CTT310: Digital Image Processing

Edge Detection

Dr. Nguyen Ngoc Thao
Department of Computer Science, FIT, HCMUS

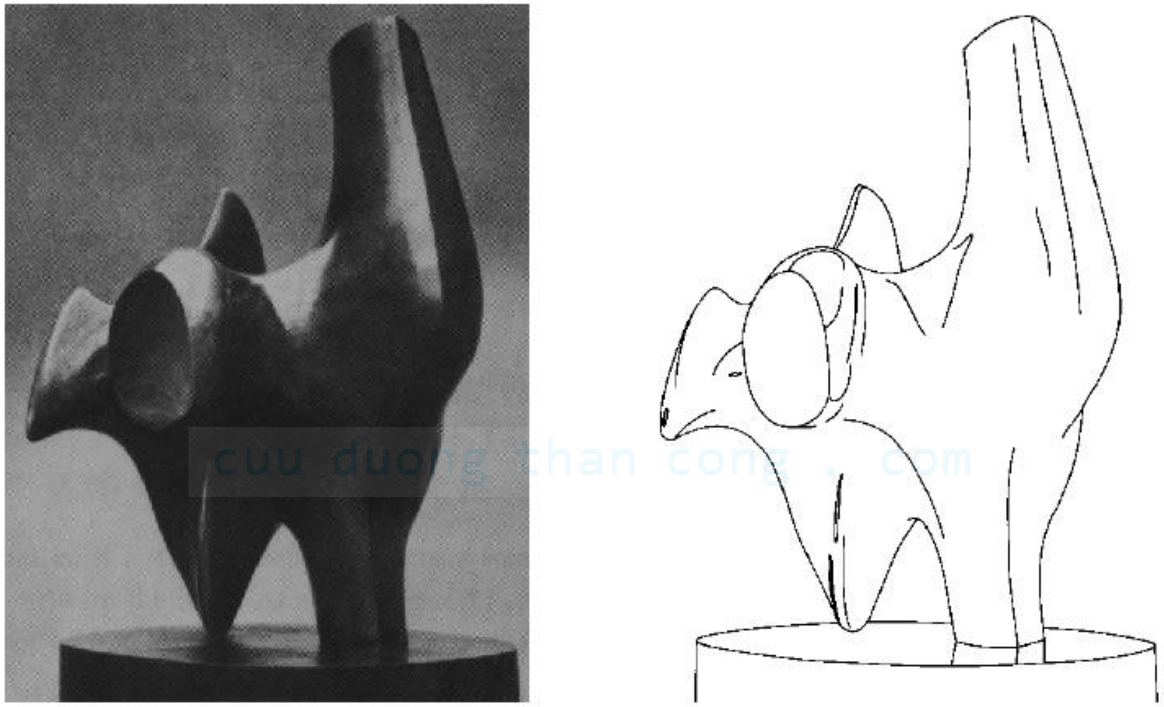
Outline

- What is edge detection?
- Common approaches for edge detection
  - Basic edge detectors: Robert, Prewitt, Sobel, Laplacian, and Kirsh edge operators
  - Laplacian of Gaussian (LoG)
  - Canny Edge Operator
- Applications of edge detection

Section 6.1

WHAT IS EDGE DETECTION?

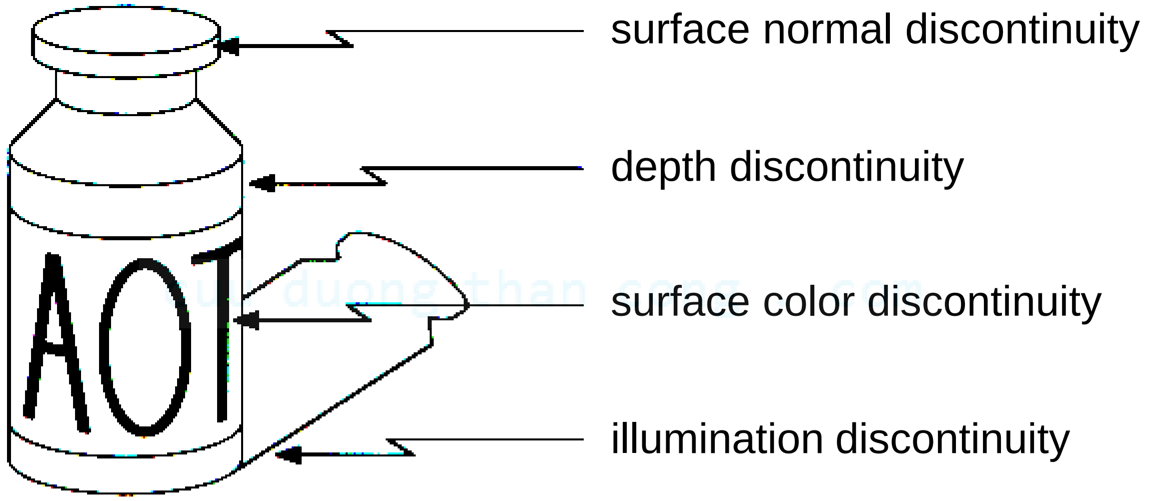
Edge detection



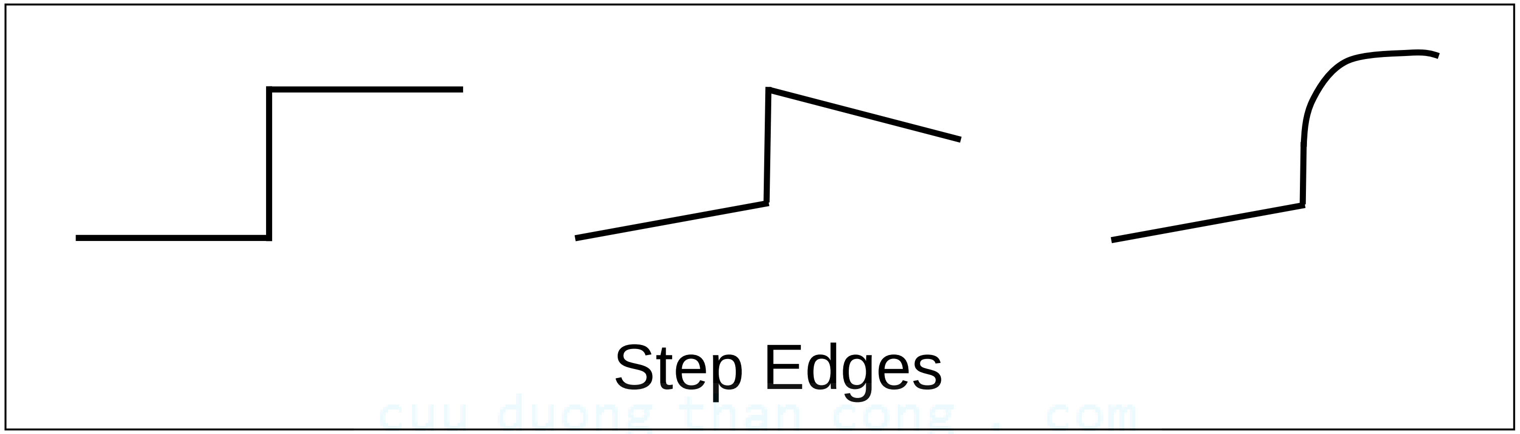
- Convert a 2D image into a set of curves
  - Extracts salient features of the scene
  - More compact than pixels
- How can you tell that a pixel is on an edge?

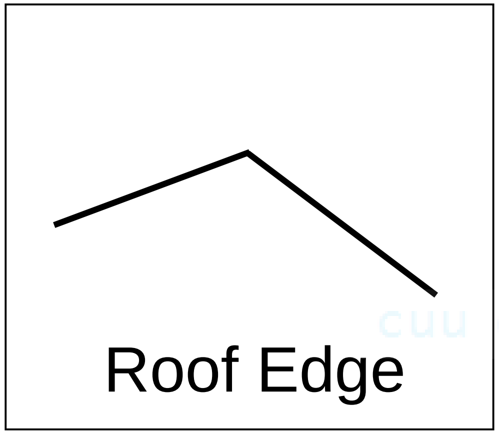
Origin of edges

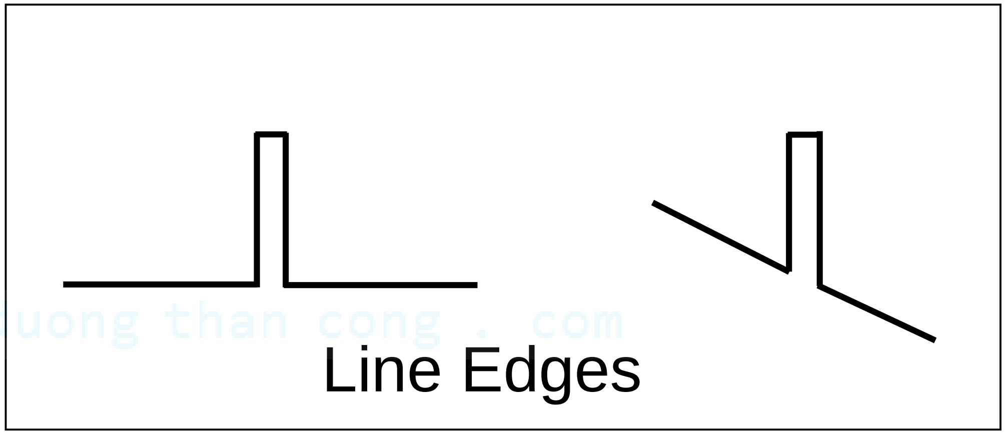
- Edges are caused by a variety of factors


surface normal discontinuity
depth discontinuity
surface color discontinuity
illumination discontinuity
AOT

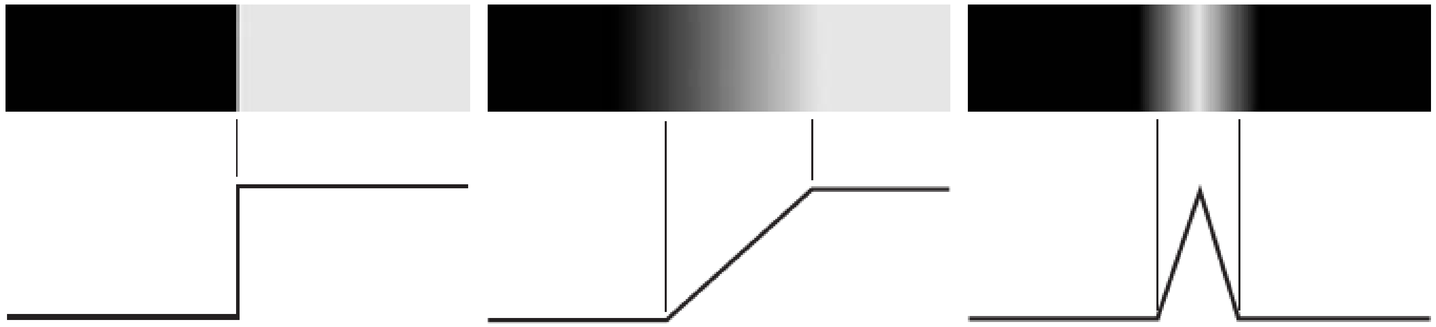
Types of edge

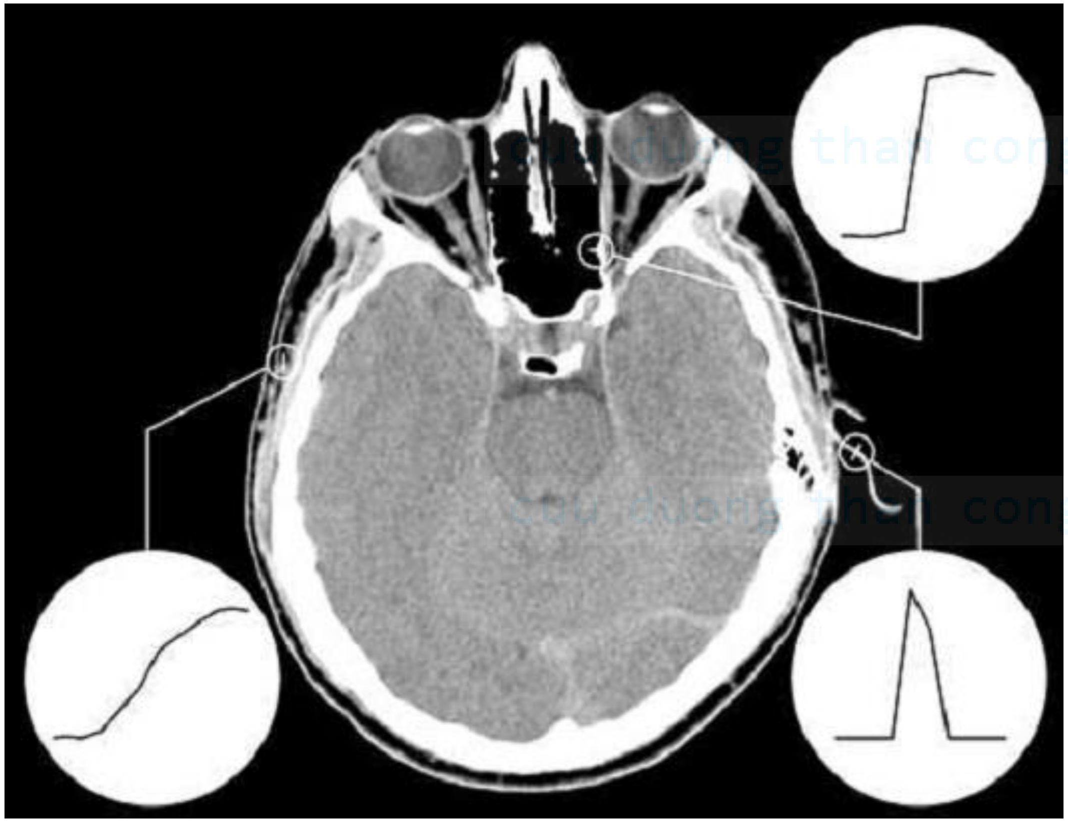






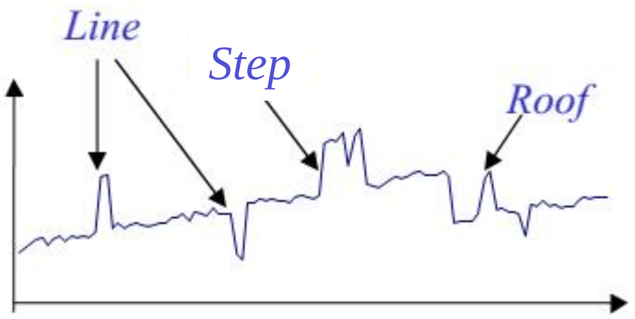
a b c From left to right, models (ideal representations) of a step, a ramp, and a roof edge, and their corresponding intensity profiles.

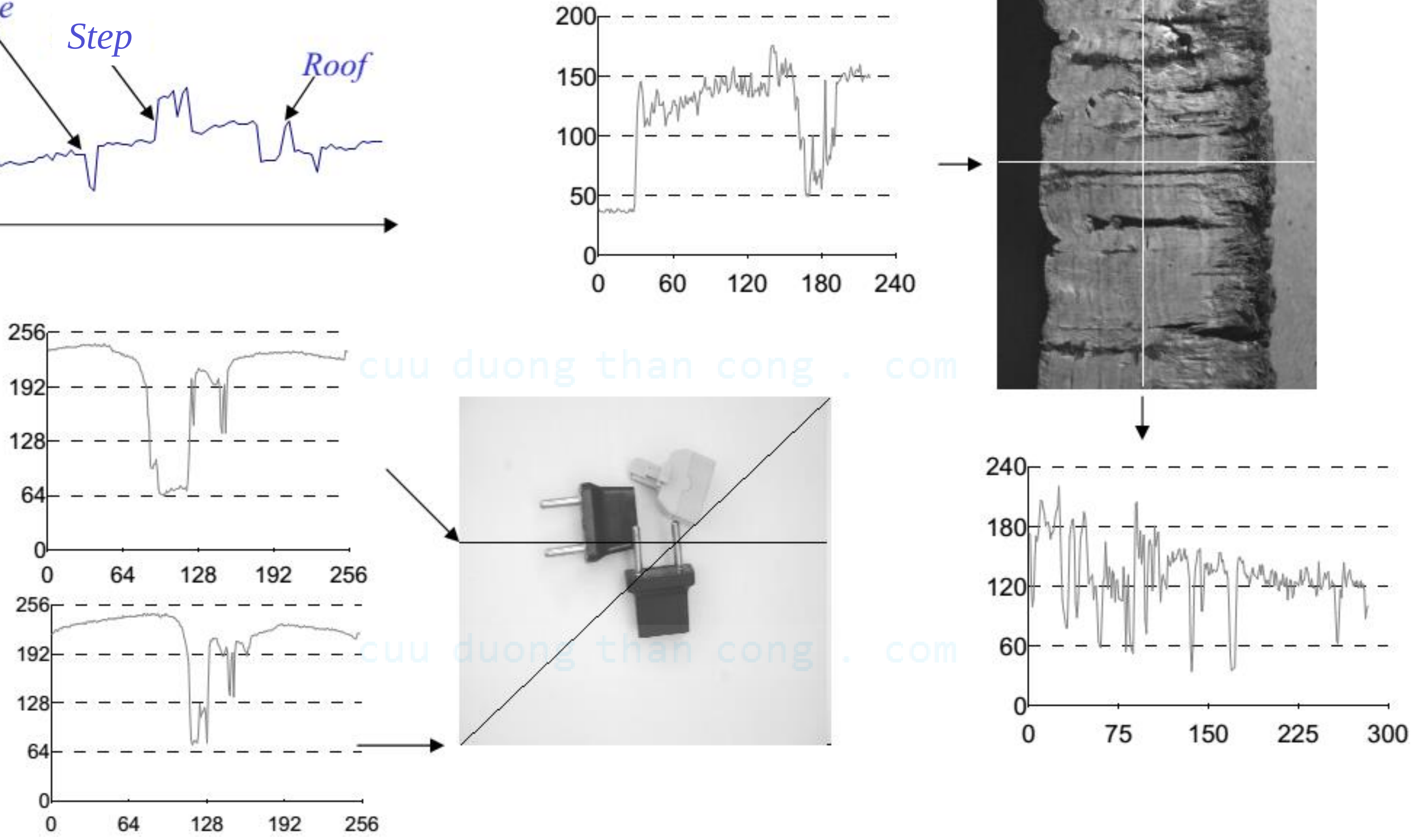




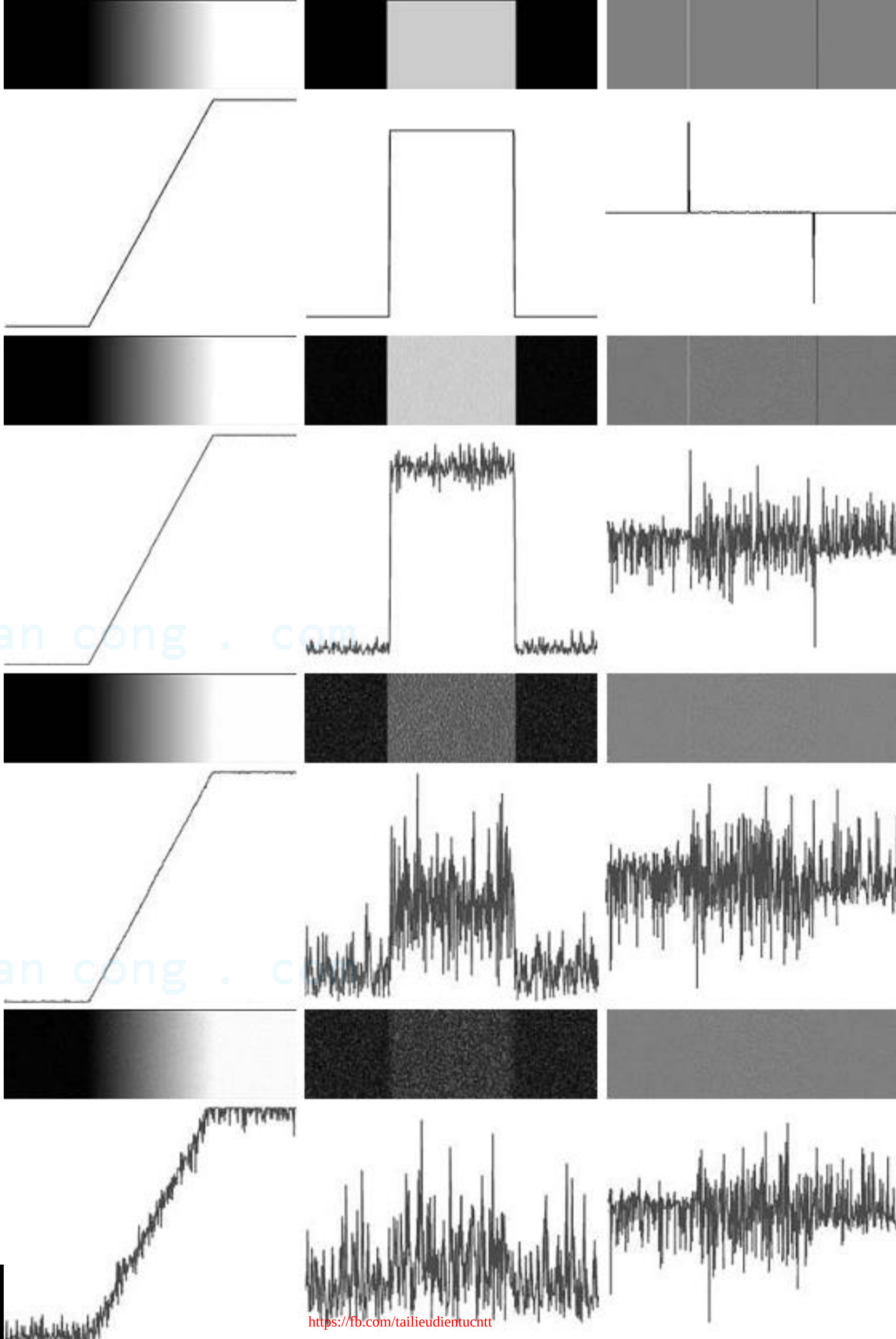
A image showing (zoomed) actual ramp (bottom, left), step (top, right), and roof edge profiles. The profiles are from dark to light, in the areas indicated by the short line segments shown in the small circles. The ramp and "step" profiles span 9 pixels and 2 pixels, respectively. The base of the roof edge is 3 pixels.

