Digital Image Processing

Basic Methods for Image Segmentation

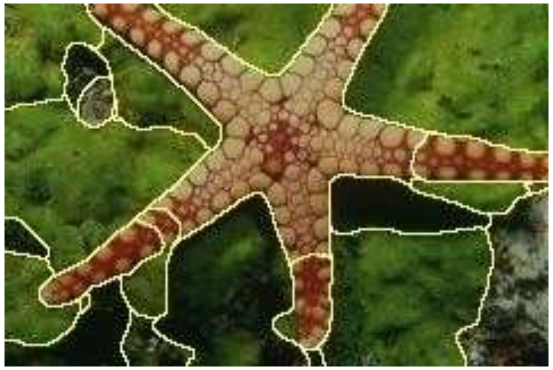
Image Segmentation

- Obtain a compact representation of the image to be used for further processing.
- Group together similar pixels
- Image intensity is not sufficient to perform *semantic* segmentation
  - Object recognition
    - Decompose objects to simple tokens (line segments, spots, corners)
  - Finding buildings in images
    - Fit polygons and determine surface orientations.
  - Video summarization
    - Shot detection

Image Segmentation (cont.)

Goal: separate an image into "coherent" regions.

- Basic methods
  - point, line, edge detection
  - thresholding
  - region growing
  - morphological watersheds
- Advanced methods
  - clustering
  - model fitting.
  - probabilistic methods.
  - …

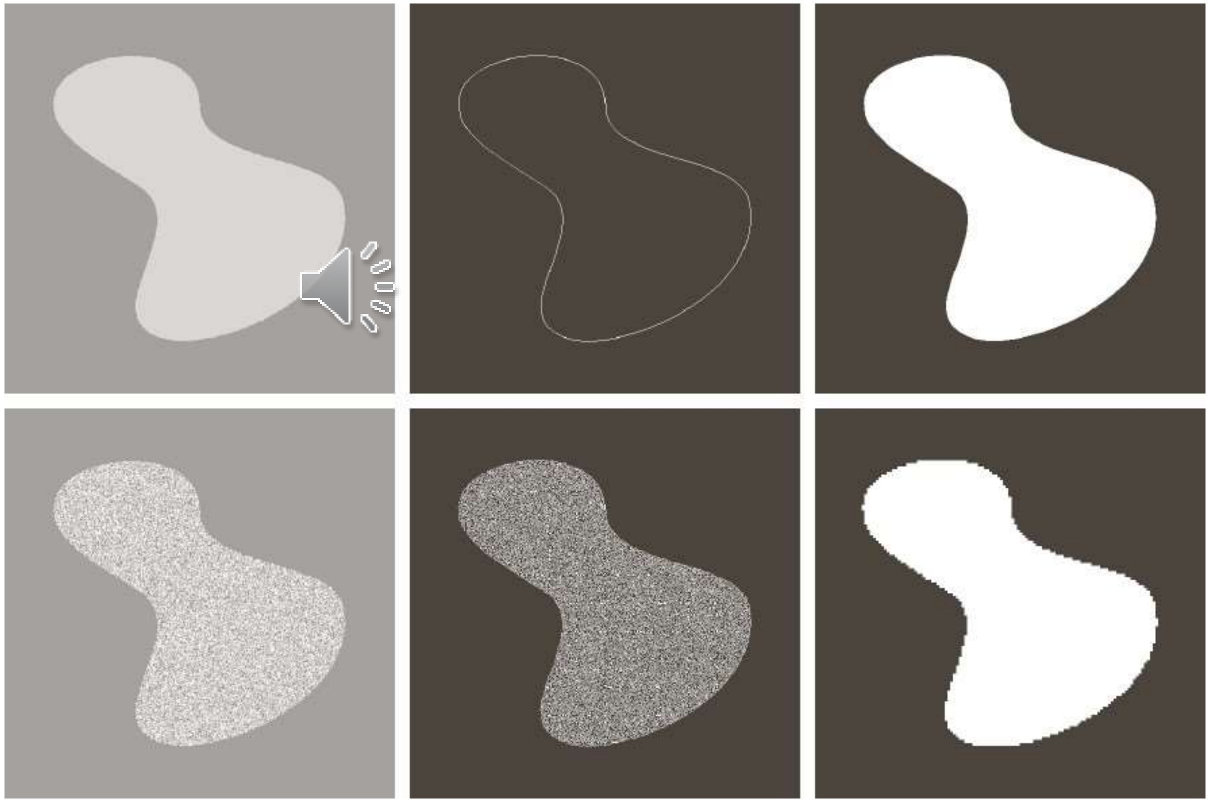


Fundamentals

- Edge information is in general not sufficient.

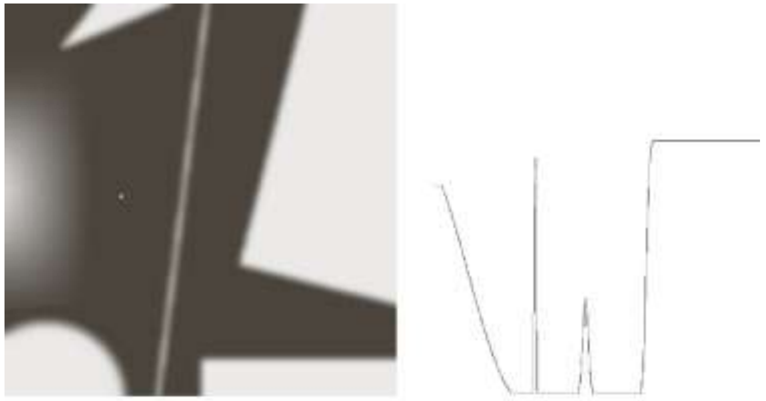
Constant intensity
(edge-based
segmentation)

Textured region
(region-based
segmantation)



Point, line and edge detection

- First order derivatives produce thick edges at ramps.
- Second order derivatives are non zero at the onset and at the end of a ramp or step edge (sign change).
- Second order derivatives respond stronger at isolated points and thin lines than first order derivatives.



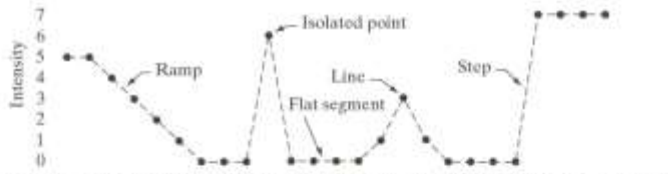


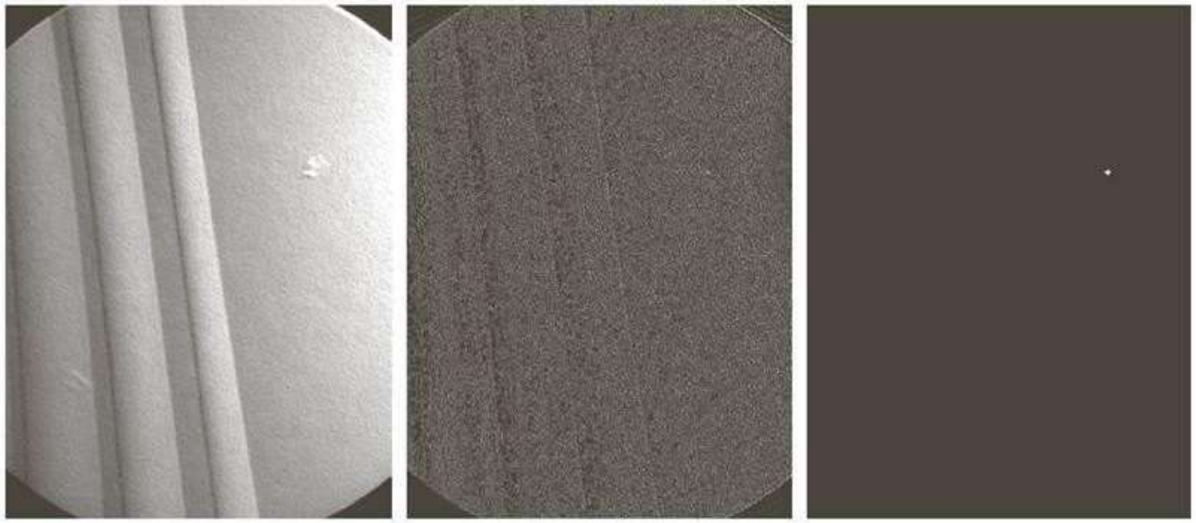
Image strip	5	5	4	3	2	1	0	0	0	6	0	0	0	0	1	3	1	0	0	0	0	7	7	7	7	•	•
First derivative		−1	−1	−1	−1	−1	0	0	6	−6	0	0	0	1	2	−2	−1	0	0	0	7	0	0	0		
Second derivative		−1	0	0	0	0	1	0	6	−12	6	0	0	1	1	−4	1	1	0	0	7	−7	0	0		

Detection of isolated points

- Simple operation using the Laplacian.

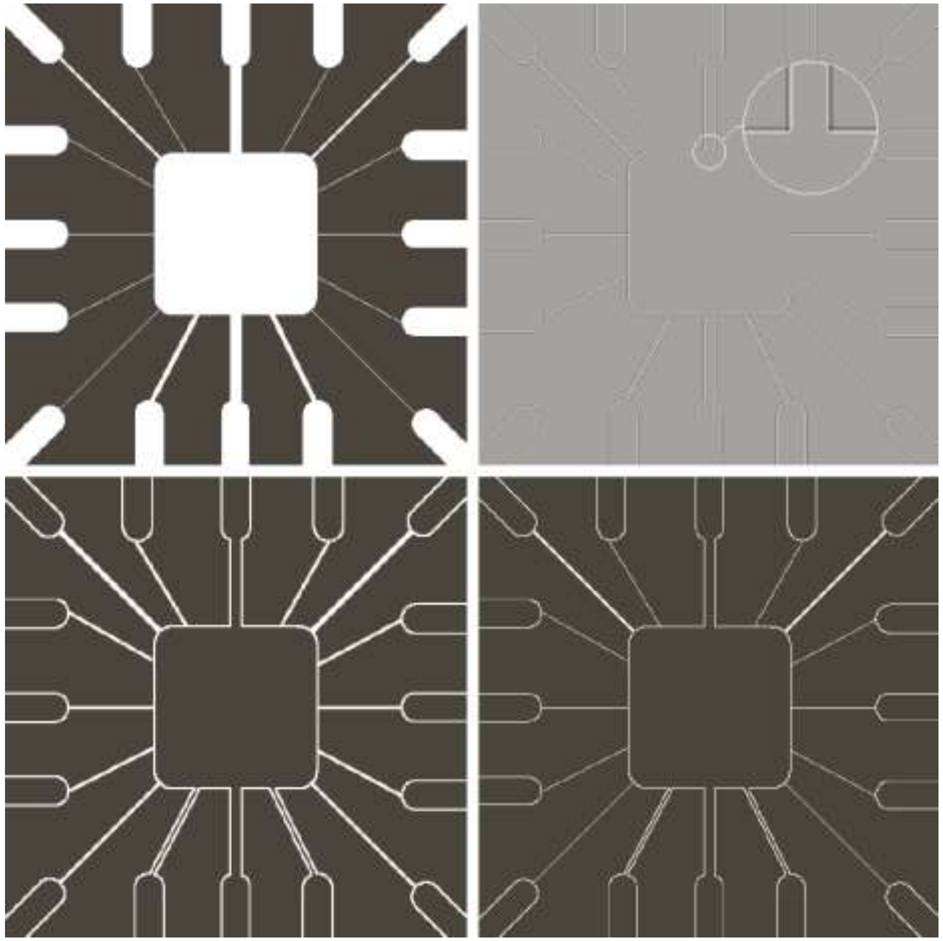
$$g(x, y) = \begin{cases} 1 & \text{if } \left|\nabla^2 f(x, y)\right| > T \\ 0 & \text{otherwise} \end{cases}$$

