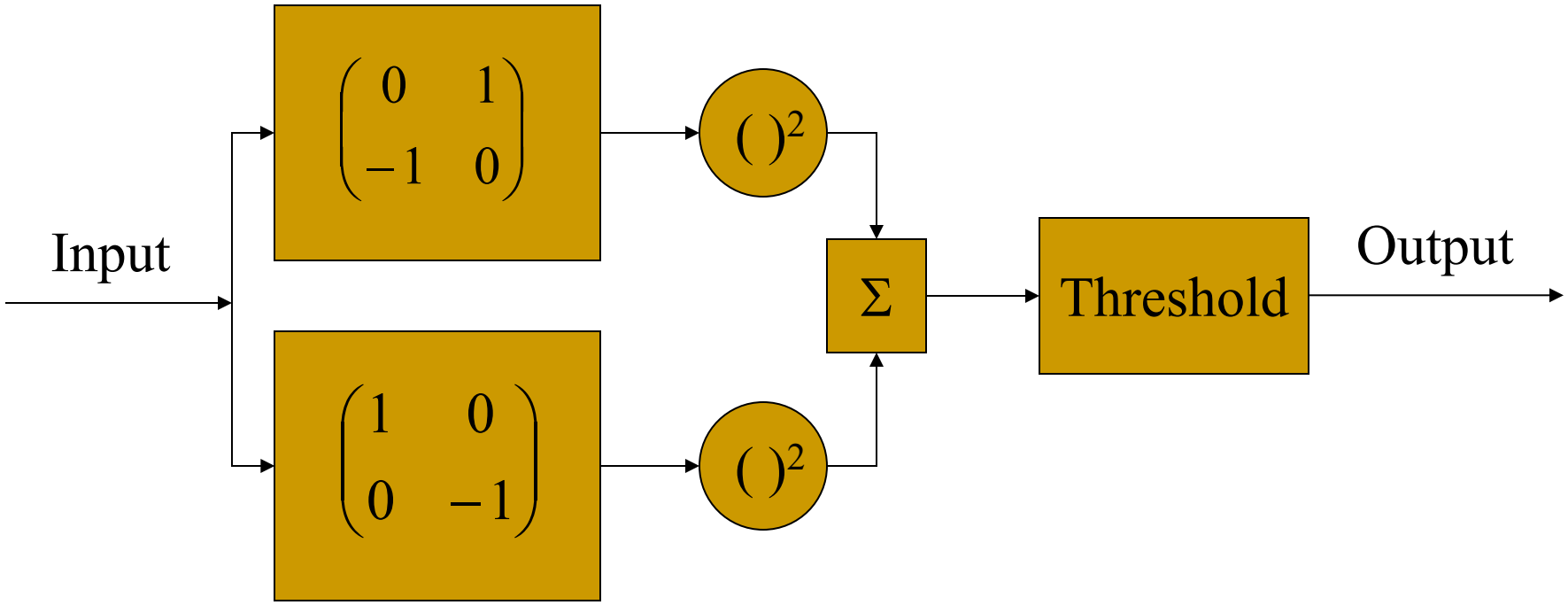
0	+1
-1	0

G_y

$$|G| = \sqrt{G_x^{\,2} + G_y^{\,2}}$$

Roberts Detector

- The Roberts detector gives an approximation to the continuous gradient at $[i+½, j+½]$. (not at $[i, j]$, why?)



Sobel Detector

- Sobel detector gives more weight to the 4-neighbors
- Emphasize the pixels closer to the center of the mask (compare with Prewitt!)