Real edges

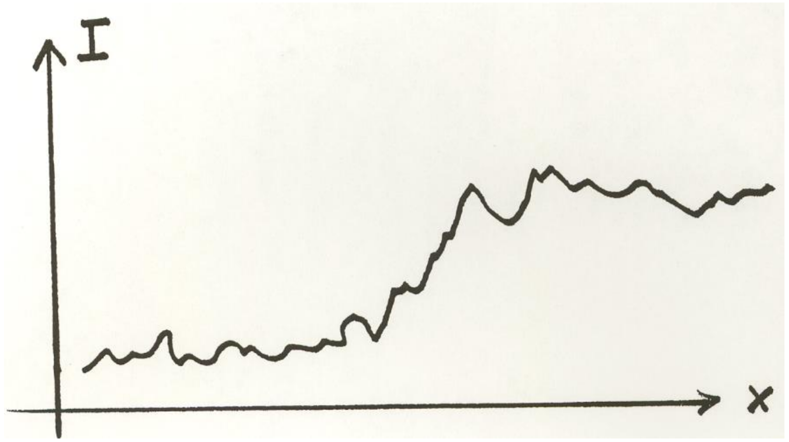

Line
Step
Roof


200
150
100
50
0
0
60
120
180
240
256
192
128
64
0
0
64
128
192
256
256
192
128
64
0
0
64
128
192
256
240
180
120
60
0
0
75
150
225
300

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



Real edges



Noisy and Discrete!

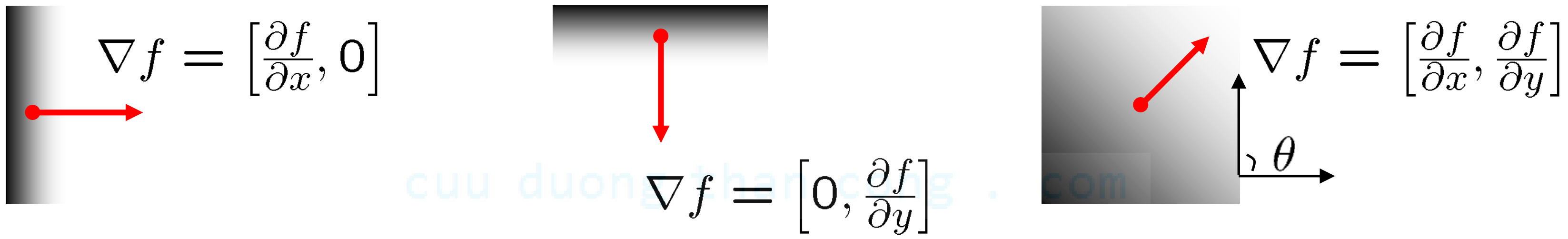
- We want an Edge Operator that produces
  - Edge Magnitude
  - Edge Orientation
  - High Detection Rate and Good Localization

Section 6.2

COMMON APPROACHES FOR EDGE DETECTION

Gradient

- Gradient equation: $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$
- Represents direction of most rapid change in intensity



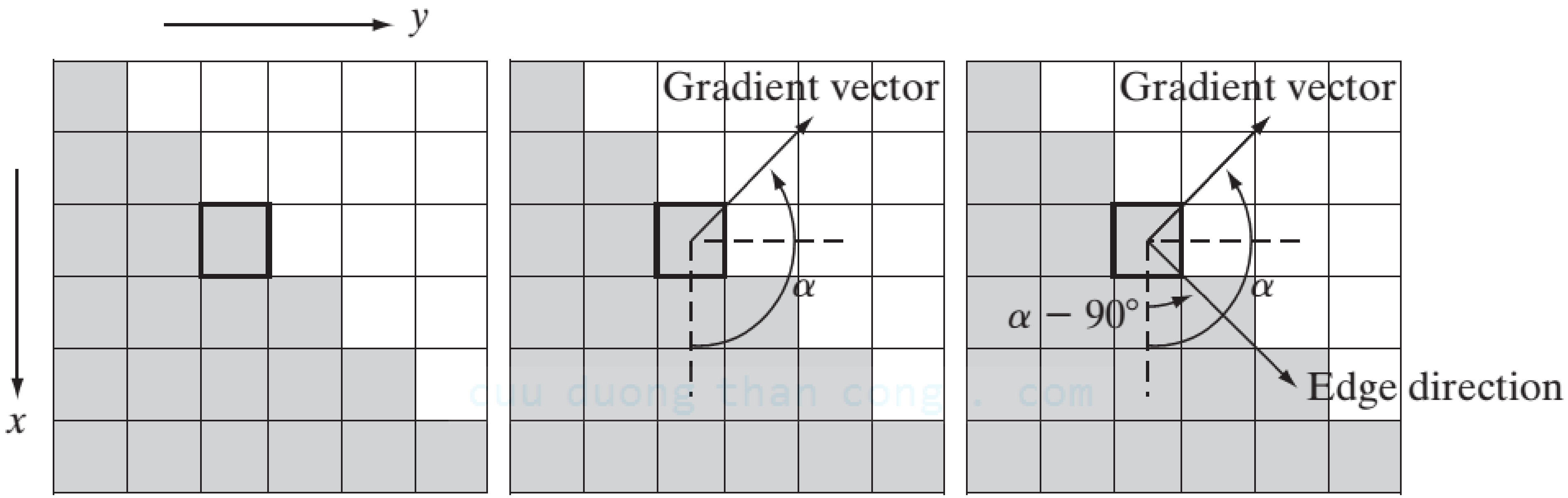
$$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$$

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

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

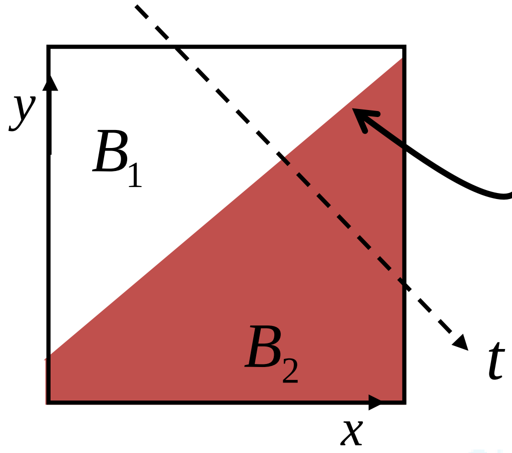
- Gradient direction: $\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$
- The edge strength is given by the gradient magnitude

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



Using the gradient to determine edge strength and direction at a point. Note that the edge is perpendicular to the direction of the gradient vector at the point where the gradient is computed. Each square in the figure represents one pixel.

Theory of edge detection



Ideal edge

$$L(x, y) = x \sin\theta - y\cos\theta + \rho = 0$$

$$B_1 : L(x, y) < 0 \qquad B_2 : L(x, y) > 0$$

Unit step function:

$$u(t) = \begin{cases} 1 & \text{for } t > 0 \\ 1/2 & \text{for } t = 0 \\ 0 & \text{for } t < 0 \end{cases} \qquad u(t) = \int_{-\infty}^{t} \delta(s) ds$$

Image intensity (brightness):

$$I(x, y) = B_1 + (B_2 - B_1) u(x \sin\theta - y\cos\theta + \rho)$$

Theory of edge detection

- Image intensity (brightness):

$$I(x,y) = B_1 + (B_2 - B_1)u(x\sin\theta - y\cos\theta + \rho)$$

- Partial derivatives (gradients):

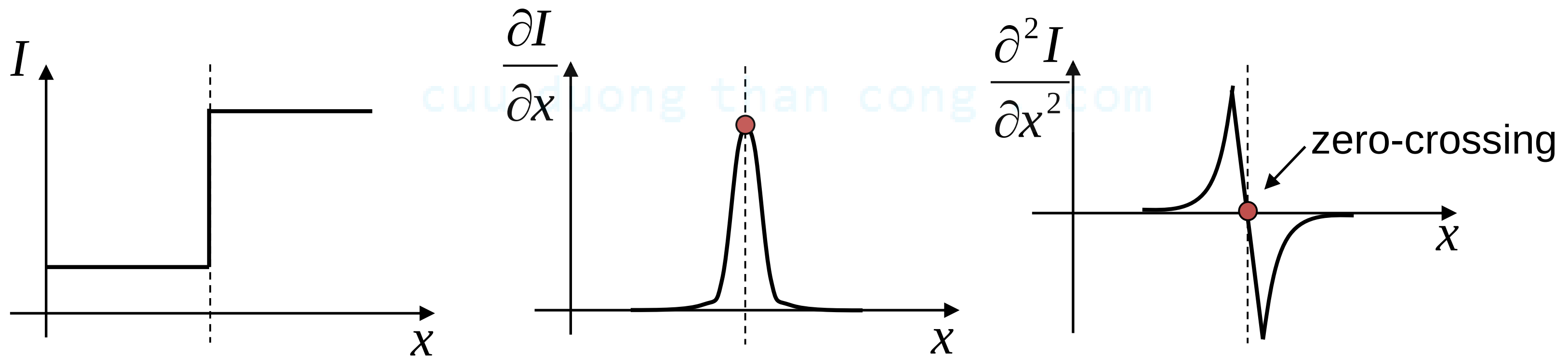
$$\frac{\partial I}{\partial x} = +\sin\theta(B_2 - B_1)\delta(x\sin\theta - y\cos\theta + \rho)$$

Directional!

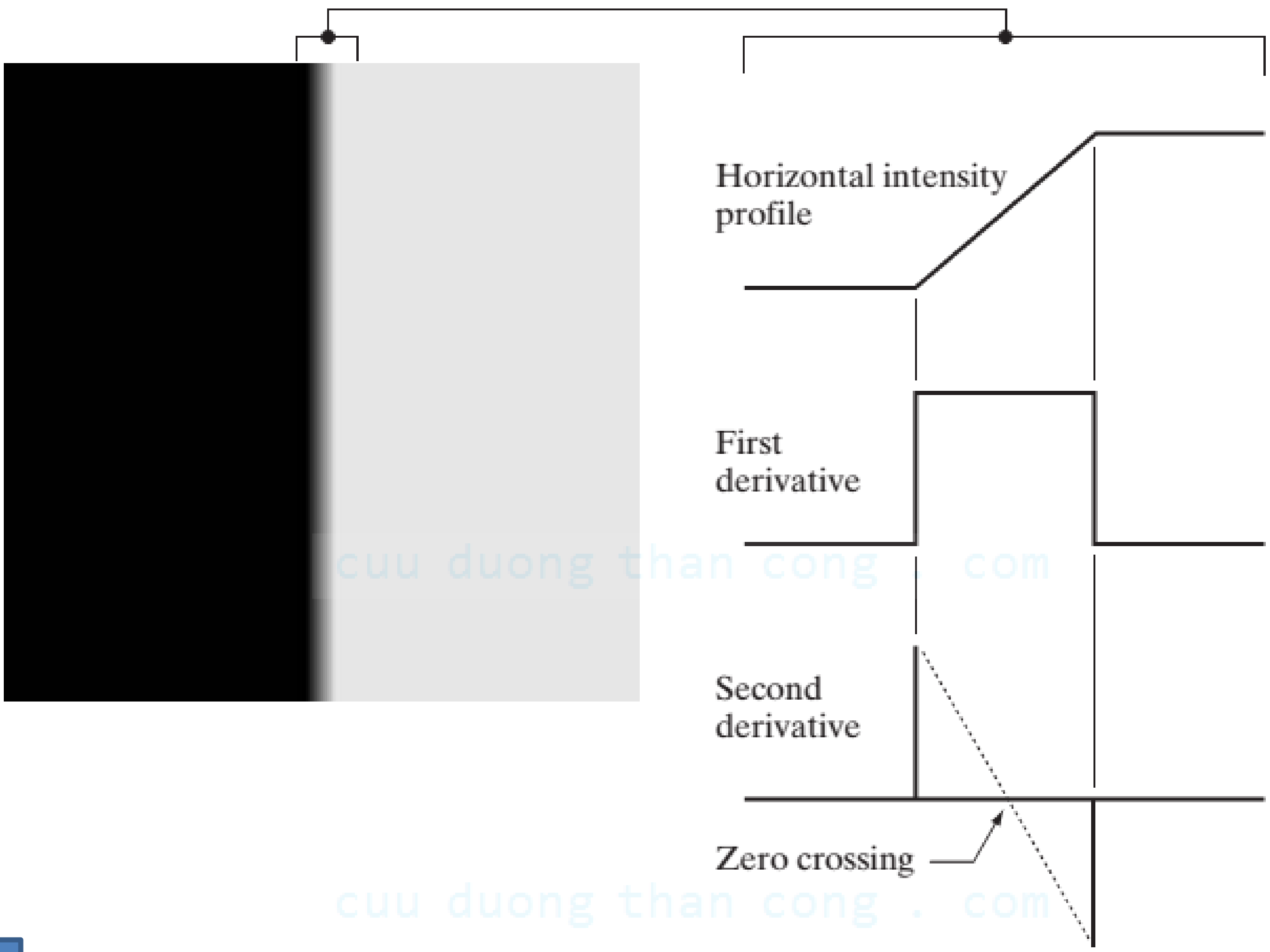
$$\frac{\partial I}{\partial y} = -\cos\theta(B_2 - B_1)\delta(x\sin\theta - y\cos\theta + \rho)$$

Theory of edge detection

- Edge Orientation (normal of the edge): $\arctan\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$
- Edge Magnitude: $\sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2} = [(B_2 - B_1)\delta(x\sin\theta - y\cos\theta + \rho)]^2$
  - Rotationally symmetric, non-linear operator
- Laplacian: $\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} = (B_2 - B_1)\delta(x\sin\theta - y\cos\theta + \rho)$
  - Rotationally symmetric, linear operator



I
x
$\frac{\partial I}{\partial x}$
x
$\frac{\partial^2 I}{\partial x^2}$
zero-crossing
x



a b

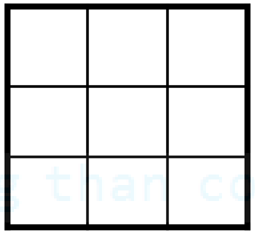
(a) Two regions of constant intensity separated by an ideal vertical ramp edge.
(b) Detail near the edge, showing a horizontal intensity profile, together with its first and second derivatives

Discrete edge operators

- How can we differentiate a discrete image?
  - Option 1: reconstruct a continuous image, then take gradient
  - Option 2: take discrete derivative (finite difference)

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

How would you implement this as a cross-correlation?



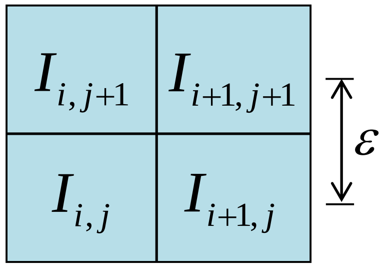
H

Discrete edge operators

- Finite difference approximations:

$$\frac{\partial I}{\partial x} \approx \frac{1}{2\varepsilon}\Big(\big(I_{i+1,j+1} - I_{i,j+1}\big) + \big(I_{i+1,j} - I_{i,j}\big)\Big)$$

$$\frac{\partial I}{\partial y} \approx \frac{1}{2\varepsilon}\Big(\big(I_{i+1,j+1} - I_{i+1,j}\big) + \big(I_{i,j+1} - I_{i,j}\big)\Big)$$



- Convolution masks:

$$\frac{\partial I}{\partial x} \approx \frac{1}{2\varepsilon}\begin{array}{|c|c|}\hline -1 & 1 \\ \hline -1 & 1 \\ \hline\end{array} \qquad \frac{\partial I}{\partial y} \approx \frac{1}{2\varepsilon}\begin{array}{|c|c|}\hline 1 & 1 \\ \hline -1 & -1 \\ \hline\end{array}$$

Discrete edge operators: Laplacian

- Second order partial derivatives:

$$\frac{\partial^2 I}{\partial x^2} \approx \frac{1}{\varepsilon^2}\left(I_{i-1,j} - 2I_{i,j} + I_{i+1,j}\right)$$

$$\frac{\partial^2 I}{\partial y^2} \approx \frac{1}{\varepsilon^2}\left(I_{i,j-1} - 2I_{i,j} + I_{i,j+1}\right)$$

$I_{i-1,j+1}$	$I_{i,j+1}$	$I_{i+1,j+1}$
$I_{i-1,j}$	$I_{i,j}$	$I_{i+1,j}$
$I_{i-1,j-1}$	$I_{i,j-1}$	$I_{i+1,j-1}$

- Laplacian operators: $\nabla^2 I = \dfrac{\partial^2 I}{\partial x^2} + \dfrac{\partial^2 I}{\partial y^2}$
  - Convolution masks:

$$\nabla^2 I \approx \frac{1}{\varepsilon^2}\begin{array}{|c|c|c|}\hline 0 & 1 & 0 \\ \hline 1 & -4 & 1 \\ \hline 0 & 1 & 0 \\ \hline\end{array} \quad \text{or} \quad \frac{1}{6\varepsilon^2}\begin{array}{|c|c|c|}\hline 1 & 4 & 1 \\ \hline 4 & -20 & 4 \\ \hline 1 & 4 & 1 \\ \hline\end{array}$$

(more accurate)

Discrete edge operators: Sobel

- Better approximations of the derivatives exist
- The **Sobel operators** below are very commonly used

$$\frac{1}{8}\begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array} \qquad \frac{1}{8}\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array}$$

$$H_x \qquad\qquad H_y$$

- The standard definition of Sobel omits the 1/8 term
  - It does not make a difference for edge detection
  - The 1/8 term is needed to get the right gradient value, however