1	1	1
1	−8	1
1	1	1



Line detection

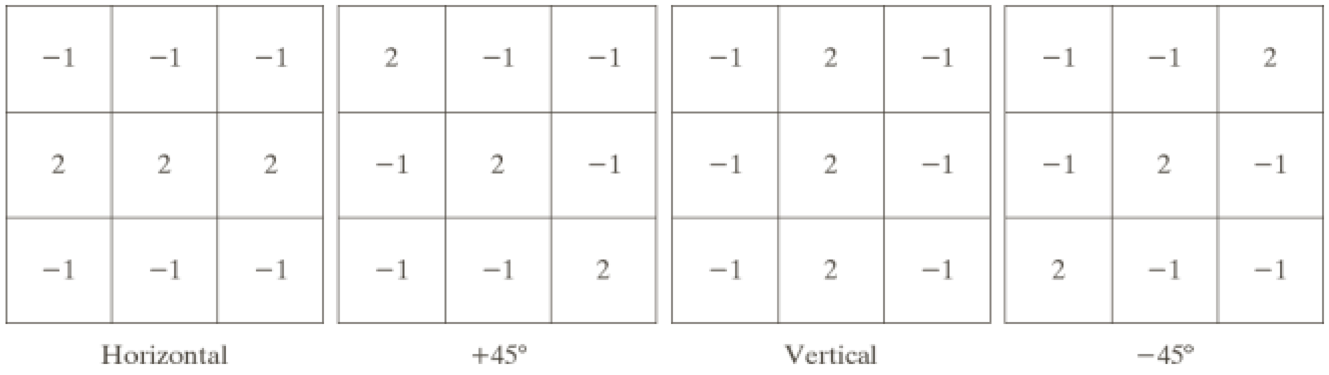
- The Laplacian is also used here.
- It has a double response
  - Positive and negative values at the beginning and end of the edges.
- Lines should be thin with respect to the size of the detector



Absolute value Positive value

Line detection (cont.)

- The Laplacian is isotropic.
- Direction dependent filters localize 1 pixel thick lines at other orientations (0, 45, 90).



−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°	

Line detection (cont.)

- The Laplacian is isotropic.
- Direction dependent filters localize 1 pixel thick lines at other orientations (0, 45, 90).

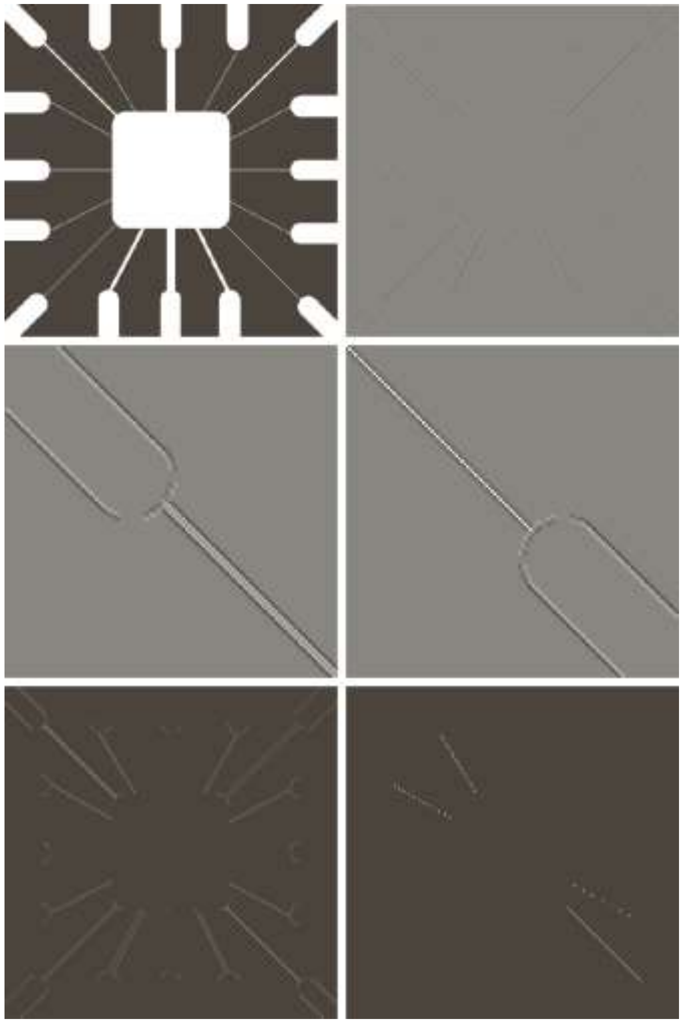
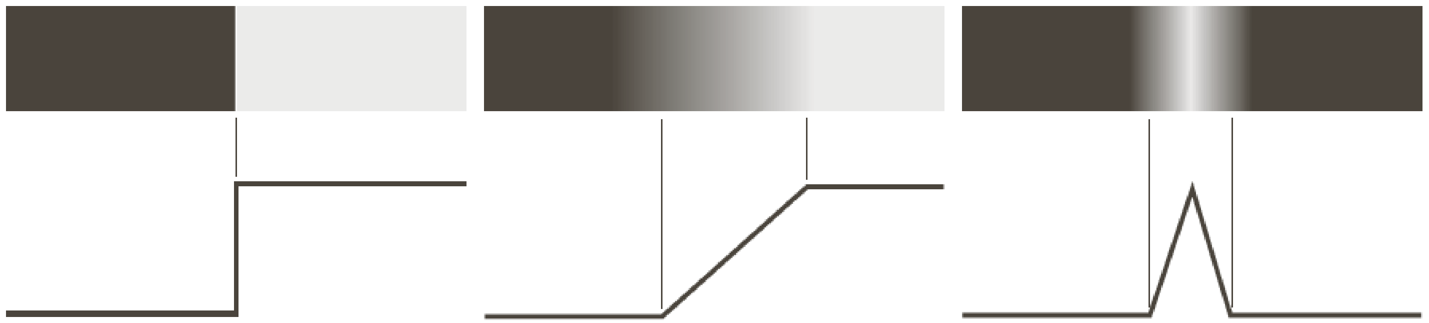

a b
c d
e f

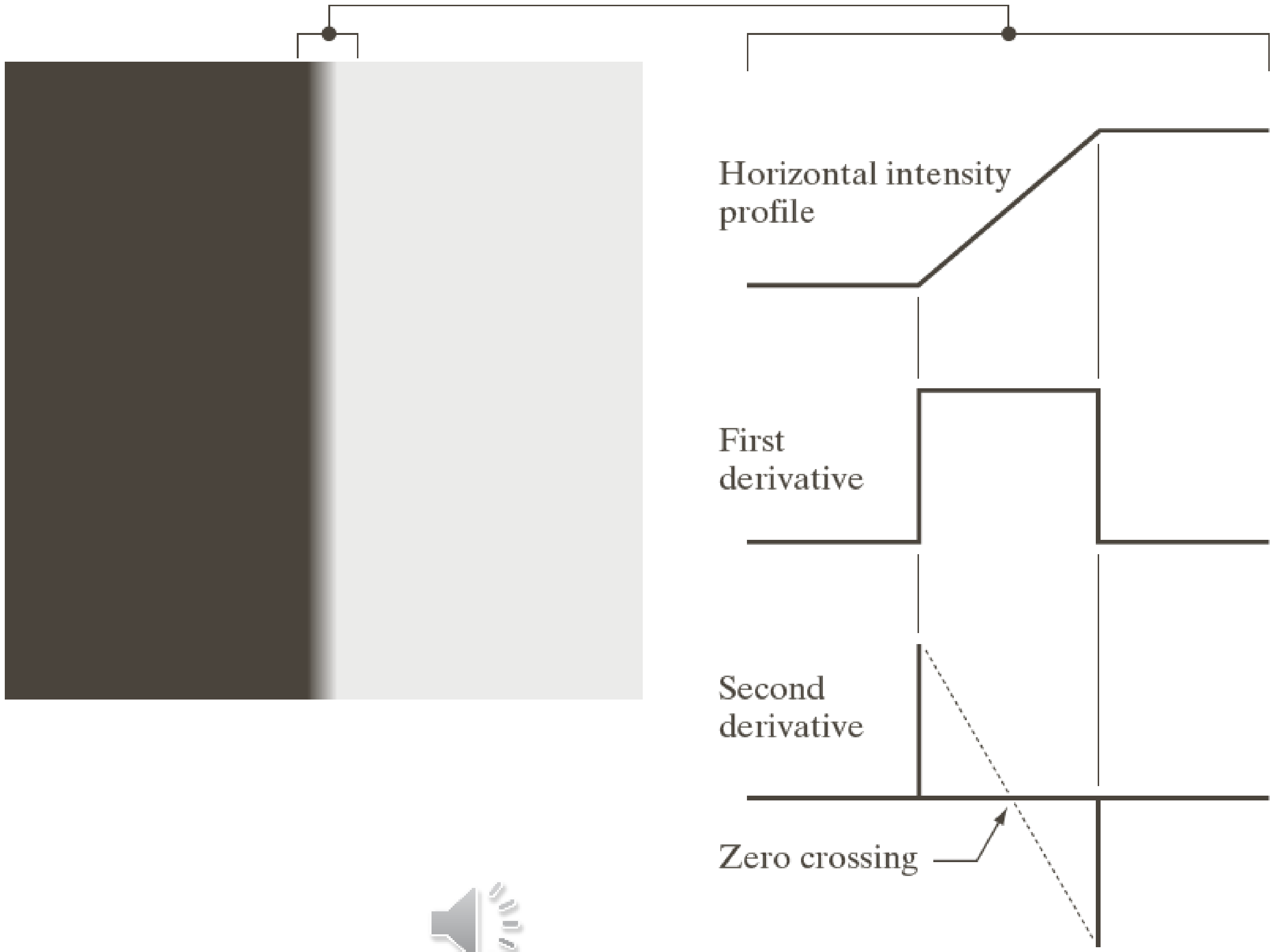
FIGURE 10.7 (a) Image of a wire-bond template. (b) Result of processing with the +45° line detector mask in Fig. 10.6. (c) Zoomed view of the top left region of (b). (d) Zoomed view of the bottom right region of (b). (e) The image in (b) with all negative values set to zero. (f) All points (in white) whose values satisfied the condition $g \geq T$, where g is the image in (e). (The points in (f) were enlarged to make them easier to see.)

Edge detection

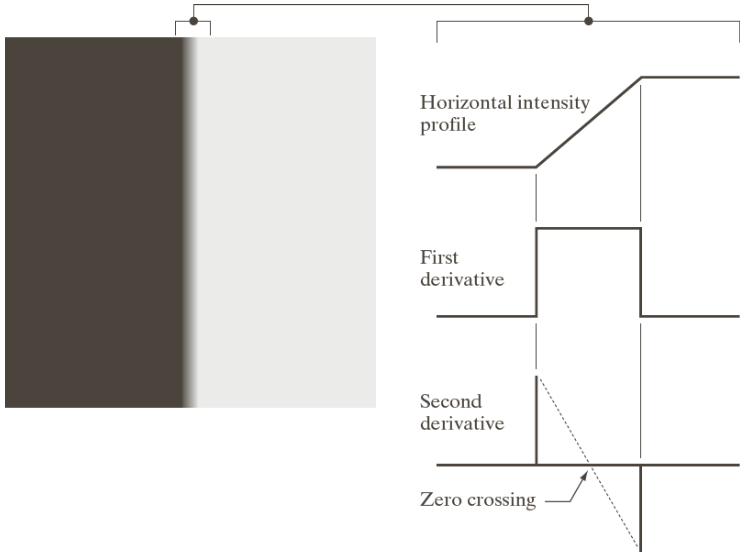
- Edge models
  - Ideally, edges should be 1 pixel thin.
  - In practice, they are blurred and noisy.



