Concepts in

Edge Detection

Dr. Sukhendu Das
Deptt. of Computer Science and Engg.,
Indian Institute of Technology, Madras
Chennai – 600036, India.

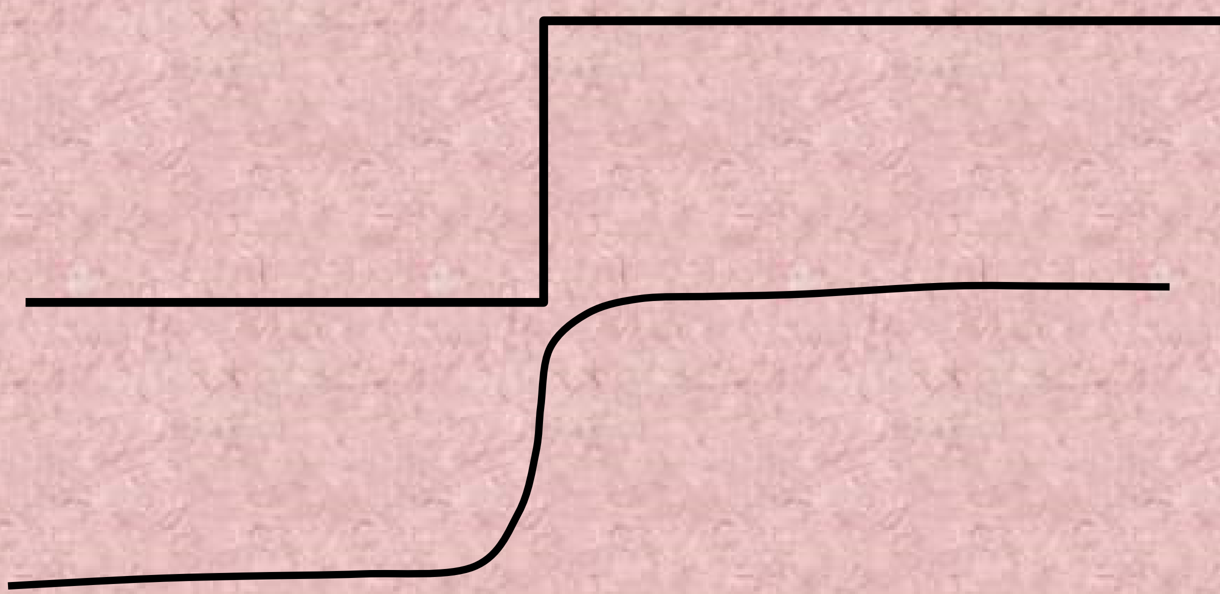
http://www.cs.iitm.ernet.in/~sdas
Email: sdas@iitm.ac.in

Edge Detection

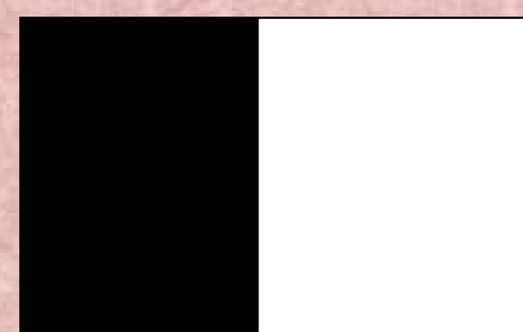
Edge is a boundary between two homogeneous regions. The gray level properties of the two regions on either side of an edge are distinct and exhibit some local uniformity or homogeneity among themselves.

An edge is typically extracted by computing the derivative of the image intensity function. This consists of two parts:

- Magnitude of the derivative: measure of the strength/contrast of the edge
- Direction of the derivative vector: edge orientation



Ideal Step edge in 1-D

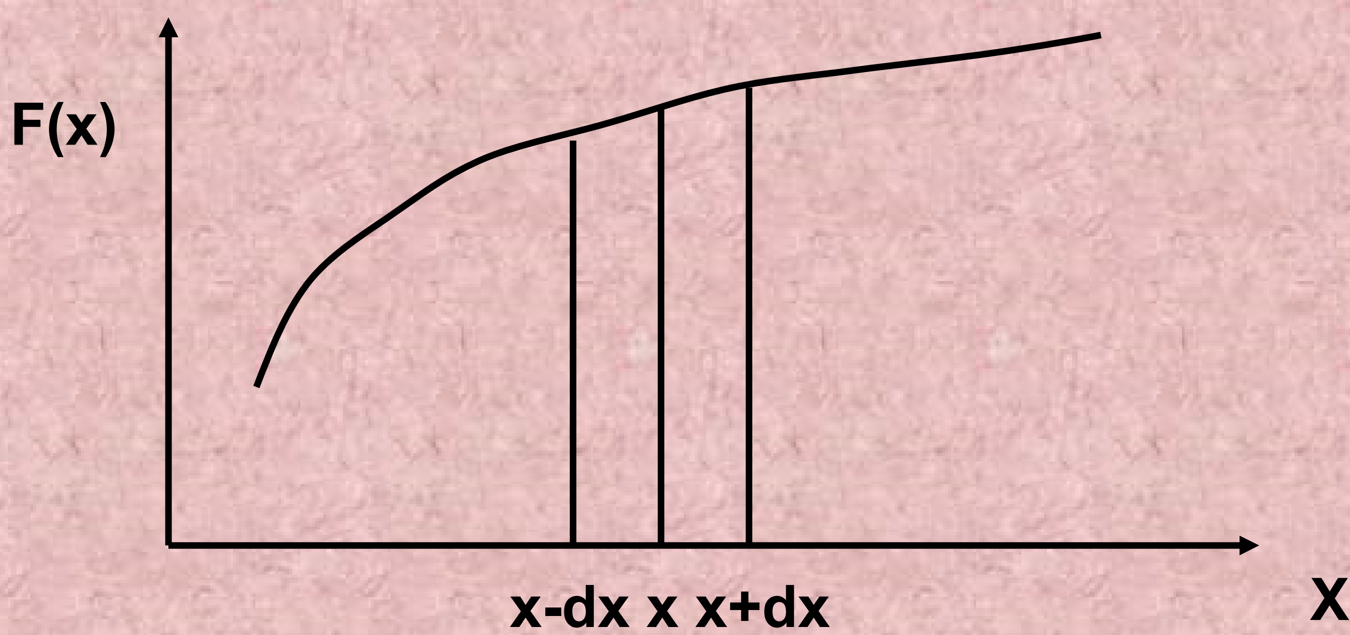


Step edge in 2-D

Computing the derivative: **Finite difference in 1-D**

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

$$\frac{d^2 f}{d^2 x} \approx \frac{f(x+dx)-2f(x)+f(x-dx)}{dx^2}$$



Original Image **Horizontal derivative** **Vertical derivative**

Differentiation using convolution:

