

# Edge and Corner Detection

# Basics

- **Edge pixels** are those at which image intensity changes abruptly.
- **Edge segments** or edges are sets of connected edge pixels.

# Digital derivatives: first versus second

- Along a ramp (i.e. intensity change with constant slope), first derivative is a non-zero constant.
- Second derivative is zero along the ramp, except at the start and end!
- Second derivative is preferable for image sharpening! Many edges in images are ramp-like, in which case **first** derivative will give **thick** edges (undesirable), whereas **second** derivative gives **thin** edges two pixels wide (desirable).
- Second derivative changes sign midway at a step edge (**zero crossing property**).

# Digital derivatives: first versus second

- At **roof edges**, second derivative gives stronger response than first derivative (i.e. has larger magnitude) – except if both parts of the roof were at 45 degrees, where both will have equal response.
- At **isolated points**, second derivative gives stronger response than first derivative
- A rotationally invariant second-derivative operator is used for edge detection: the Laplacian, in fact zero-crossings of a Laplacian.

# Laplacian of an image

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \longrightarrow \text{Rotationally symmetric operator (in the continuous domain)}$$

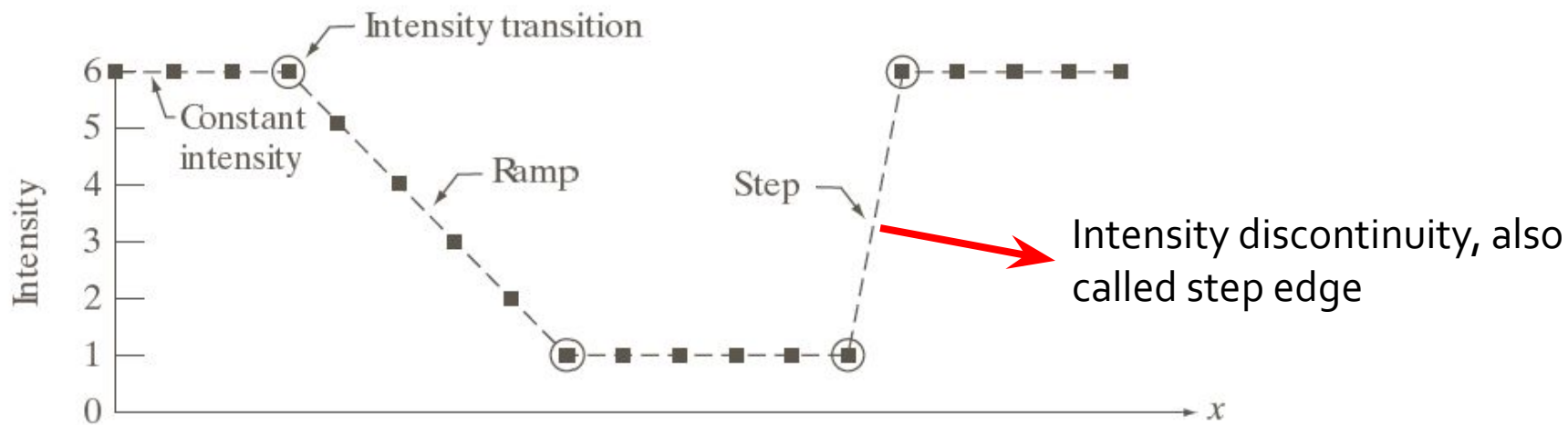
$$= f(x+1, y) + f(x-1, y) - 2f(x, y)$$

$$+ f(x, y+1) + f(x, y-1) - 2f(x, y)$$

$$= f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Laplacian operators: second operator is obtained by adding second derivatives along both the diagonals, to the first operator