Edge detection (cont.)



Edge detection (cont.)



Edge point detection

- Magnitude of the first derivative.
- Sign change of the second derivative.

Observations:

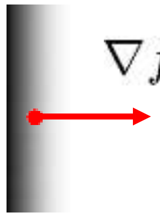
- Second derivative produces two values for an edge (undesirable).
- Its zero crossings may be used to locate the centres of thick edges.

Fundamental steps in edge detection

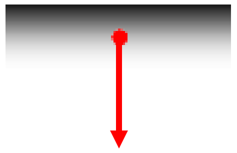
- Image smoothing for noise reduction.
  - Derivatives are very sensitive to noise.
- Detection of candidate edge points.
- Edge localization.
  - Selection of the points that are true members of the set of points comprising the edge.

Image gradient

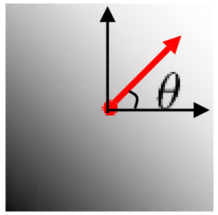
- The gradient of an image: $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$

- 

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



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



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

The gradient points in the direction of most rapid increase 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

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

Source: Steve Seitz

Gradient operators

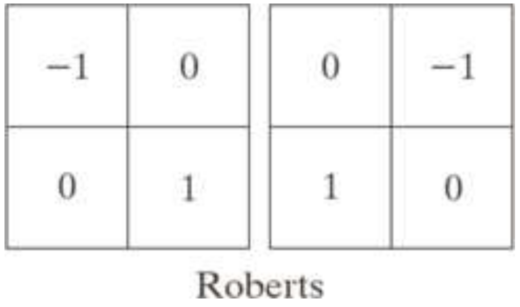
$$\frac{\partial f(x,y)}{\partial x} = f(x+1,y) - f(x,y)$$

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

Roberts operators

−1
1

−1	1

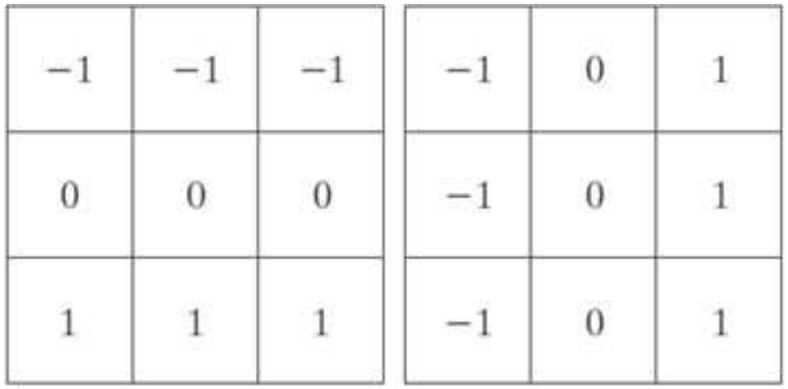


−1	0
0	1

0	−1
1	0

Roberts

Gradient operators (cont.)

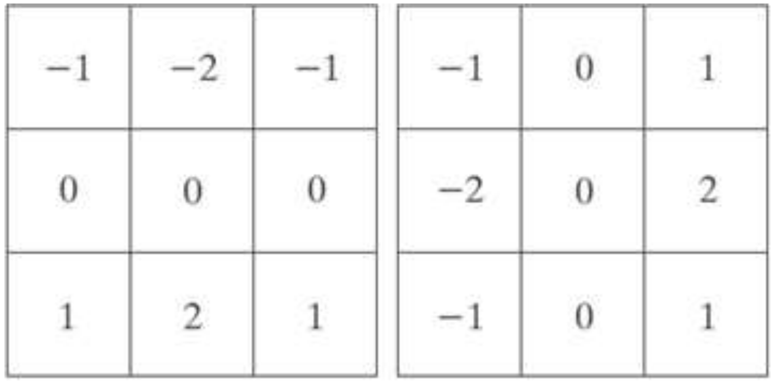


−1	−1	−1
0	0	0
1	1	1

−1	0	1
−1	0	1
−1	0	1

Prewitt

Integrates image smoothing



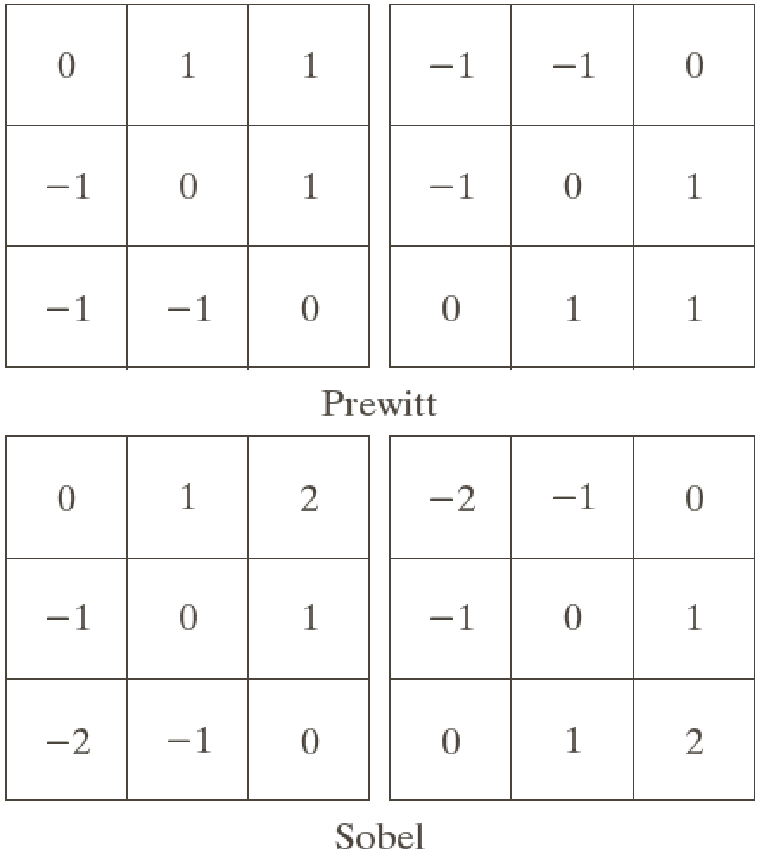
−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

Sobel

Gradient operators (cont.)

Diagonal edges



0	1	1
−1	0	1
−1	−1	0

−1	−1	0
−1	0	1
0	1	1

Prewitt

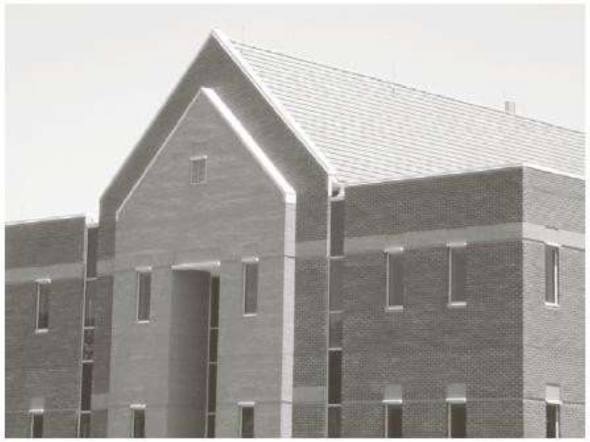
0	1	2
−1	0	1
−2	−1	0

−2	−1	0
−1	0	1
0	1	2

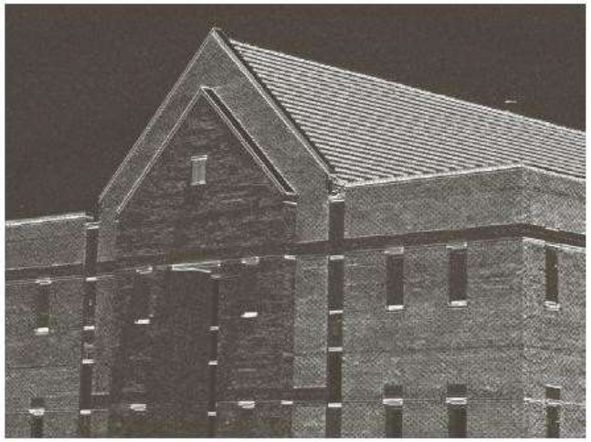
Sobel

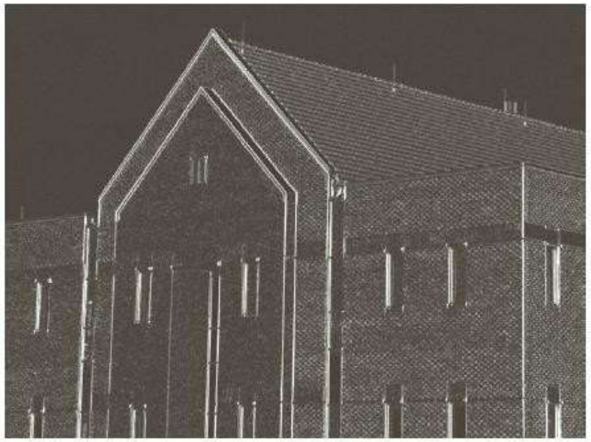
Gradient operators (cont.)