$\delta f/\delta x = [-1\ 1]$; $\delta f/\delta y = [-1\ 1]^T$;

$\delta^2 f/\delta x^2 = [1\ -2\ 1]$; $\delta^2 f/\delta y^2 = [1\ -2\ 1]^T$;

Need to use wider masks to add an element of smoothing and better response. The traditional derivative operators used were:

Roberts

$$\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Prewitt

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

Sobel

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

Laplacian

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

-1	-1	-1
-1	8	-1
-1	-1	-1

Laplacian Operator

$$\frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad \frac{1}{8}\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

$$\boldsymbol{S}_x \qquad\qquad \boldsymbol{S}_y$$

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

Sobel

weighted average and scaling

x-derivative

-5	-4	0	4	5
-8	-10	0	10	8
-10	-20	0	20	10
-8	-10	0	10	8
-5	-4	0	4	5

5	8	10	8	5
4	10	20	10	4
0	0	0	0	0
-4	-10	-20	-10	-4
-5	-8	-10	-8	-5

Run filter over image

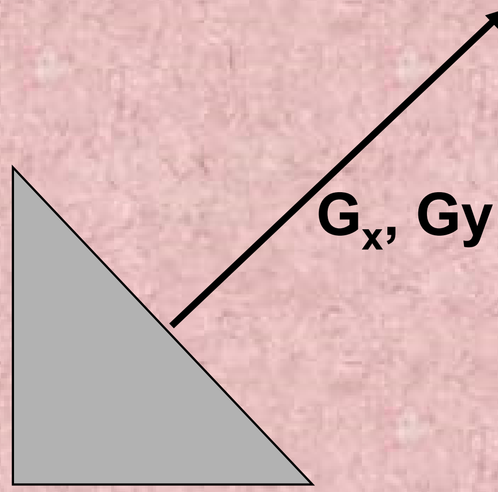
$$\frac{\partial \boldsymbol{f}}{\partial x} = \boldsymbol{S}_x \otimes \boldsymbol{f} \qquad \frac{\partial \boldsymbol{f}}{\partial y} = \boldsymbol{S}_y \otimes \boldsymbol{f}$$

Image gradient

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

Two components of the edge values computed are:

Gradient values: $G_x = \delta f/\delta x$; $Gy = \delta f/\delta y$.



The ***magnitude*** of the edge is calculated as:

$$|G| = [G_x^2 + G_y^2]^{1/2}$$

and ***orientation*** as:

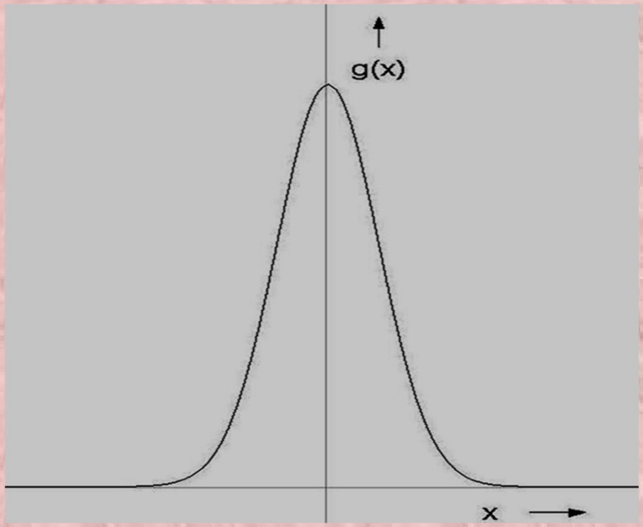
$$\theta = \arctan(G_y/G_x)$$

Most of these partial derivative operators are sensitive to noise. Use of these masks resulted in thick edges or boundaries, in addition to spurious edge pixels due to noise.

Laplacian mask is highly sensitive to spike noise. Use of noise smoothing became mandatory before edge detection, specifically for noisy images. But noise smoothing, typically by the use of a ***Gaussian*** function, caused a blurring or smearing of the edge information or gradient values.

A Gaussian function is shown below. The width of the Gaussian depends on the variance σ. The value of σ dictates the amount of smoothing. The expression of the Gaussian function is given as:

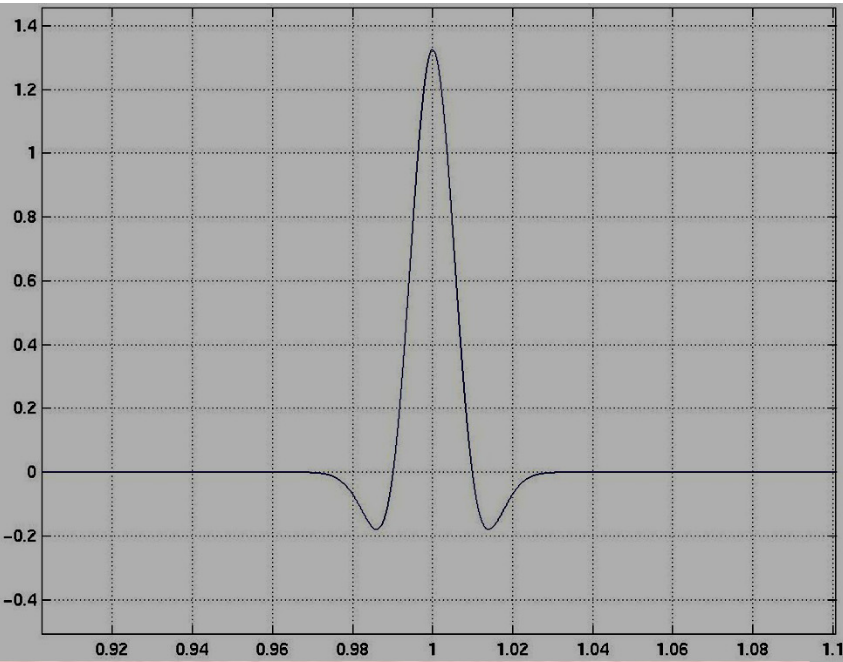
$$g(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2}{2\sigma^2}}$$



Gaussian Function

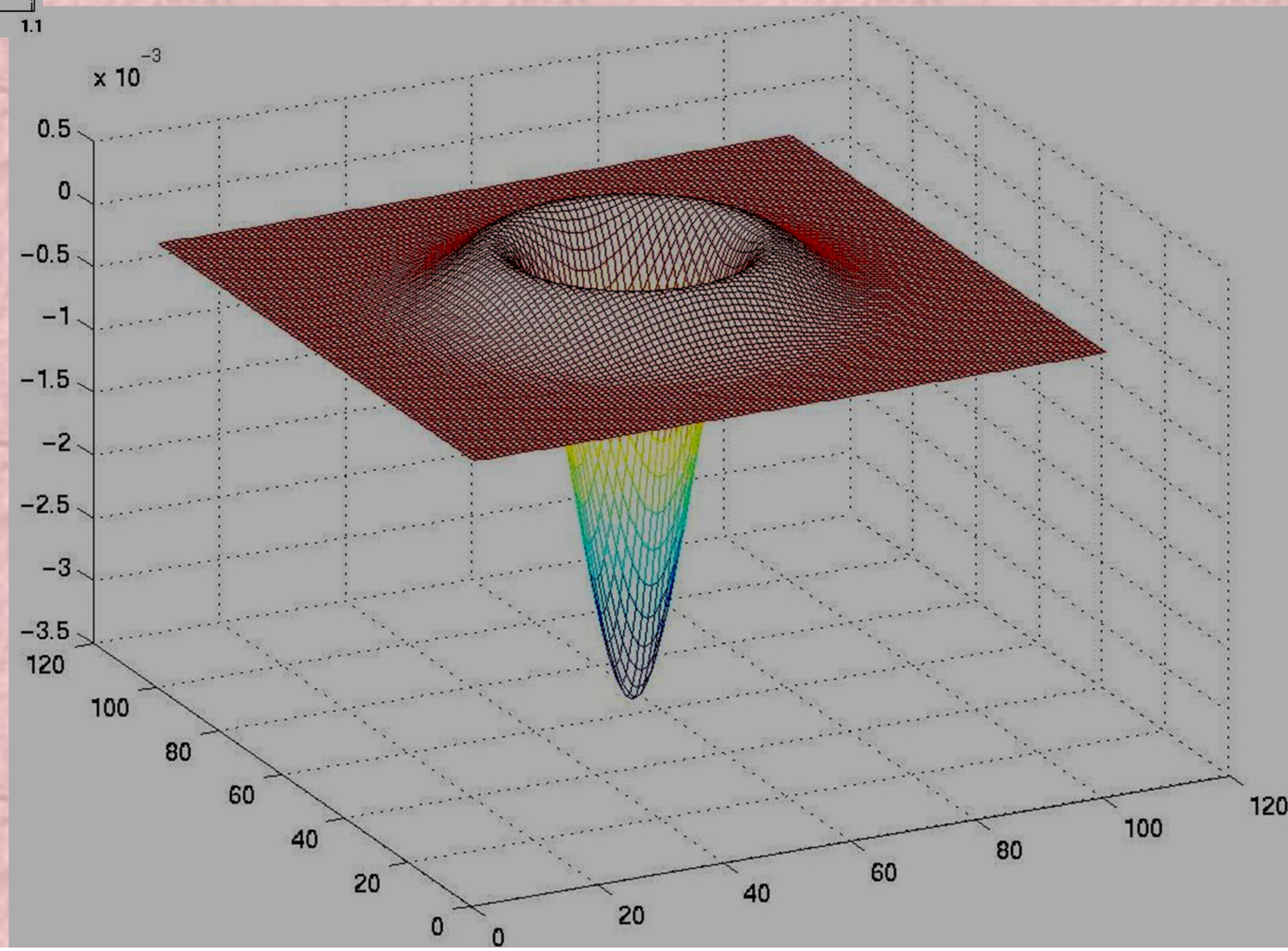
Marr and Hildreth (1980) suggested the use of the "Laplacian of the Gaussian" (LOG) operator to detect edges. This produced edges as Zero-Crossings (ZC's) in the output function - why??