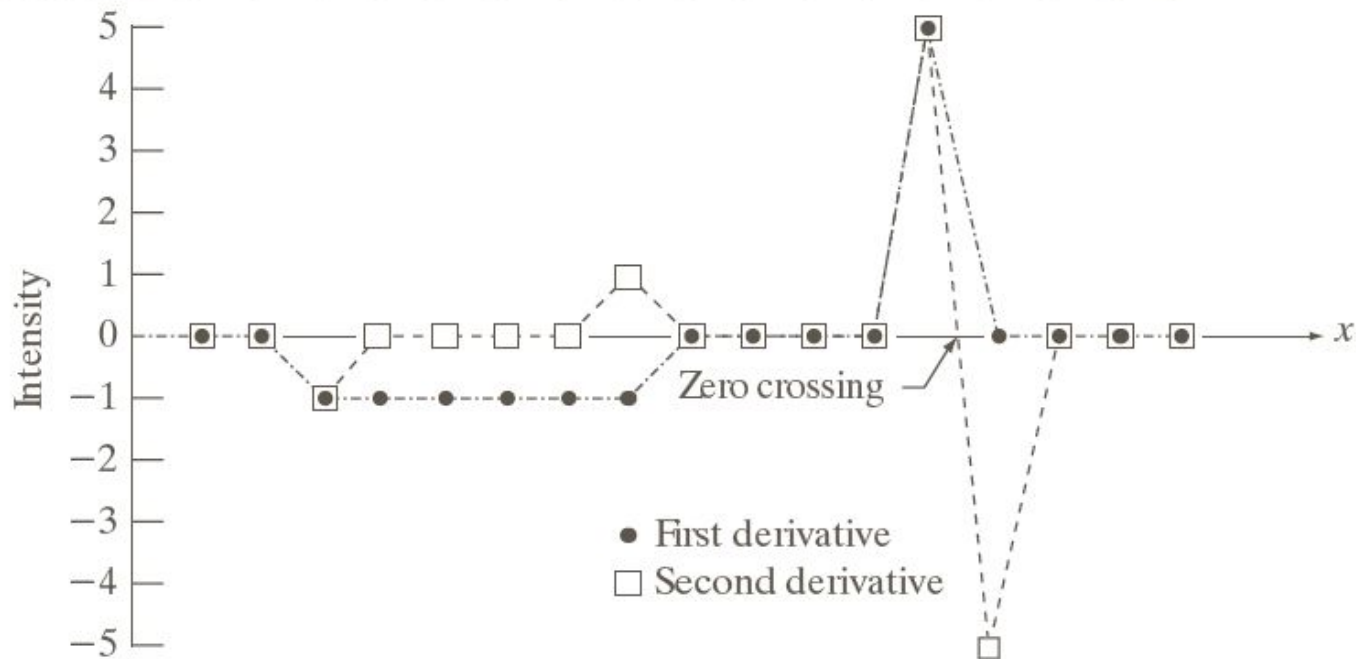


Scan line

	6	6	6	6	5	4	3	2	1	1	1	1	1	1	6	6	6	6	6
	$x$																		

1st derivative 0 0 -1 -1 -1 -1 -1 0 0 0 0 0 0 5 0 0 0 0

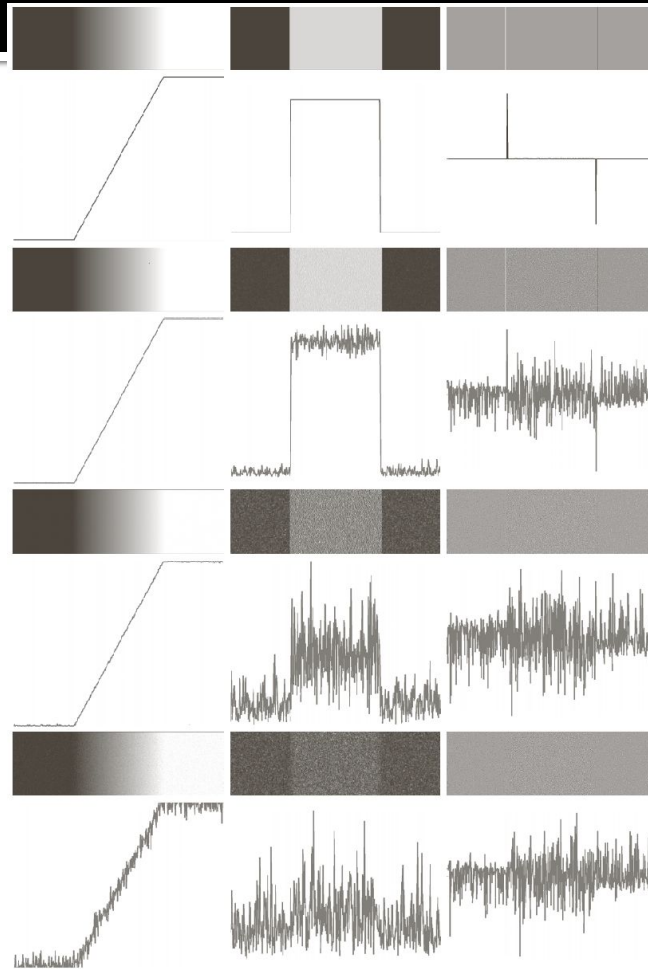
2nd derivative 0 0 -1 0 0 0 0 1 0 0 0 0 0 5 -5 0 0 0



From book  
by Gonzalez  
and Woods

# Edge detection under noise

First and second derivatives are very sensitive to noise!  
**Left to right in each row:** original image, first derivative, second derivative  
**In any column:** increase in noise level



**FIGURE 10.11** First column: Images and intensity profiles of a ramp edge corrupted by random Gaussian noise of zero mean and standard deviations of 0.0, 0.1, 1.0, and 10.0 intensity levels, respectively. Second column: First-derivative images and intensity profiles. Third column: Second-derivative images and intensity profiles.

# Steps in robust edge detection

- Image smoothing to reduce noise: because first and second derivatives are noisy
- Actual detection of edge points (may produce thick or disconnected edges)
- Pruning or linking of the edge points



# Edge detection: image gradient

- An image gradient is defined as follows:

$$\nabla f = \begin{pmatrix} \partial f / \partial x \\ \partial f / \partial y \end{pmatrix} = \begin{pmatrix} f_x \\ f_y \end{pmatrix}$$

$$\text{Gradient magnitude} = \|\nabla f\|_2 = \sqrt{f_x^2 + f_y^2}$$

$$\text{Gradient direction} = \theta = \tan^{-1}(f_y / f_x)$$

Gradient operator :

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

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

# Robust edge operators

These operators: Prewitt and Sobel perform some smoothing prior to difference computation, and hence are more robust than the simple finite differences from the previous slide.

$$f_y = \frac{\partial f}{\partial y} = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$$

$$f_x = \frac{\partial f}{\partial x} = (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)$$

$$f_y = \frac{\partial f}{\partial y} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

$$f_x = \frac{\partial f}{\partial x} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

$z_1$	$z_2$	$z_3$
$z_4$	$z_5$	$z_6$
$z_7$	$z_8$	$z_9$

-1	0	0	-1
0	1	1	0

Roberts

-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

a
b c
d e
f g

**FIGURE 10.14**  
A  $3 \times 3$  region of an image (the  $z$ 's are intensity values) and various masks used to compute the gradient at the point labeled  $z_5$ .

# Prewitt and Sobel

-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

- Apply the differencing mask  $[-1 \ 0 \ 1]^T$  and then apply the smoothing mask  $[1 \ 1 \ 1]$  on the result. This is equivalent to applying the first Prewitt mask (i.e. for  $f_y$ ).
- For the Prewitt operator for  $f_x$ , apply  $[-1 \ 0 \ 1]$  and then  $[1 \ 1 \ 1]^T$ .
- Apply the differencing mask  $[-1 \ 0 \ 1]^T$  and then apply the smoothing mask  $[1 \ 2 \ 1]$  on the result. This is equivalent to applying the first Sobel mask (i.e. for  $f_y$ ).
- For the Sobel operator for  $f_x$ , apply  $[-1 \ 0 \ 1]$  and then  $[1 \ 2 \ 1]^T$ .