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

G_x

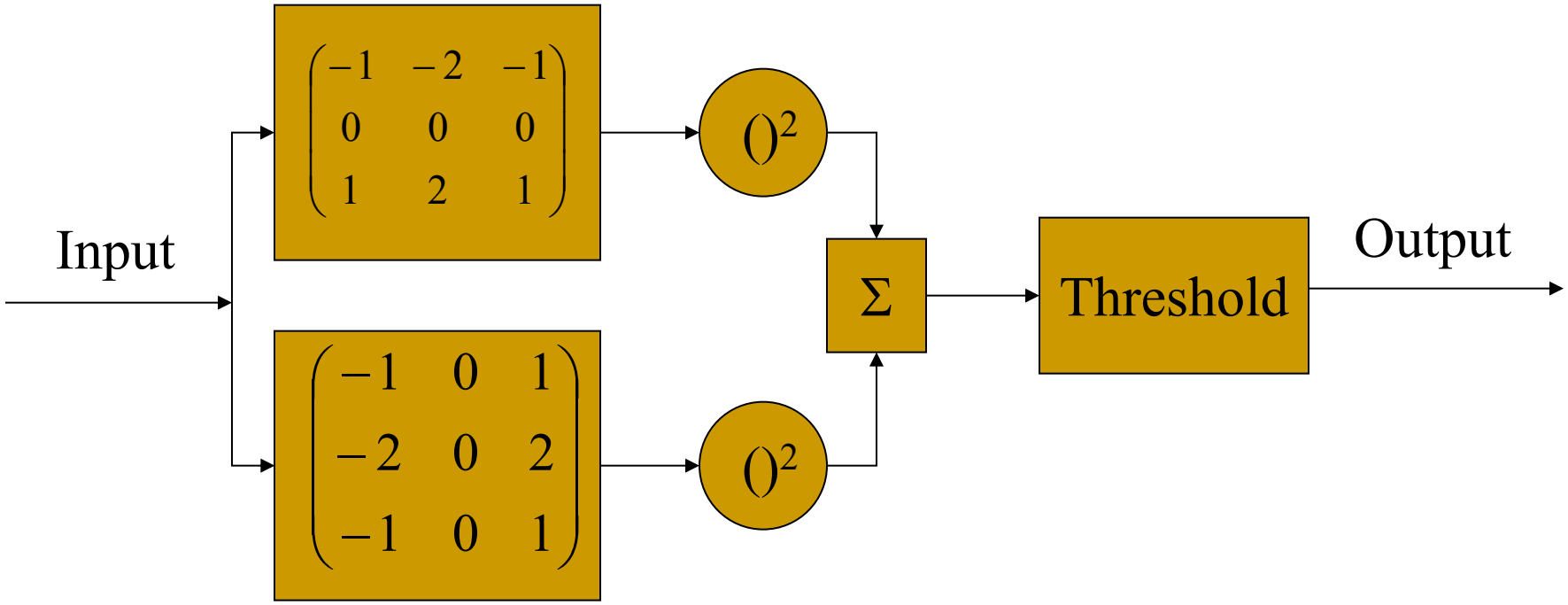
-1	-2	-1
0	0	0
1	2	1

G_y

$$\theta = \tan^{-1}(G_y / G_x)$$

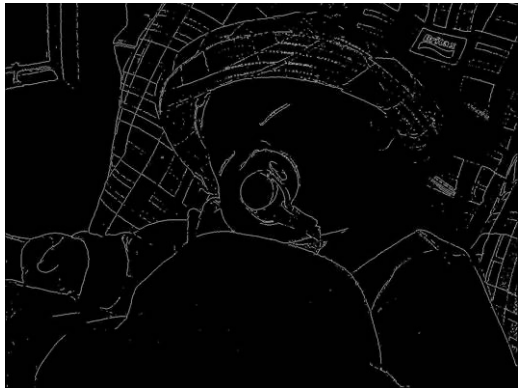
Sobel Detector

- Sobel operator is one of the most commonly used edge detector



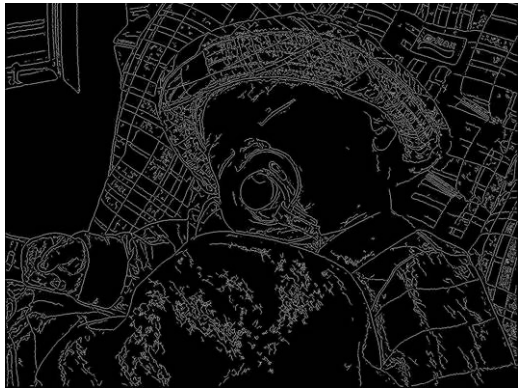
Examples

Original



Sobel

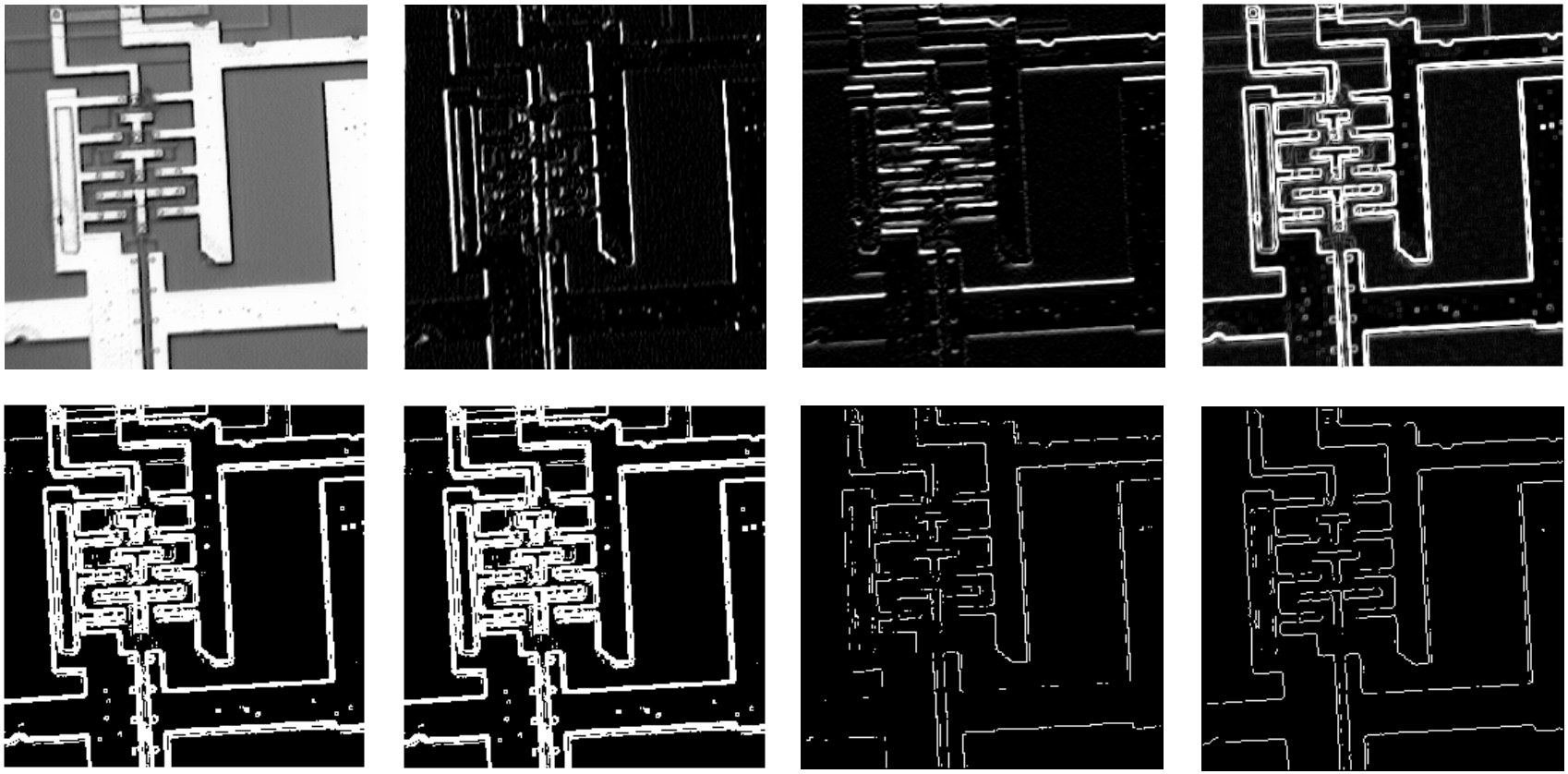
Roberts



Canny

Prewitt

Example



Zero Crossing Detector

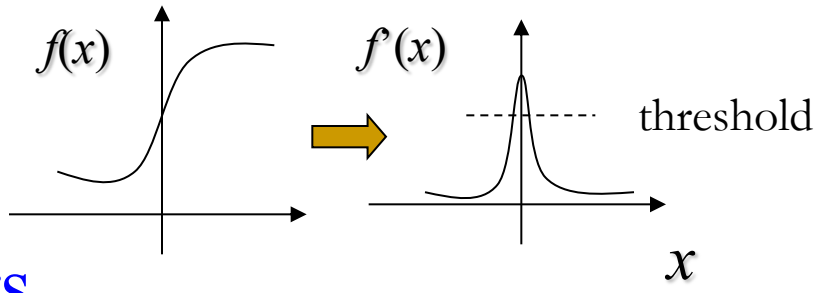
- Problem with first-order derivative edge detectors:
  - Using a threshold may result in detection of too many edge points
- Another approach:
  - Look for zero crossings of the second derivative since these will correspond to maxima (or minima) of the derivative
- $f''(x) = 0$ is not enough!
  - $f'(x) = c$ has $f''(x) = 0$, but there is no edge
- The second-derivative must change sign, i.e. from (+) to (–) or from (–) to (+)
- The sign transition depends on the intensity change of the image – i.e. from dark to bright or vice versa