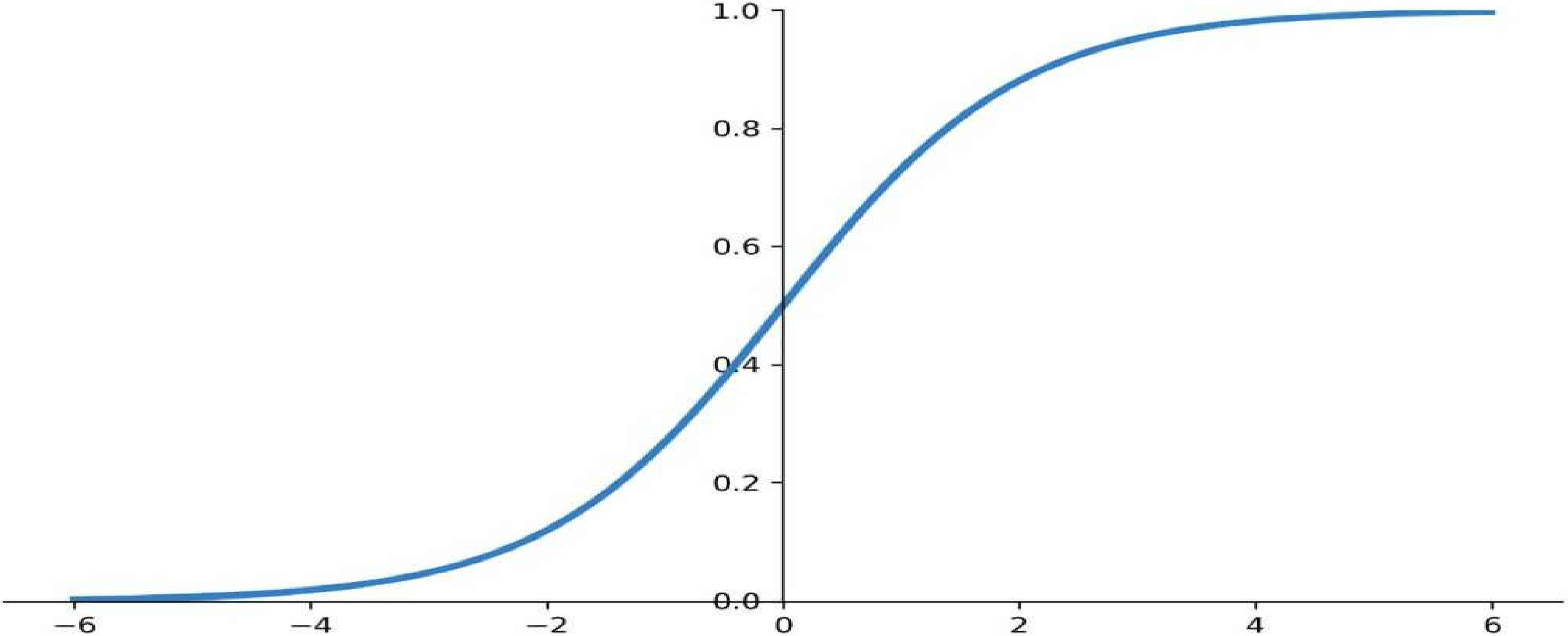
However, it did not give any idea of the gradient magnitude or orientation of the edges. But ZC's were spread through-out an image. How do one detect true edges from ZC's??