Comparing edge operators

Good Localization
Noise Sensitive
Poor Detection

Gradient

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

Roberts 2×2

-1	0
0	1

0	-1
1	0

Sobel 3×3

1	2	1
0	0	0
-1	-2	1

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

Sobel 5×5

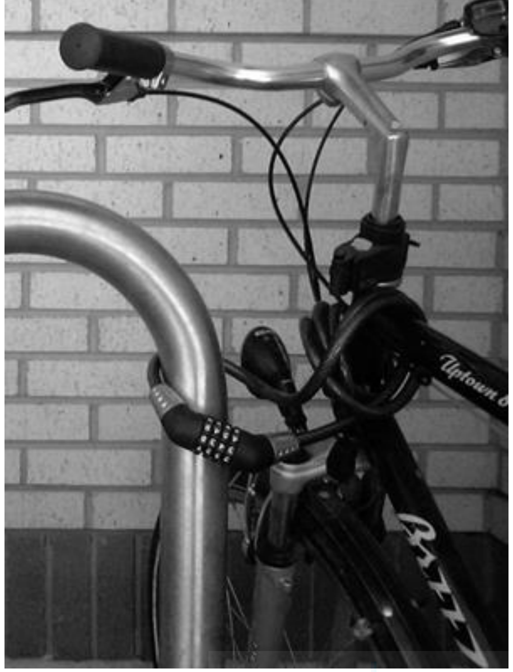
1	2	3	2	1
2	3	5	3	2
0	0	0	0	0
-2	-3	-5	-3	-2
-1	-2	-3	-2	-1

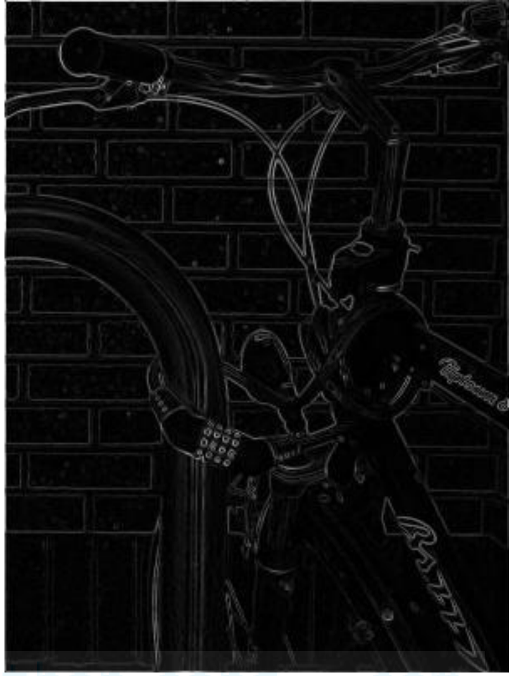
-1	-2	0	2	1
-2	-3	0	3	2
-3	-5	0	5	3
-2	-3	0	3	2
-1	-2	0	2	1

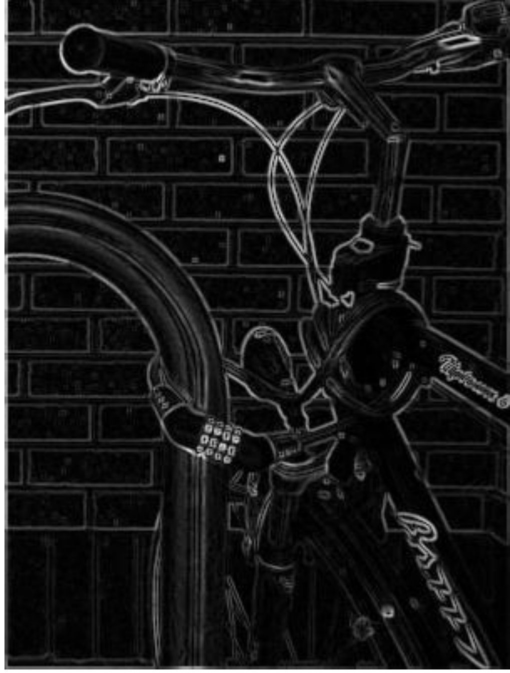
Poor Localization
Less Noise Sensitive
Good Detection

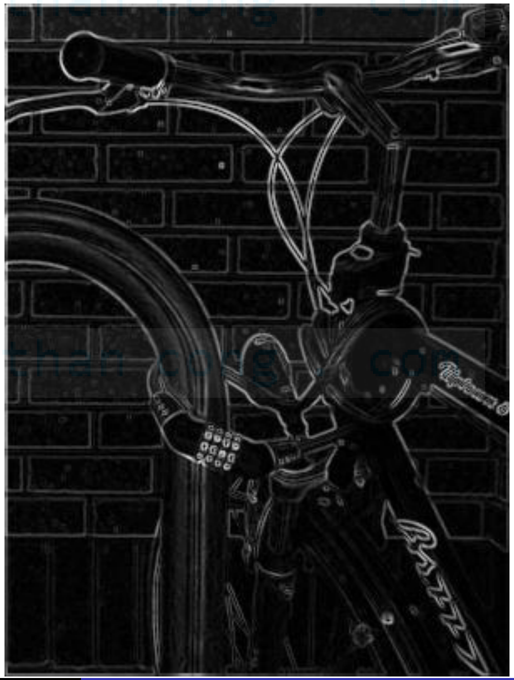
Common edge operators

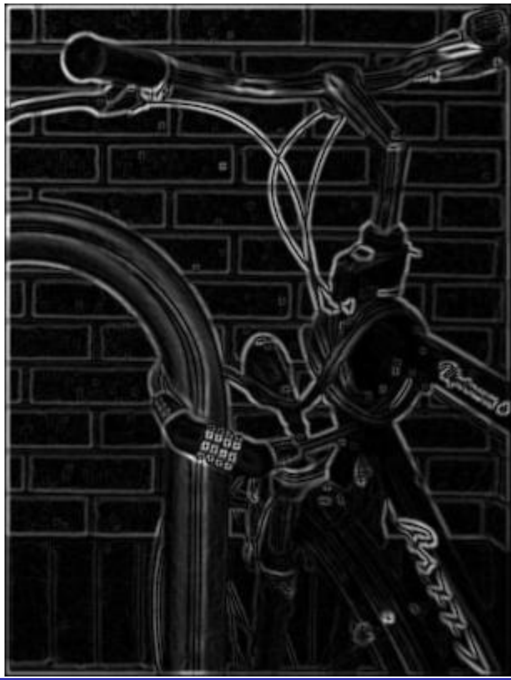
	g_x	g_y	Diagonals	
Roberts 2×2	$\begin{array}{\|c\|c\|}\hline -1 & 0 \\ \hline 0 & 1 \\ \hline\end{array}$	$\begin{array}{\|c\|c\|}\hline 0 & -1 \\ \hline 1 & 0 \\ \hline\end{array}$		
Prewitt 3×3	$\begin{array}{\|c\|c\|c\|}\hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -1 & 1 \\ \hline\end{array}$	$\begin{array}{\|c\|c\|c\|}\hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline\end{array}$	$\begin{array}{\|c\|c\|c\|}\hline 0 & 1 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & -1 & 0 \\ \hline\end{array}$	$\begin{array}{\|c\|c\|c\|}\hline -1 & -1 & 0 \\ \hline -1 & 0 & 1 \\ \hline 0 & 1 & 1 \\ \hline\end{array}$
Sobel 3×3	$\begin{array}{\|c\|c\|c\|}\hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & 1 \\ \hline\end{array}$	$\begin{array}{\|c\|c\|c\|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline\end{array}$	$\begin{array}{\|c\|c\|c\|}\hline 0 & 1 & 2 \\ \hline -1 & 0 & 1 \\ \hline -2 & -1 & 0 \\ \hline\end{array}$	$\begin{array}{\|c\|c\|c\|}\hline -2 & -1 & 0 \\ \hline -1 & 0 & 1 \\ \hline 0 & 1 & 2 \\ \hline\end{array}$





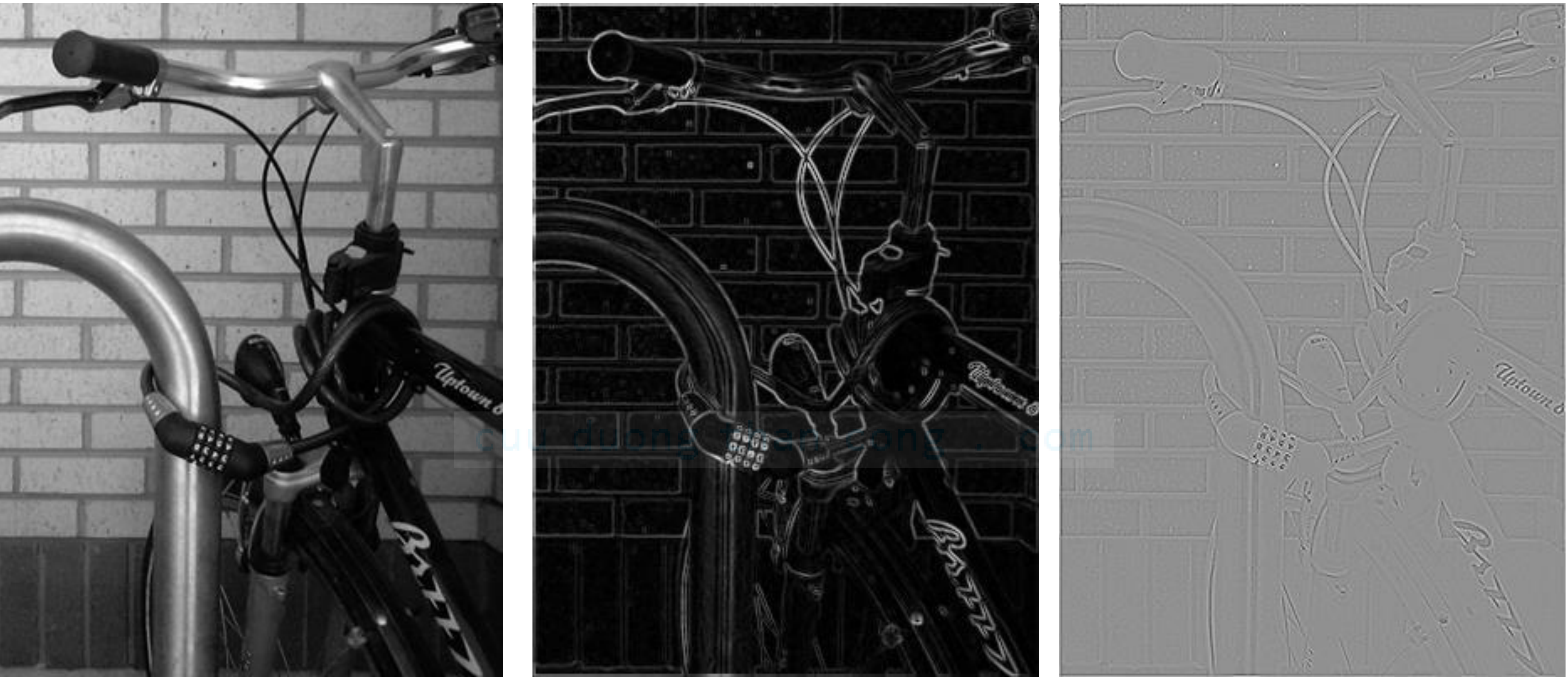






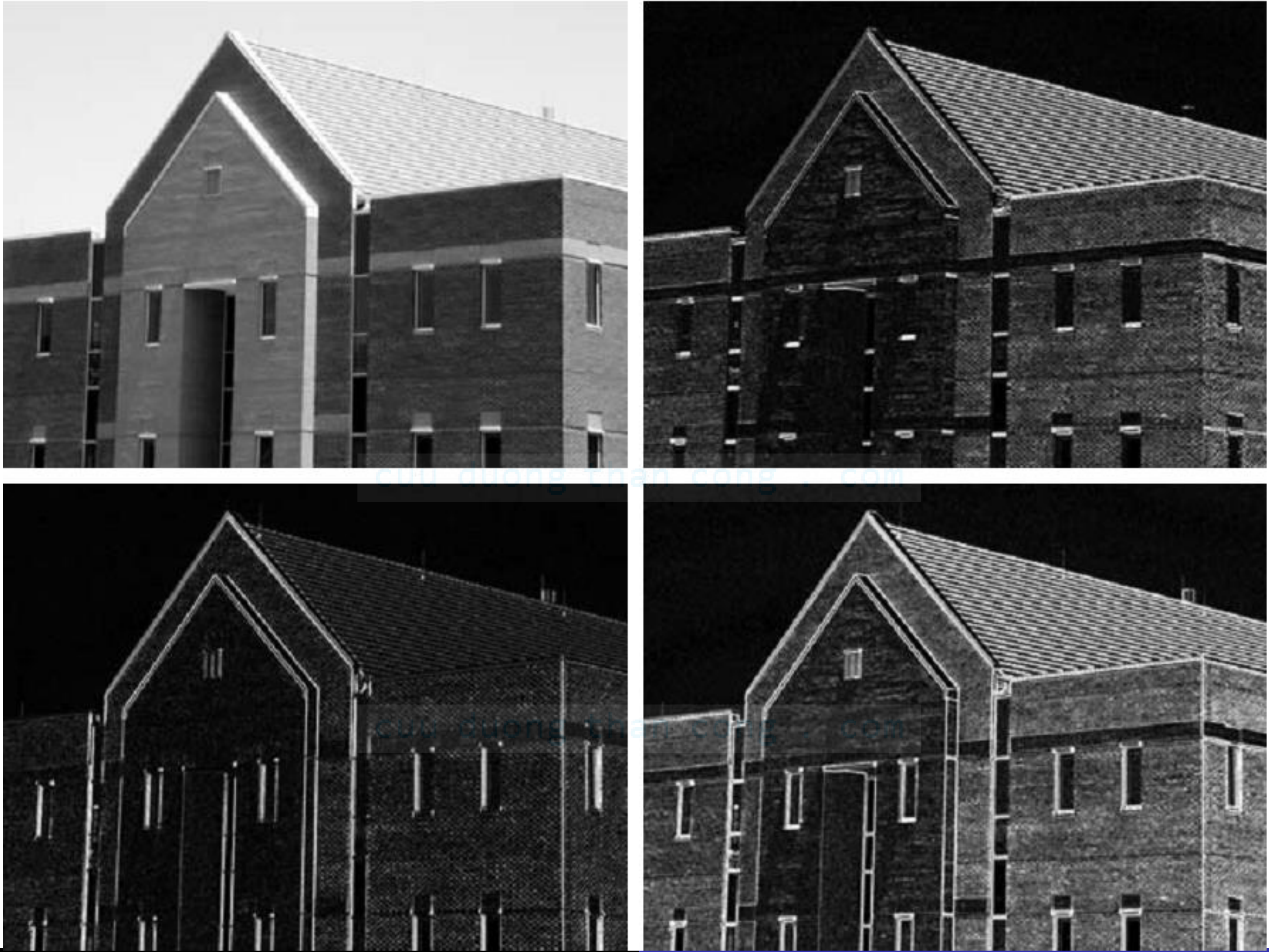
a	b	c
	d	e

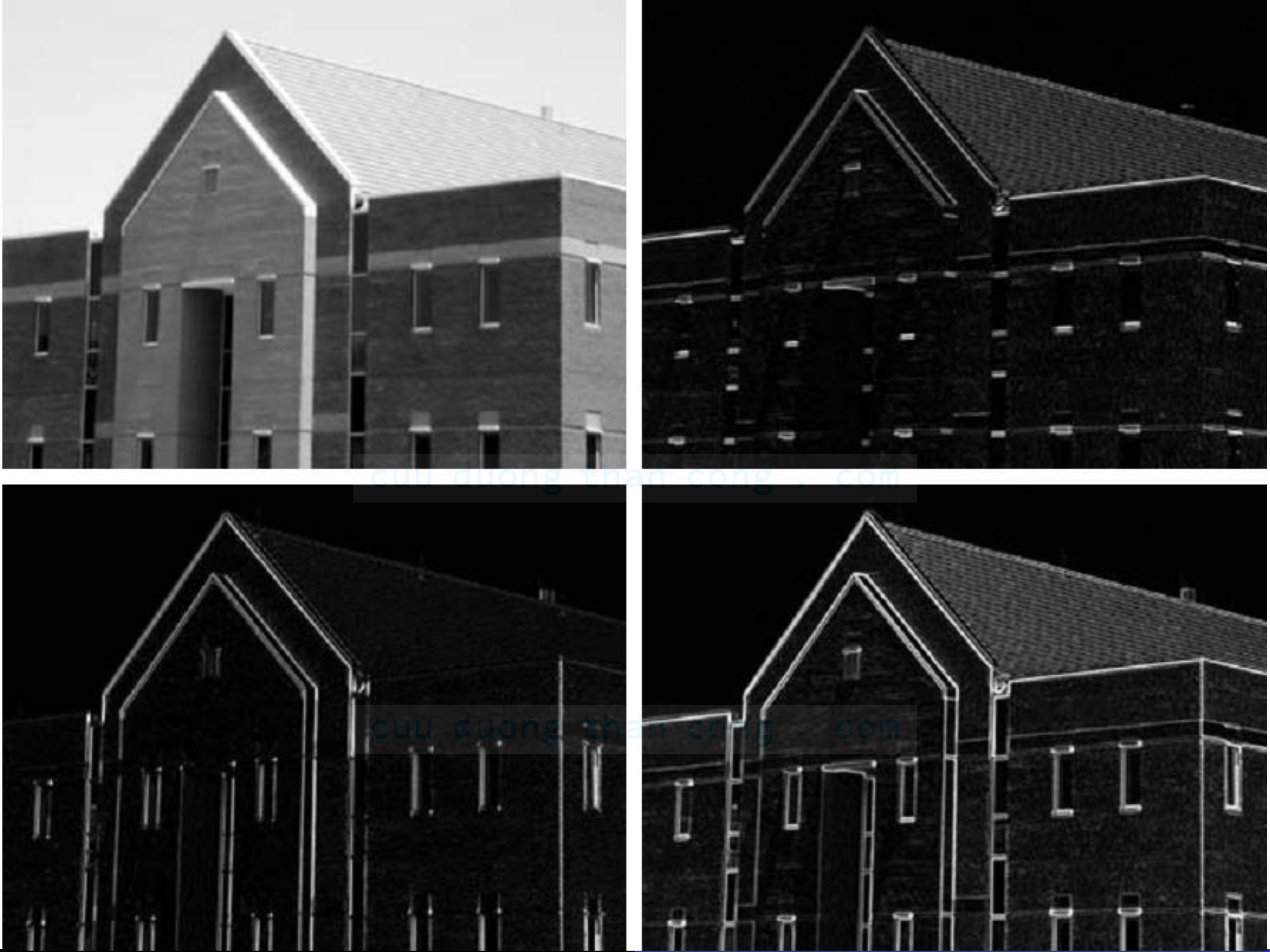
(a) Original image.
Edges detected by (b) Roberts 2×2, c) Prewitts 3×3, d) Sobel 3×3 and e) Sobel 5×5 operators



a b c

(a) Original image. Edges detected by (b) Sobel 3×3 and Laplacian (center = -4) operators





Slide 26

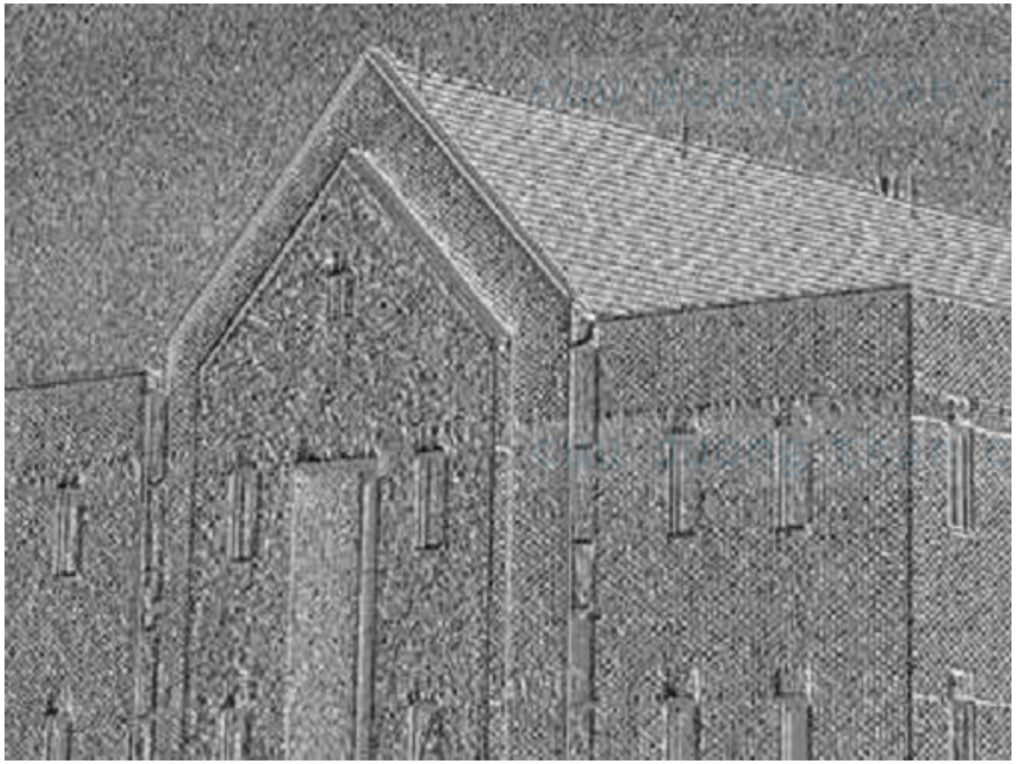
a	b
c	d

(a) Original image of size 834×1114 pixels, with intensity values scaled to the range [0, 1]. (b)-(c) $|g_x|$ and $|g_y|$, the component of the gradient in the *x*-direction and y-direction using the Sobel operators. (d) The gradient image, $|g_x| + |g_y|$