Zero Crossing Detector

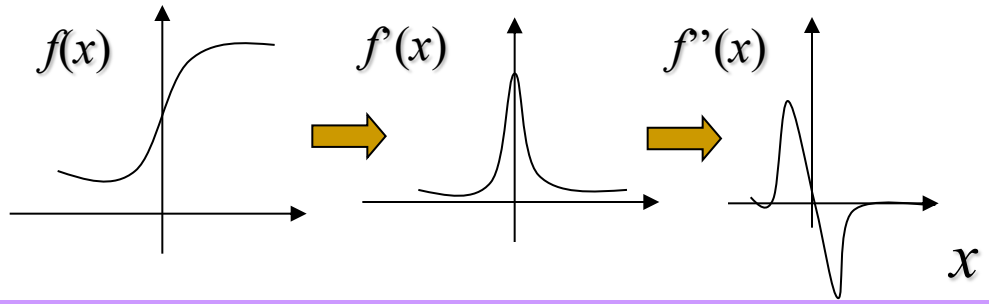
- There are two operators in 2-D corresponding to second derivative
  - Laplacian
  - Second directional derivative

Edge Detectors

- First-order derivative filters
  - Sharp changes in gray level of the input image correspond to "peaks" of the first-derivative of the input signal



- Second-order derivative filters
  - Peaks of the first-derivative of the input signal, correspond to "zero-crossings" of the second-derivative of the input signal



Zero Crossings of the Laplacian

- The 2-D analog to the second derivative is called the Laplacian

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

Laplacian

- Laplacian of f is defined as

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

- Numerical approximation is given by

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

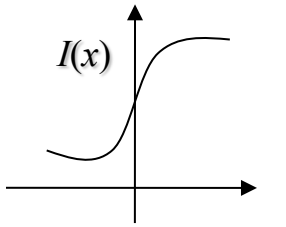
$$\frac{\partial^2 f}{\partial y^2} = f(i+1, j) - 2f(i, j) + f(i-1, j)$$

