Image Segmentation

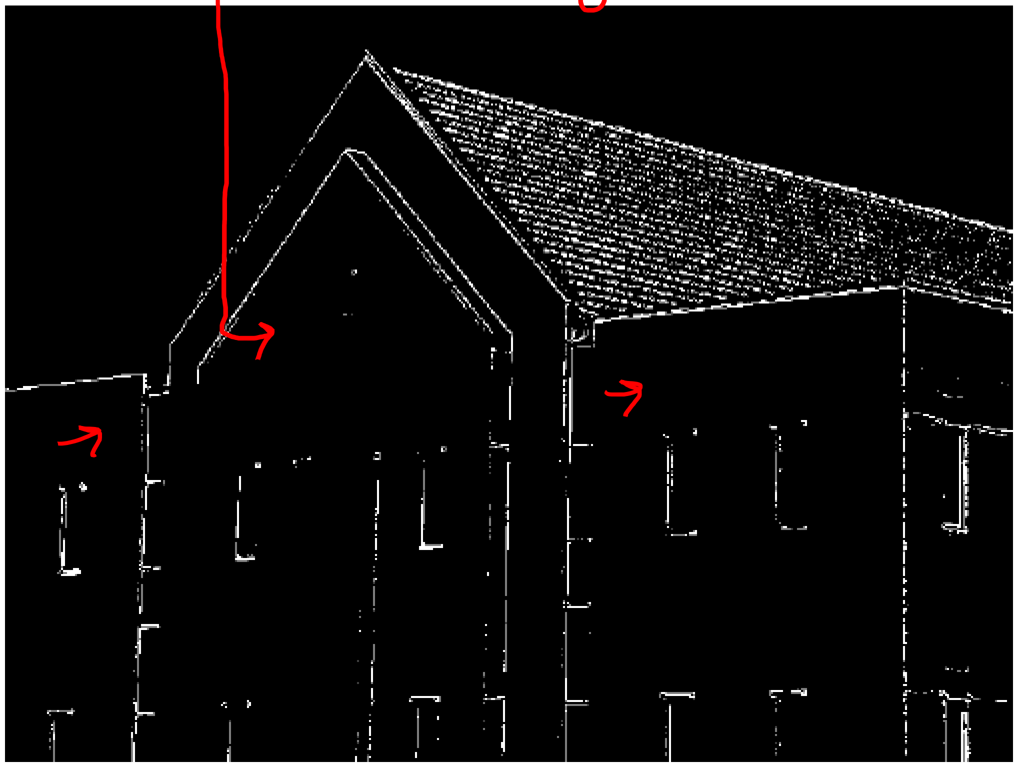
Lecture 2

Combining the Gradient with Threshold

- edge detection can be made more selective by using thresholding with gradient.

brick edges are removed

a) $|g_x| + |g_y|$ of unblurred original image.



b) Threshold image with threshold selected as 33% of highest value in fig a)

Reduced no. of broken edges than previous

a) $|g_x|+|g_y|$ of blurred original image

b) Threshold image with threshold selected at 33% of maximum value in fig a)

More Advanced Techniques for Edge Detection
The Marr – Hildreth Edge Detector

- Scale Dependency
- Marr and Hildreth argued that
  1. intensity changes are not independent of image scale and so their detection requires the use of operators of different sizes; and
  2. that a sudden intensity change will give rise to a peak or trough in the first derivative or, equivalently, to a zero crossing in the second derivative
- These ideas suggest that an operator used for edge detection should have two salient features.
  1. it should be a differential operator capable of computing a digital approximation of the first or second derivative at every point in the image.
  2. it should be capable of being “tuned” to act at any desired scale, so that large operators can be used to detect blurry edges and small operators to detect sharply focused fine detail.

- Large operator used to detect blurry edges and small operators to detect sharply focused fine details

Marr and Hildreth Operator

- Consider the function

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

- The most satisfactory operator satisfying the previous two conditions is $\nabla^2 G$
- ∇^2 is the Laplacian Operator and *G* is the Gaussian Function.
- *LoG :* Laplacian of Gaussian

Marr-Hildreth Edge Detector

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

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

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

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

Marr-Hildreth Edge Detector

Zero crossing
at $x^2 + y^2 = 2\sigma^2$
circle of radius $= \sqrt{2}\sigma$

- The Final expression is given as:

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

$\rightarrow$ -ve value

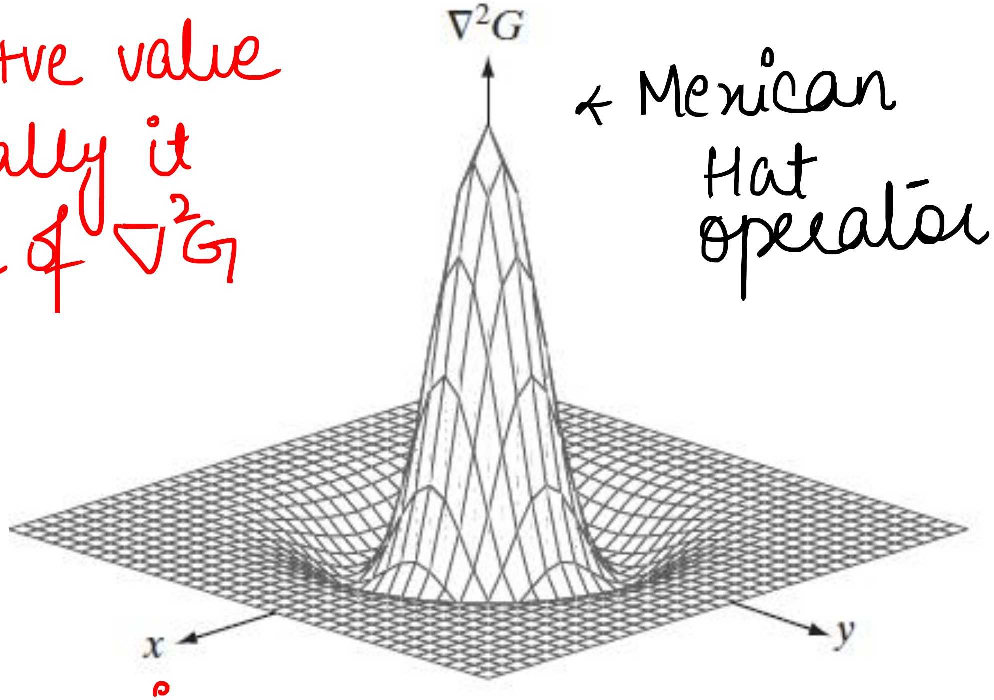
- This expression is called the *Laplacian of a Gaussian* (LoG).