Image

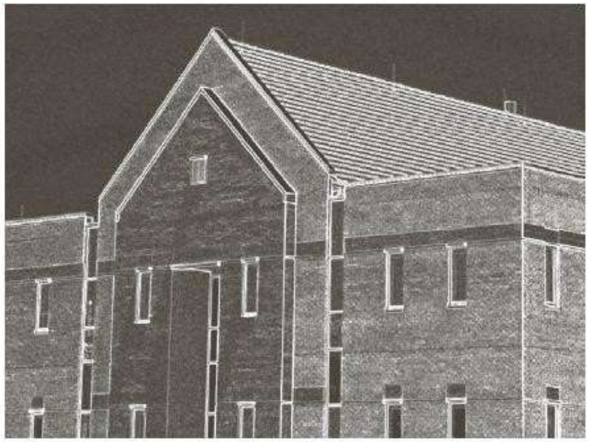


Sobel $|g_y|$





Sobel $|g_x|$



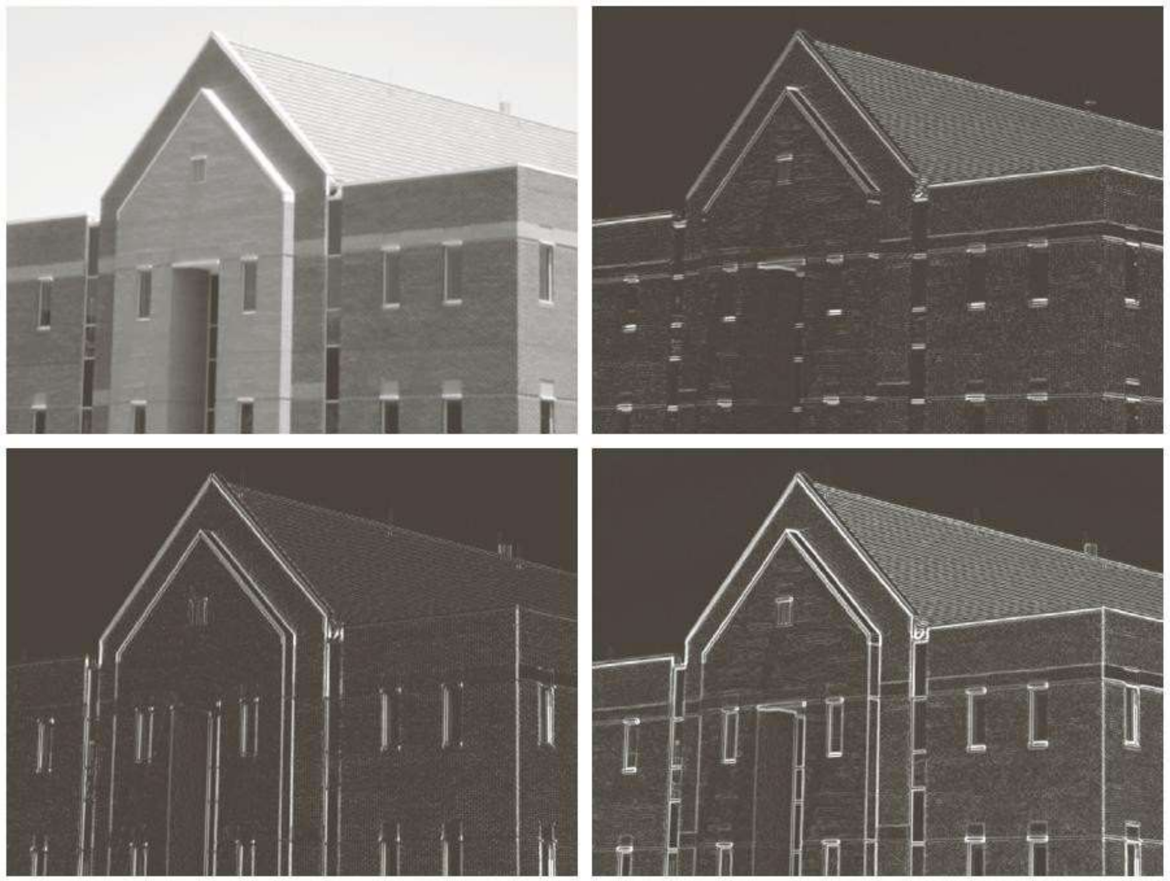
Sobel $|g_x|+|g_x|$

Gradient operators (cont.)

Image smoothed prior to edge detection.

The wall bricks are smoothed out.

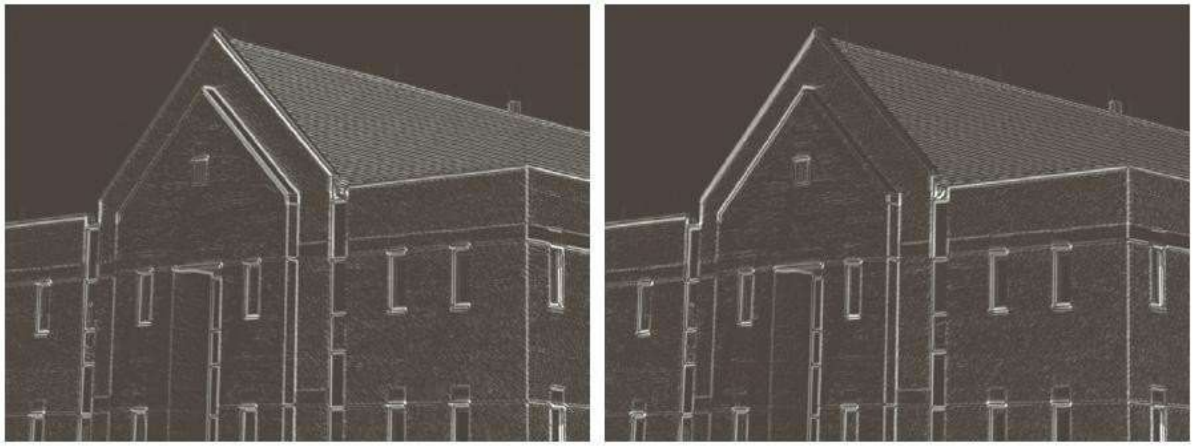
Image　　Sobel $|g_y|$



Sobel $|g_x|$　　Sobel $|g_x|+|g_x|$

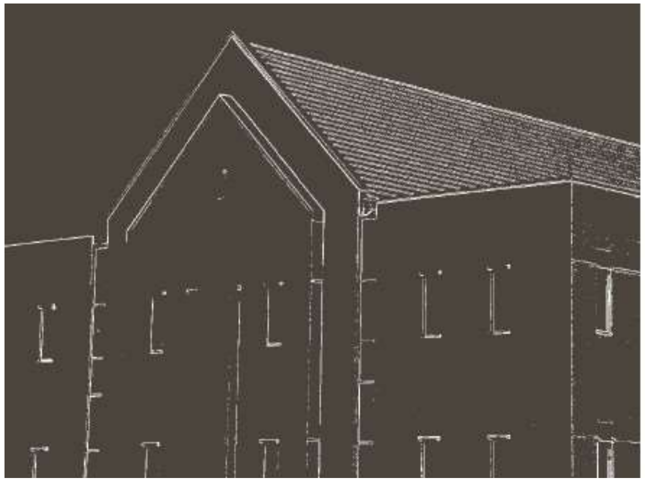
Gradient operators (cont.)

Diagonal Sobel filters



Gradient operators (cont.)

Thresholded Sobel gradient amplitydes at 33% of max value



Thresholded gradient

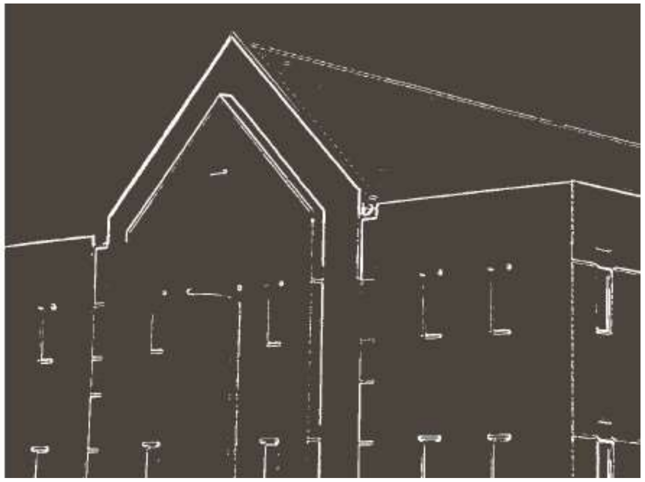


Image smoothing prior to gradient thresholding

The LoG operator

- A good place to look for edges is the maxima of the first derivative or the zeros of the second derivative.
- The 2D extension approximates the second derivative by the Laplacian operator (which is rotationally invariant):

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

- Marr and Hildreth [1980] argued that a satisfactory operator that could be tuned in scale to detect edges is the Laplacian of the Gaussian (LoG).

The LoG operator (cont.)

$$\nabla^2\left(G(x,y) * f(x,y)\right) = \nabla^2 G(x,y) * f(x,y)$$

- The LoG operator is given by:

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

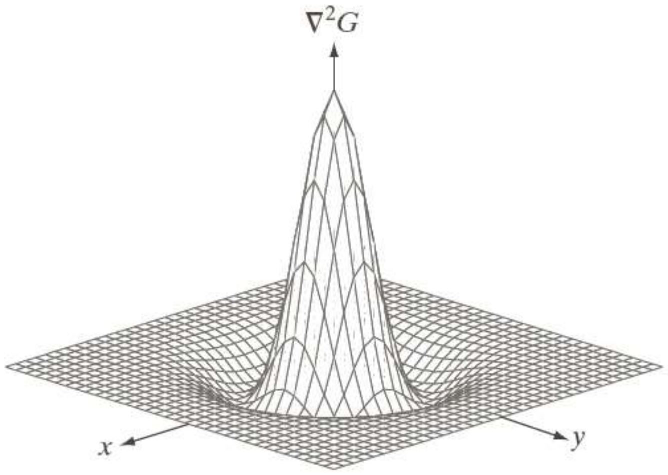
$$\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right) e^{-\frac{x^2+y^2}{2\sigma^2}} = \left(\frac{x^2+y^2-2\sigma^2}{o^4}\right) e^{-\frac{x^2+y^2}{2\sigma^2}}$$

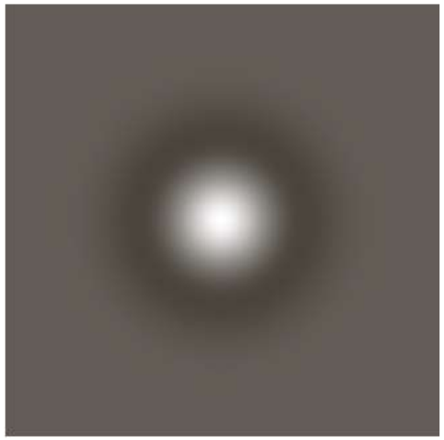
The LoG operator (cont.)

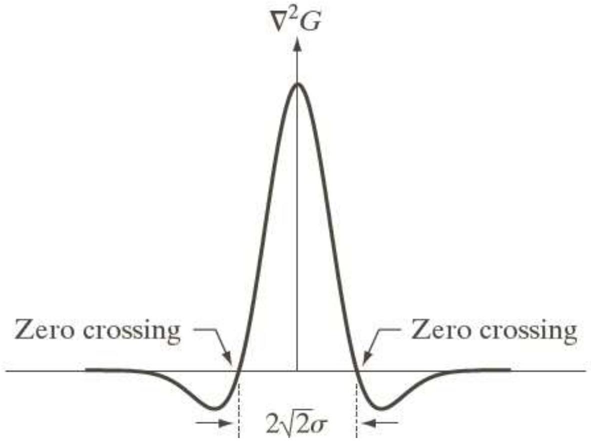
$$\nabla^2 G(x,y) = \frac{\partial^2 G}{\partial x^2} + \frac{\partial^2 G}{\partial y^2} = \left(\frac{x^2 + y^2 - 2\sigma^2}{\sigma^4} \right) e^{-\frac{x^2+y^2}{2\sigma^2}}$$