0	1	0
1	-4	1
0	1	0

1	4	1
4	-20	4
1	4	1

Laplacian Edge Detection

1D



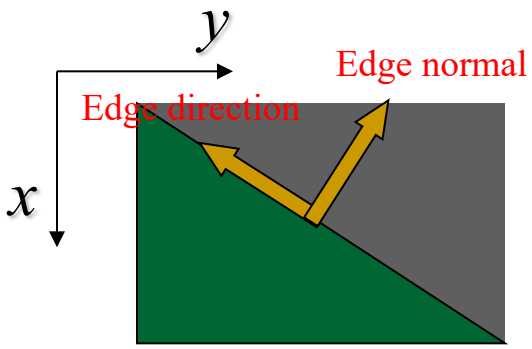
$I(x)$

$$\frac{dI(x)}{dx}$$

$$\left|\frac{dI(x)}{dx}\right| > threshold$$

$$\frac{d^2 I(x)}{dx^2} = 0$$

2D



$I(x,y)$

$$\nabla I(x,y) = [\partial I(x,y)/\partial x \quad \partial I(x,y)/\partial y]^T = [I_x(x,y) \quad I_y(x,y)]^T$$

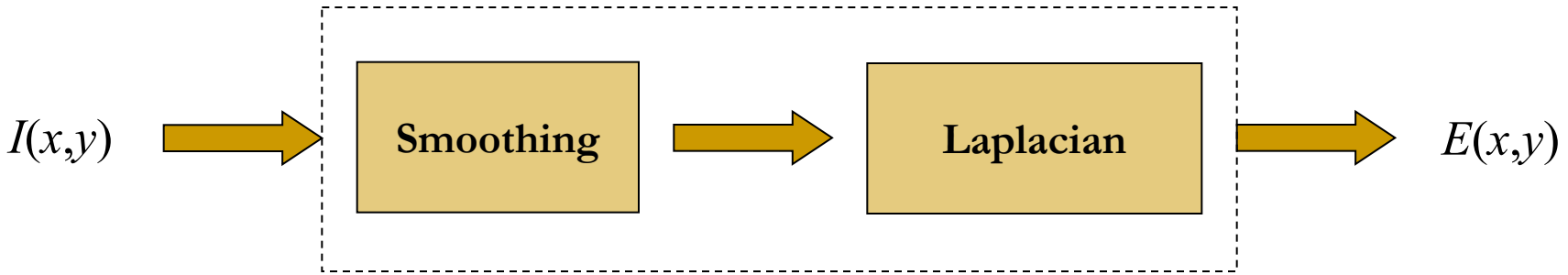
$$|\nabla I(x,y)| = \sqrt{I_x(x,y)^2 + I_y(x,y)^2} > threshold$$

$$\tan\theta = \frac{I_x(x,y)}{I_y(x,y)}$$

$$\nabla^2 I(x,y) = I_{xx}(x,y) + I_{yy}(x,y) = 0$$

Notes about Laplacian Filter

- $\nabla^2 I(x,y)$ is a scalar
  - Plus: Can be found using a "single" mask
  - Minus: Orientation information is lost
- $\nabla^2 I(x,y)$ is the sum of second-order derivatives
  - Taking derivatives increases noise
  - Very noise sensitive!
- It is "always" combined with a smoothing operation:



Combined linear filter

Approximating the Laplacian

- The second derivative along the x and y directions are approximated using difference equations*

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

$$\frac{\partial^2 f}{\partial y^2} = \frac{\partial G_y}{\partial y} = f[i+1, j] - 2f[i, j] + f[i-1, j]$$

- Combine the equations into a single operator, the Laplacian can be approximated by

∇^2 :

0	1	0
1	-4	1
0	1	0

Approximating the Laplacian

- Various discrete Laplacian kernels

0	1	0
1	-4	1
0	1	0

1	1	1
1	-8	1
1	1	1

-1	2	-1
2	-4	2
-1	2	-1

- For 1-D case:

1	-2	1

Zero-Crossing Edge Detector

- The zero-crossing corresponding to the edge could lie exactly halfway between the two pixels (the values, -7, 7, for instance)
  - The edge should be consistently marked at the pixel to the left or right of the edge
  - It rarely happens in most cases (usually -4, 6, etc.)
  - The actual edge location must be determined by interpolating the pixel values on either side of the zero-crossing
- The zero-crossing could correspond to a pixel in the image, but rarely happens (e.g. -3, 0, 3)

Laplacian of Gaussian (LoG)

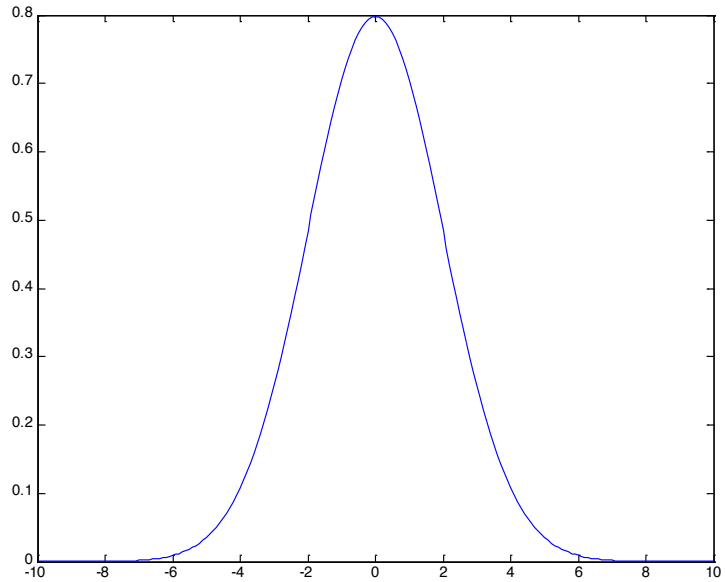
- Before applying the Laplacian, the image is often pre-filtered with a Gaussian kernel
  - $O(x,y) = \nabla^2(G(x,y)*I(x,y))$
  - By linearity: $O(x,y) = \nabla^2 G(x,y)*I(x,y)$
- Both stages can be combined into a single filter (The Mexican hat filter)