## Edges with Sobel Mask

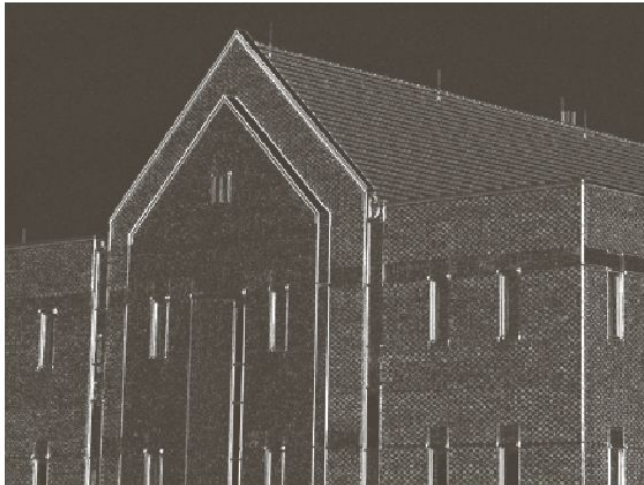
Original  
image



Derivative  
s in y  
direction:  
 $f_y$



Derivative  
s in x  
direction:  
 $f_x$



$|f_x| + |f_y|$

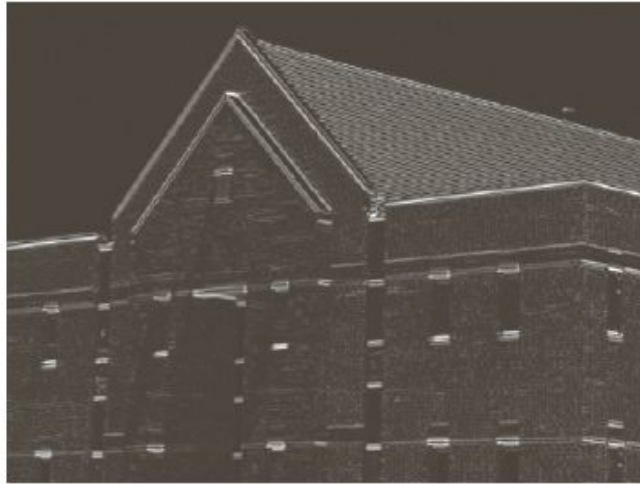


## Edges with 5 x 5 averaging followed by Sobel Mask

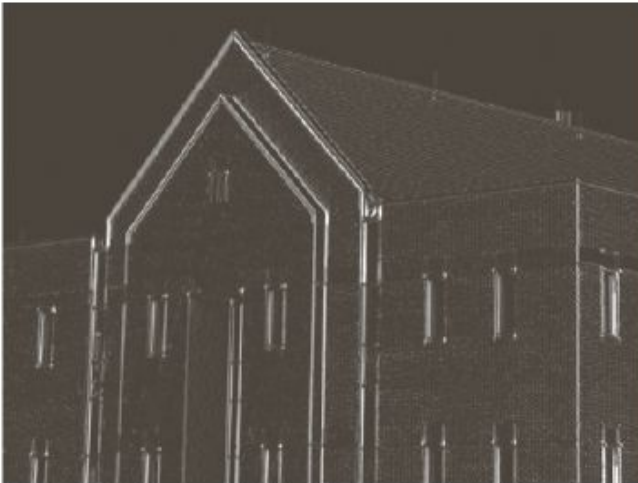
Original  
image



Derivative  
s in y  
direction:  
 $f_y$



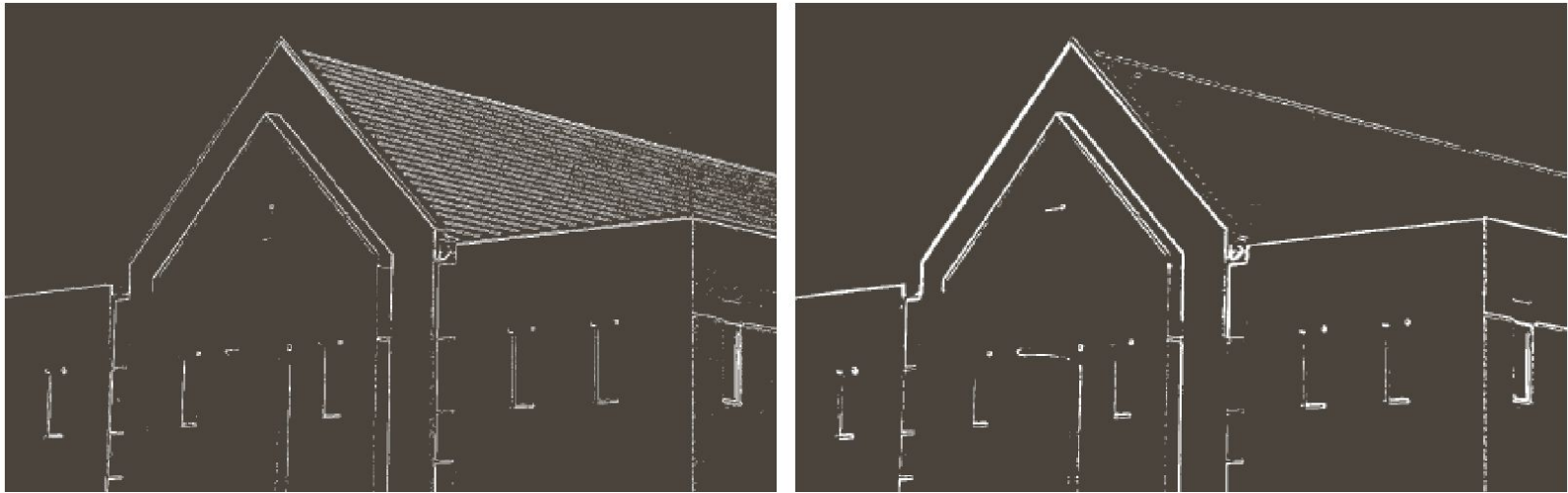
Derivative  
s in x  
direction:  
 $f_x$



$|f_x| + |f_y|$



Notice: finer details removed



a b

**FIGURE 10.20** (a) Thresholded version of the image in Fig. 10.16(d), with the threshold selected as 33% of the highest value in the image; this threshold was just high enough to eliminate most of the brick edges in the gradient image. (b) Thresholded version of the image in Fig. 10.18(d), obtained using a threshold equal to 33% of the highest value in that image.