The zero crossings are at:

$$x^2 + y^2 = 2\sigma^2$$





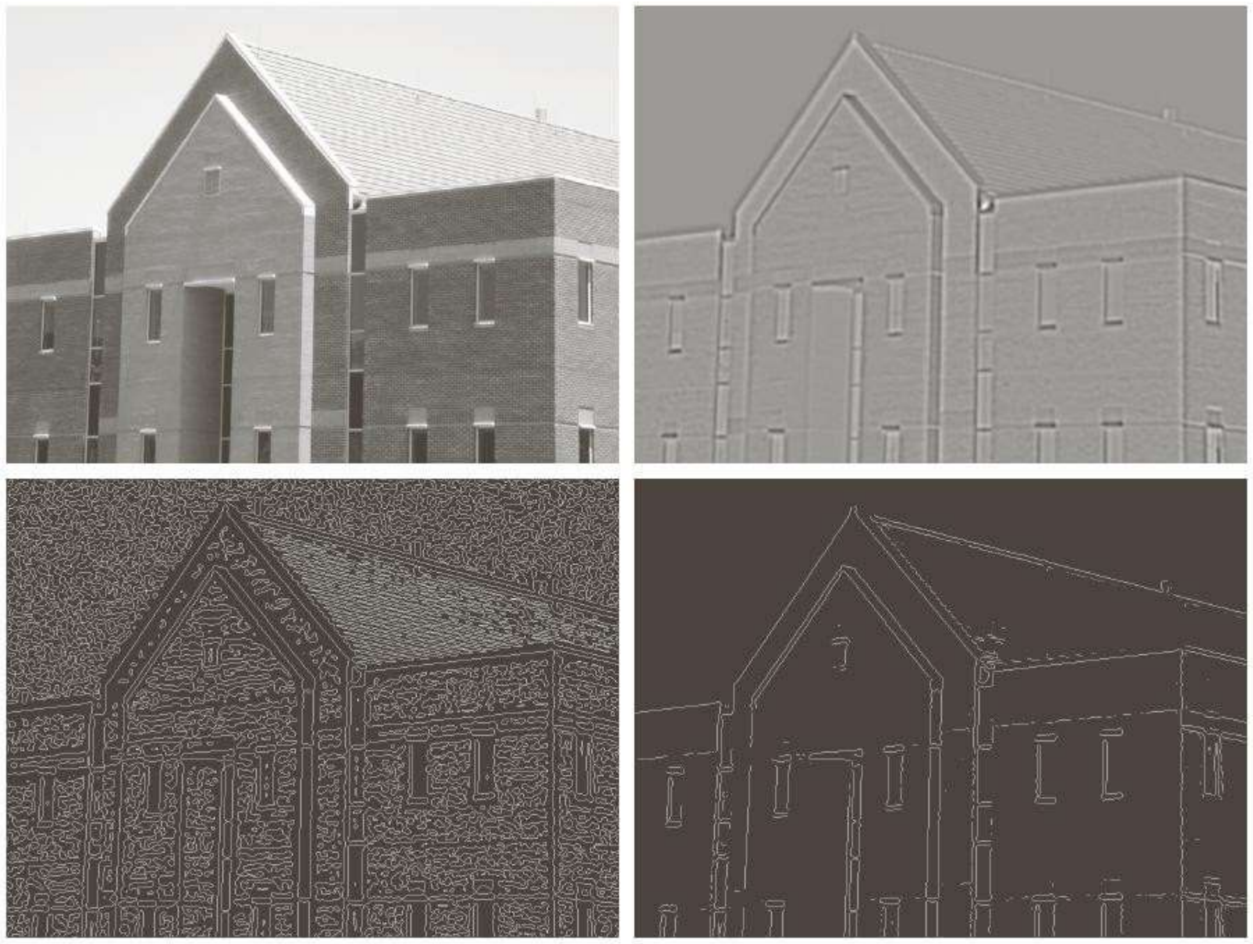


0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

The LoG operator (cont.)

- Fundamental ideas
  - The Gaussian blurs the image. Iт reduces the intensity of structures at scales much smaller than σ.
  - The Laplacian is isotropic and no other directional mask is needed.
- The zero crossings of the operator indicate edge pixels. They may be computed by using a 3x3 window around a pixel and detect if two of its opposite neighbors have different signs (and their difference is significant compared to a threshold).

The LoG operator (cont.)



Image

LoG

Zero crossings

Zero crossings with a threshold of 4% of the image max

The LoG operator (cont.)

- Filter the image at various scales and keep the zero crossings that are common to all responses.
- Marr and Hildreth [1980] showed that LoG may be approximated by a difference of Gaussinas (DOG):

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

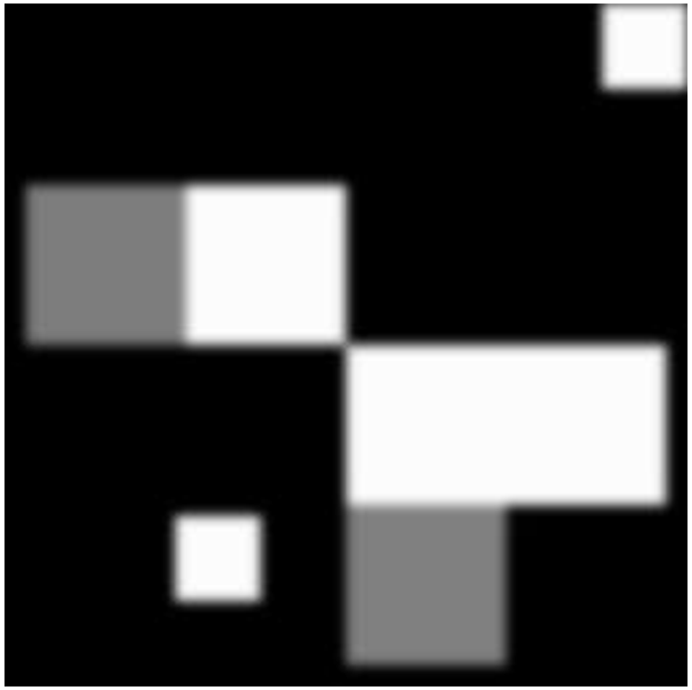
- Certain channels of the human visual system are selective with respect to orientation and frequency and can be modeled by a DoG with a ratio of standard deviations of 1.75.

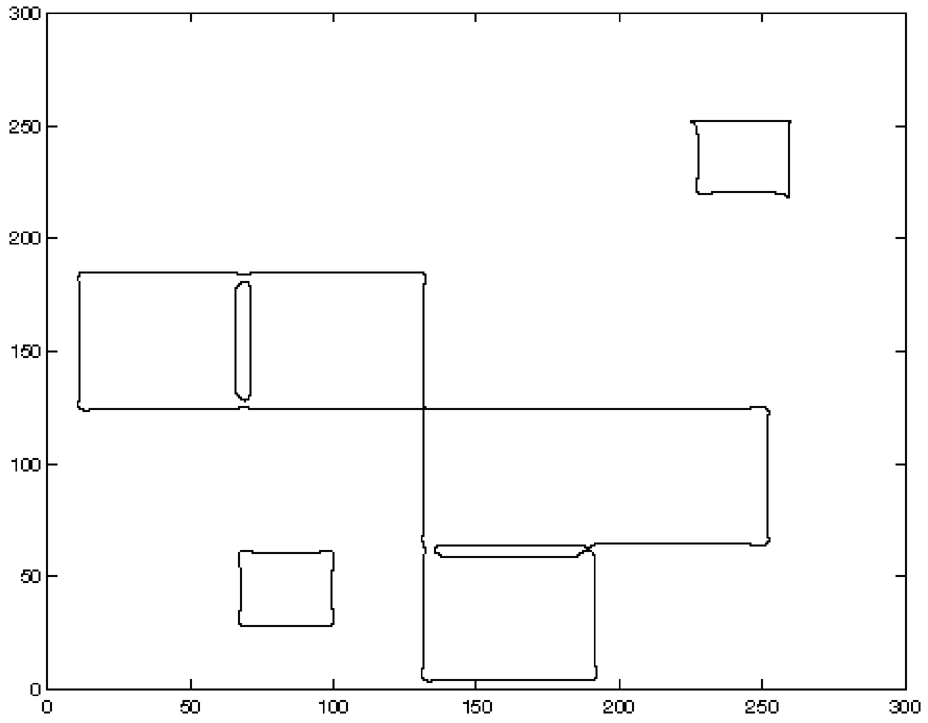
The LoG operator (cont.)

- LoG has fallen to some disfavour.
- It is not oriented and its response averages the responses across the two directions.
  - Poor response at corners.
  - Difficulty in recording the topology of T-junctions (trihedral vertices).
- Several studies showed that image components along an edge contribute to the response of LoG to noise but not to a true edge. Thus, zero crossings may not lie exactly on an edge.

The LoG operator (cont.)

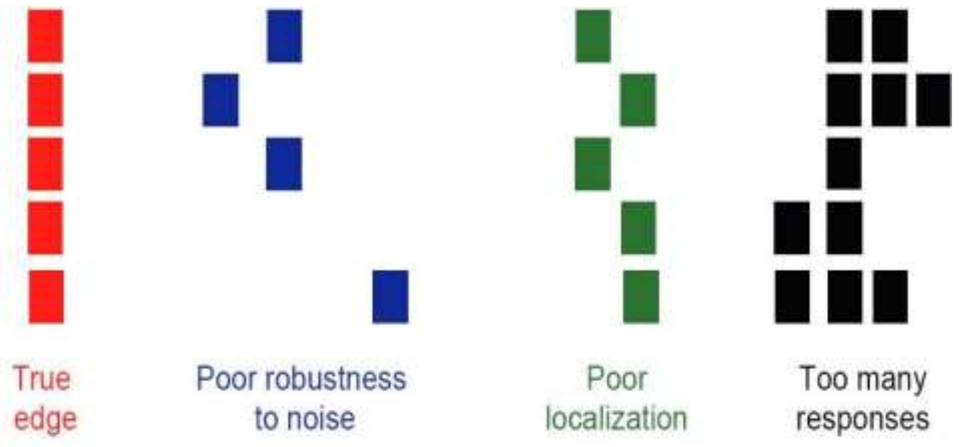
Poor corner and trihedral vertices detection





Designing an optimal edge detector

- Criteria for an “optimal” edge detector [Canny 1986]:
  - **Good detection:** the optimal detector must minimize the probability of false positives (detecting spurious edges caused by noise), as well as that of false negatives (missing real edges).
  - **Good localization:** the edges detected must be as close as possible to the true edges
  - **Single response:** the detector must return one point only for each true edge point; that is, minimize the number of local maxima around the true edge.



Canny edge detector

1. Convolution with derivative of Gaussian
2. Non-maximum Suppression
3. Hysteresis Thresholding

Canny edge detector

Step 1: **Convolution with derivative of Gaussian**

Smooth by Gaussian

$$S = G_\sigma * I \qquad G_\sigma = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Compute *x* and *y* derivatives

$$\nabla S = \left[\frac{\partial}{\partial x} S \quad \frac{\partial}{\partial y} S \right]^T = \left[S_x \quad S_y \right]^T$$

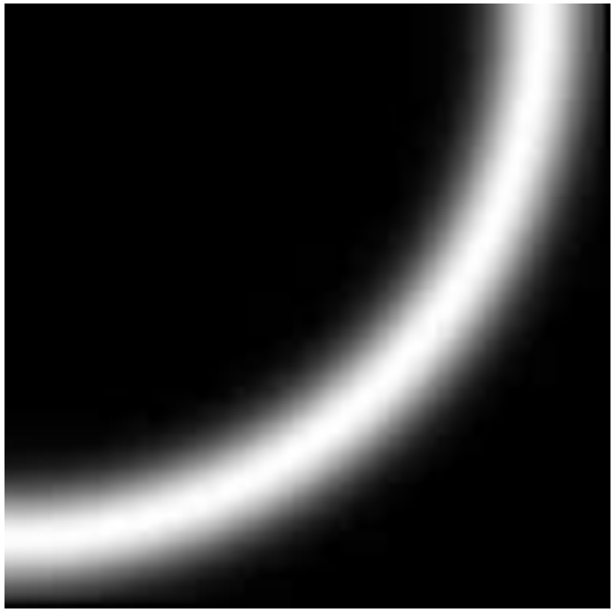
Compute gradient magnitude and orientation

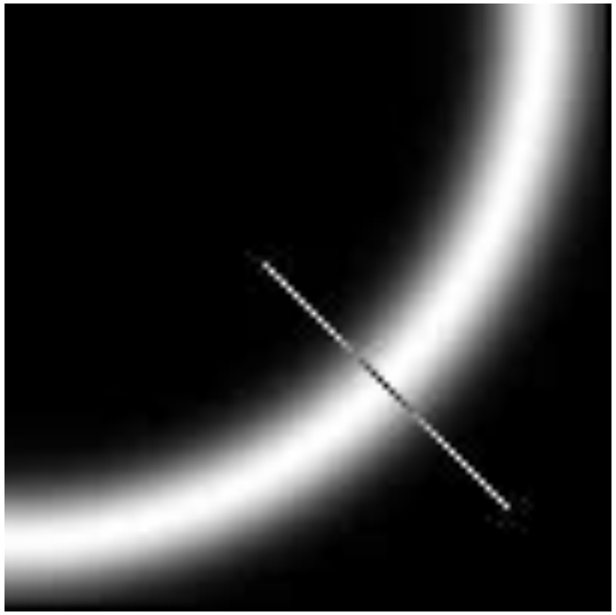
$$|\nabla S| = \sqrt{S_x^2 + S_y^2}$$

$$\theta = \tan^{-1} \frac{S_y}{S_x} \qquad \nabla S = \nabla (G_\sigma * I) = \nabla G_\sigma * I$$

Canny edge detector (cont.)