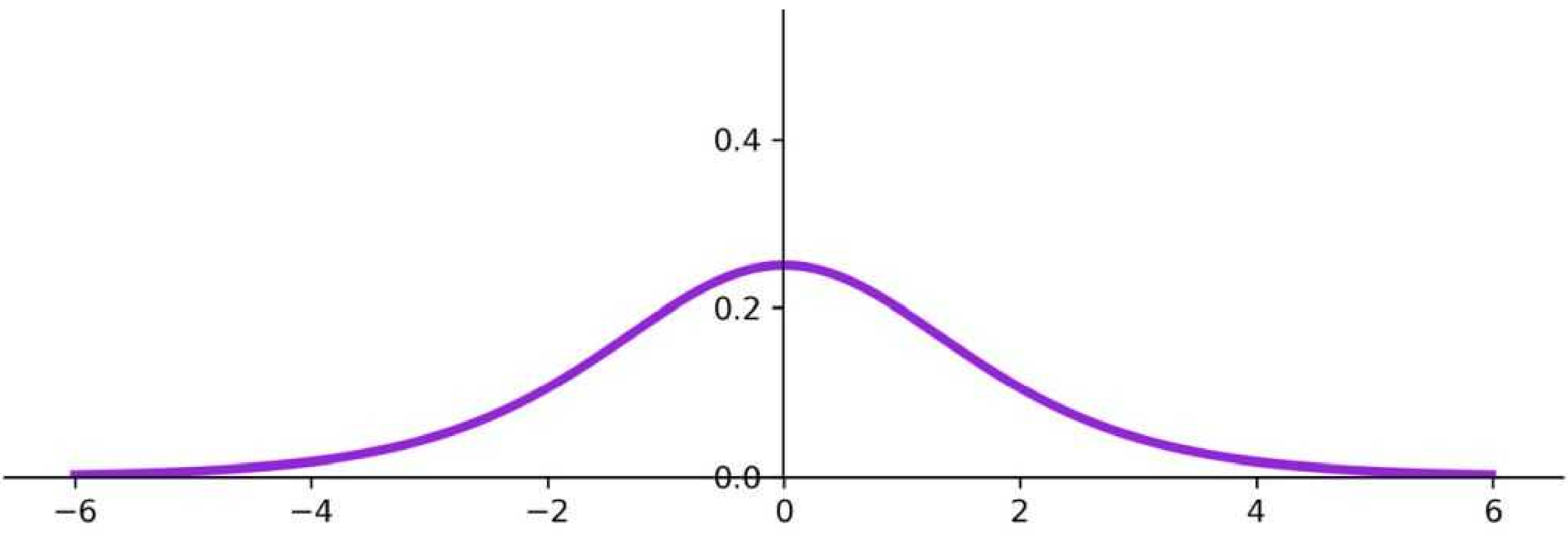


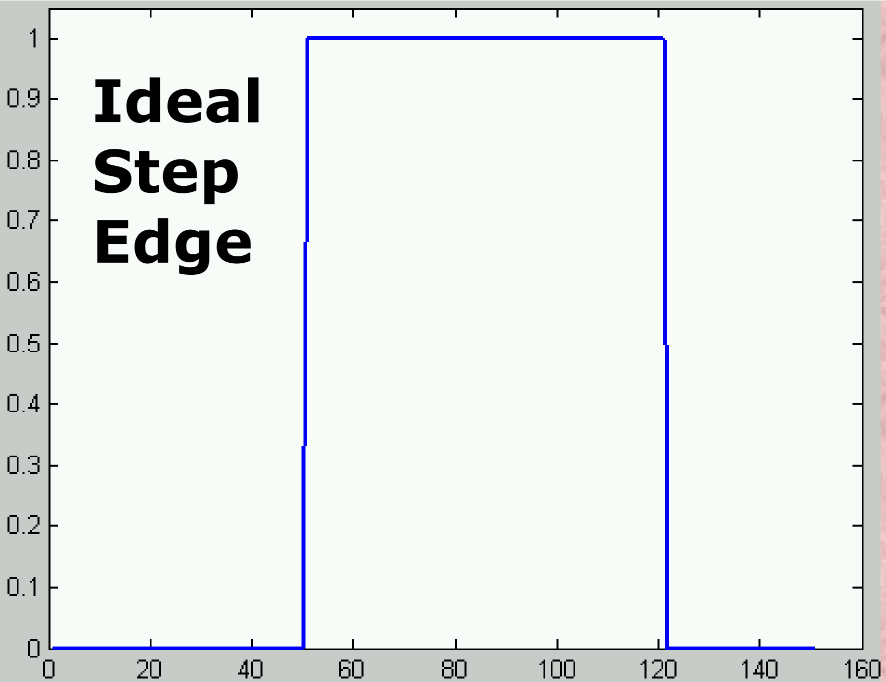
LOG operator in 2-D

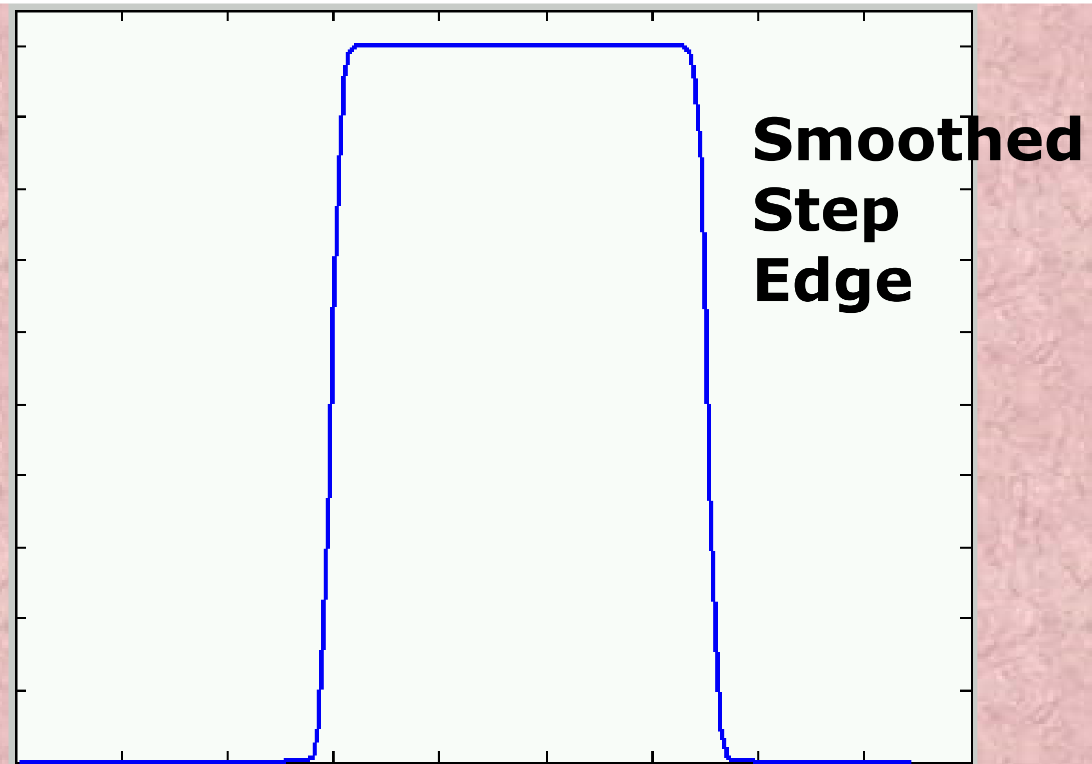
LOG operator in 1-D

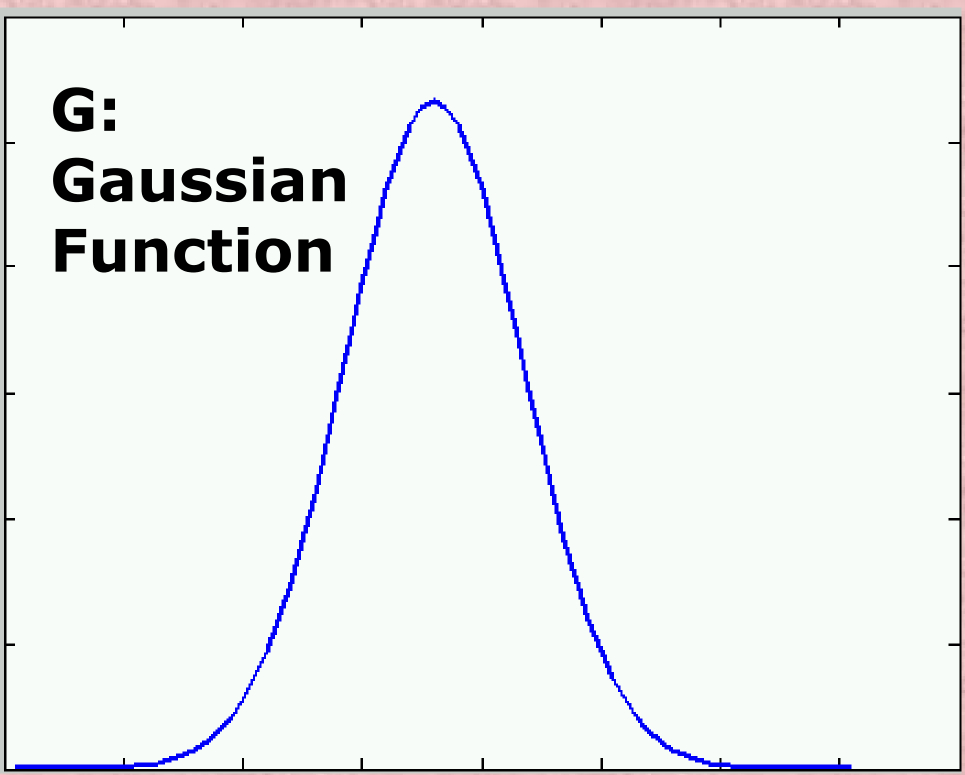