Edge map after Thresholding the earlier map.



# Marr-Hildreth edge detector

- Principled approach to combined edge detection and image smoothing
- An edge operator should satisfy two criteria:
  - (1) Notion of **scale** in edge detection (operators of different sizes)
  - (2) Differential operator to compute first or second derivatives of the intensity
- Uses the **Laplacian of Gaussian** operator on an image satisfies both criteria

# Laplacian of Gaussian (LoG or Mexican hat)

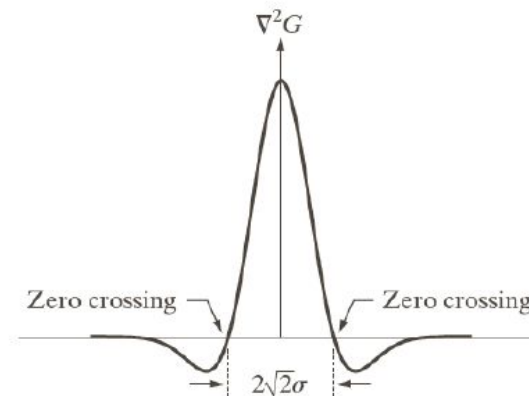
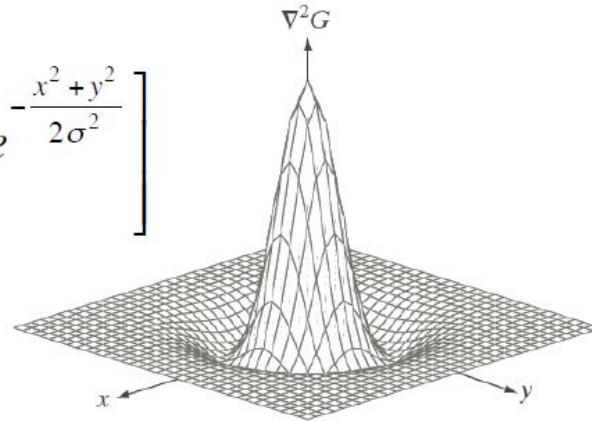
$$\nabla^2 G(x,y) = \frac{\partial^2 G(x,y)}{\partial x^2} + \frac{\partial^2 G(x,y)}{\partial y^2}$$

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

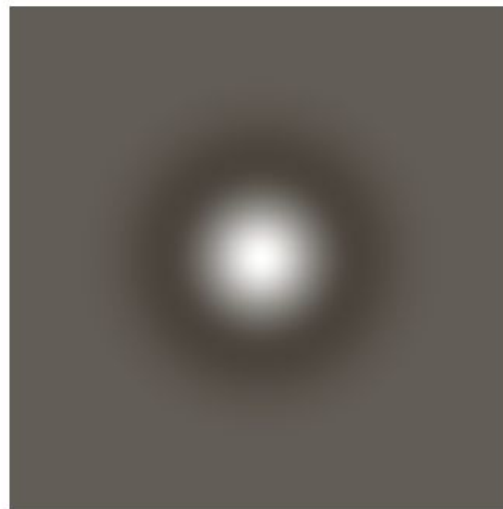
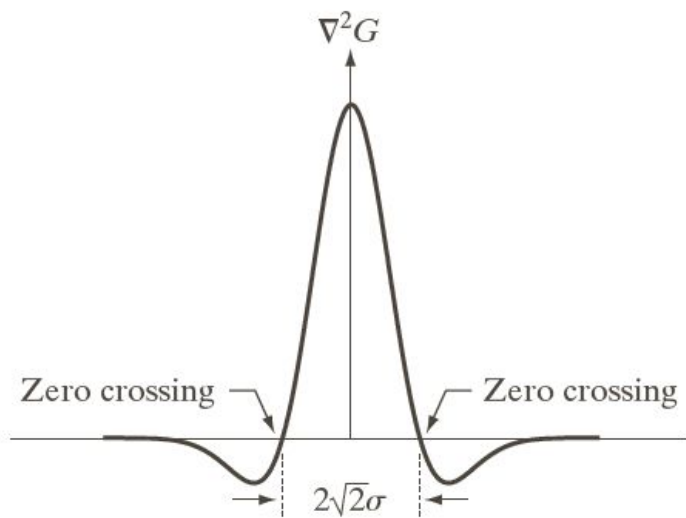
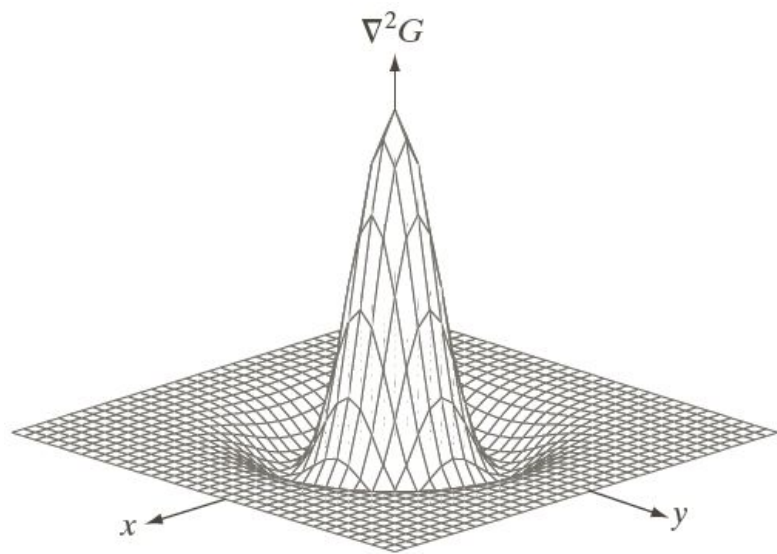
$$= \left( \frac{x^2}{\sigma^4} - \frac{1}{\sigma^2} \right) e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$\frac{\partial^2 G(x,y)}{\partial y^2} = \left( \frac{y^2}{\sigma^4} - \frac{1}{\sigma^2} \right) e^{-\frac{x^2+y^2}{2\sigma^2}}$$