Laplacian operator $\rightarrow$ $\begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ or $\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

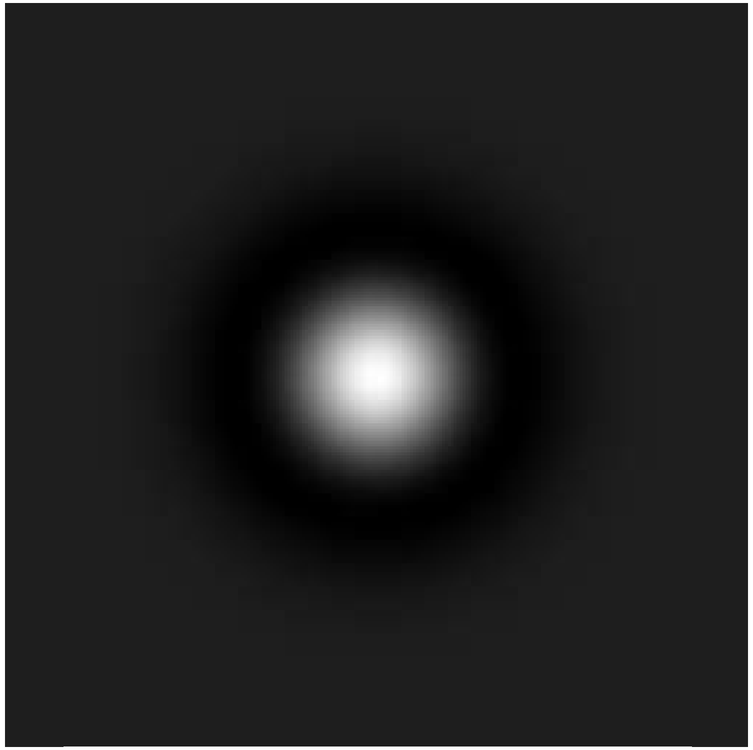
-ve peak

Convention to take +ve value

→ Actually it is -ve of $\nabla^2 G$

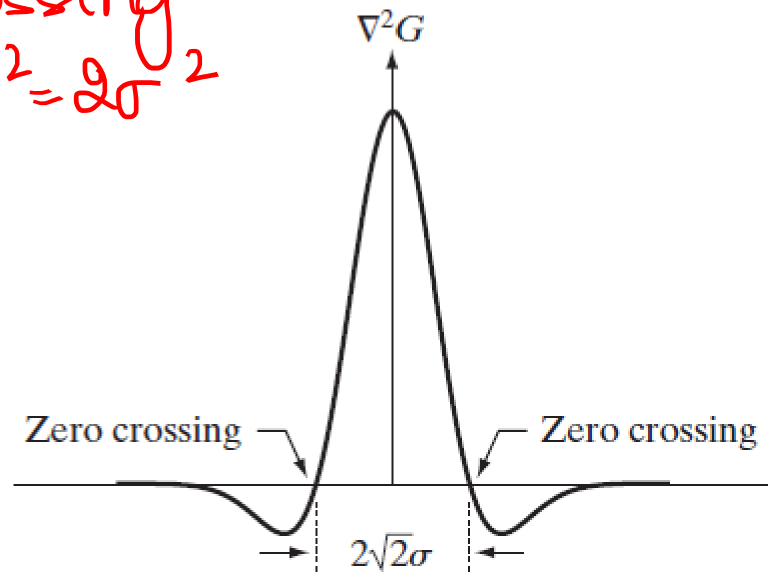


← Mexican Hat operator



zero crossing at $x^2+y^2=2\sigma^2$

circle of radius $\sqrt{2}\,\sigma$



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) 5×5 mask approximation to the shape in (a). The negative of this mask would be used in practice.

Why this operator?

- There are two fundamental ideas behind the selection of the operator $\nabla^2 G$

1. the Gaussian part of the operator blurs the image, thus reducing the intensity of structures (including noise) at scales much smaller than unlike averaging. Gaussian function is smooth in both the spatial and frequency domains and is thus less likely to introduce artifacts (e.g., ringing) not present in the original image.
2. Although first derivatives can be used for detecting abrupt changes in intensity, they are directional operators. The Laplacian, on the other hand, has the important advantage of being isotropic (invariant to rotation), which not only corresponds to characteristics of the human visual system (Marr [1982]) but also responds equally to changes in intensity in any mask direction, thus avoiding having to use multiple masks to calculate the strongest response at any point in the image.

Marr- Hildreth Edge Detection Algorithm

- The Marr-Hildreth algorithm consists of convolving the LoG filter with an input image, $f(x, y)$

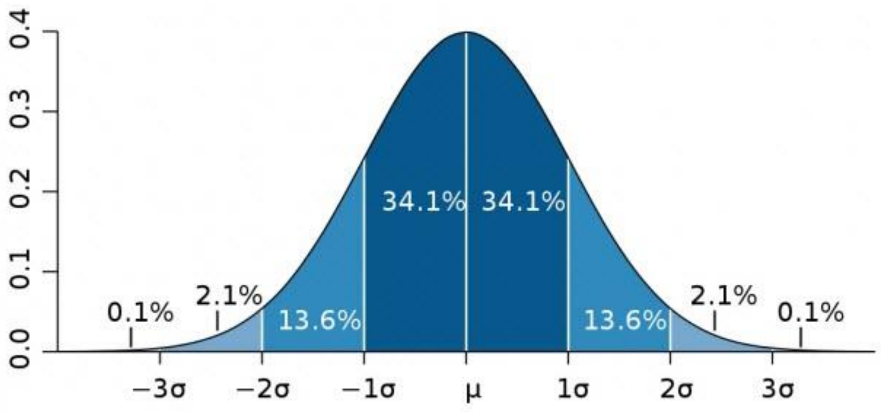
$$g(x, y) = [\nabla^2 G(x, y)] \star f(x, y)$$