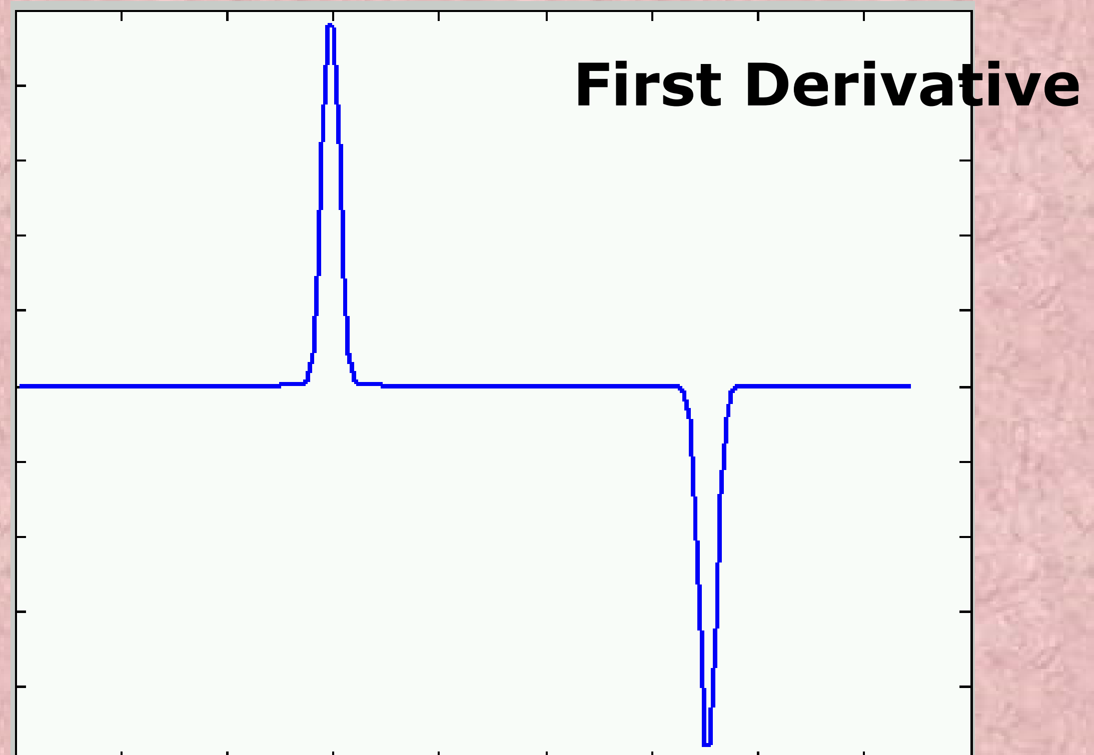

1.0
0.8
0.6
0.4
0.2
0.0
−6
−4
−2
0
2
4
6


0.4
0.2
0.0
−6
−4
−2
0
2
4
6


Ideal
Step
Edge
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
20
40
60
80
100
120
140
160