Slide 27

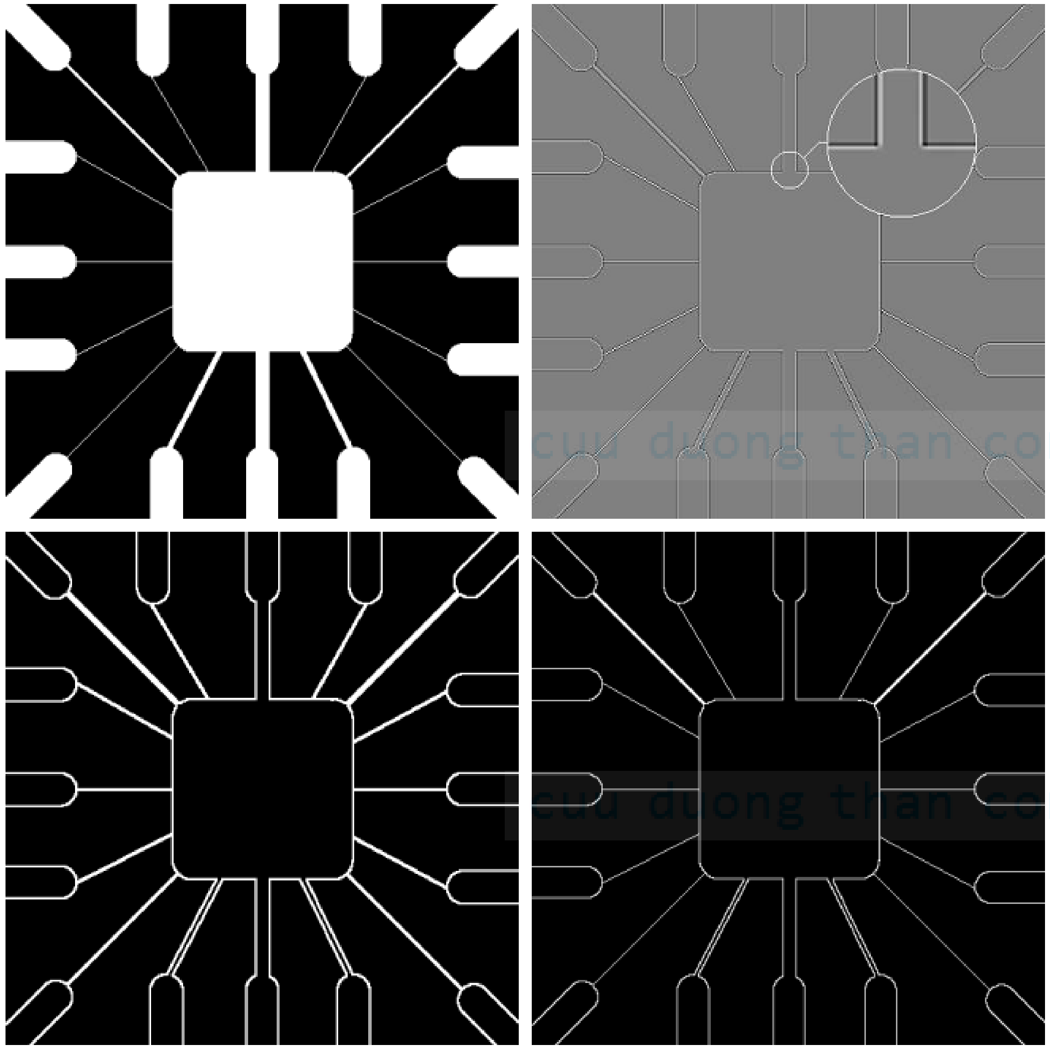
a	b
c	d

Same sequence as above, but with the original image smoothed using a 5×5 averaging filter prior to edge detection



Gradient angle image computed using $\arctan\left(g_y/g_x\right)$. Areas of constant intensity in this image indicate that the direction of the gradient vector is the same at all the pixellocations in those regions.

Double-line effect of the Laplacian



(a) Original image. (b) Laplacian image; the magnified section shows the positive/negative double-line effect characteristic of the Laplacian. (c) Absolute value of the Laplacian. (d) Positive values of the Laplacian.

Common edge operators: Kirsch

- The Kirsch operator (Kirsch compass kernel) is a non-linear edge detector that finds the maximum edge strength in a few predetermined directions

5	5	5
-3	0	-3
-3	-3	-3

g_1 (north)

5	5	-3
5	0	-3
-3	-3	-3

g_2 (northwest)

5	-3	-3
5	0	-3
5	-3	-3

g_3 (west)

-3	-3	-3
5	0	-3
5	5	-3

g_4 (southwest)

-3	-3	-3
-3	0	-3
5	5	5

g_5 (south)

-3	-3	-3
-3	0	5
-3	5	5

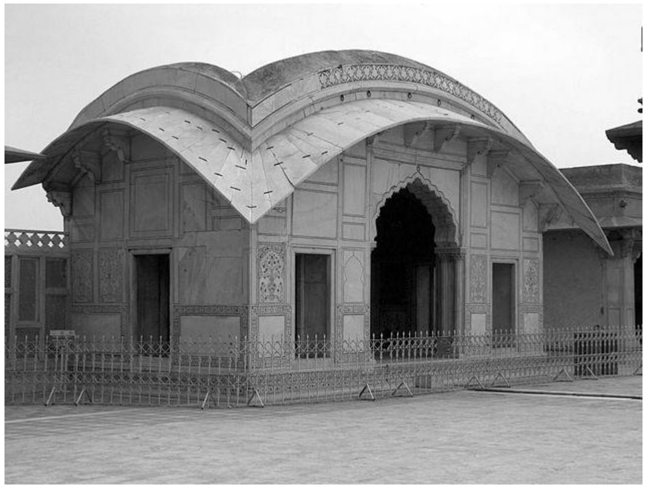
g_2 (southeast)

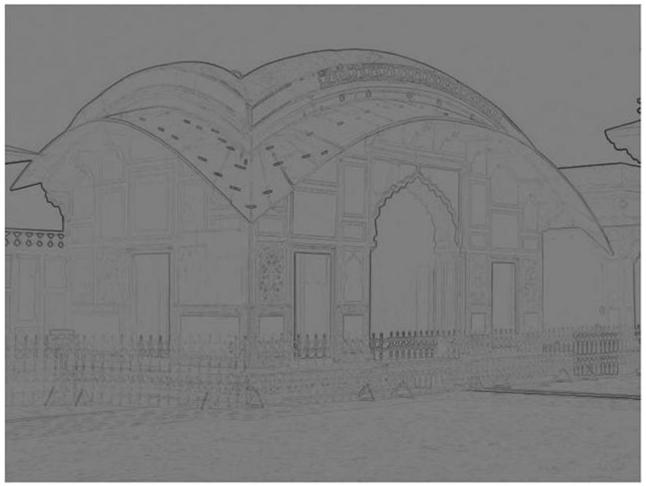
-3	-3	5
-3	0	5
-3	-3	5

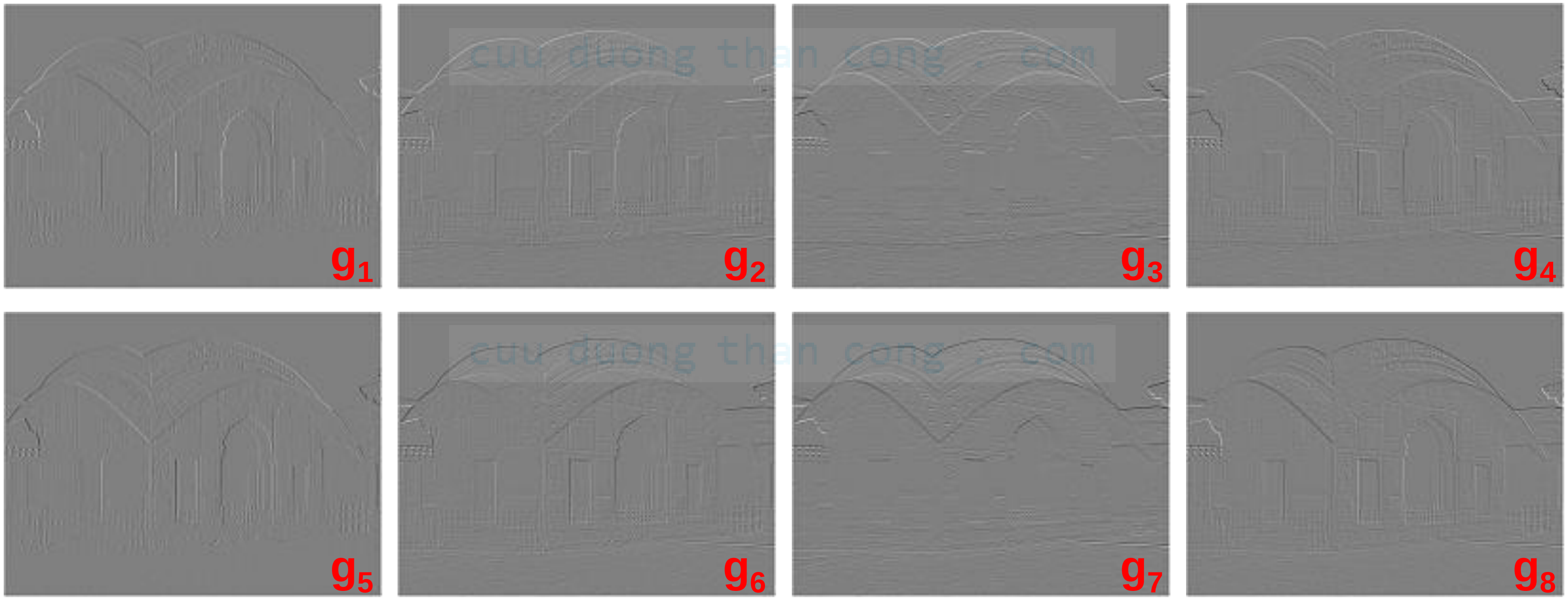
g_3 (east)

-3	5	5
-3	0	5
-3	-3	-3

g_4 (northeast)

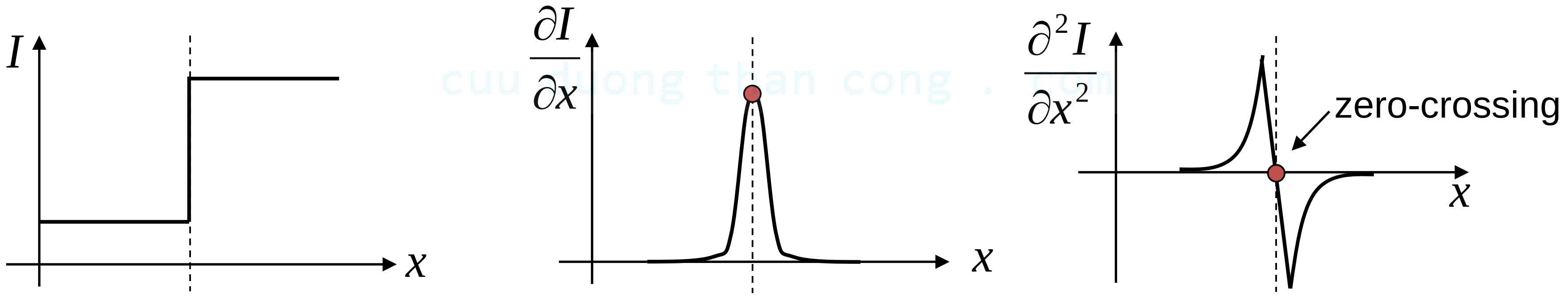

Original image


Maximum gradient in 8 directions



First-order vs. second-order derivatives

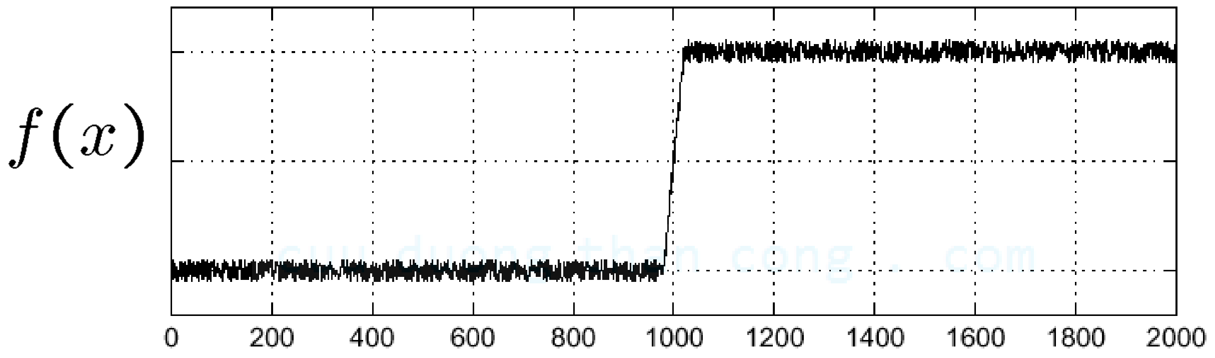
- First-order derivatives
  - generally produce thicker edges in an image
  - generally have a stronger response to a gray-level step
- Second-order derivatives
  - have a stronger response to fine detail, such as thin lines and isolated points
    - Laplacian is sensitive to noise. First smooth the image with Gaussian before applying the Laplacian
  - produce a double response at step changes in gray level

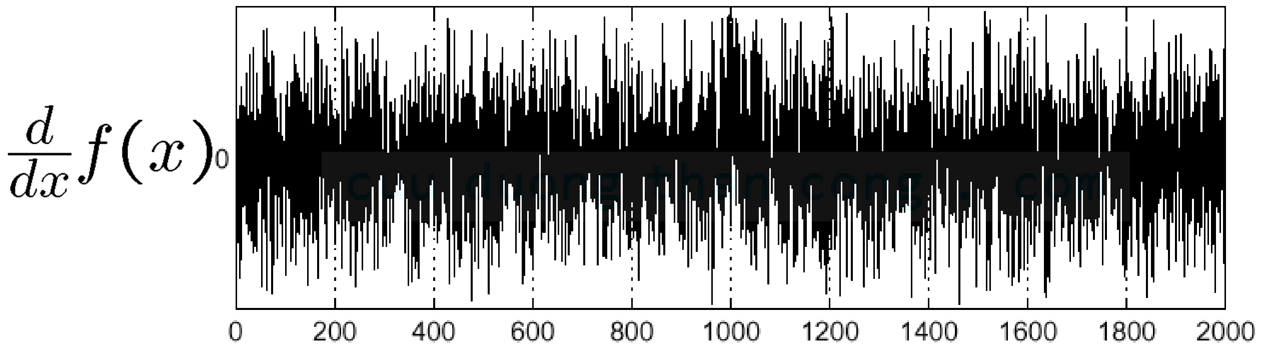


I
x
$\frac{\partial I}{\partial x}$
x
$\frac{\partial^2 I}{\partial x^2}$
zero-crossing
x

Effects of noise

- Consider a single row or column of the image
  - Plotting intensity as a function of position gives a *signal*





Where is the edge?

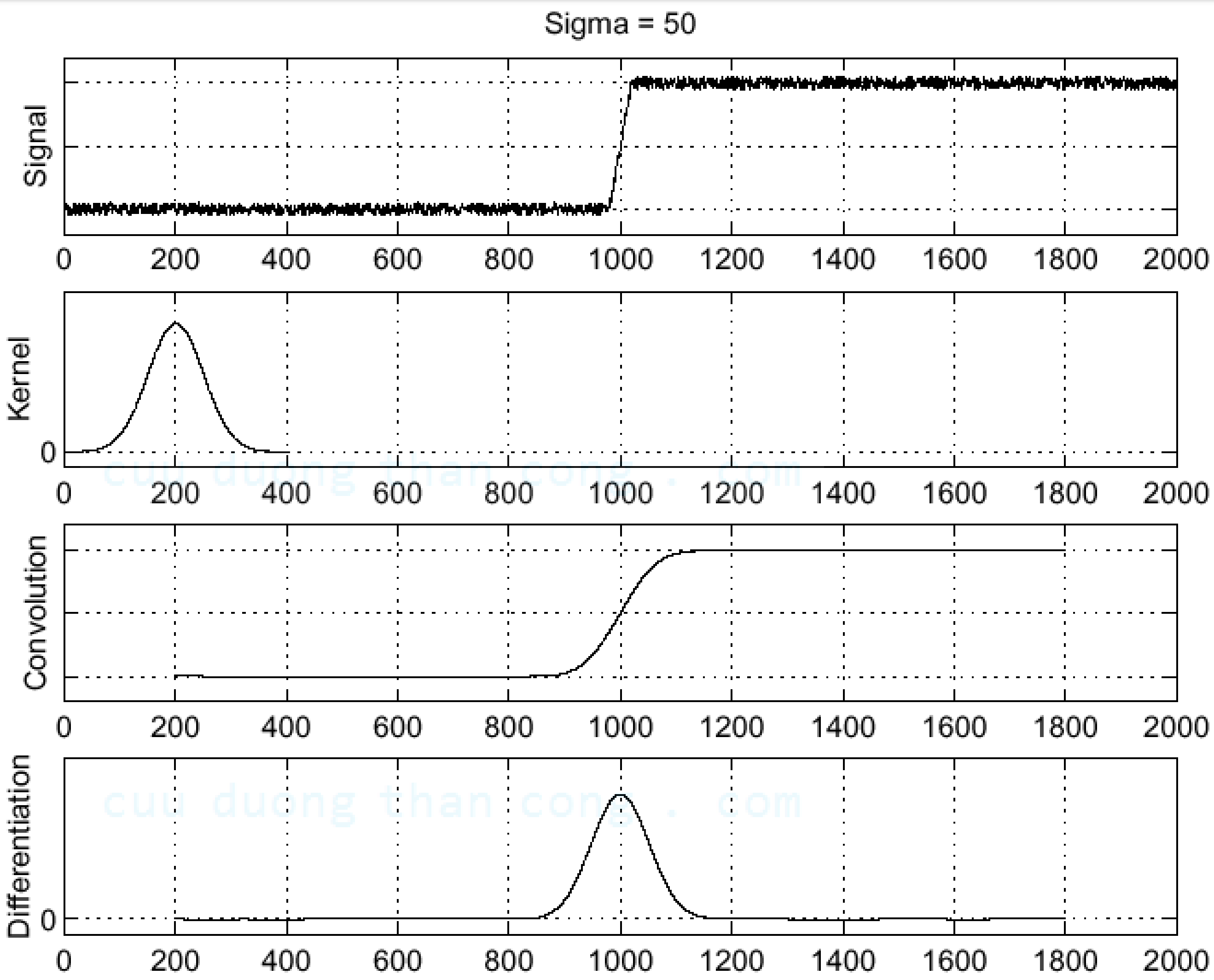
Solution for Noise effects: Smooth first