$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1-\frac{x^2+y^2}{2\sigma^2}\right]e^{-\frac{x^2+y^2}{2\sigma^2}}$$

1-D Gaussian

- 1-D Gaussian with zero mean is given by

$$g(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$$



- Ignore the scale factor,

$$g(x) = e^{-\frac{x^2}{2\sigma^2}}$$

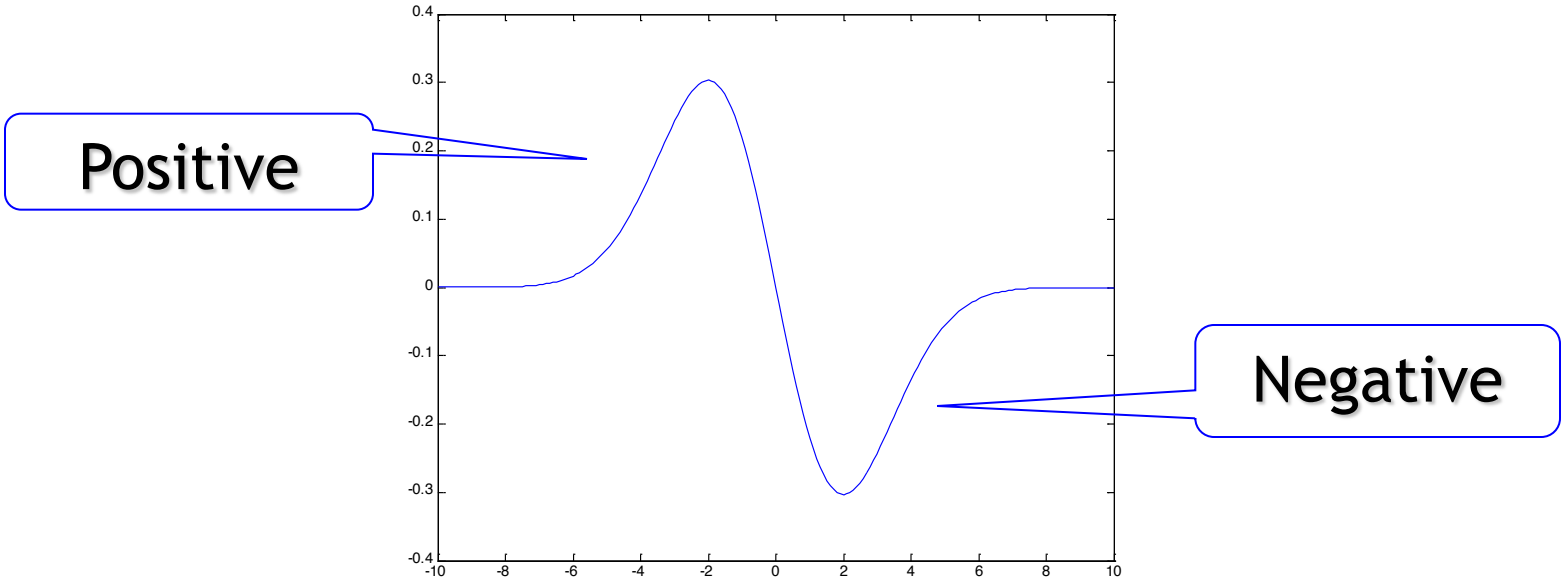
- Then,

$$g'(x) = -\frac{1}{2\sigma^2} 2xe^{-\frac{x^2}{2\sigma^2}} = -\frac{x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}$$

First Derivative of Gaussian

- The first derivative of Gaussian is given by

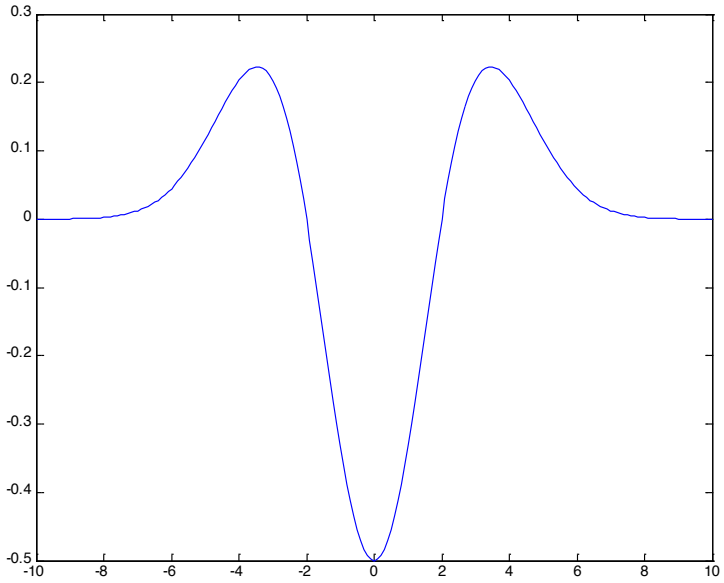
$$g'(x) = -\frac{1}{2\sigma^2} 2xe^{-\frac{x^2}{2\sigma^2}} = -\frac{x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}$$