Smoothed
Step
Edge

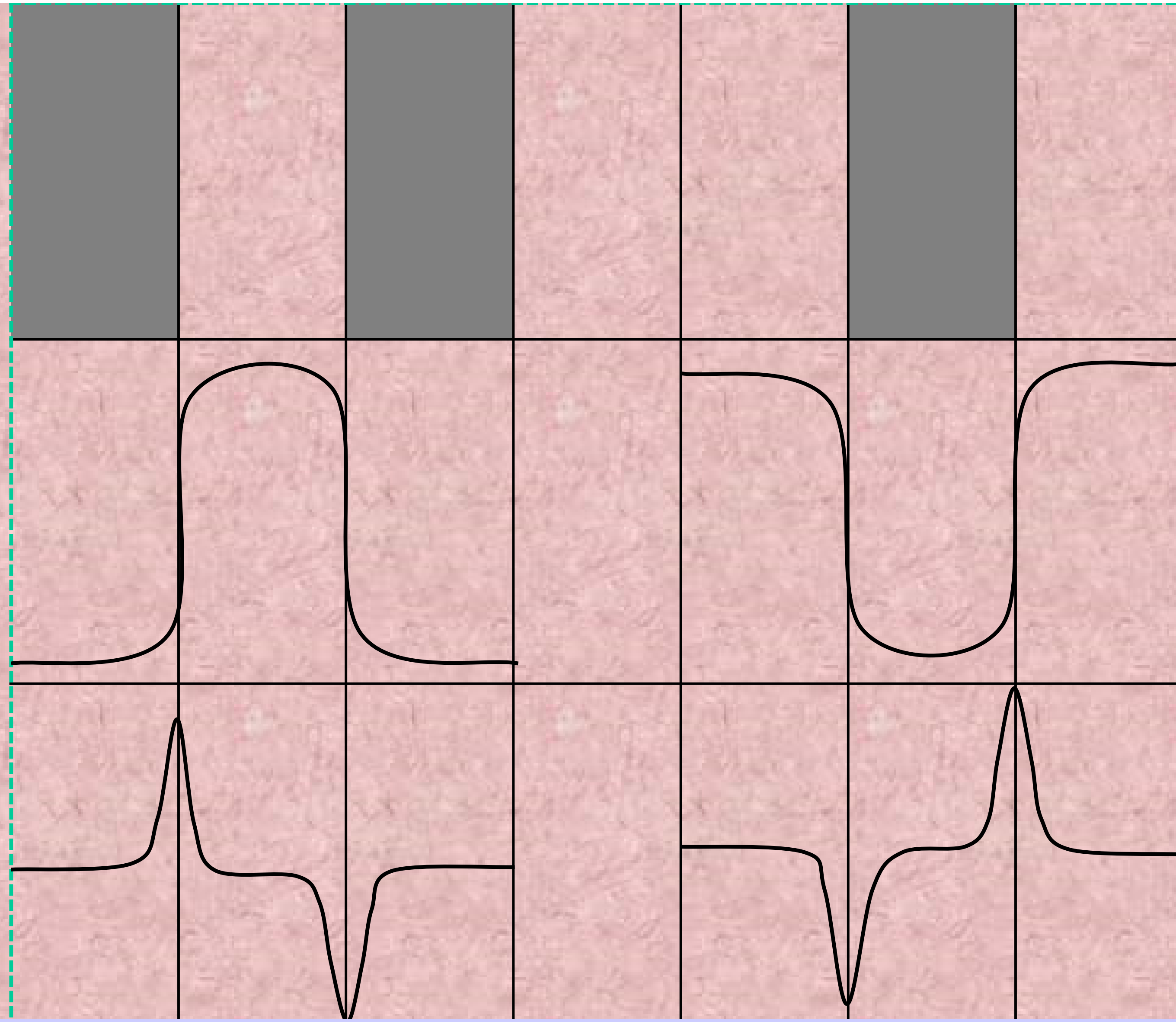

G:
Gaussian
Function


First Derivative

Images

Horizontal Intensity Profiles

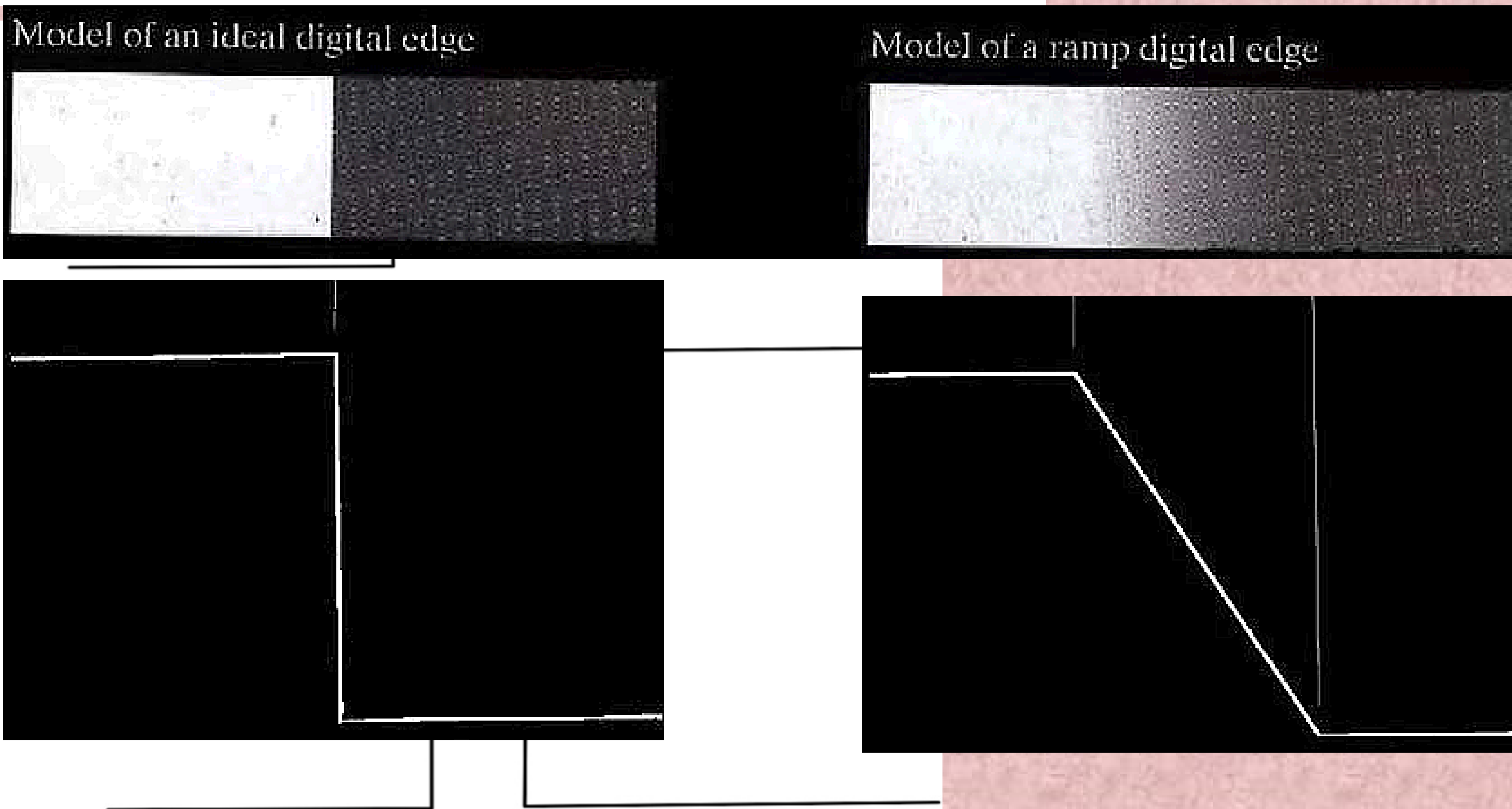
First Derivative



One-dimensional edge profiles

STEP

RAMP

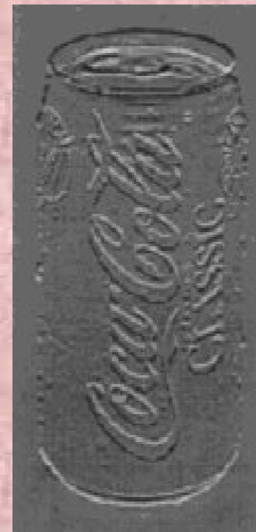


LINE

ROOF