f

h

$h \star f$

$\frac{\partial}{\partial x}(h \star f)$



Where is the edge? Look for peaks in $\frac{\partial}{\partial x}(h \star f)$

Derivative theorem of convolution

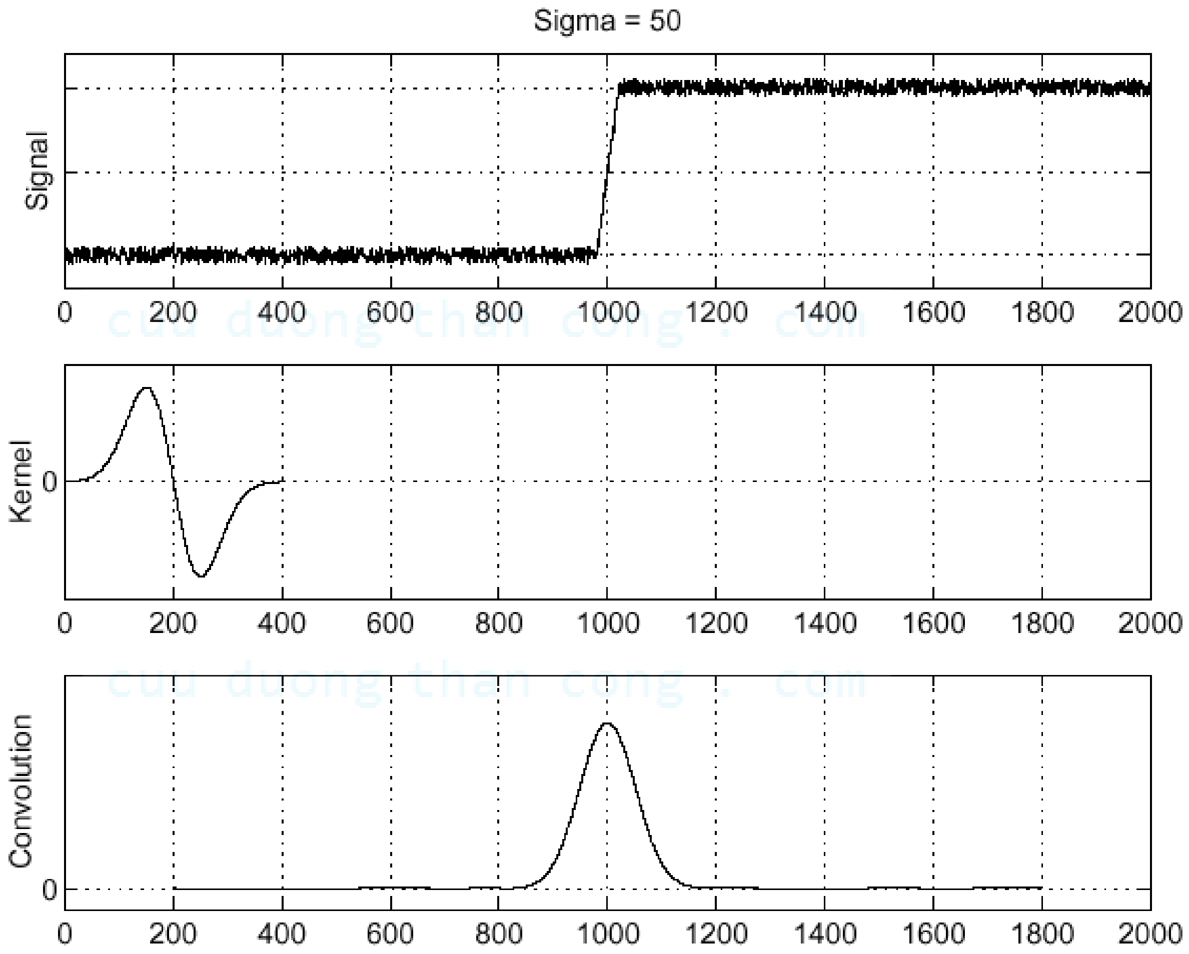
$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$

…saves us one operation.

f

$\frac{\partial}{\partial x}h$

$(\frac{\partial}{\partial x}h) \star f$



Laplacian of Gaussian (LoG)

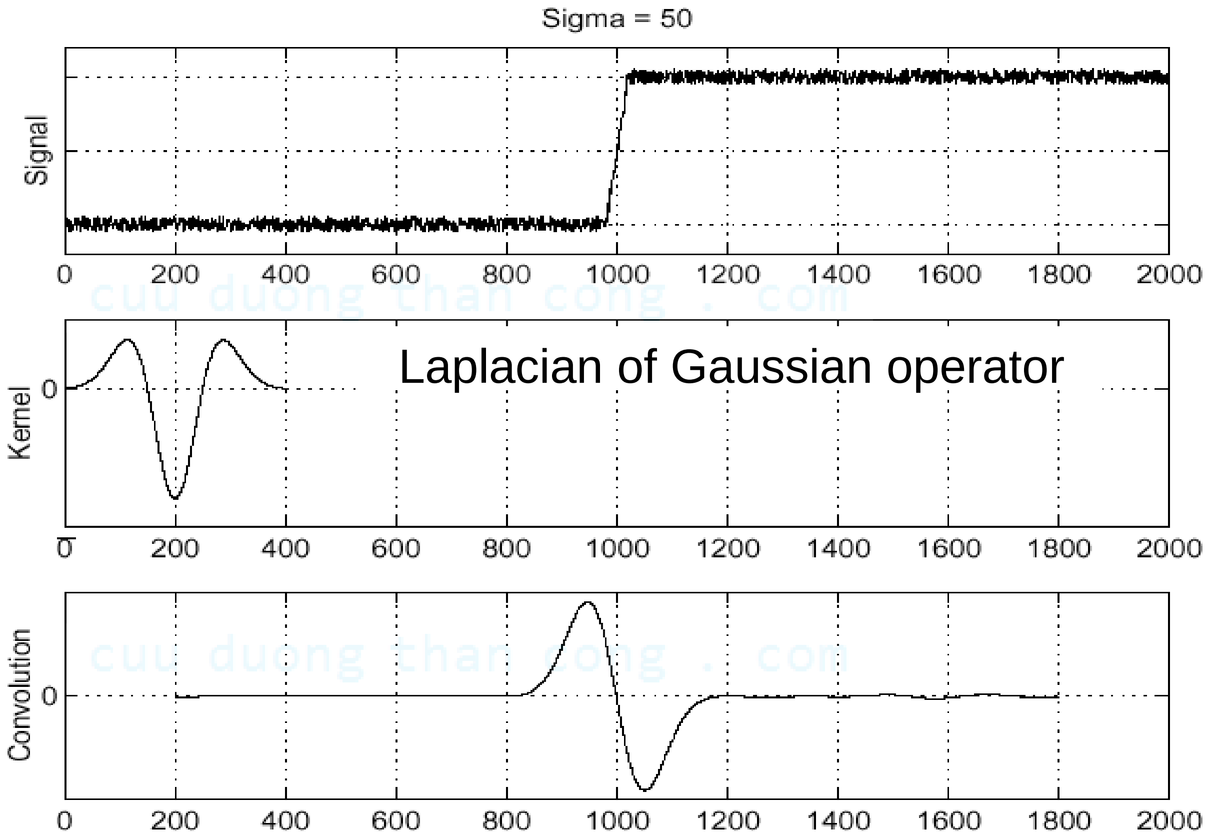
$$\frac{\partial^2}{\partial x^2}(h \star f) = \boxed{\left(\frac{\partial^2}{\partial x^2}\right)} \star f$$

Laplacian of Gaussian

f

$\frac{\partial^2}{\partial x^2}h$

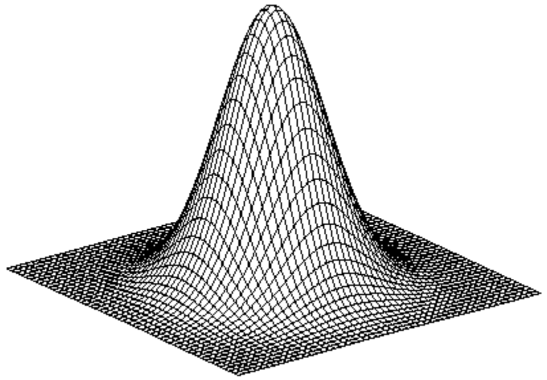
$(\frac{\partial^2}{\partial x^2}h) \star f$



Where is the edge?

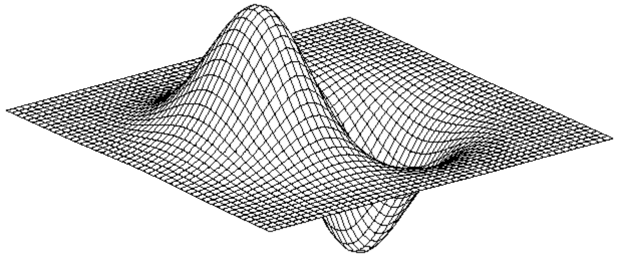
Zero-crossings of bottom graph !

2D Gaussian edge operators



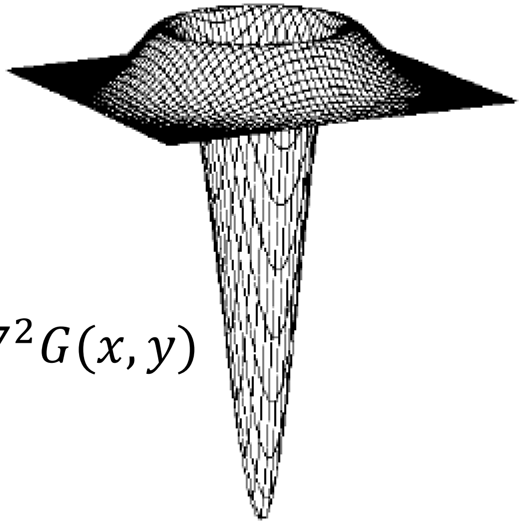
$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Gaussian



$$\frac{\partial}{\partial x} G(x,y)$$

Derivative of Gaussian



$$\nabla^2 G(x,y)$$

Laplacian of Gaussian

Mexican Hat (Sombrero)

- ∇^2 is the Laplacian operator: $\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$

Laplacian of Gaussian (LoG)

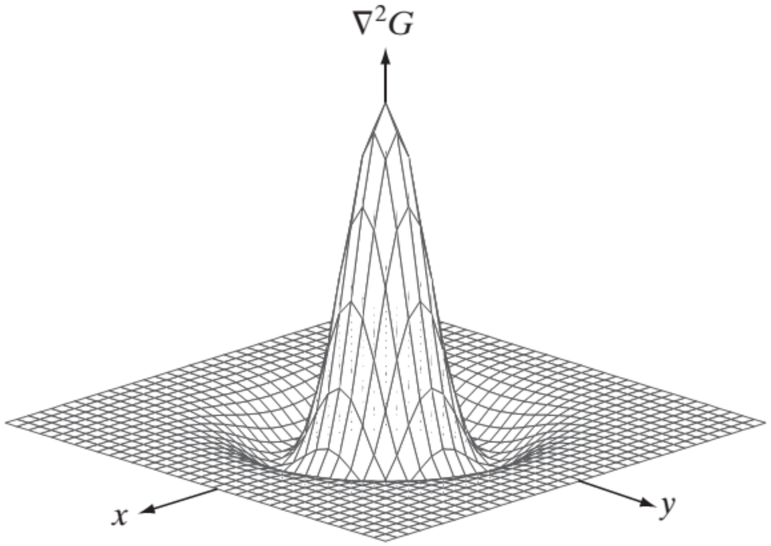
- Assume that $G(x, y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$ is the 2-D Gaussian function
  - Note that the constant term $\frac{1}{2\pi\sigma^2}$ was simplified without any violation

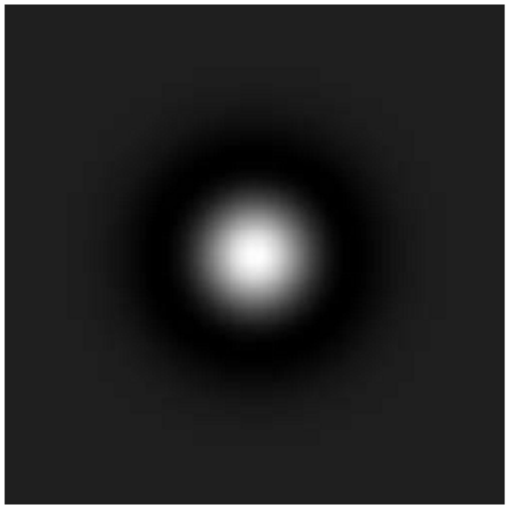
- To find **Laplacian of Gaussian (LoG)** $\nabla^2 G$, we perform the following differentiations

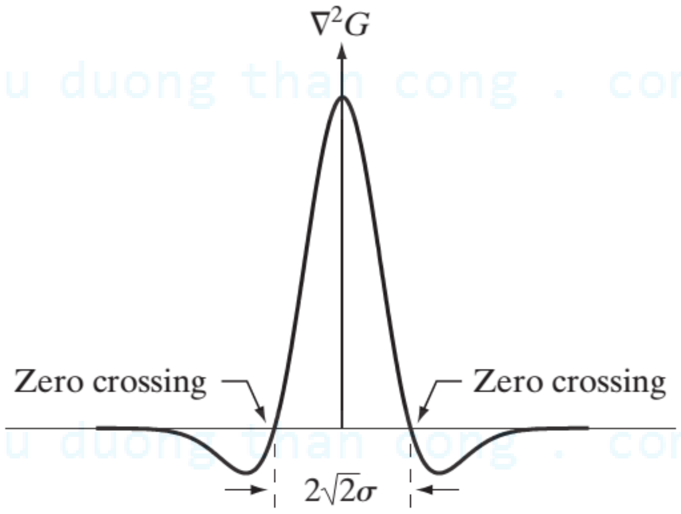
$$\nabla^2 G(x, y) = \frac{\partial^2 G(x,y)}{\partial x^2} + \frac{\partial^2 G(x,y)}{\partial y^2} = \frac{\partial}{\partial x}\left[\frac{-x}{\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}\right] + \frac{\partial}{\partial y}\left[\frac{-y}{\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}\right]$$

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

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







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

a	b
c	d

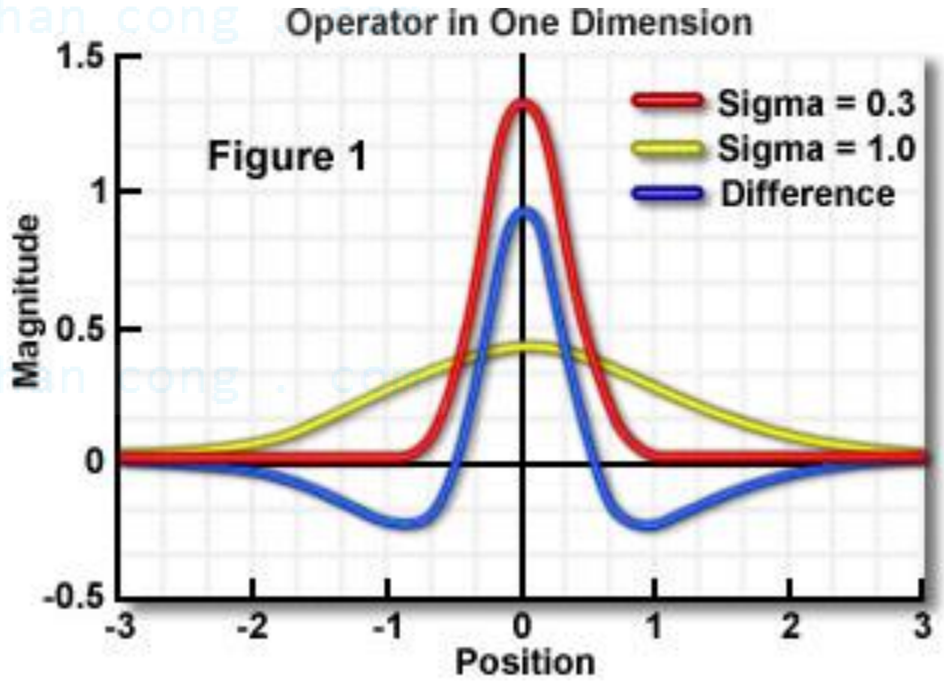
(a) Three dimensional plot of the *negative* of the LoG. (b) Negative of the LoG displayed as an image. (c) Cross section of (a) showing zero crossings. (d) Mask approximation to the shape in (a). The negative of this mask would be used in practice.

Difference of Gaussians (DoG)

- LoG can be approximated by the difference between two different Gaussians (**Difference of Gaussians – DoG**)

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

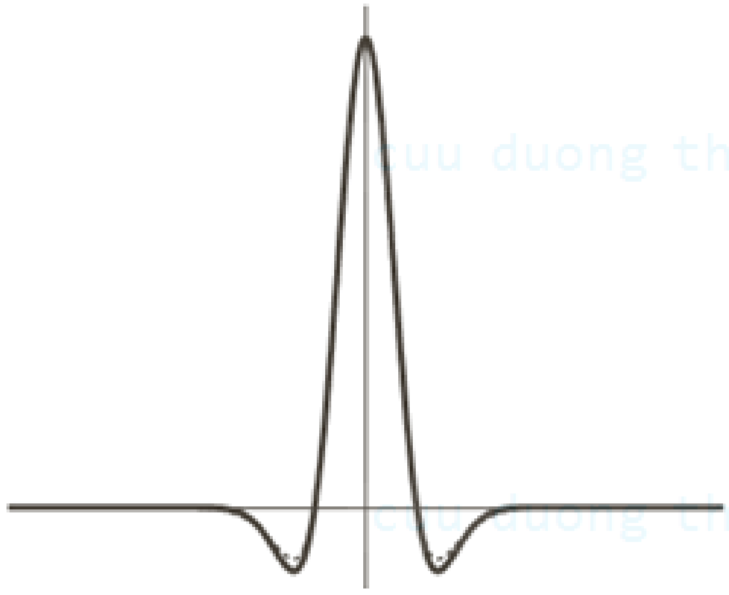
  - where $\sigma_1 = k\sigma_2$, typically $k > 0$

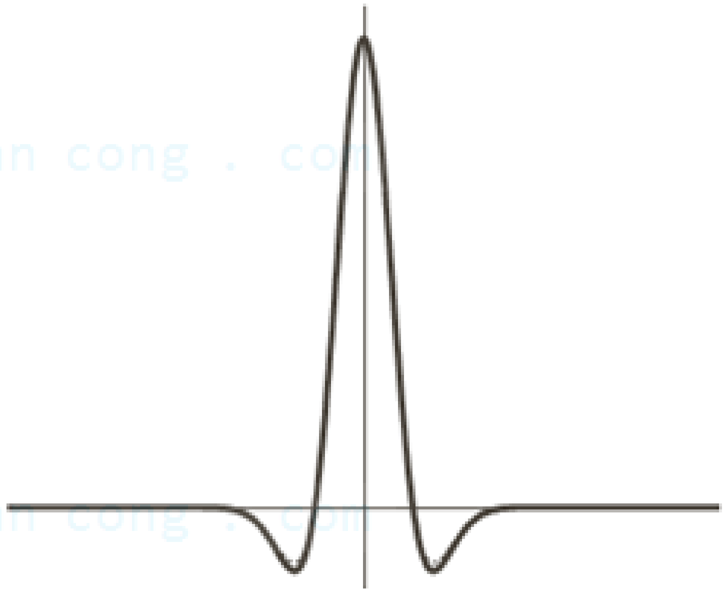


DoG vs. LoG

- LoG and DoG have the same zero crossings when

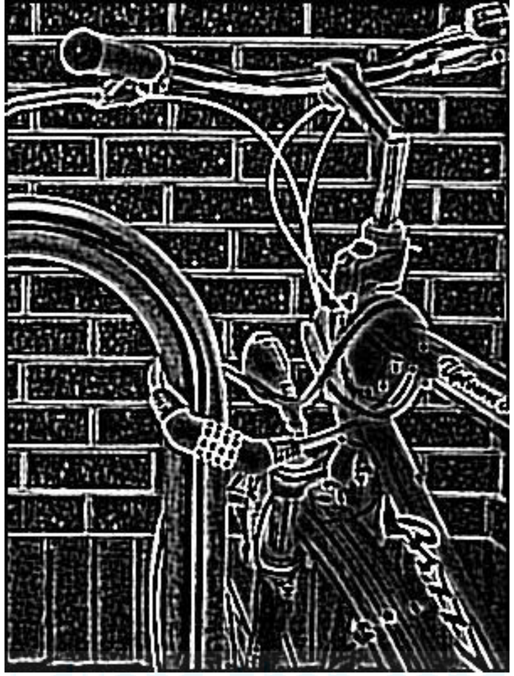
$$\sigma^2 = \frac{\sigma_1^2\sigma_2^2}{\sigma_1^2 - \sigma_2^2} = \ln\left[\frac{\sigma_1^2}{\sigma_2^2}\right]$$