Step 2:Non-Maximum Suppression



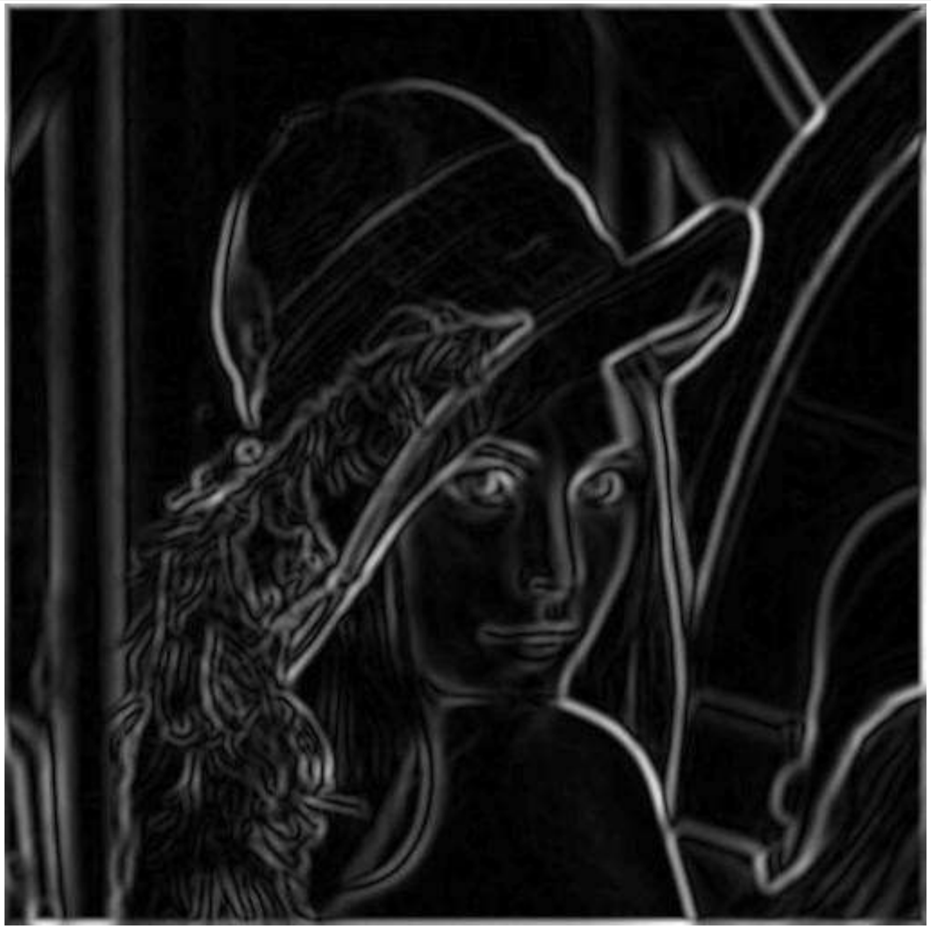


We wish to mark points along the curve where the magnitude is biggest. We can do this by looking for a maximum along a slice normal to the curve (non-maximum suppression). These points should form a curve. There are then two algorithmic issues: at which point is the maximum, and where is the next one?

Canny edge detector (cont.)

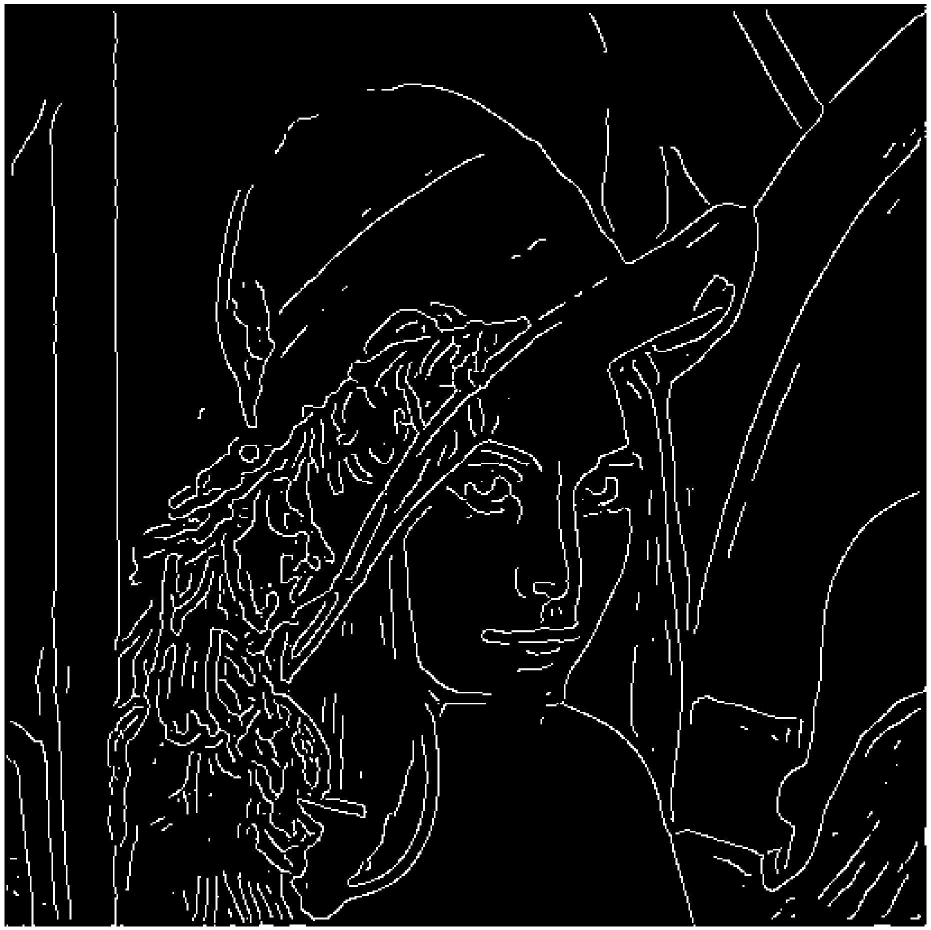
original image

Canny edge detector (cont.)



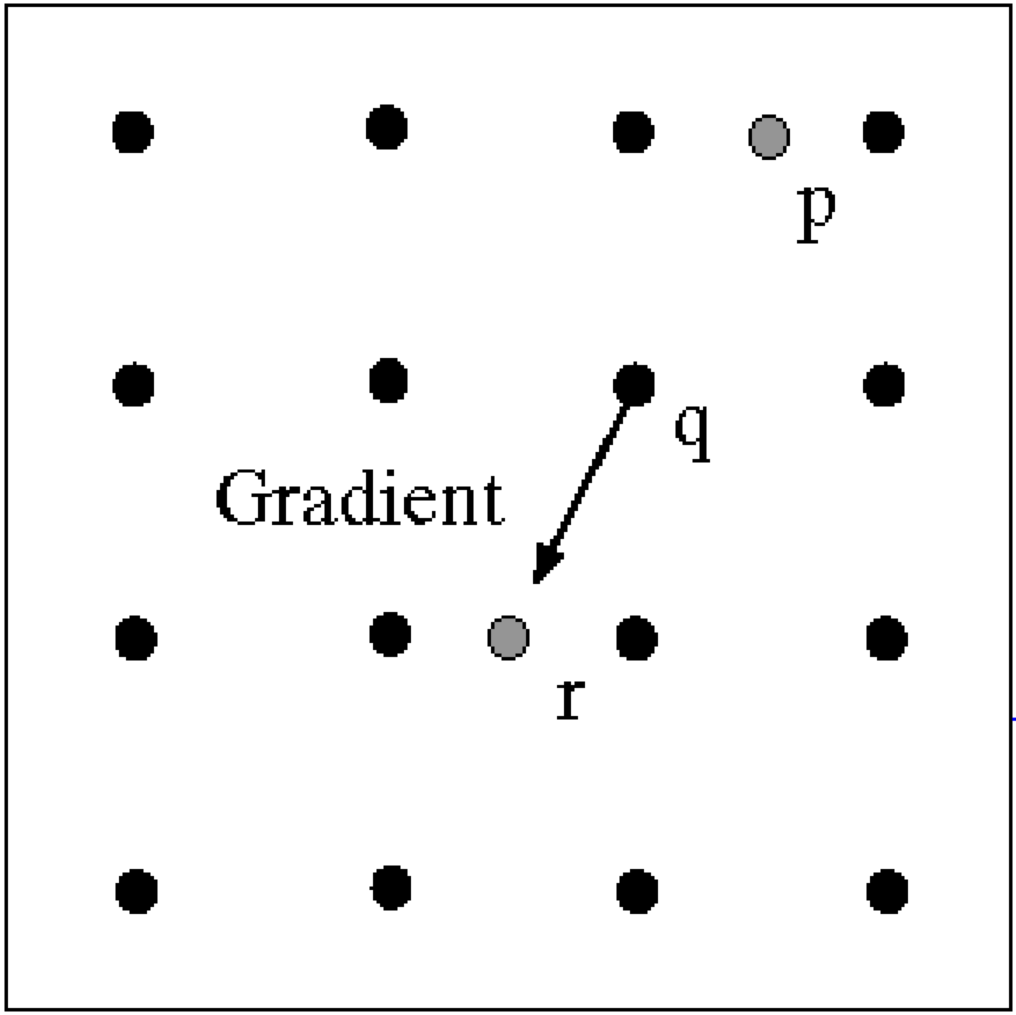
Gradient magnitude

Canny edge detector (cont.)



Non-maximum suppression and hysteresis thresholding

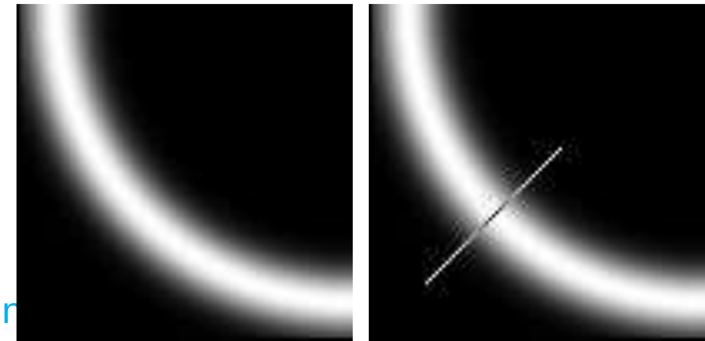
Non-maximum suppression (cont.)