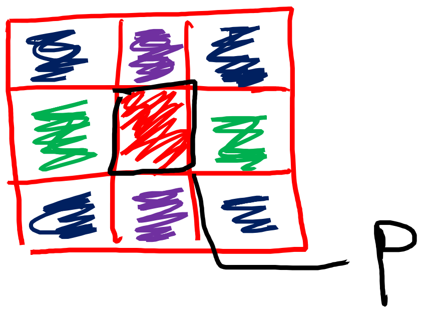
- and then finding the zero crossings of $g(x, y)$ to determine the locations of edges in $f(x, y)$. Because these are linear processes,

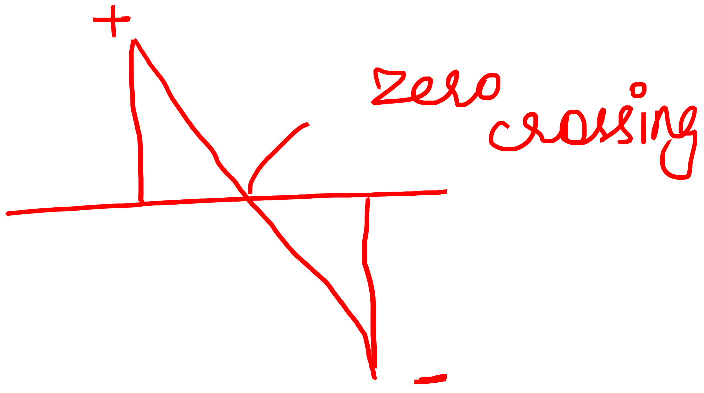
$$g(x, y) = \nabla^2 [G(x, y) \star f(x, y)]$$

- The Marr-Hildreth edge-detection algorithm may be summarized as follows:
  1. Filter the input image with an Gaussian lowpass filter.
  2. Compute the Laplacian of the image resulting from Step 1 using, for example, the 3 x 3 mask
  3. Find the zero crossings of the image from Step 2.

- What should be the size of the Gaussian filter mask n x n ?
  - Most of the volume of Gaussian surface lies between $\pm 3\sigma$, so n is a smallest odd integer greater than or equal to 6σ Mask smaller than this will tend to truncate the LOG function.
- How do we find the zero crossings?
  - Consider 3x3 neighborhood across *p*, Zero crossing at *p* implies that signs of at least two of its opposing neighbour pixels must differ.
  - Four cases to test-(up-down),(left-right),two diagonals.
  - Absolute value of their numerical difference must also exceed the threshold.