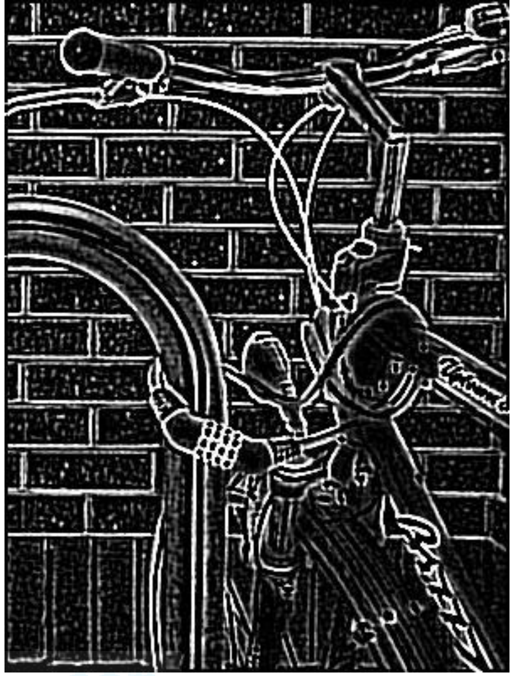

(a)


(b)

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

LOG





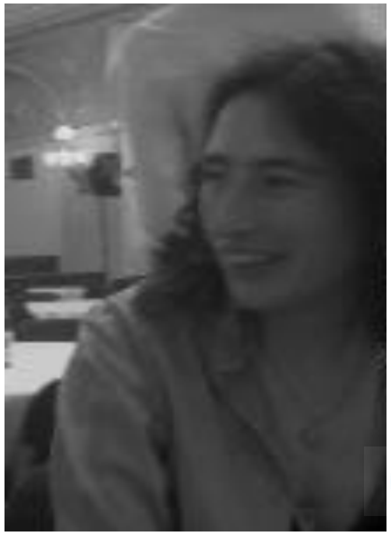
$\sigma_1 - \sigma_2$

Original image

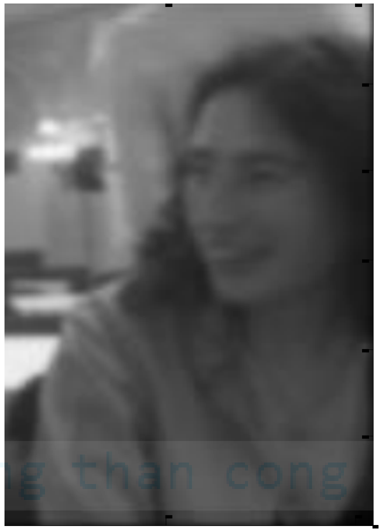
Gaussian $\sigma_1 = 3$

Gaussian $\sigma_2 = 2$

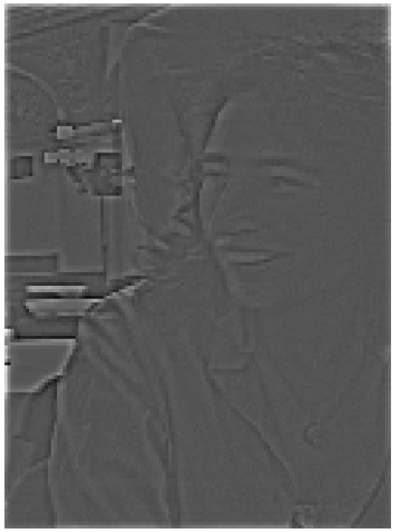
Unsharp Masking

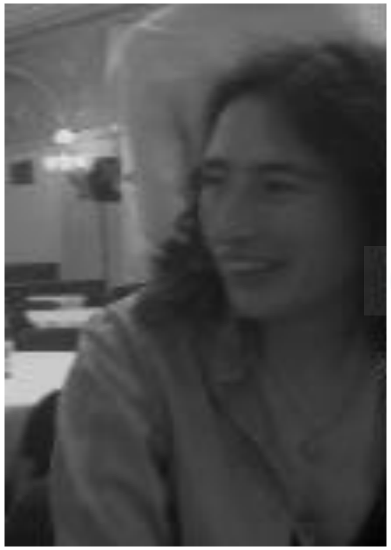


−

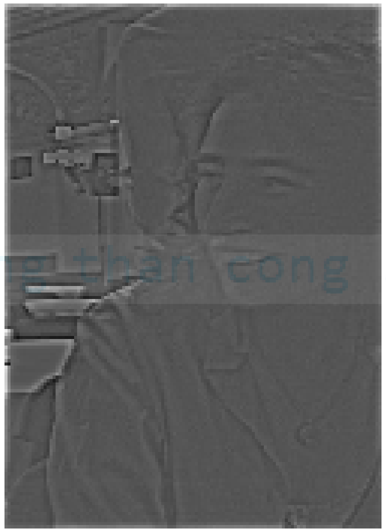


=

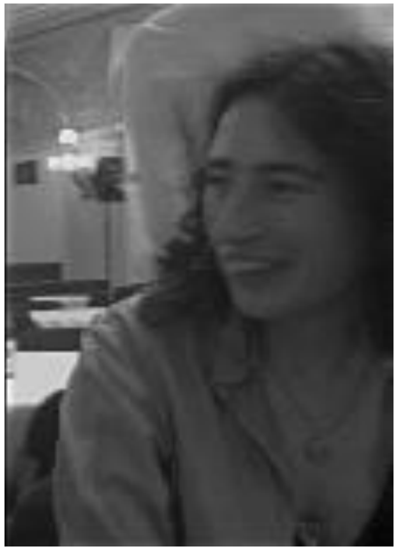




+ a



=



Canny Edge Operator

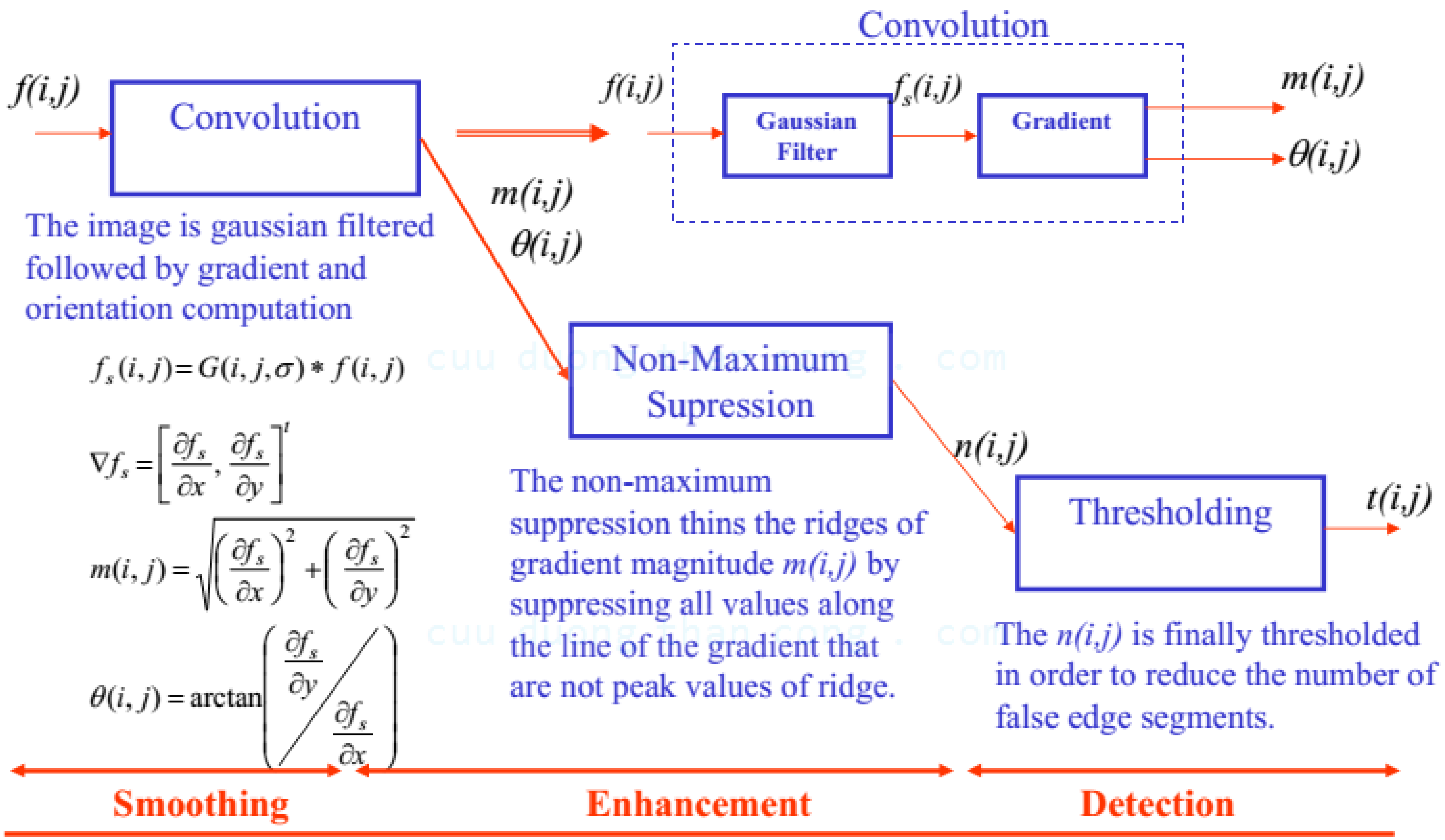
- Smooth image *I* with 2D Gaussian: $G \star I$
- Find local edge normal directions for each pixel

$$\bar{\mathrm{n}} = \frac{\nabla(G \star I)}{|\nabla(G \star I)|}$$

- Compute edge magnitudes $|\nabla(G \star I)|$
- Locate edges by finding zero-crossings along the edge normal directions (non-maximum suppression)

$$\frac{\partial^2 (G \star I)}{\partial \bar{\mathrm{n}}^2} = 0$$

The process of Canny Edge Operator



The image is gaussian filtered followed by gradient and orientation computation

$$f_s(i,j) = G(i,j,\sigma) * f(i,j)$$

$$\nabla f_s = \left[\frac{\partial f_s}{\partial x}, \frac{\partial f_s}{\partial y}\right]^t$$

$$m(i,j) = \sqrt{\left(\frac{\partial f_s}{\partial x}\right)^2 + \left(\frac{\partial f_s}{\partial y}\right)^2}$$

$$\theta(i,j) = \arctan\left(\frac{\partial f_s}{\partial y} \Big/ \frac{\partial f_s}{\partial x}\right)$$

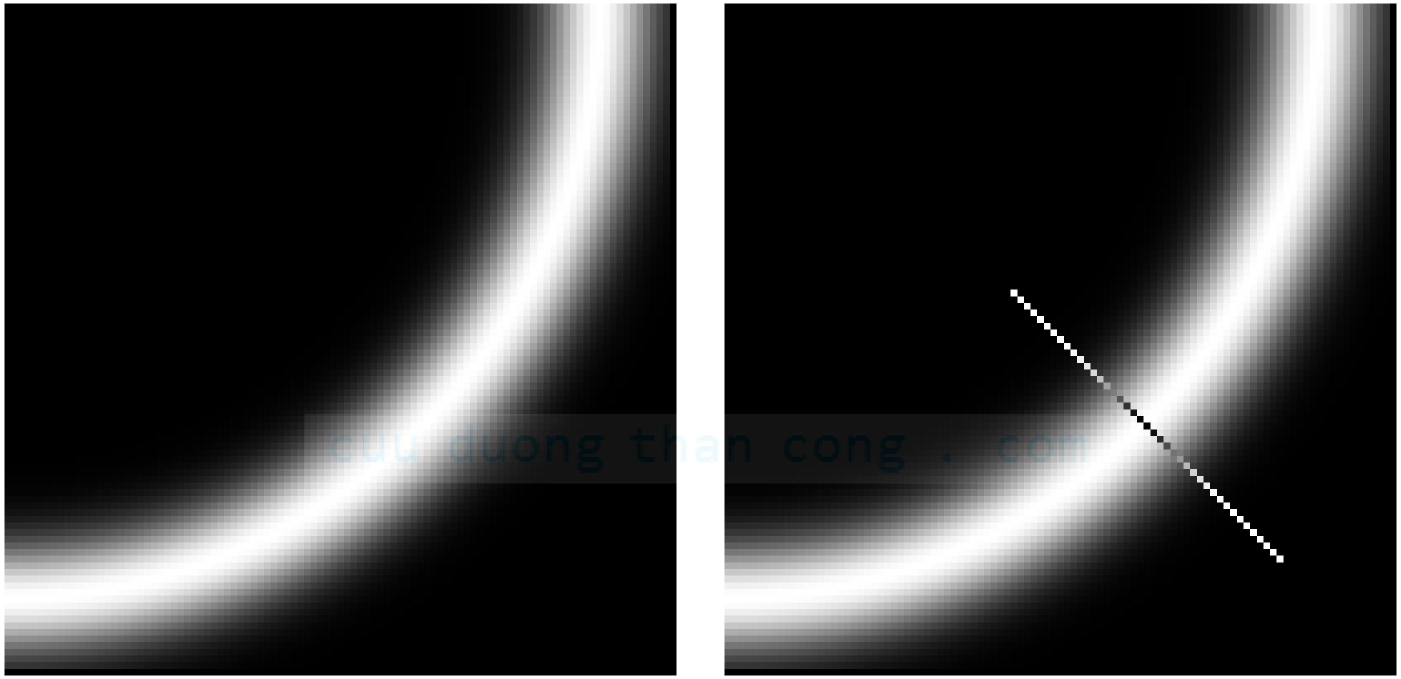
The non-maximum suppression thins the ridges of gradient magnitude *m(i,j)* by suppressing all values along the line of the gradient that are not peak values of ridge.

The *n(i,j)* is finally thresholded in order to reduce the number of false edge segments.

General criteria for edge detection

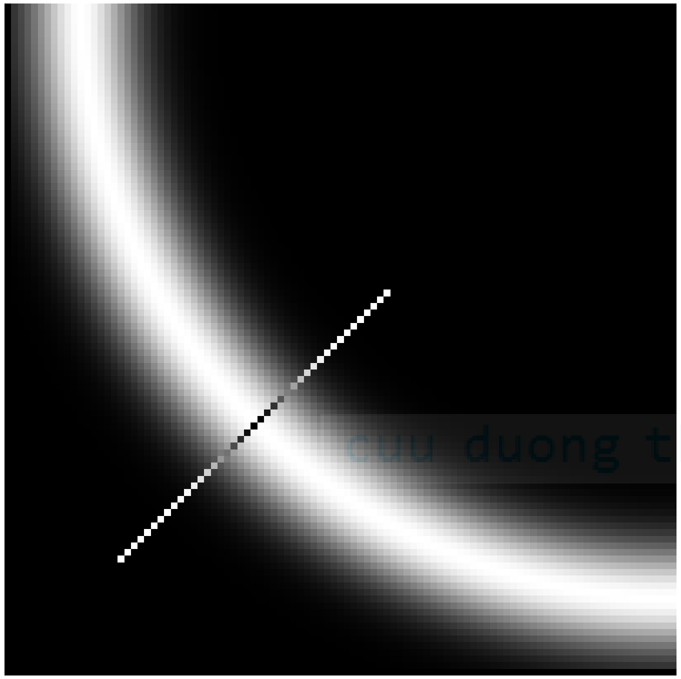
- Detection of edge with low error rate, which means that the detection should accurately catch as many edges shown in the image as possible
- The edge point detected from the operator should accurately localize on the center of the edge
- A given edge in the image should only be marked once, and where possible, image noise should not create false edges

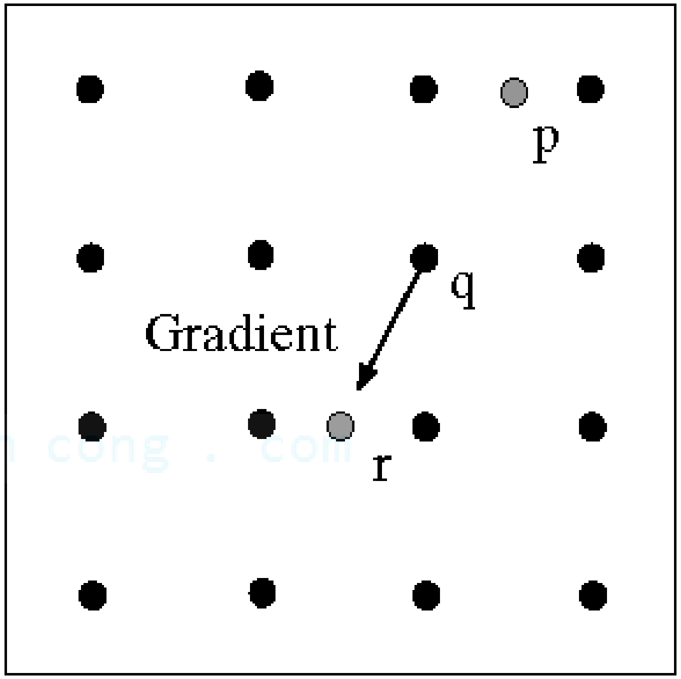
Non-maximum suppression



- Mark points along the curve where the magnitude is biggest
  $\Rightarrow$ suppress all gradient values except the local maximal
- At which point is the maximum? Where is the next one?