At pixel q, we have a maximum if the value of the gradient magnitude is larger than the values at both p and at r.

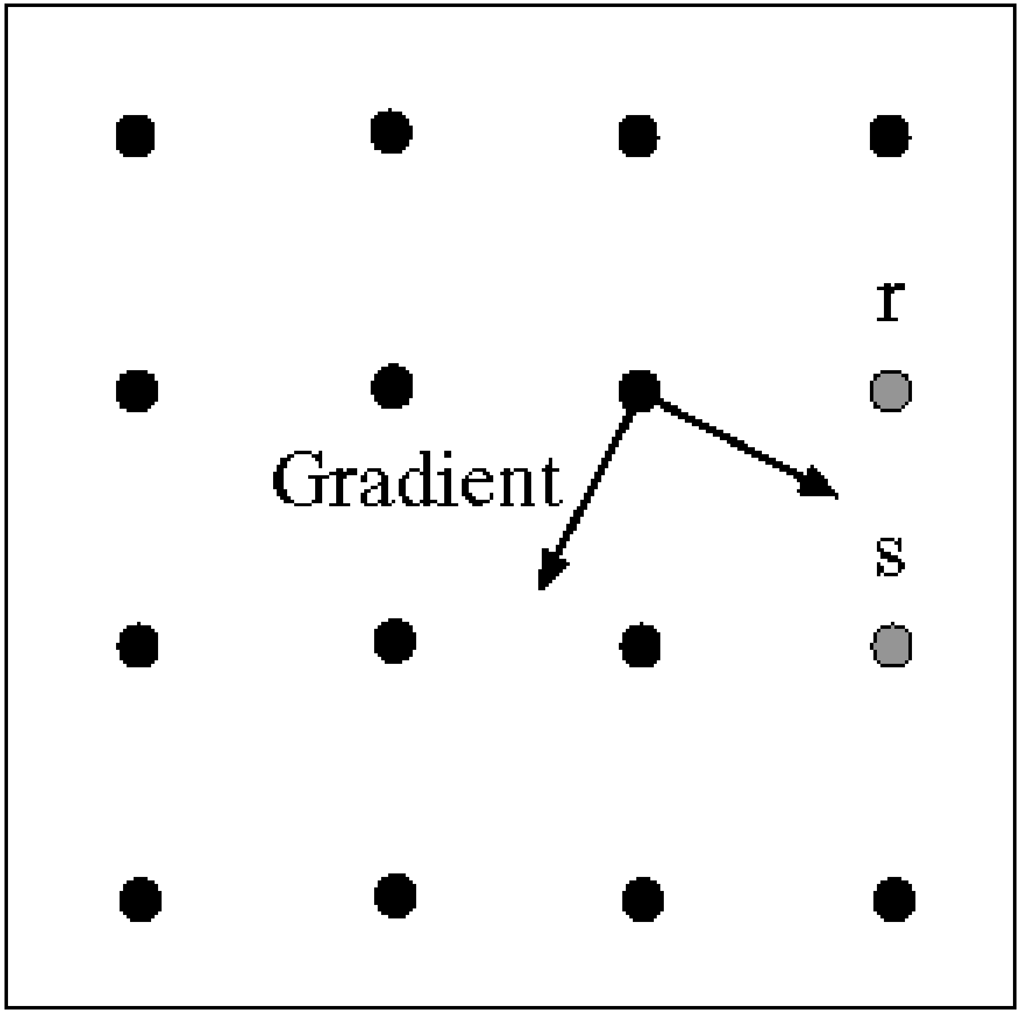
Interpolation provides these values.

http://justin-liang.com/tutorials/canny/

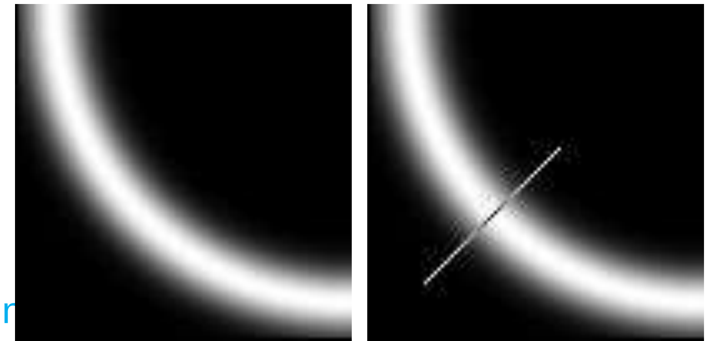


Source: D. Forsyth

Predict the next edge point



Assume the marked point is an edge point. Then we construct the tangent to the edge curve (which is normal to the gradient at that point) and use this to predict the next points (here either r or s).



Source: D. Forsyth

Hysteresis Thresholding

Reduces false edge pixels. It uses a low (T_L) and a high threshold (T_H) to create two additional images from the gradient magnitude image $g(x,y)$:

$$g_L(x,y) = \begin{cases} g(x,y) & g(x,y) \geq T_L \\ 0 & \text{otherwise} \end{cases}, \quad g_H(x,y) = \begin{cases} g(x,y) & g(x,y) \geq T_H \\ 0 & \text{otherwise} \end{cases}$$

$g_L(x,y)$ has more non zero pixels than $g_H(x,y)$.

We eliminate from $g_L(x,y)$ all the common non zero pixels:

$$g_L(x,y) = g_L(x,y) - g_H(x,y)$$

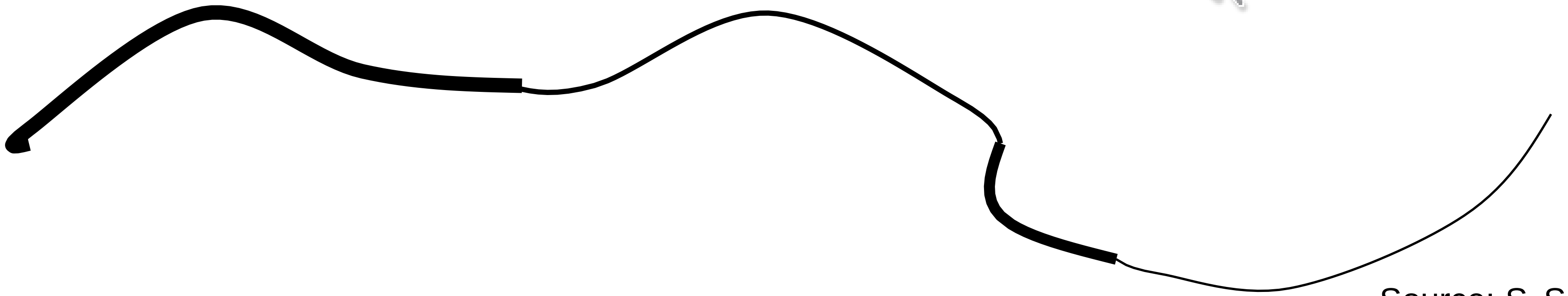
$g_L(x,y)$ and $g_H(x,y)$ may be viewed as *weak* and *strong* edge pixels.

Canny suggested a ratio of 2:1 to 3:1 between the thresholds.

Source: S. Seitz

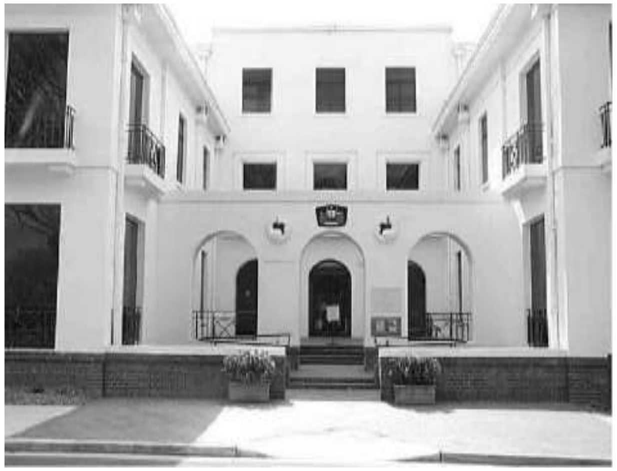
Hysteresis Thresholding (cont.)

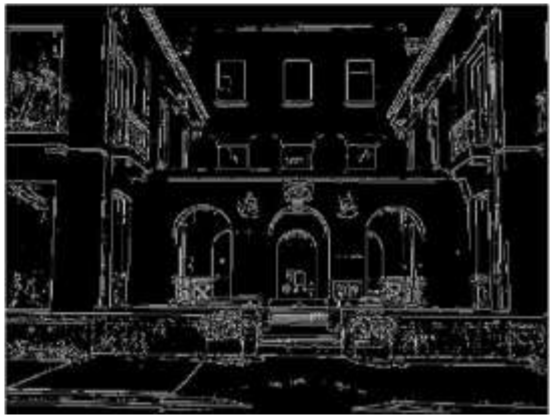
- After the thresholdings, all strong pixels are assumed to be valid edge pixels. Depending on the value of T_H, the edges in $g_H(x,y)$ typically have gaps.
- All pixels in $g_L(x,y)$ are considered valid edge pixels if they are 8-connected to a valid edge pixel in $g_H(x,y)$.

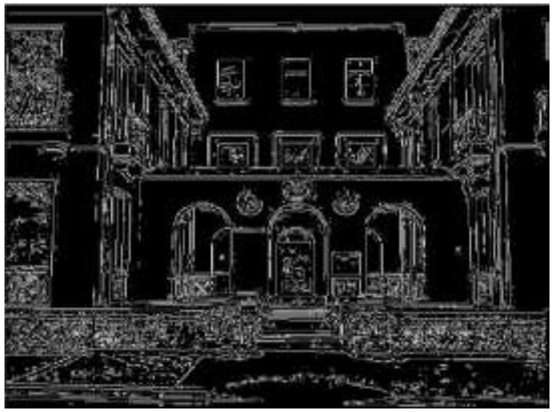


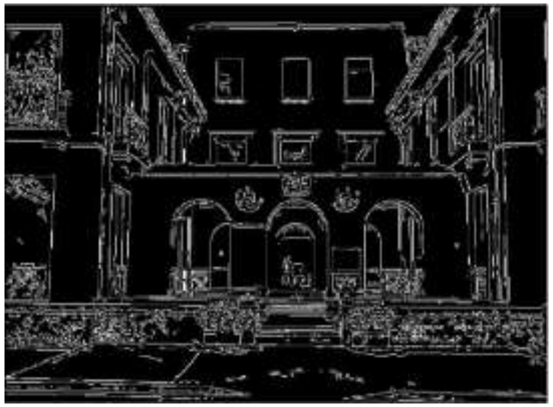
Source: S. Seitz

Hysteresis thresholding (cont.)