Second Derivative of Gaussian

- The second derivative of Gaussian is given by

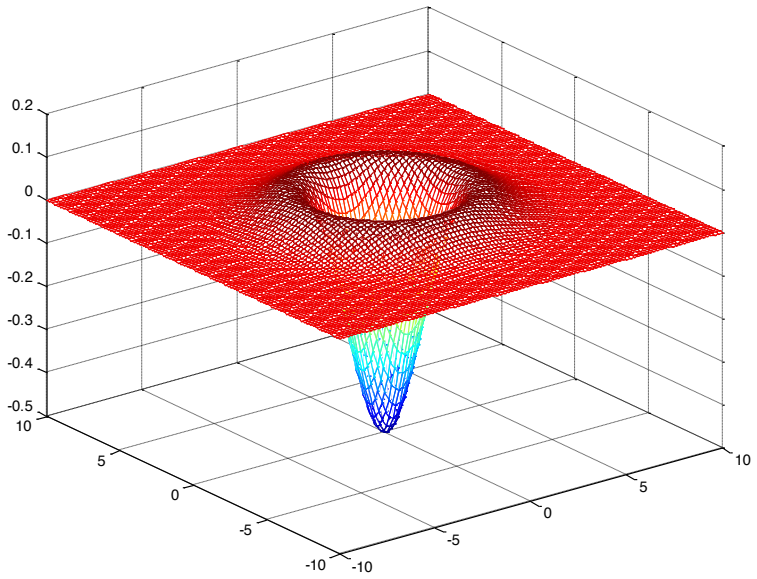
$$g''(x) = \left(\frac{x^2}{\sigma^3} - \frac{1}{\sigma}\right) e^{-\frac{x^2}{2\sigma^2}}$$



Laplacian of the Gaussian

- For the 2-D case, LoG is given by

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



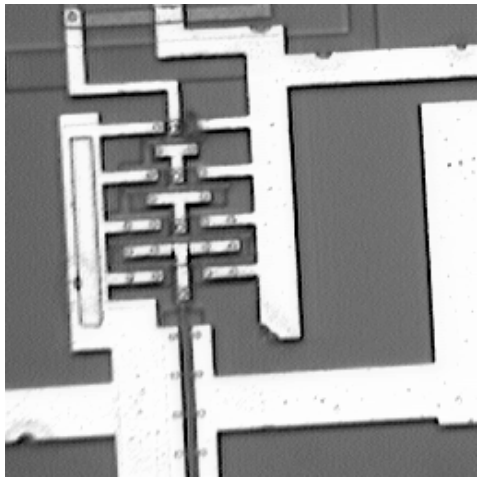
"Mexican Hat"

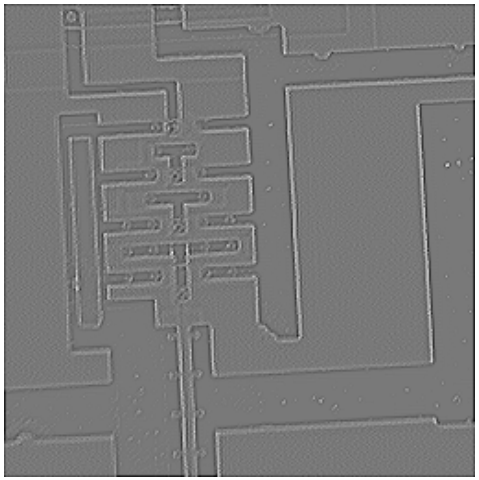
- LoG masks given in Fig. 5.12
- Scale space used in Gaussian smoothing
  - Ideal similar to double thresholding

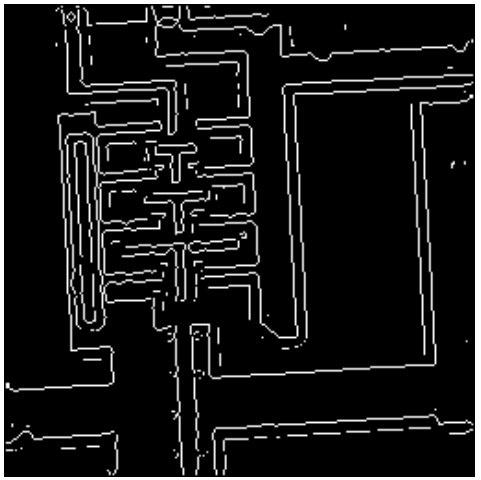
LoG Edge Detector

- The fundamental characteristics of LoG
  - The smoothing filter is a Gaussian
  - The enhancement step is the second derivative (Laplacian in two dimensions)
  - The detection criterion is the presence of a zero crossing in the second derivative with a corresponding large peak in the first derivative
  - The edge location can be estimated with "subpixel" resolution using linear interpolation
- There are two ways of implementation
  - Apply Gaussian filter and compute the Laplacian
  - Apply Laplacian of Gaussian filter directly