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







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

a b  
c d

**FIGURE 10.21**

(a) Three-dimensional plot of the *negative* of the LoG. (b) Negative of the LoG displayed as an image. (c) Cross section of (a) showing zero crossings. (d)  $5 \times 5$  mask approximation to the shape in (a). The negative of this mask would be used in practice.

# Why LoG for edge detection?

- Remember: we are interesting in zero-crossings of the second derivative!
- We want the second derivative operator to be rotationally invariant. The Laplacian is the simplest (though not the only) such operator.
- The Gaussian is a low-pass filter, which blurs structures with intensity difference smaller than  $\sigma$ .
- Gaussian is smooth in spatial and frequency domains.

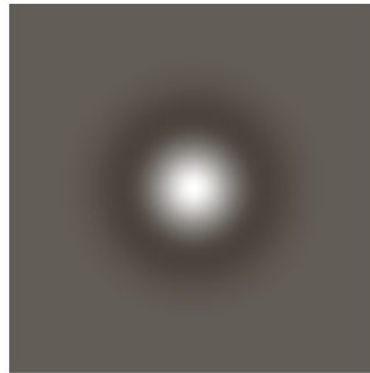
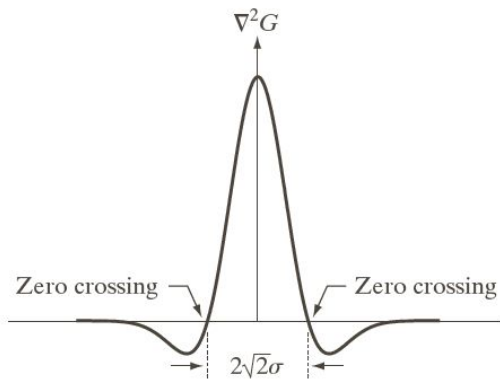
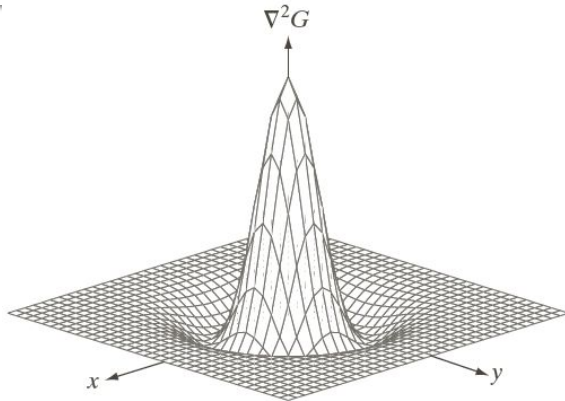
# Using LoG for edge detection

- (1) Convolve the image with the LoG operator, i.e.:

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

- (2) Find the zero-crossings in the resultant image.

# Implementing the convolution



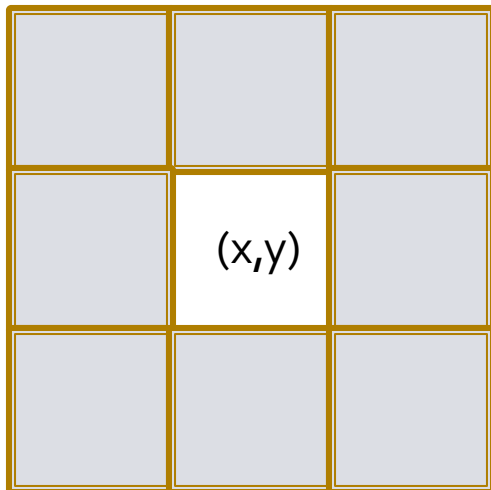
0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

Use an  $n \times n$  mask, created by sampling the formula for LoG on a discrete Cartesian grid of size  $n \times n$ .

How to choose  $n$ ? Depends on  $\sigma$ :  $n$  should be  $\geq 6\sigma$ , because more than 99% of the volume underneath a 2D Gaussian lies between  $\mu - 3\sigma$  and  $\mu + 3\sigma$ .

# Finding zero-crossings

- Look at a  $3 \times 3$  window around a pixel.
- There exists a zero-crossing at that pixel if the signs of at least one pair of opposing neighbors (top & bottom, left & right, top-left & right-bottom, top-right & left-bottom) differ and the difference in their values must exceed a threshold  $p$ .