Example

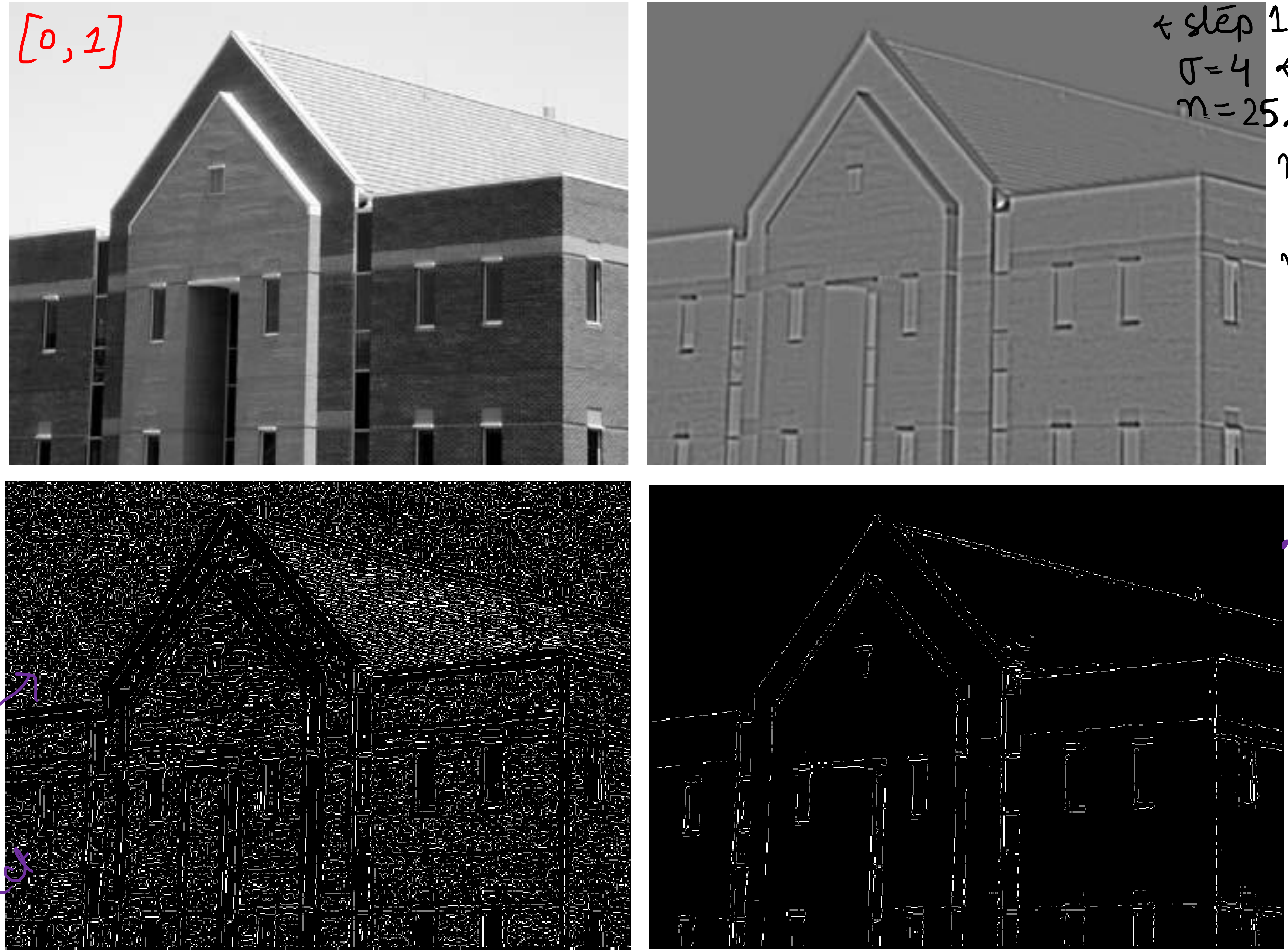


FIGURE 10.22 (a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1]. (b) Results of Steps 1 and 2 of the Marr-Hildreth algorithm using $\sigma = 4$ and $n = 25$. (c) Zero crossings of (b) using a threshold of 0 (note the closed-loop edges). (d) Zero crossings found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

← step 1 & 2
$\sigma = 4$ ← G
$n = 25$ ← ∇^2
$n => 6\sigma$
n is odd

spaghetti effect

← threshold 4% of max value of LoG

- To consider scale dependency-Filter the image for various values of sigma and keep the zero crossings that are common to all responses-complex procedure.
- Marr and Hildreth [1980] showed that LoG may be approximated by a difference of Gaussians (DOG).

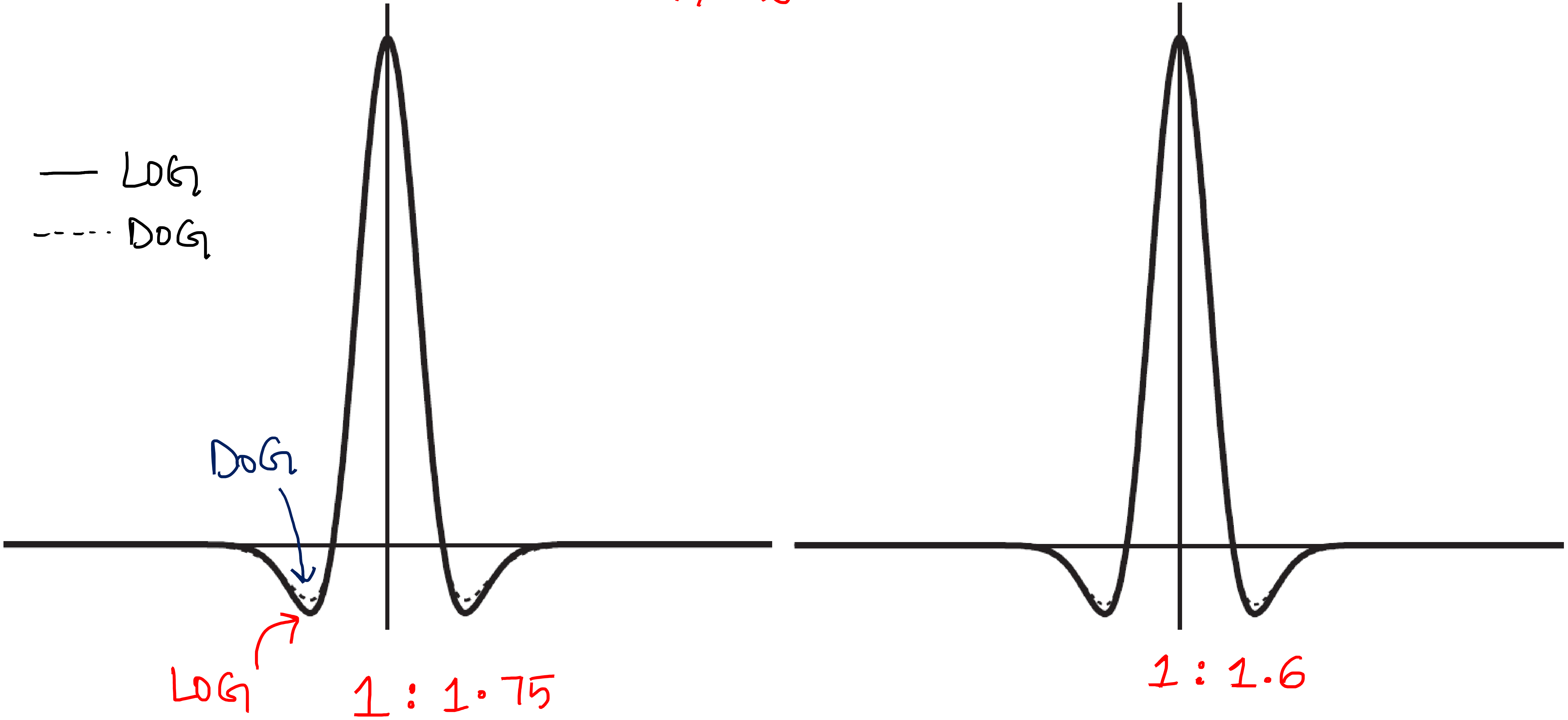
$$\mathrm{DoG}(x, y) = \frac{1}{2\pi\sigma_1^2} e^{-\frac{x^2+y^2}{2\sigma_1^2}} - \frac{1}{2\pi\sigma_2^2} e^{-\frac{x^2+y^2}{2\sigma_2^2}}$$

- Meaningful comparison between LoG and DoG may be obtained after selecting the value of σ for LoG so that LoG has the same zero crossings as DoG:

$$\sigma^2 = \frac{\sigma_1^2\sigma_2^2}{\sigma_1^2 - \sigma_2^2} \ln\left[\frac{\sigma_1^2}{\sigma_2^2}\right]$$

- By selecting this sigma LoG and DoG will have same zero crossings but to be compatible in amplitude scaling(normalization) is needed.