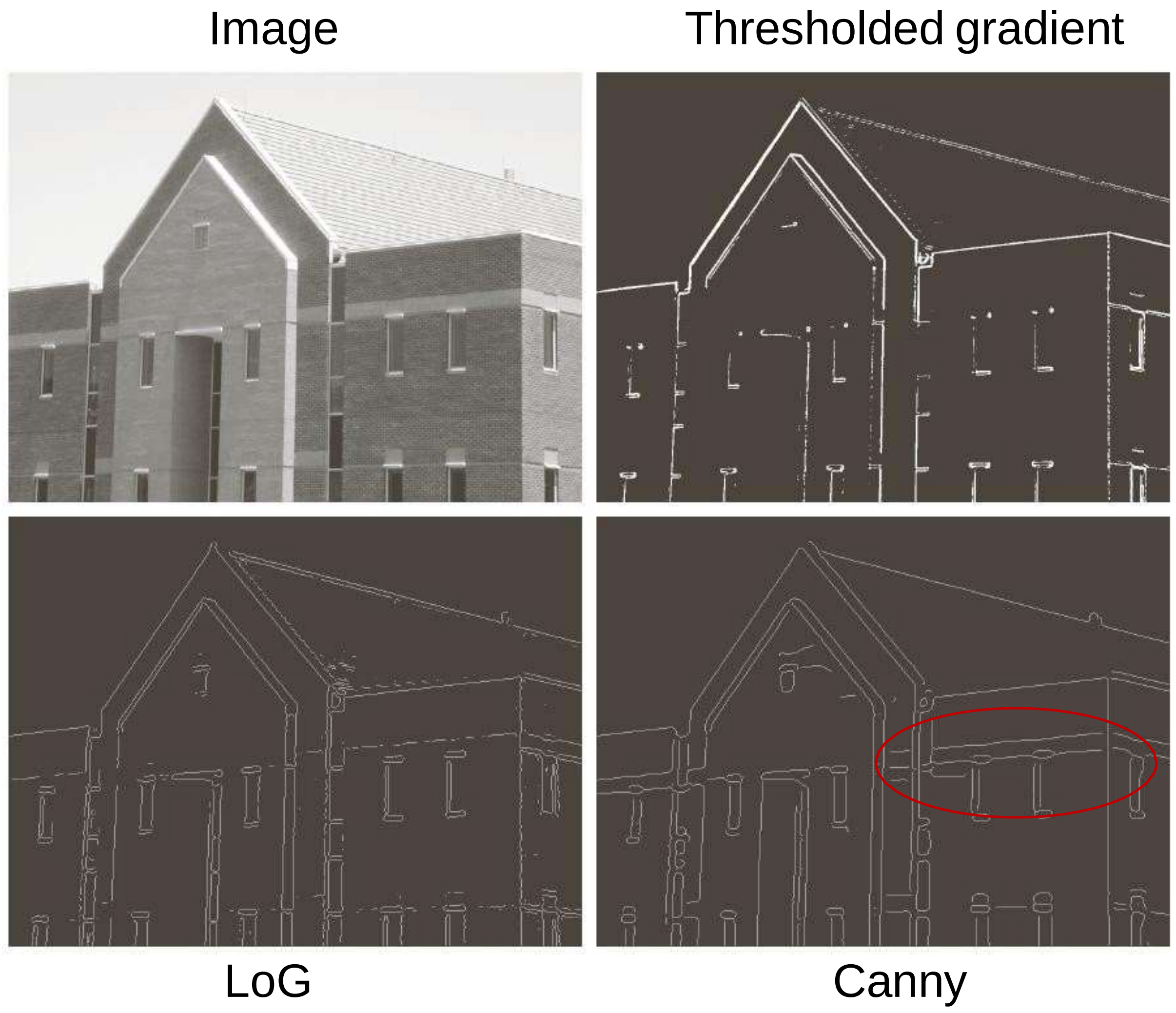

high threshold
(strong edges)


low threshold
(weak edges)


hysteresis threshold

Source: L. Fei-Fei

Canny vs LoG



Both edges of the concrete band lining the bricks were preserved.

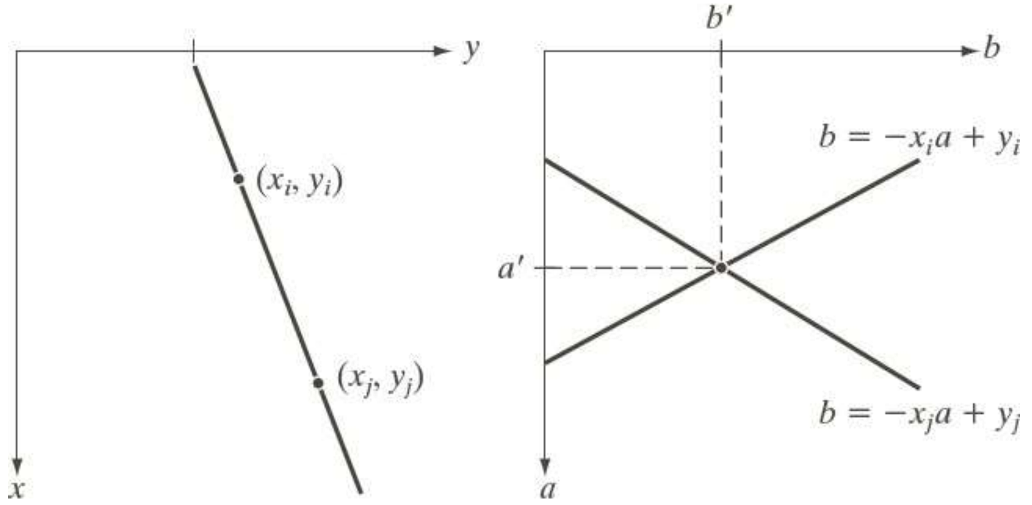
Edge Linkin

- Even after hysteresis thresholding, the detected pixels do not completely characterize edges completely due to occlusions, non-uniform illumination and noise. Edge linking may be:
  - **Local**: requiring knowledge of edge points in a small neighborhood.
  - **Regional**: requiring knowledge of edge points on the boundary of a region.
  - **Global**: the Hough transform, involving the entire edge image.

Hough transform

- An early type of voting scheme.
- Each line is defined by two points (x_i,y_i) and (x_j,y_j).
- Each point (x_i,y_i) has a line parameter space (a,b) because it belongs to an infinite number of lines $y_i=ax_i+b$.
- All the points (x,y) on a line $y=a'x+b'$ have lines in parameter space that intersect at $(a'x+b')$.

Image space

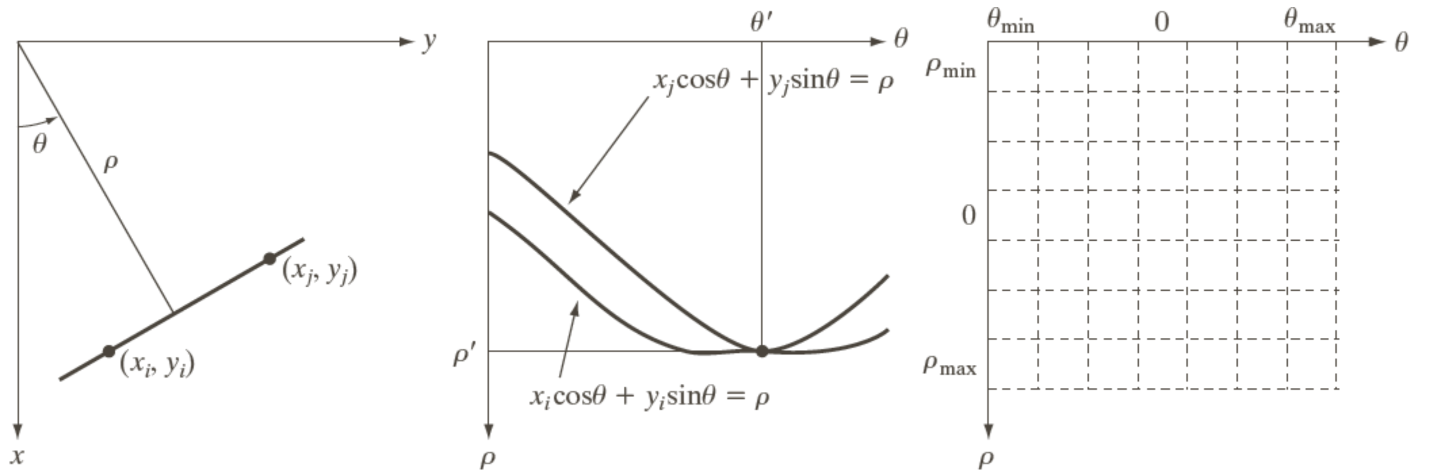


Parameter space

Hough transform (cont.)

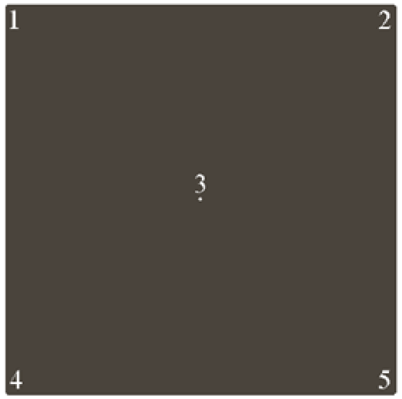
- Problems with the (a,b) parameter space:
  - Unbounded parameter domain
  - Vertical lines require infinite a.
- Polar (normal) representation of a line:

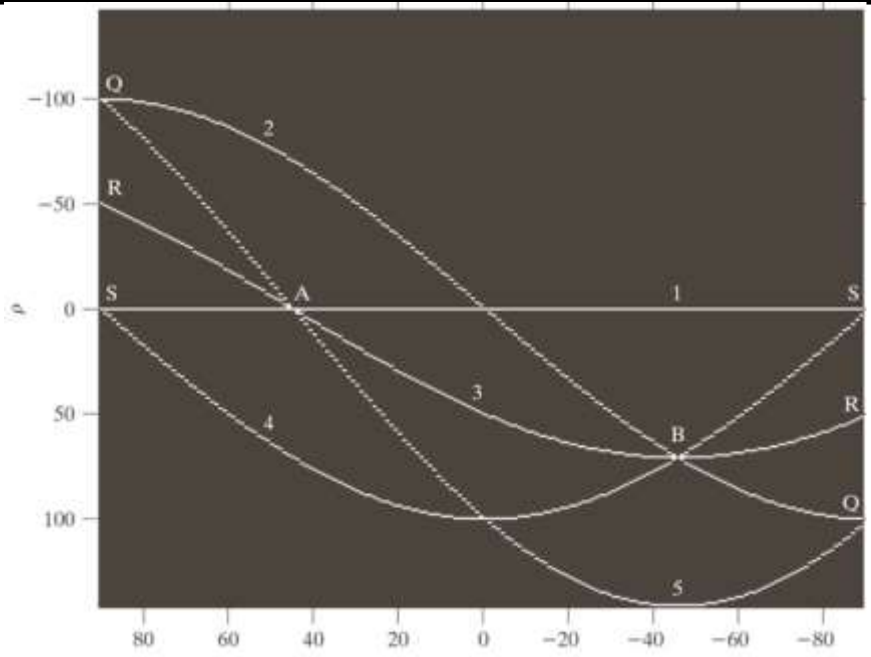
$$x\cos\theta + y\sin\theta = \rho$$



Accumulator array

Hough transform (cont.)





- A: intersection of curves corresponding to points 1, 3, 5.
- B: intersection of curves corresponding to points 2, 3, 4.
- Q, R and S show the reflective adjacency at the edges of the parameter space. They do not correspond to points.

Practical details

- Try to get rid of irrelevant features
  - Take only edge points with significant gradient magnitude.
- Choose a good grid / discretization
  - Too coarse: large votes obtained when too many different lines correspond to a single bucket.
  - Too fine: miss lines because some points that are not exactly collinear cast votes for different buckets.
- Increment also neighboring bins (smoothing in accumulator array).