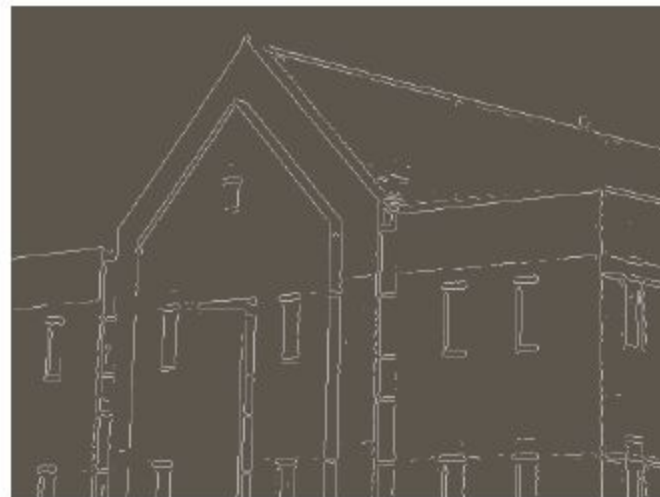
$\sigma = 4, n$   
 $= 25$



Convolut  
-ion with  
LoG



Zero-crossings



Zero-crossings  
+ thresholding

# Some properties of the Marr-Hildreth Edge Detector

- Detected edges are 1-pixel in thickness
- Can be used for different sigma values, keeping only those edge points that are common to multiple scales

# Canny Edge Detector: using 1<sup>st</sup> derivatives

- Three aims/objectives:
  - (a) Low error (no spurious responses due to noise)
  - (b) Well-localized edge points
  - (c) Single pixel wide edges

Canny's detector was formulated by analyzing step edges in 1D under zero-mean i.i.d. Gaussian noise, and expressing the preceding three criteria mathematically and deriving an (approximate) operator that obeys all criteria.



# Canny's Edge Detector

- In 1D, the approximate operator was the first derivative of the Gaussian:  $\frac{d}{dx} e^{-\frac{x^2}{2\sigma^2}} = \frac{-x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}$
- In 2D, the approximate operator was the first derivative of the Gaussian in the direction of the local image gradient (i.e. perpendicular to the edge).
- But the true gradient is not known, so we first smooth the image with a 2D Gaussian and then use its gradient magnitude and direction.

# Canny's Edge Detector

- Gradient magnitude and direction:

$$g(x, y) = G(x, y) * f(x, y)$$

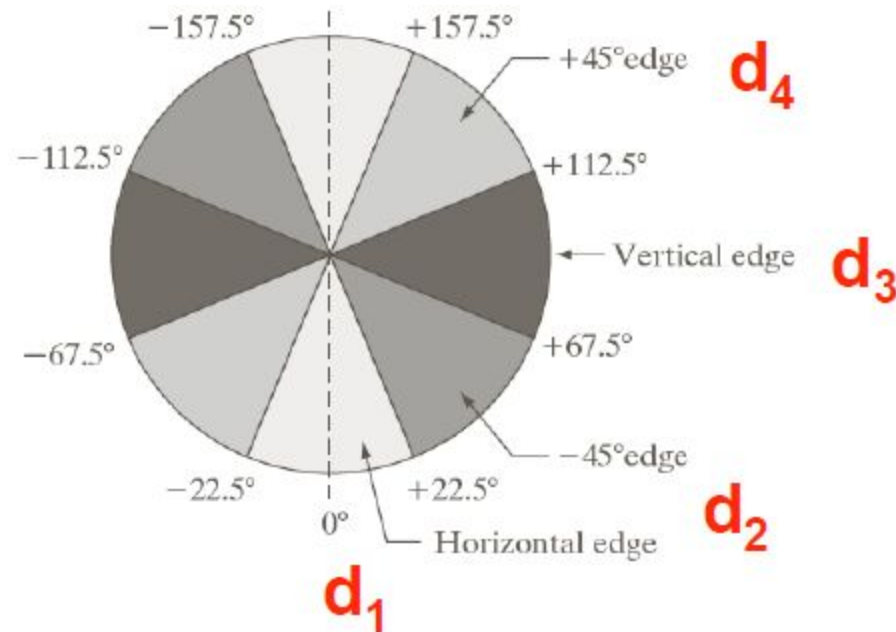
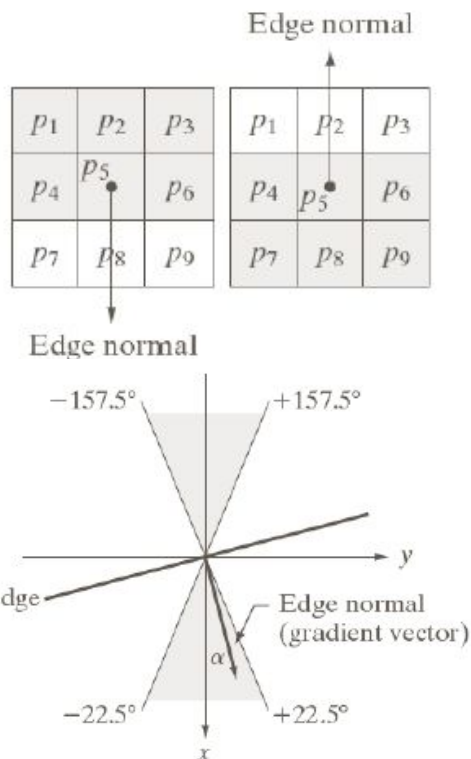
$$M(x, y) = |\nabla g(x, y)| = \sqrt{g_x^2(x, y) + g_y^2(x, y)}$$

$$\theta(x, y) = \tan^{-1} \left( \frac{g_y(x, y)}{g_x(x, y)} \right)$$

- $M(x, y)$  contains wide ridges along the true edges of the image.
- Though the maximum values are along the edge, those values don't immediately fall off to 0 on either side of the edge.
- Recall ramp edges.

# Canny edge detector

- This requires us to perform a **thinning** operation – called as **non-maximal suppression** (i.e. we set to zero those values of  $M(x,y)$  which are likely to NOT be local maxima).