$\sigma_1/\sigma_2 \leftarrow$ DoG std ratios



The Canny Edge Detector

- Objectives

1. *Low error rate.* All edges should be found, and there should be no spurious responses. That is, the edges detected must be as close as possible to the true edges.

2. *Edge points should be well localized.* The edges located must be as close as possible to the true edges. That is, the distance between a point marked as an edge by the detector and the center of the true edge should be minimum.

3. *Single edge point response.* The detector should return only one point for each true edge point. That is, the number of local maxima around the true edge should be minimum. This means that the detector should not identify multiple edge pixels where only a single edge point exists.

Canny Edge Detection Algorithm

1. Smooth the input image with a Gaussian filter.

2. Compute the gradient magnitude and angle images.

3. Apply non-maxima suppression to the gradient magnitude image.

4. Use double thresholding and connectivity analysis to detect and link edges.

Step 1

- Let $f(x, y)$ denote the input image and denote the Gaussian function:

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

- We form a smoothed image, $f_s(x, y)$, by convolving G and f:

$$f_s(x, y) = G(x, y) \star f(x, y)$$

Step 2

- This operation is followed by computing the gradient magnitude and direction (angle),

$$M(x, y) = \sqrt{g_x^2 + g_y^2} \qquad \alpha(x, y) = \tan^{-1}\left[\frac{g_y}{g_x}\right]$$