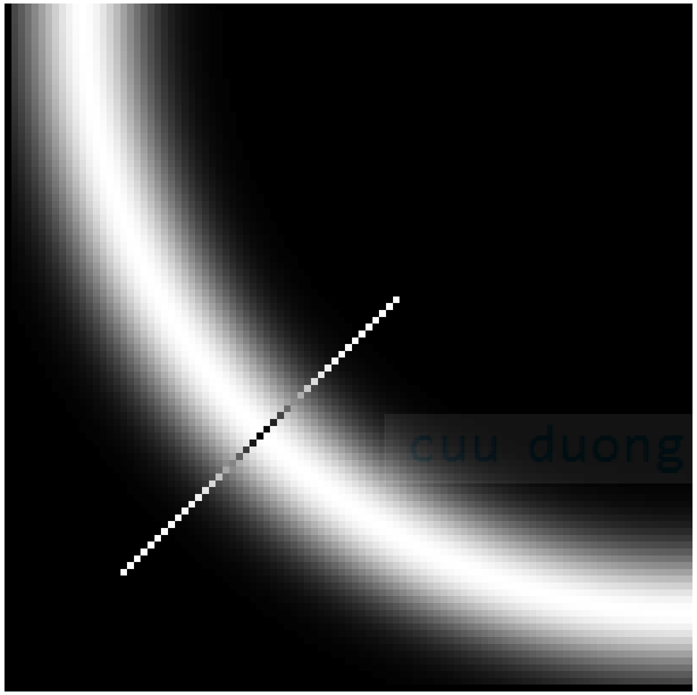
Non-maximum suppression

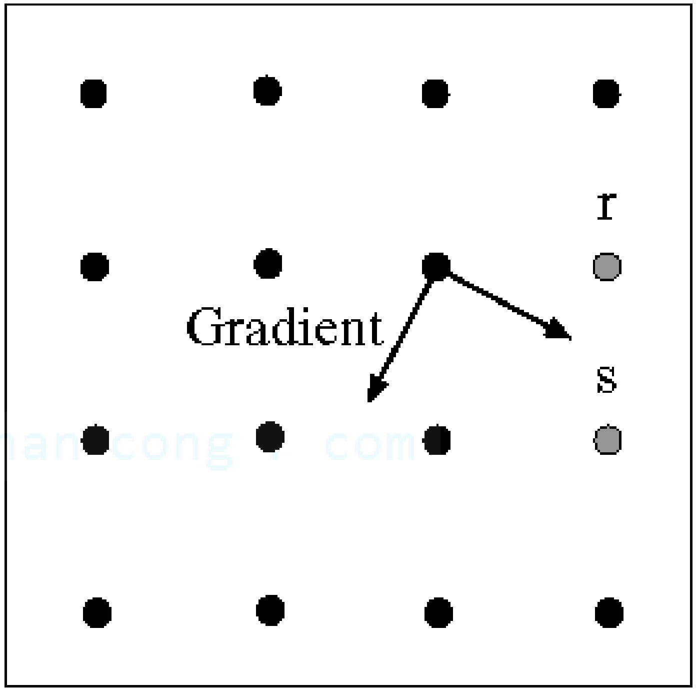




- Check if a pixel q is local maximum along gradient direction
  - requires checking interpolated pixels p and r

Non-maximum suppression

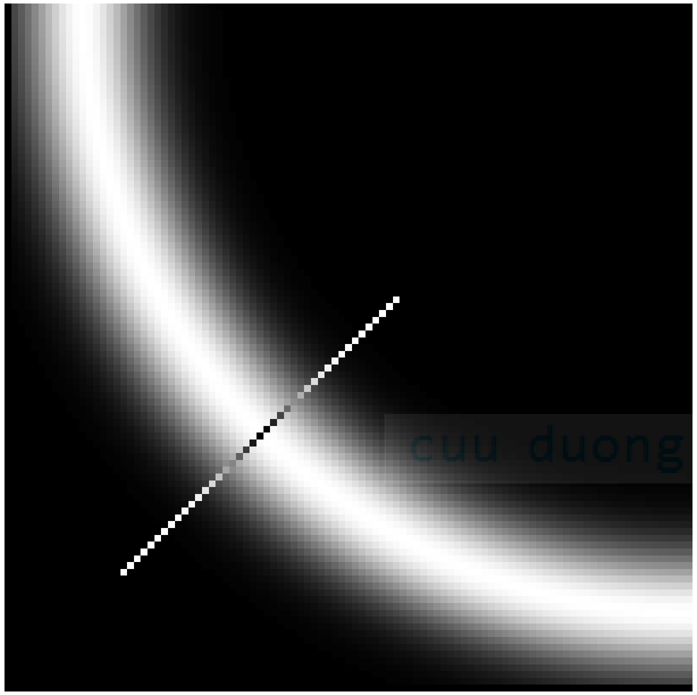


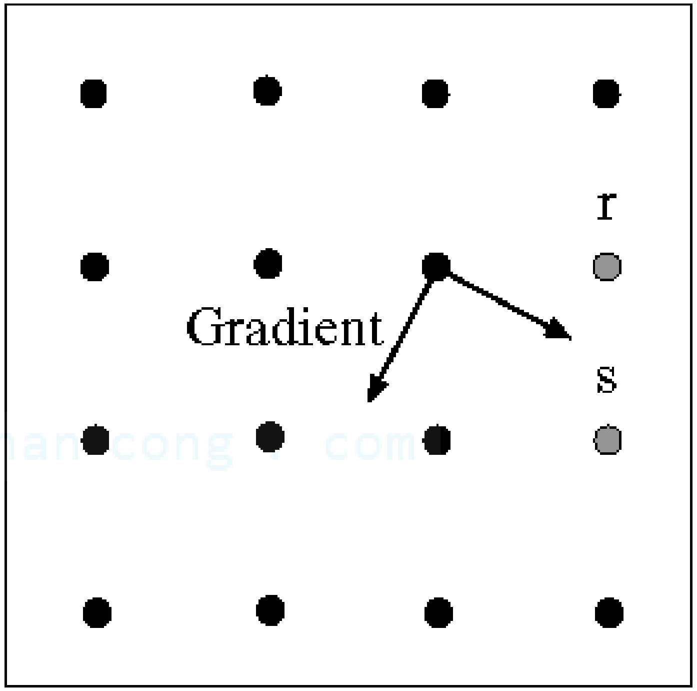


(the next edge point is either r or s)

- Predicting the next edge point
  - Assume the marked point is an edge point, Construct the tangent to the edge curve (which is normal to the gradient at that point) and use this to predict the next points.

Non-maximum suppression

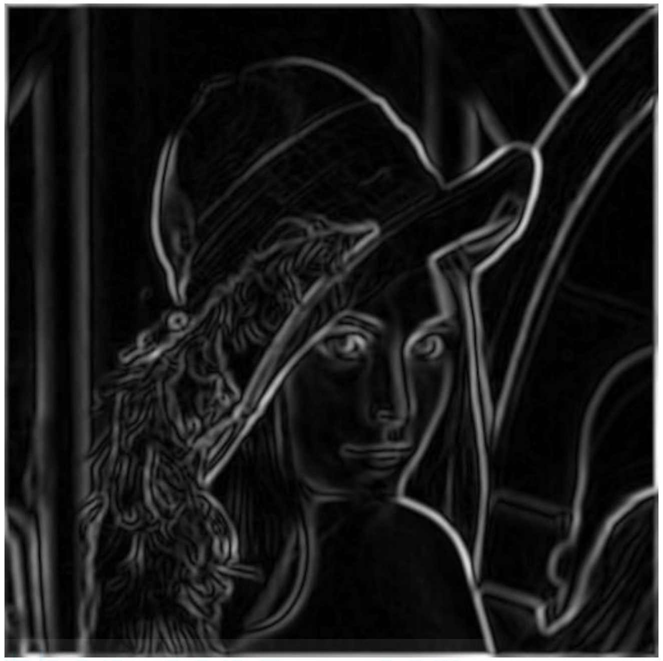


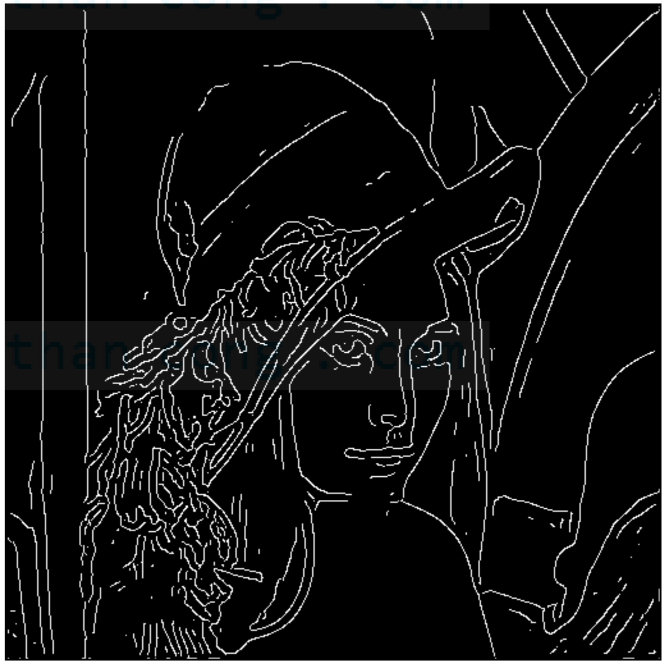


(the next edge point is either r or s)

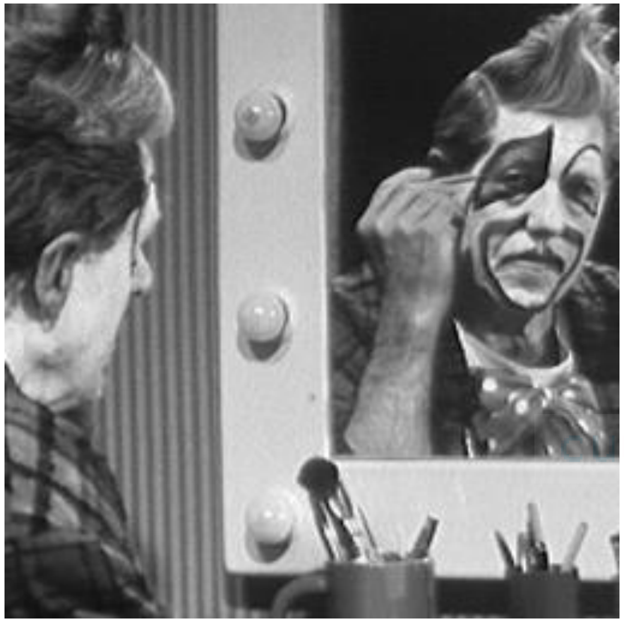
- Edge following: edge points occur along curve like chains

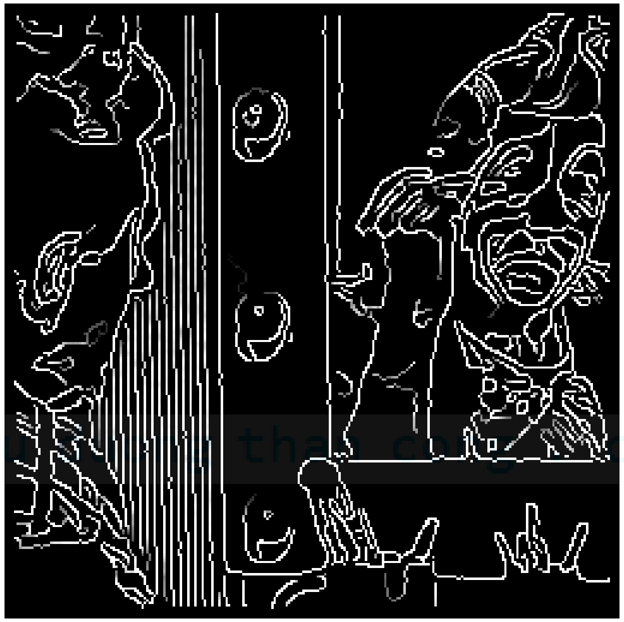
Original image

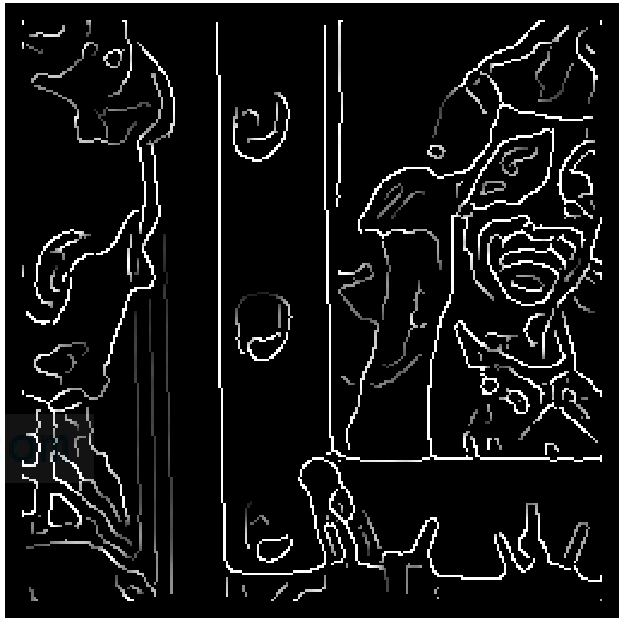

magnitude of the gradient


after non-maximum suppression

Canny Edge operator


original


Canny with $\sigma = 1$


Canny with $\sigma = 2$

- The choice of σ depends on desired behavior
  - large σ detects large scale edges
  - small σ detects fine features

Edge thresholding

- A common step in edge detection is to follow the gradient operator by a magnitude thresholding operation

$$E(x,y) = \begin{cases} 1 & \text{if } \|\nabla f(x,y)\| > T \text{ for some threshold } T \\ 0 & \text{otherwise} \end{cases}$$

  - $\{(x,y): E(x,y) = 1\}$ is called the set of edge pixels

- Thresholding the gradient magnitude leaves only "strong" edges, but it makes no guarantees of continuity

Edge thresholding

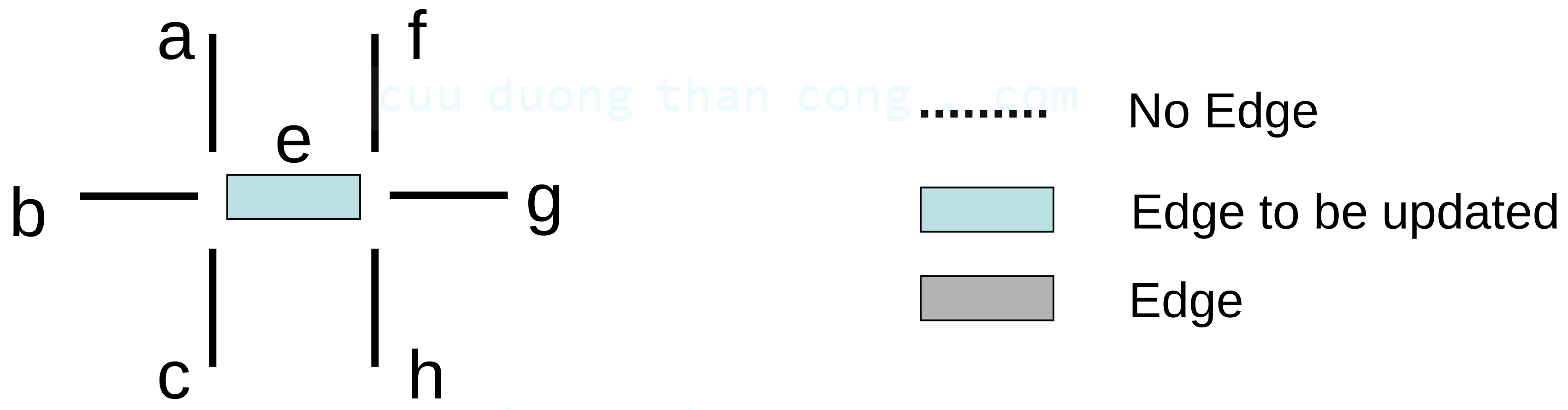
- **Thresholding with hysteresis** uses two thresholds

$$\begin{array}{rl} \|\nabla f(x,y)\| \geq t_1 & \text{definitely an edge} \\ t_0 \geq \|\nabla f(x,y)\| < t_1 & \text{maybe an edge, depends on context} \\ \|\nabla f(x,y)\| < t_0 & \text{definitely not an edge} \end{array}$$

- First identify all definite edge pixels. Then, add all “maybe” pixels only if they are next to an already-labeled edge pixel. This process is repeated until it converges

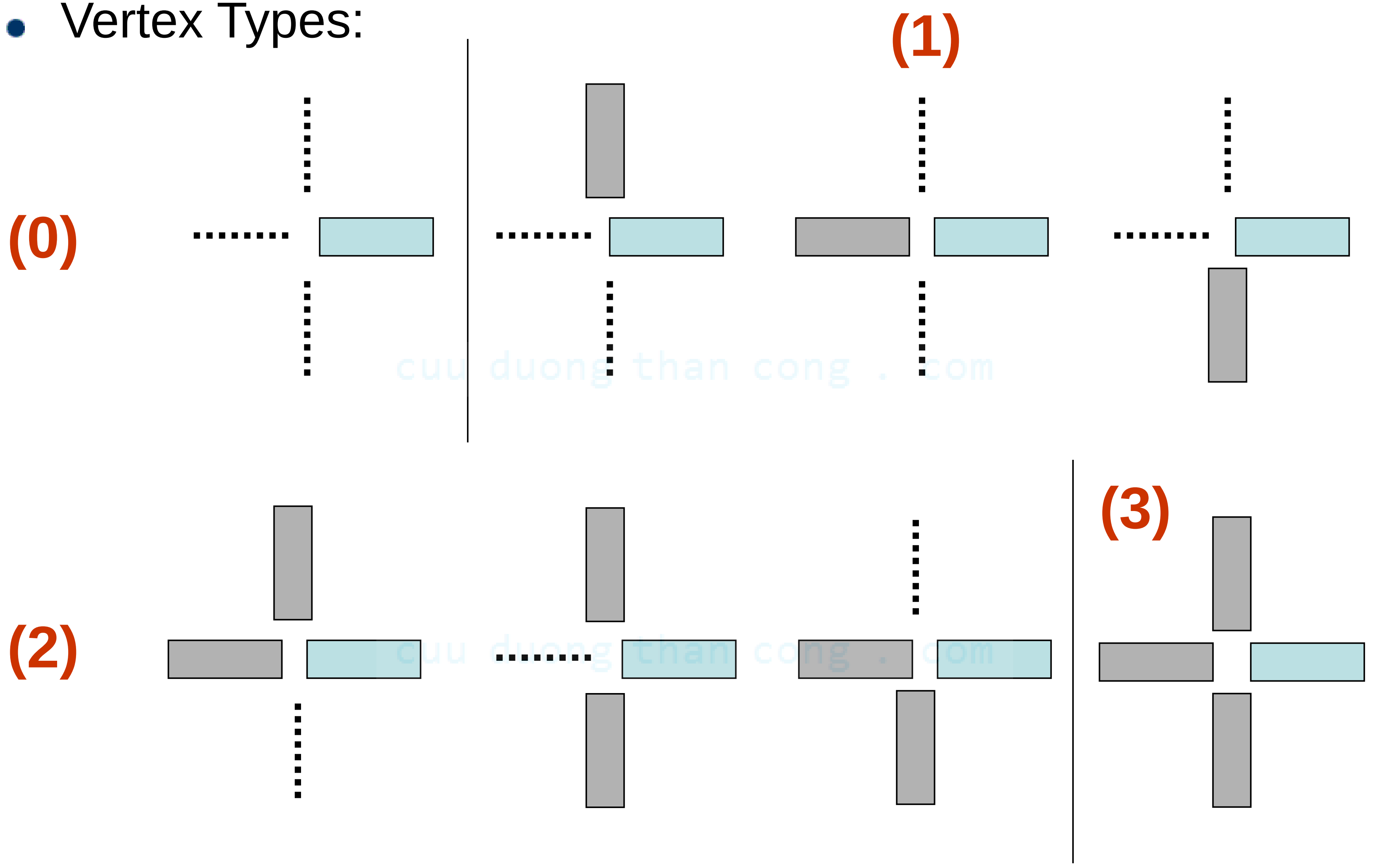
Edge thresholding: Edge relaxation

- Idea: Not simply add pixels if they are next to other edge pixels but to consider the context as well
- Parallel – Iterative method to adjust edge values on the basis of neighboring edges



Edge thresholding: Edge relaxation

- Vertex Types:



Edge thresholding: Edge relaxation

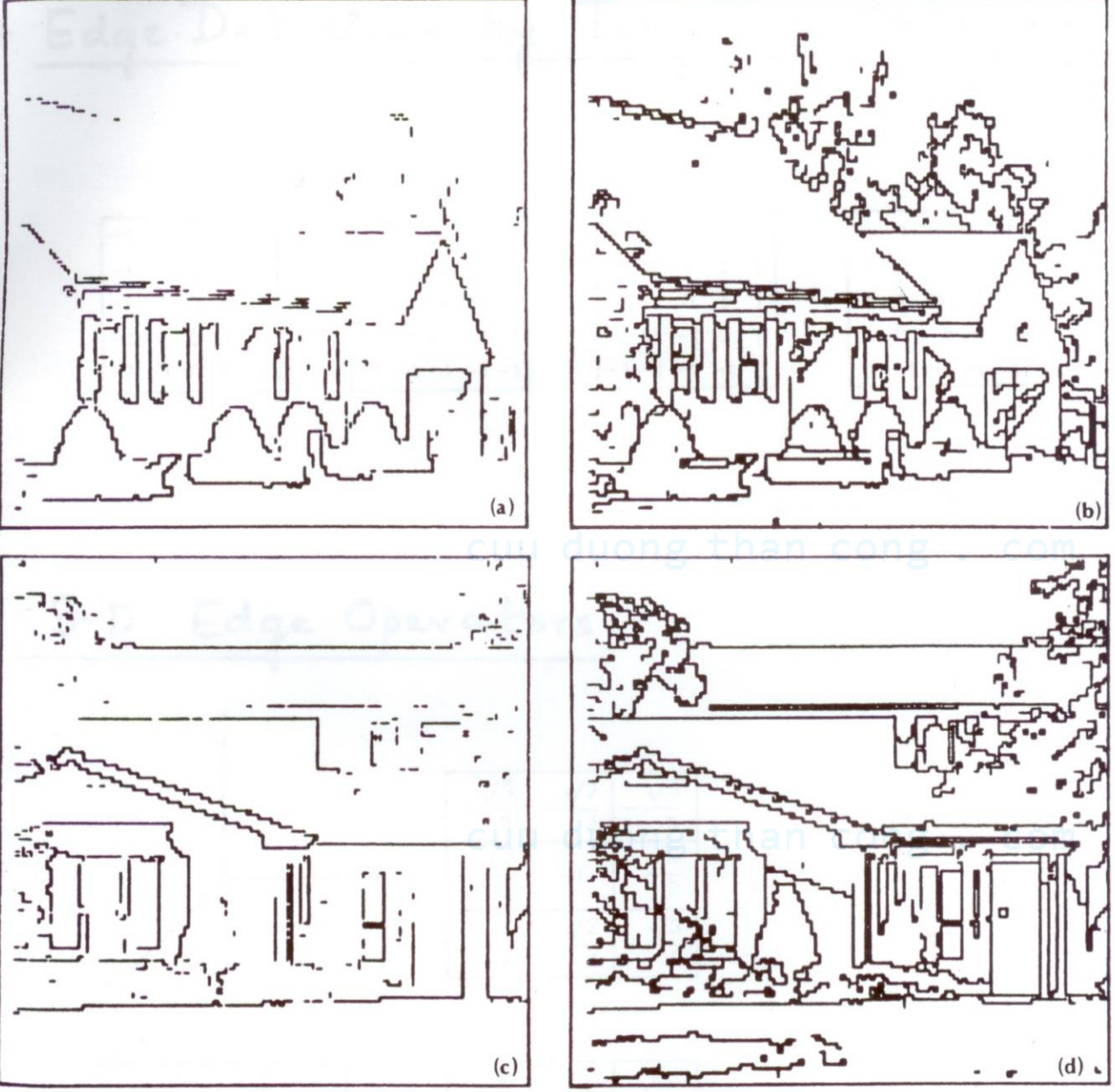
- Action Table:

	Decrement	Increment	Leave as is
Edge Type	0 - 0	1 - 1	0 - 1
	0 - 2	1 - 2	2 - 2
	0 - 3	1 - 3	2 - 3
			3 - 3

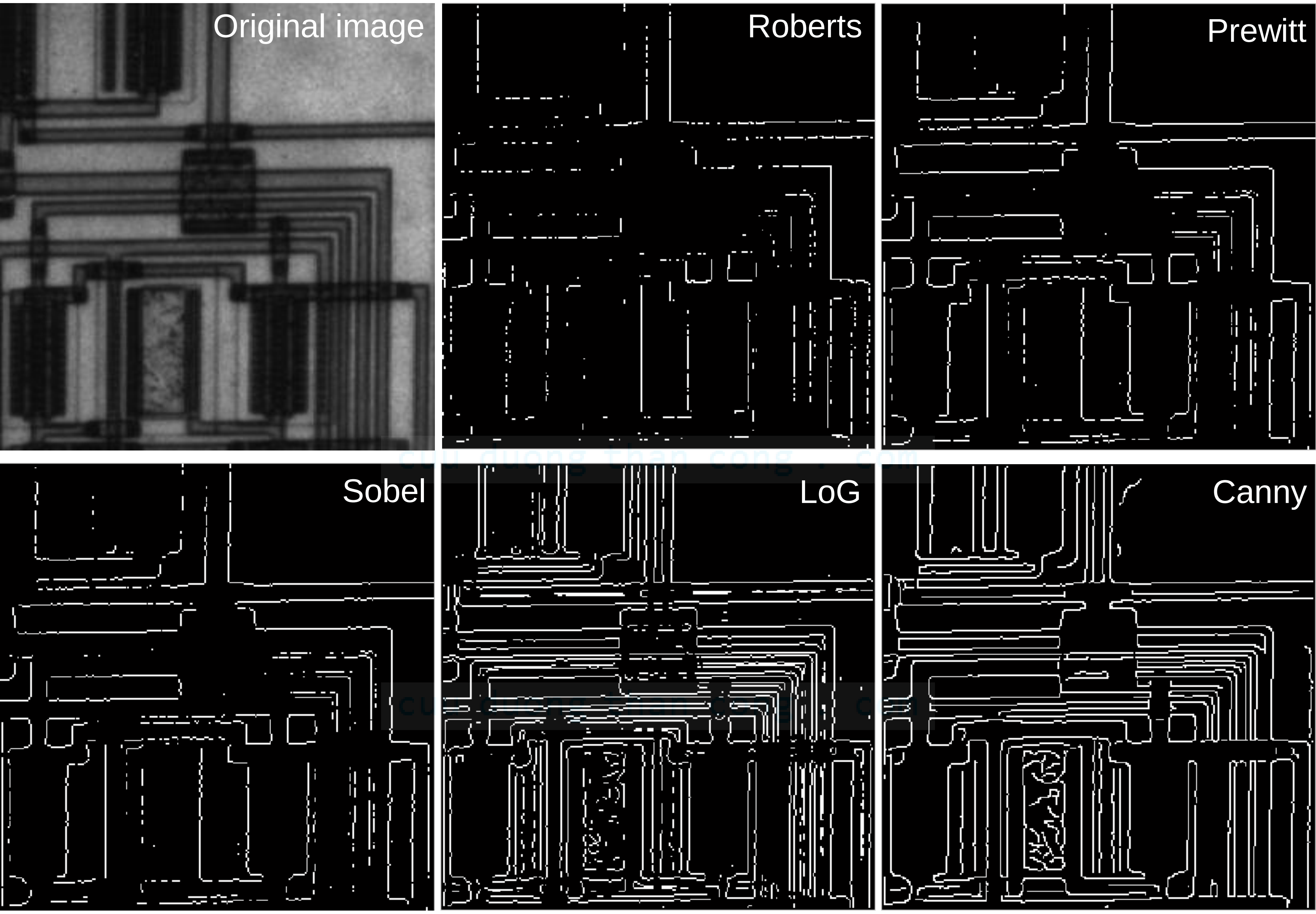
- Algorithm:
  - Step 0: Compute Initial Confidence of edge edge e

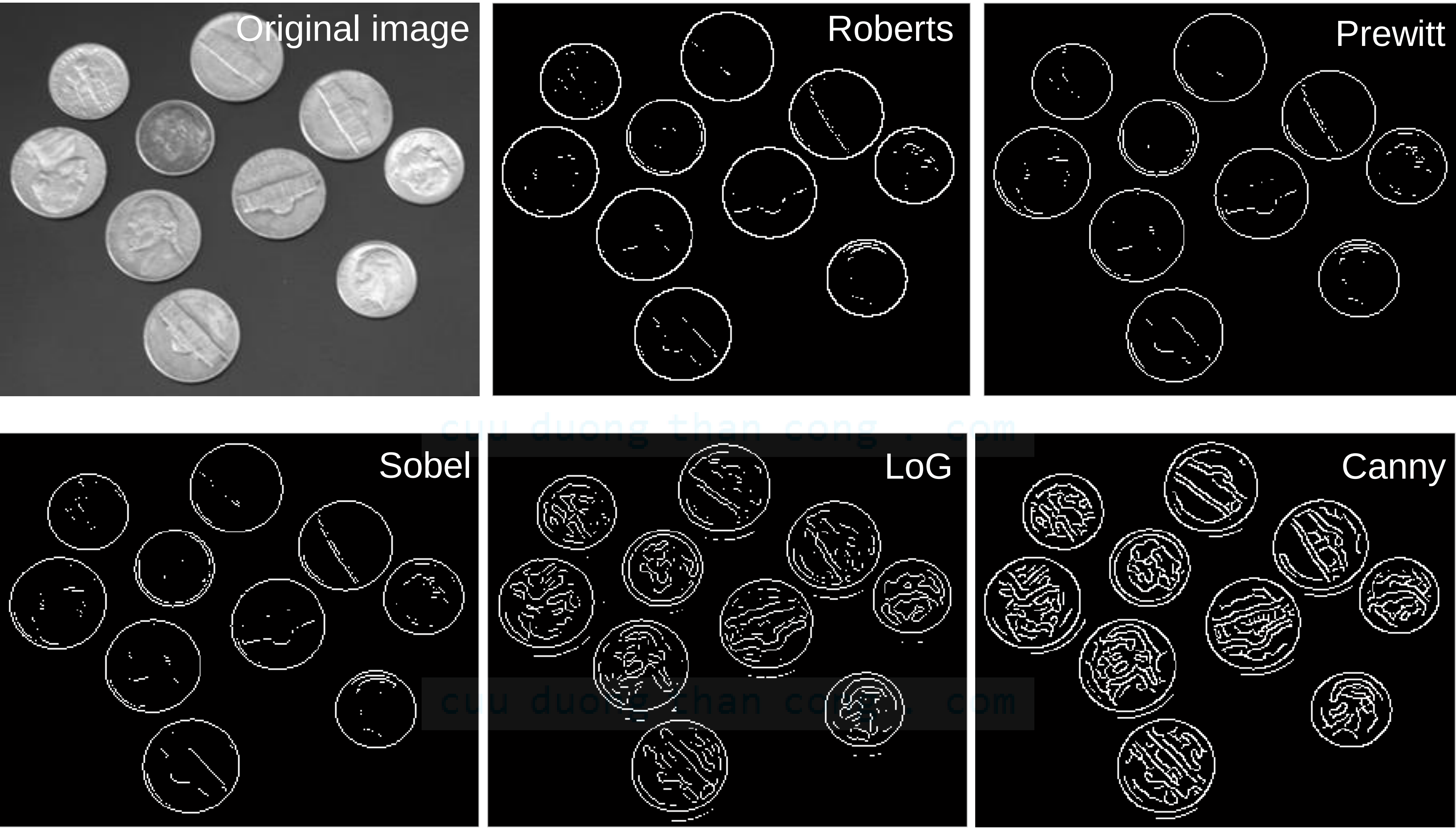
$$C^0(e) = \frac{Magnitude\ of\ e}{Maximum\ Gradient\ in\ image}$$

  - Step 1: Initialize k = 1
  - Step 2: Compute Edge Type of each edge e
  - Step 3: Modify confidence $C^k(e)$ based on $C^{k-1}(e)$ and Edge Type
  - Step 4: Test to see if all $C^k(e)$'s have CONVERGED to either 1 or 0. Else go to Step 2



Edge relaxation results. (a) Raw edge data. Edge strengths have been thresholded at 0.25 for display purposes only. (b) Results after five iterations of relaxation applied to (a). (c) Different version of (a). Edge strengths have been thresholded at 0.25 for display purposes only. (d) Results after five iterations of relaxation applied to (c)


Original image
Roberts
Prewitt
Sobel
LoG
Canny

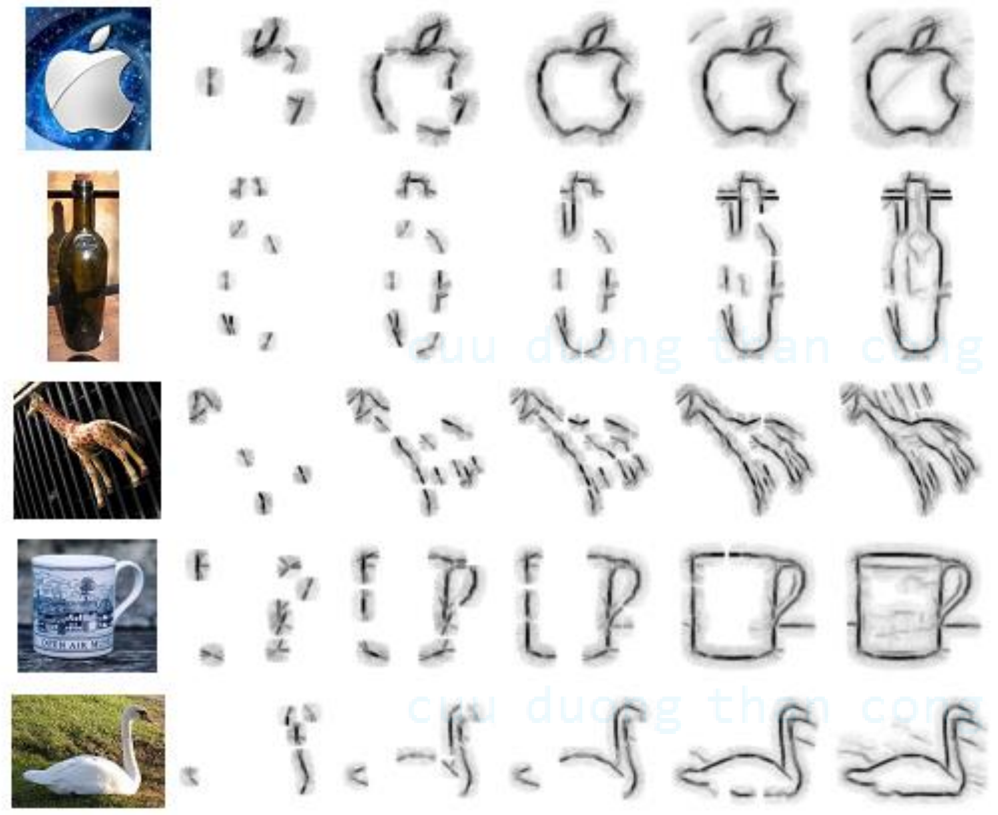

Original image
Roberts
Prewitt
Sobel
LoG
Canny

Section 6.3

APPLICATIONS OF EDGE DETECTION

Edges for object recognition

- Edges can serve as components for object recognition

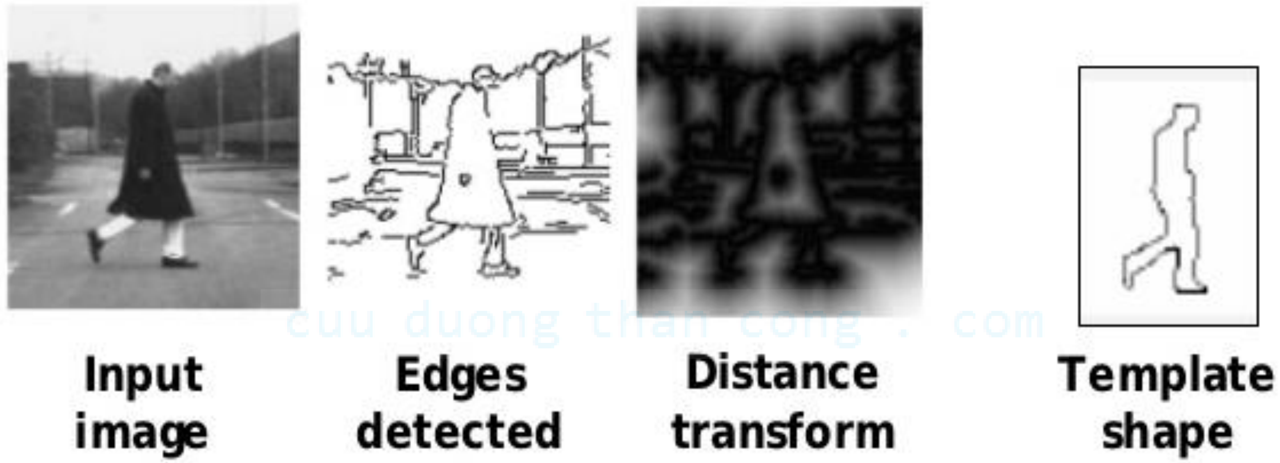


Schlecht, Joseph, and Björn Ommer. "Contour-based object detection." *BMVC*. 2011.

Figure 2: Shape information is concentrated at points of high curvature. The oriented bar features are ordered and rendered by contour strength and amount of curvature. The first column shows images from the ETHZ shape dataset. The second column shows 10 oriented bar features and continues to the right with 30, 50, 100 and 200 features. Notice that with

Edges for template matching

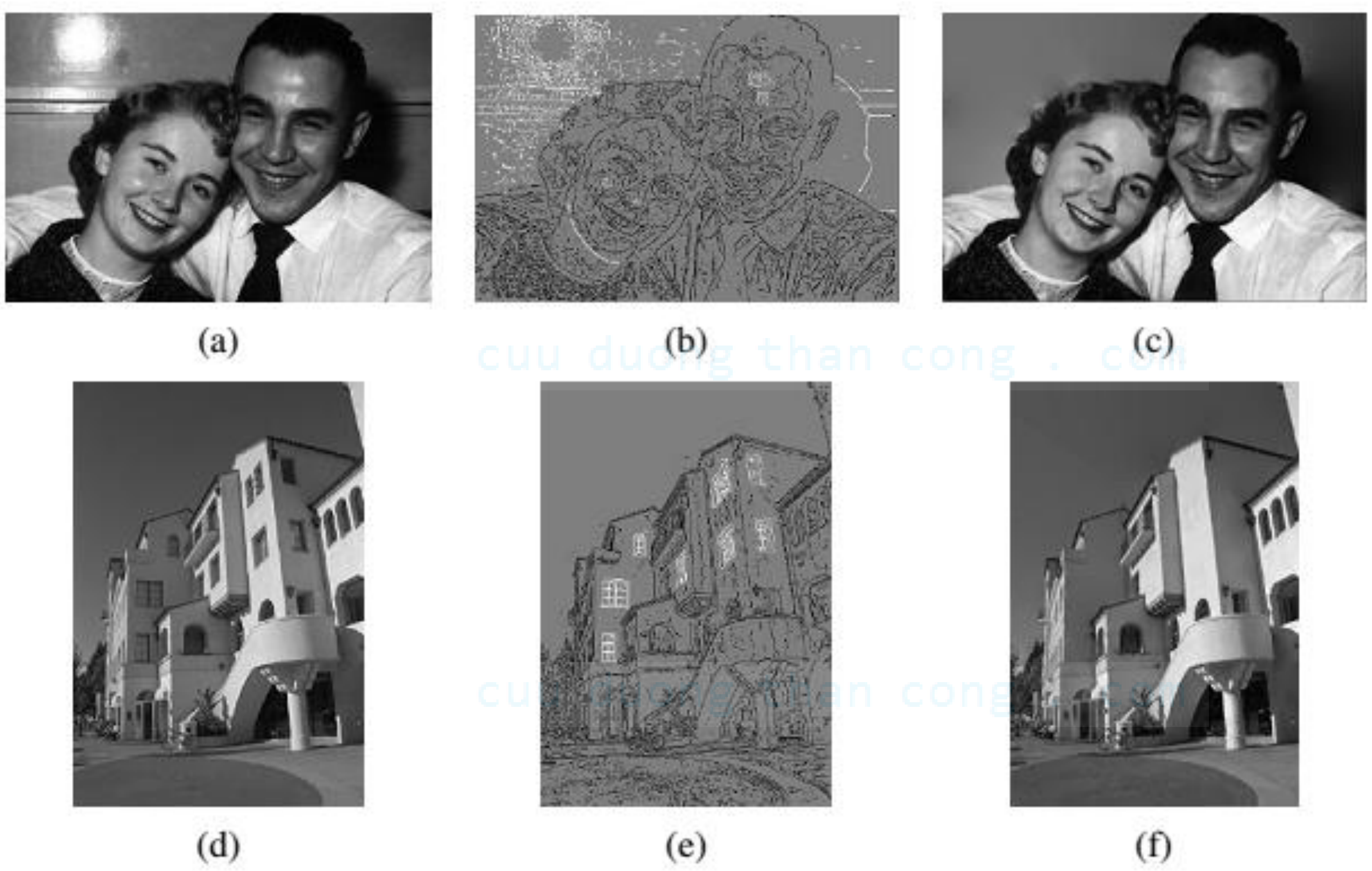
- Edges can serve as features for matching



Gavrila, D.M.; Philomin, V., "Real-time object detection for "smart" vehicles," in *ICCV 1999* , vol.1, pp.87-93, 1999

Edges for image editing

- Edges can also be used directly for image editing



Elder, J. H. and Goldberg, R. M. (2001). Image editing in the contour domain. *PAMI*, 23(3):291–296.

(a) and (d) original images. (b) and (c) extracted edges (edges to be deleted are marked in white). (c) and (f) reconstructed edited images

References

- Rafael C. Gonzalez, Richard E. Woods, “Digital Image Processing”, 3rd edition, 2008. Chapter 10.
- Richard Szeliski, “Computer Vision: Algorithms and Applications”, 2011. Part 4.2.
- Lecture of Edge Detection, Computer Vision Lecture Notes, Carnegie Mellon School of Computer Science, Carnegie Mellon University

  https://www.cs.cmu.edu/afs/cs/academic/class/15385-s06/lectures/ppts/lec-7.ppt
- Tutorials of Canny Edge Operators on Internet
  - E.g. http://justin-liang.com/tutorials/canny/