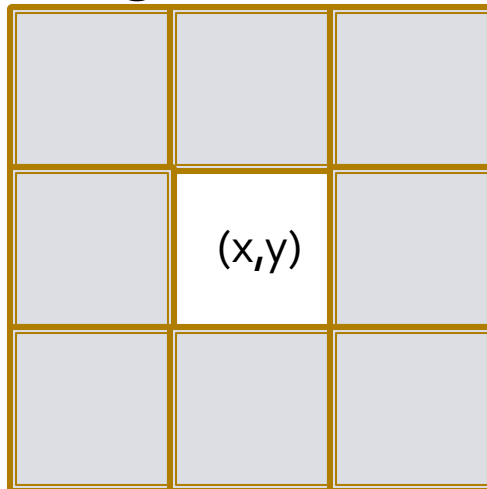


# Canny edge detector

- Consider the four quantized directions shown on the previous slide. Pick the direction  $d_{k'}$  which is closest to  $\Theta(x,y)$ .
- If  $M(x,y)$  is less than either of its two neighbors along the direction  $d_{k'}$ , then set it to 0 (i.e. suppress the non-maximum), otherwise we leave it as is.
- After this processing,  $M(x,y)$  is the non-maximal suppressed image.

# Canny Edge Detector

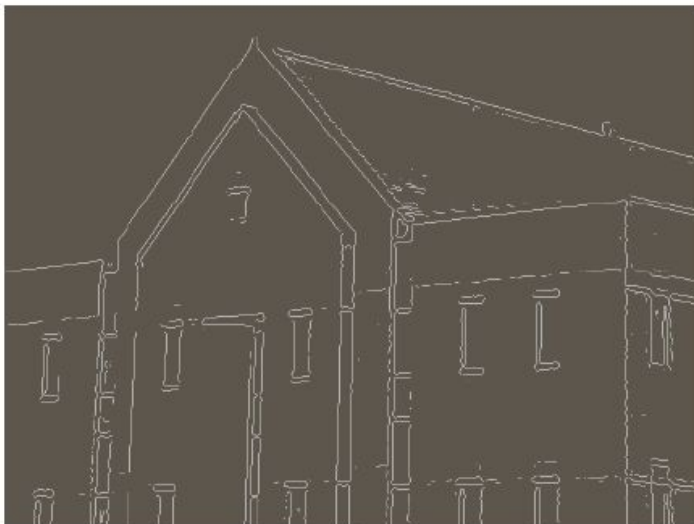
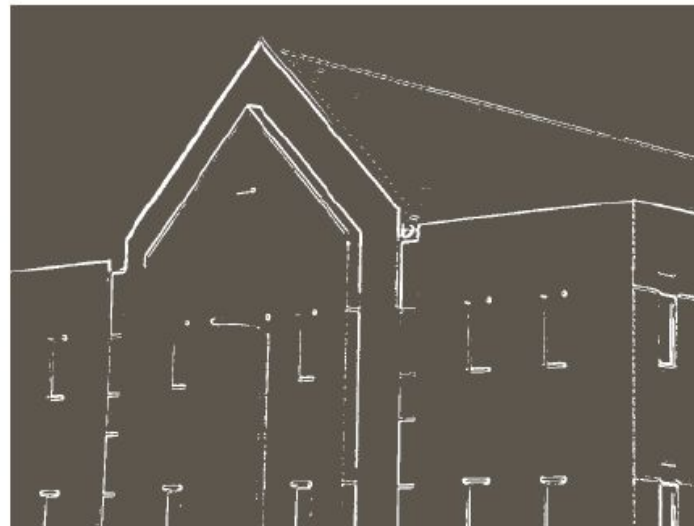
- The (thinned) edge map so far may still contain some false positives. This is taken care of (to some extent) by **hysteresis thresholding**.
- This uses a low-threshold TL and a high threshold TH.
- We keep only those pixels for which
  - $M(x,y) > TH$ , OR
  - $M(x,y) > TL$  AND  $(x,y)$  is a neighbor of at least one pixel with  $M(x,y) > TH$ .



The 8 neighbors  
of pixel  $(x,y)$

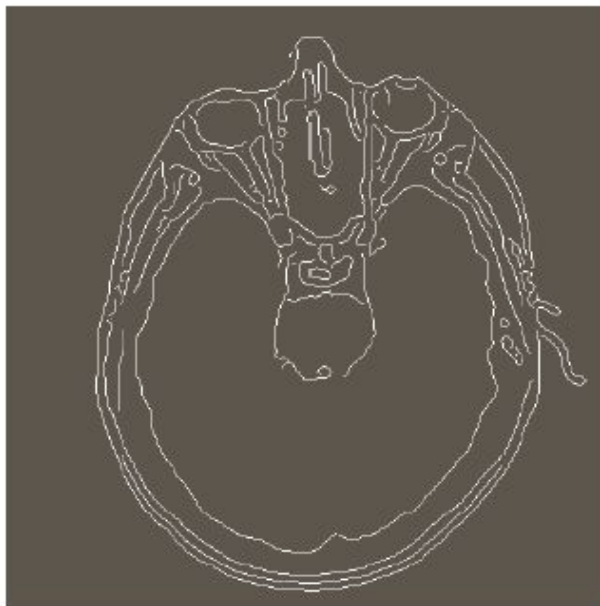
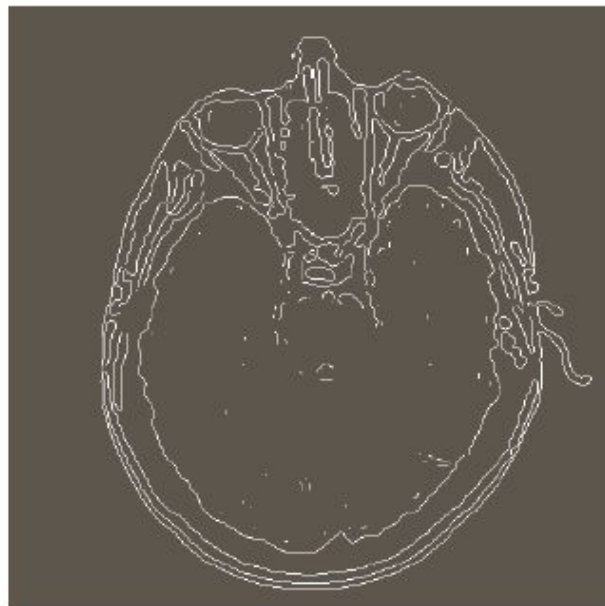
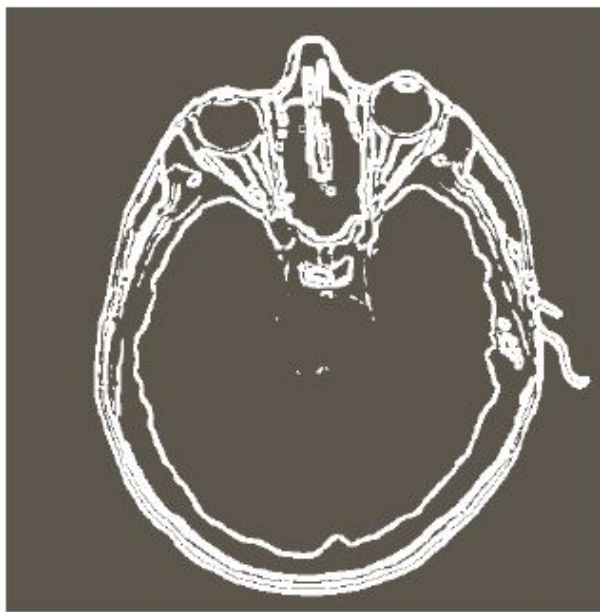
# Summary: Canny Edge detector