STEP 1: Filter the image using Gaussian filter

Original grayscale image

Blurred image using Gaussian filter

Courtesy: https://justin-liang.com/tutorials/canny/

STEP 2. Find Gradient of image

Sobel operator

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

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

G_x

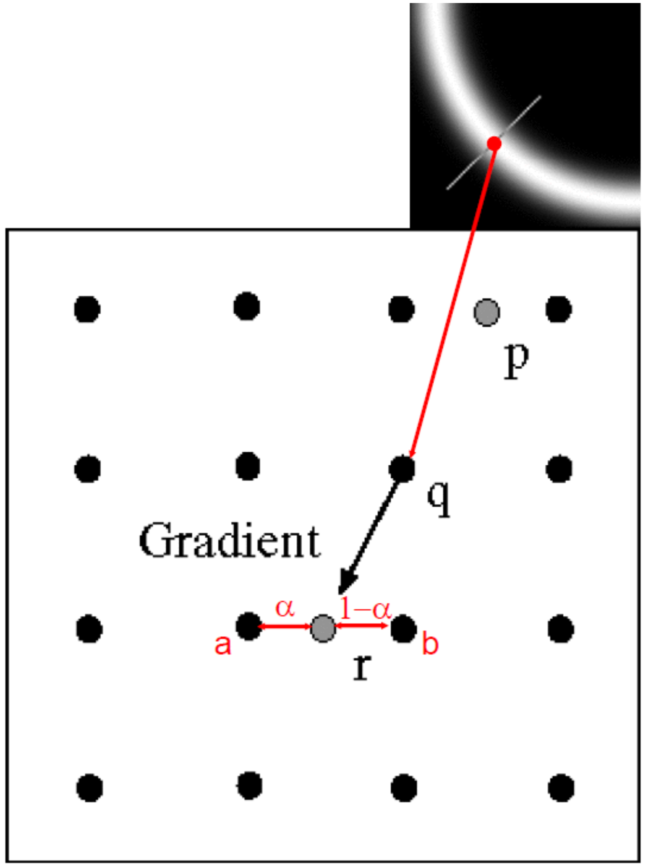
G_y

Courtesy: https://justin-liang.com/tutorials/canny/

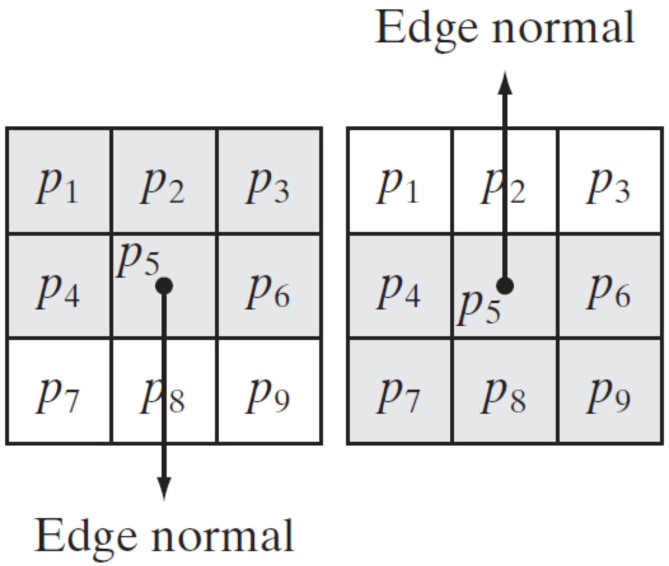
STEP 3: Non-maximum Suppression

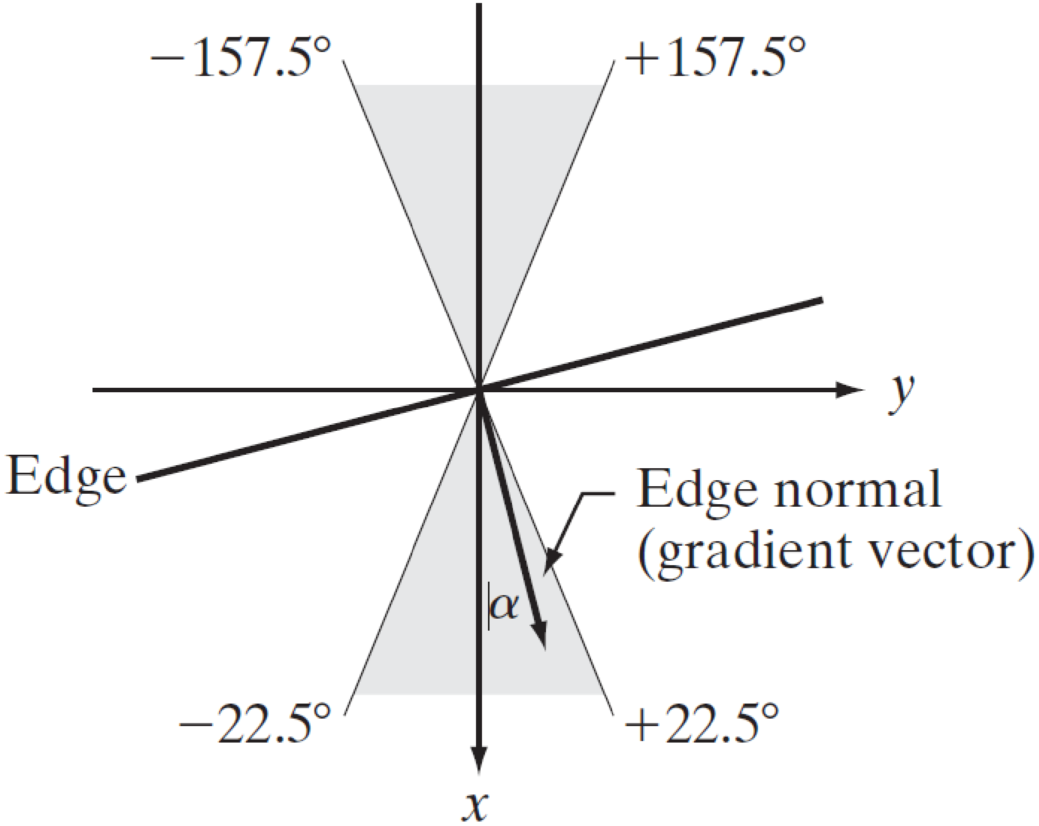
$M(x,y) = \sqrt{G_x^2 + G_y^2}$

The image magnitude produced results in thick edges. Ideally, the final image should have thin edges. Thus, we must perform non maximum suppression to thin out the edges.

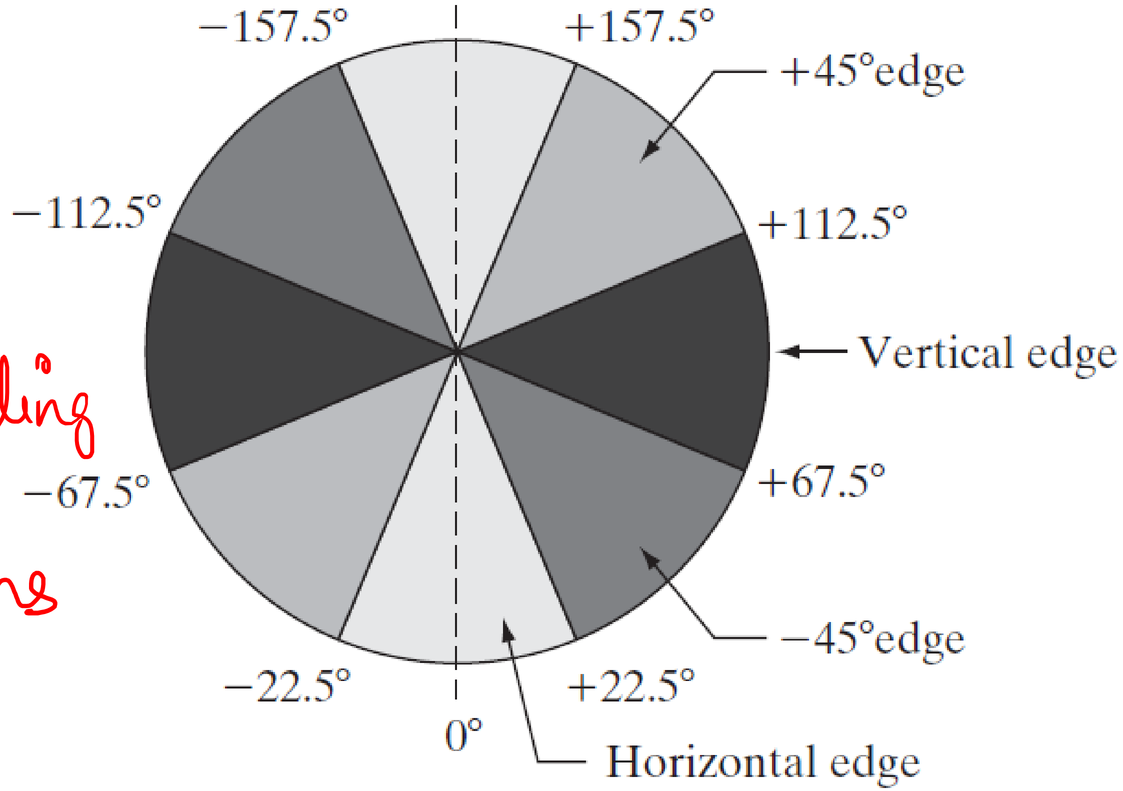


Non maximum suppression works by finding the pixel with the maximum value in an edge. In the above image, it occurs when pixel q has an intensity that is larger than both p and r where pixels p and r are the pixels in the gradient direction of q. If this condition is true, then we keep the pixel, otherwise we set the pixel to zero (make it a black pixel).





Angle range corresponding to four directions



Because we have to quantize all possible edge directions into four, we have to define a range of directions over which we consider an edge to be horizontal

Non- Maxima Suppression

Let d_1, d_2, d_3, and d_4 denote the four basic edge directions just discussed for a 3×3 region: horizontal, $-45°$, vertical, and $+45°$, respectively. We can formulate the following nonmaxima suppression scheme for a 3×3 region centered at *every* point (x, y) in $\alpha(x, y)$:

1. Find the direction d_k that is closest to $\alpha(x, y)$.
2. If the value of $M(x, y)$ is less than at least one of its two neighbors along d_k, let $g_N(x, y) = 0$ (suppression); otherwise, let $g_N(x, y) = M(x, y)$