Example







Disclaimer

- No edge detection scheme is going to work perfectly in all cases
- This is due to the fact that our notion of what constitutes a salient edge in the image is actually somewhat subtle

Detecting Corners

- We are often interested in detecting point features in an image
- These features are usually defined as regions in the image where there is significant edge strength in two or more directions
- They can be used for
  - Object tracking
  - 3D triangulation (stereo)
  - Object recognition

Detecting Corners

- Need two strong edges
- If E_x and E_y denote the gradients of the image in the x and y directions, then the behavior of the gradients in a region around a point can be obtained by considering the following matrix

$$C = \sum \begin{pmatrix} E_x \\ E_y \end{pmatrix} \begin{pmatrix} E_x & E_y \end{pmatrix} = \sum \begin{pmatrix} E_x^2 & E_x E_y \\ E_x E_y & E_y^2 \end{pmatrix}$$

Examining The Matrix

- One way to decide on the presence of a corner is to look at the eigenvalues of the 2 by 2 matrix C
  - If the area is a region of "constant intensity" we would expect both eigenvalues to be small (or zero)
  - If it contains a edge we expect one large eigenvalue and one small one
  - If it contains edges at two or more orientations we expect two large eigenvalues
- If $\min(\lambda_1, \lambda_2) > T$, then there is a corner!

Finding Corner

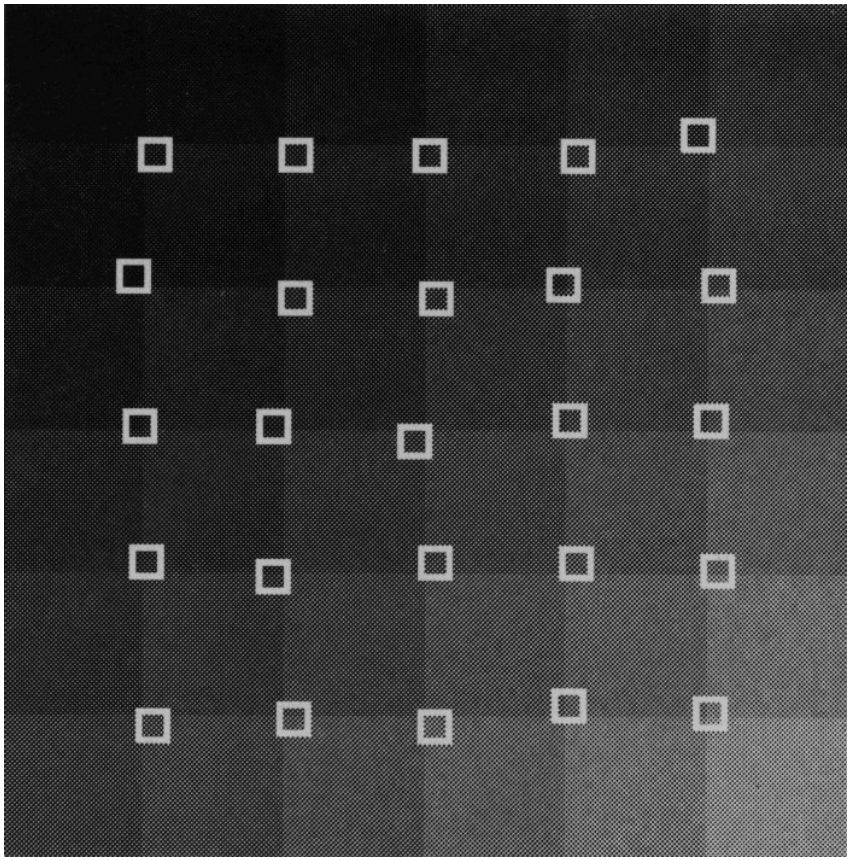
- One approach to finding corners is to find locations where the smaller eigenvalue is greater than some threshold
- We could also consider the ratio of the two eigenvalues
- Issues:
  - Localization – It can be difficult to precisely localize the corner in the intensity image
  - Modeling – It can be helpful to have a model of the corners you are trying to find in order to detect and localize them more systematically

Corner Detection

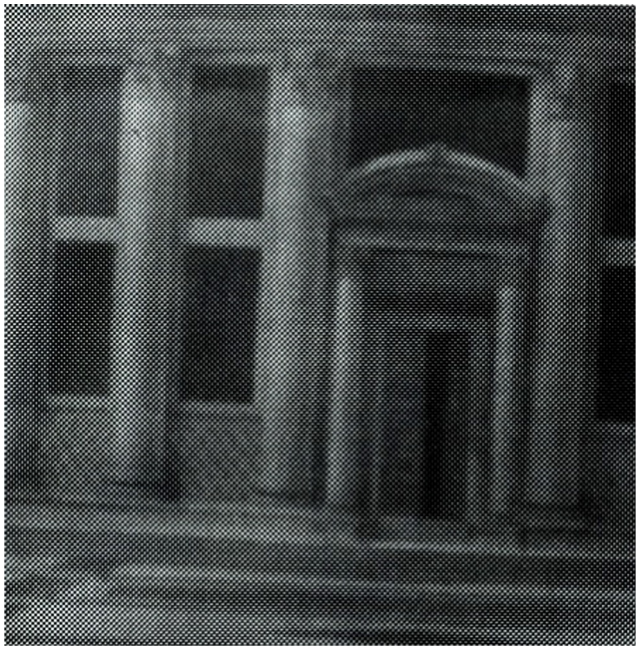
- Compute image gradient
- For each $m \times m$ neighborhood, compute matrix C
- If smaller eigenvalue λ_2 is greater than threshold τ, record a corner
- Nonmaximum suppression: only keep strongest corner in each $m \times m$ window