- (1) Convolve the image with a Gaussian Filter. Compute the gradient magnitudes and orientations.
- (2) Perform non-maximal suppression.
- (3) Perform hysteresis thresholding.



a	b
c	d

**FIGURE 10.25**  
 (a) Original image of size  $834 \times 1114$  pixels, with intensity values scaled to the range  $[0, 1]$ .  
 (b) Thresholded gradient of smoothed image.  
 (c) Image obtained using the Marr-Hildreth algorithm.  
 (d) Image obtained using the Canny algorithm. Note the significant improvement of the Canny image compared to the other two.



a	b
c	d

**FIGURE 10.26**

(a) Original head CT image of size  $512 \times 512$  pixels, with intensity values scaled to the range  $[0, 1]$ .

(b) Thresholded gradient of smoothed image.

(c) Image obtained using the Marr-Hildreth algorithm.

(d) Image obtained using the Canny algorithm.

(Original image courtesy of Dr. David R. Pickens, Vanderbilt University.)

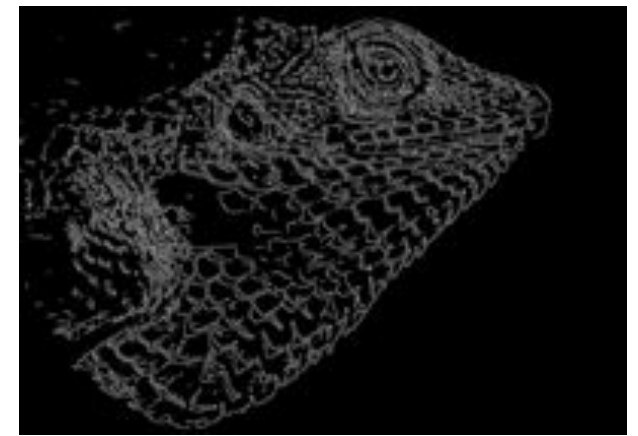
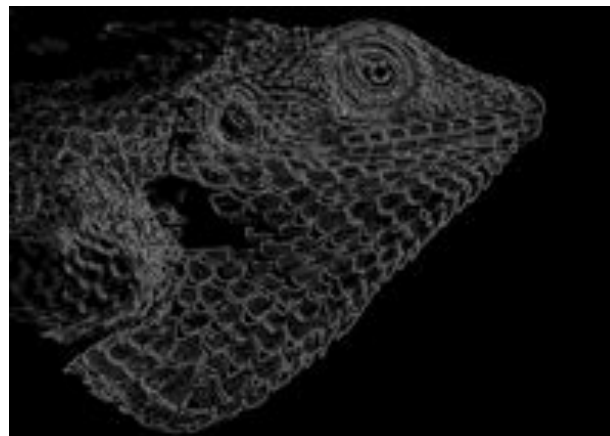
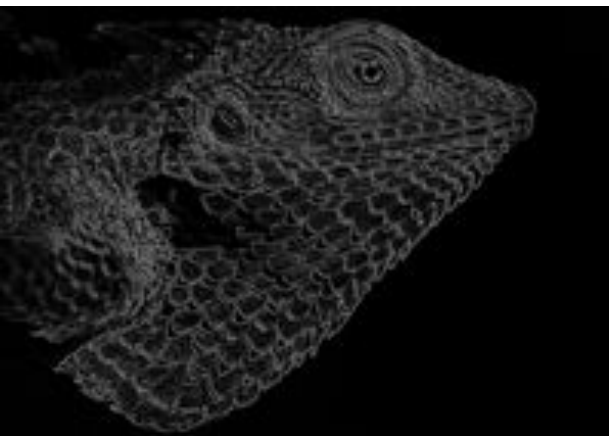




Gaussian filtered  
(sigma = 1.4, 5 × 5  
mask)

Intensity gradients  
superimposed

[https://en.wikipedia.org/wiki/Canny\\_edge\\_detector](https://en.wikipedia.org/wiki/Canny_edge_detector)



Non-maximum  
suppression

Double thresholding  
(TL = 0.1, TH = 0.3)

After edge-linking  
(hysteresis)