$\alpha(x,y)$ is angle matrix of gradient

$M(x,y) \leftarrow$ magnitude matrix of Gradient

$g_N(x,y) \leftarrow$ non-maxima suppressed matrix

STEP 3: Non-maximum Supression Result

→ Non-maxima suppressed image

$M(x,y) = \sqrt{G_x^2 + G_y^2}$

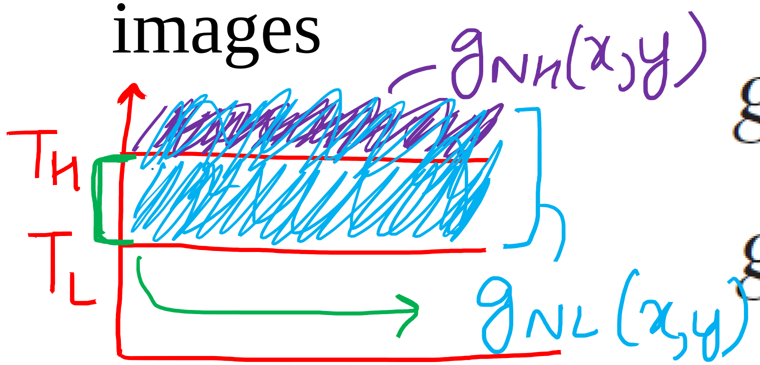
$g_N(x, y)$

Courtesy: https://justin-liang.com/tutorials/canny/

Step 4 Thresholding for Canny Edge detection (Hystresis Thresholding)

- Single threshold-
  - Low value- false positive
  - High value- false negative
- Canny’s algorithm implements hysteresis thresholding.- uses two thresholds.
- Canny suggested that ratio of the high to low thresholds should be 2:1 or 3:1

- We can visualize the thresholding operation as creating two additional images



$$g_{NH}(x, y) = g_N(x, y) \geq T_H$$

$$g_{NL}(x, y) = g_N(x, y) \geq T_L$$

- where, initially, both $g_{NH}(x, y)$ and $g_{NL}(x, y)$ are set to 0. After thresholding, $g_{NH}(x, y)$ will have fewer nonzero pixels than $g_{NL}(x, y)$ in general, but all the nonzero pixels in $g_{NH}(x, y)$ will be contained in $g_{NL}(x, y)$ because the latter image is formed with a lower threshold. We eliminate from $g_{NL}(x, y)$ all the nonzero pixels from $g_{NH}(x, y)$ by letting

$$g_{NL}(x, y) = g_{NL}(x, y) - g_{NH}(x, y)$$

- The nonzero pixels in $g_{NH}(x, y)$ and $g_{NL}(x, y)$ may be viewed as being “strong” and “weak” edge pixels, respectively.

STEP 4 : Double Thresholding

a)

Non-maximum suppressed

Double Threshold image

Courtesy: https://justin-liang.com/tutorials/canny/

STEP 4b:

Edge Linking

Now that we have determined what the strong edges and weak edges are, we need to determine which weak edges are actual edges. To do this, we perform an edge tracking algorithm. Weak edges that are connected to strong edges will be actual/real edges. Weak edges that are not connected to strong edges will be removed.

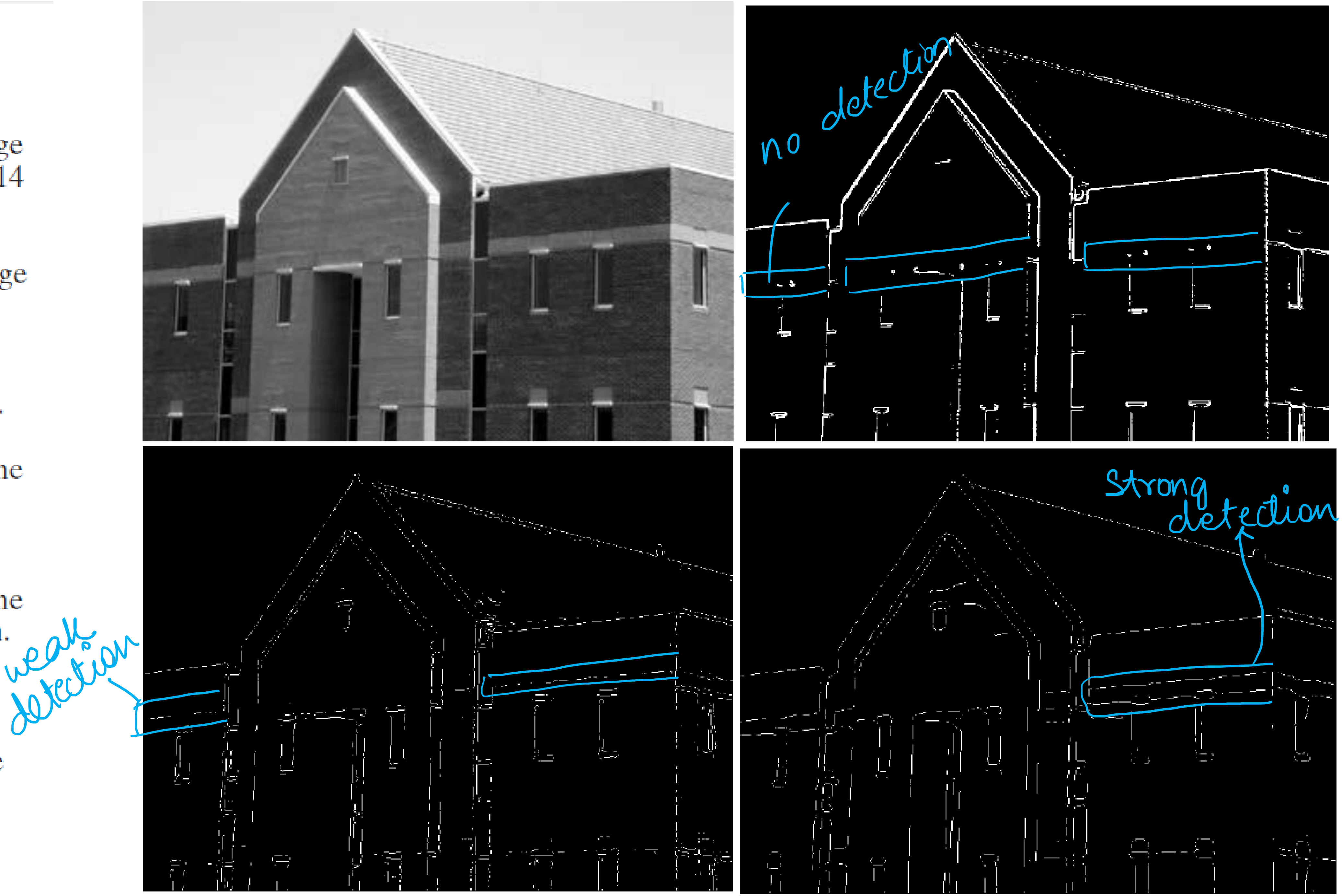
Courtesy: https://justin-liang.com/tutorials/canny/