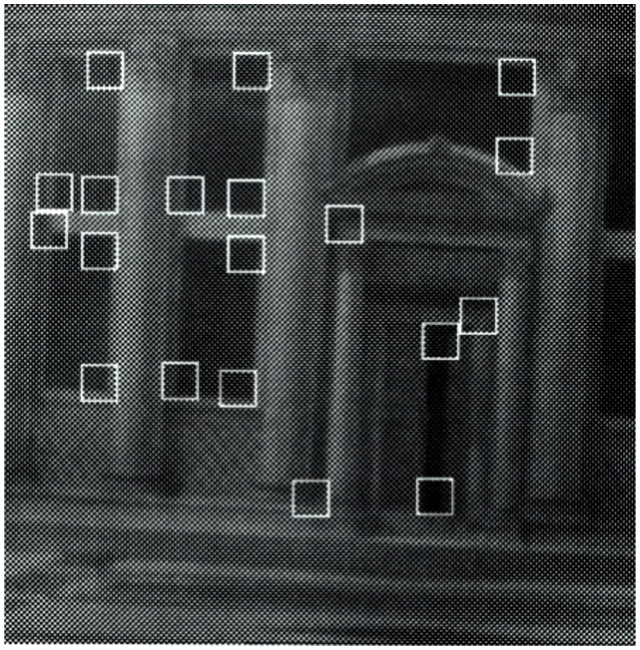
Corner Detection Results

- Checkerboard with noise

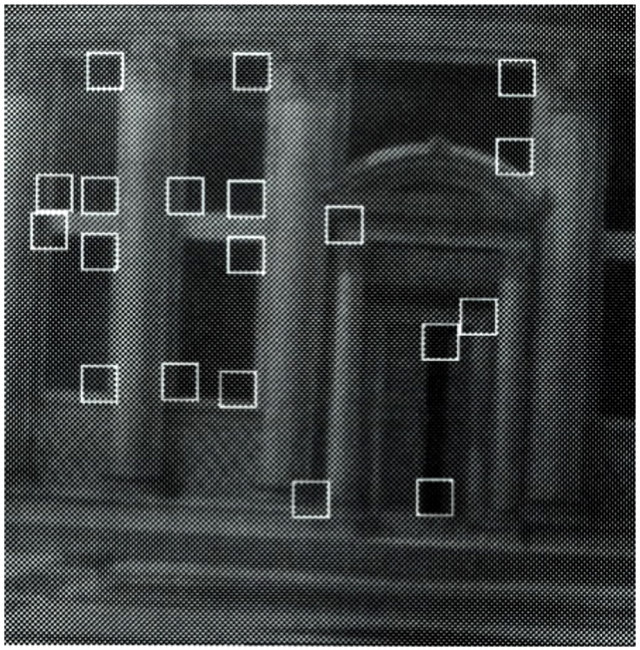


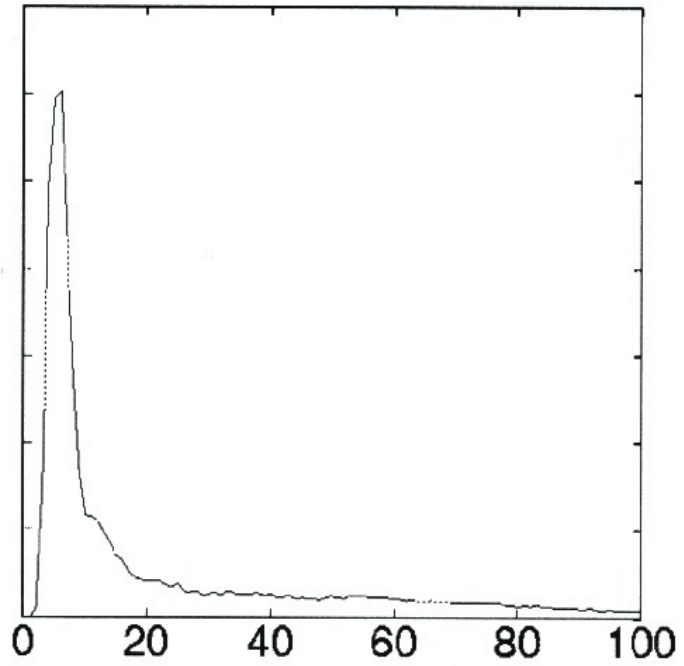
Corner Detection Results





Corner Detection Results





Histogram of λ_2 (smaller eigenvalue)