STEP 4: EDGE LINKING

- After the thresholding operations, all strong pixels in $g_{NH}(x, y)$ are assumed to be valid edge pixels and are so marked immediately.
- Depending on the value of T_H the edges in $g_{NH}(x, y)$ typically have gaps.
- Longer edges are formed using the following procedure:

1. Locate the next unvisited edge pixel, p, in $g_{NH}(x, \mathrm{y})$.
2. Mark as valid edge pixels all the weak pixels in $g_{NL}(x, y)$ that are connected to p using, say, 8-connectivity.
3. If all nonzero pixels in $g_{NH}(x, \mathrm{y})$ have been visited go to Step 4. Else, return to Step 1.
4. Set to zero all pixels in $g_{NL}(x, y)$ that were not marked as valid edge pixels.

- At the end of this procedure, the final image output by the Canny algorithm is formed by appending to $g_{NH}(x, \mathrm{y})$ all the nonzero pixels from $g_{NL}(x, y)$.



a b
c d

FIGURE 10.25
(a) Original image of size 834 × 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.

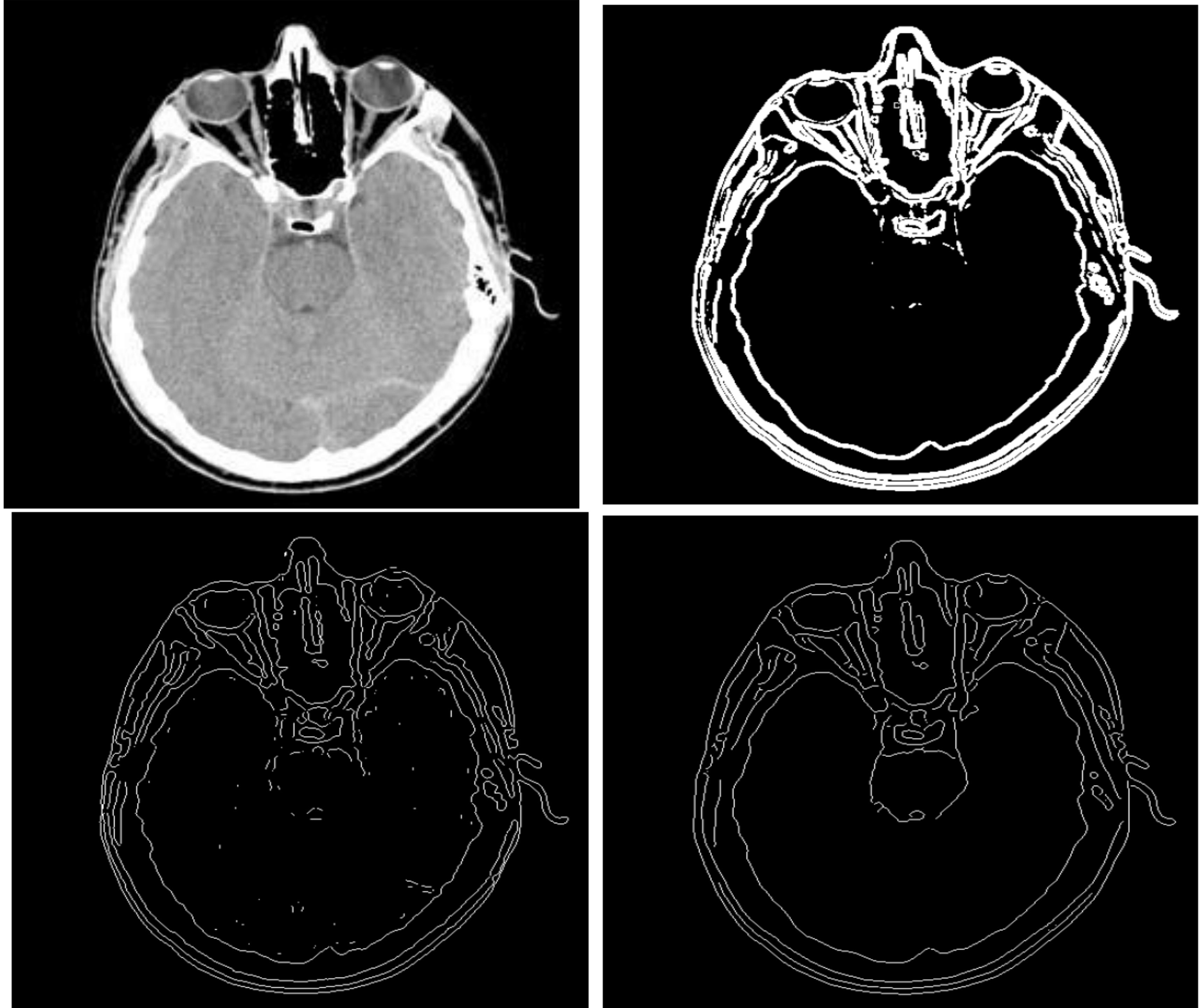


FIGURE 10.26
(a) Original head CT image of size 512×512 pixels, with intensity values scaled to the range [0, 1].
(b) Thresholded gradient of smoothed image.
(c) Image obtained using the Marr-Hildreth algorithm.
(d) Image obtained using the Canny algorithm.
(Original image courtesy of Dr. David R. Pickens, Vanderbilt University.)