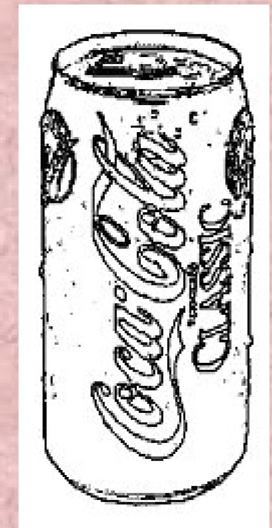
Original Gray scale Image

X-gradient



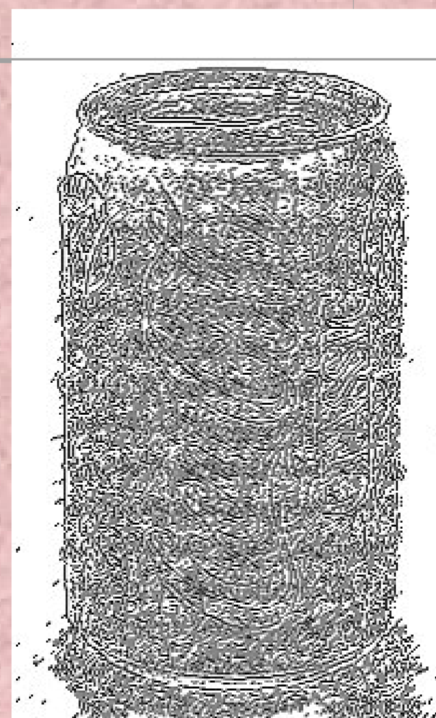
Y-gradient

Total Gradient Magnitude



Bi level Thresholding

Original Grey level Image

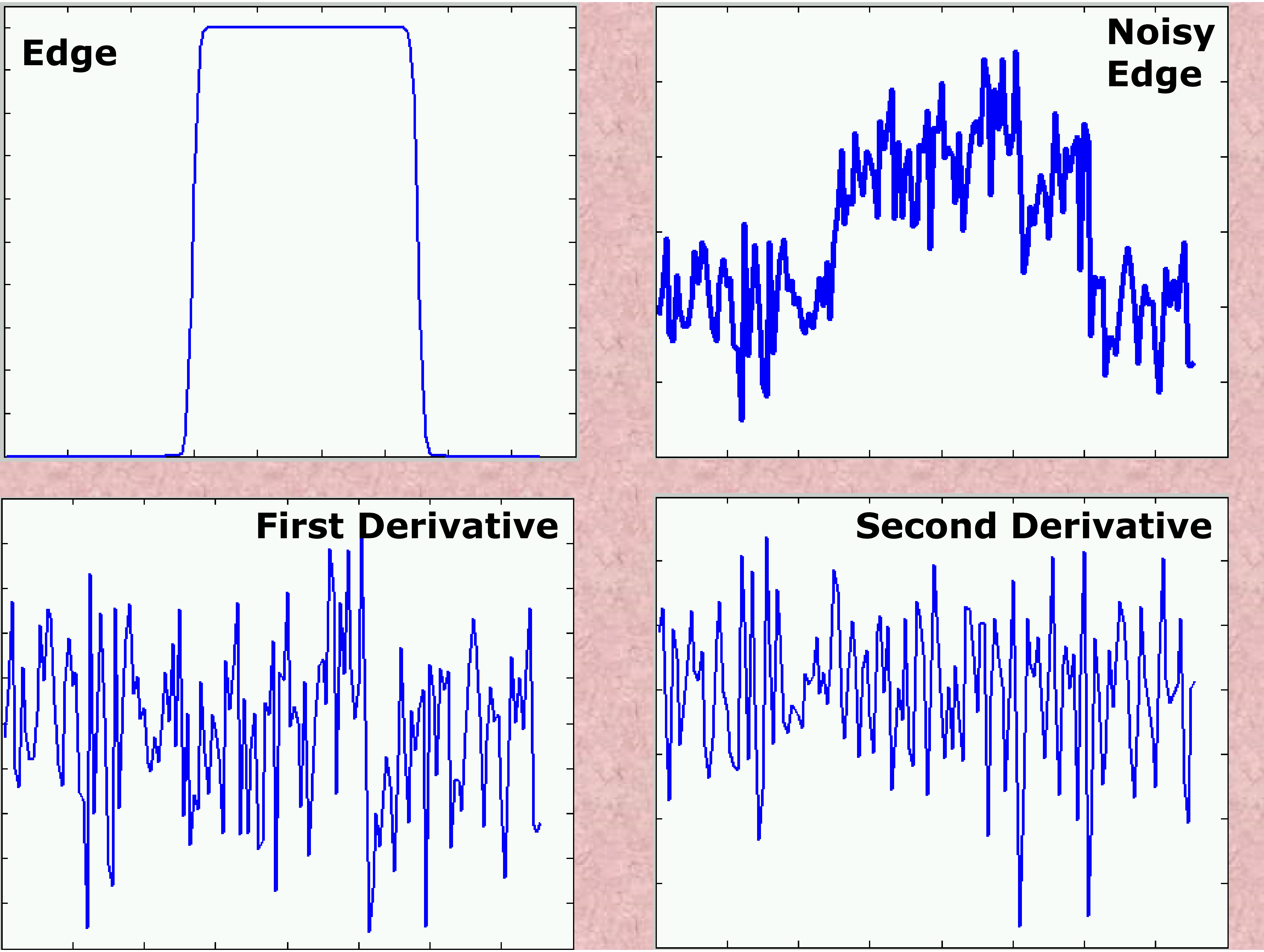


After Laplace Operator

After Zero-crossing

δF

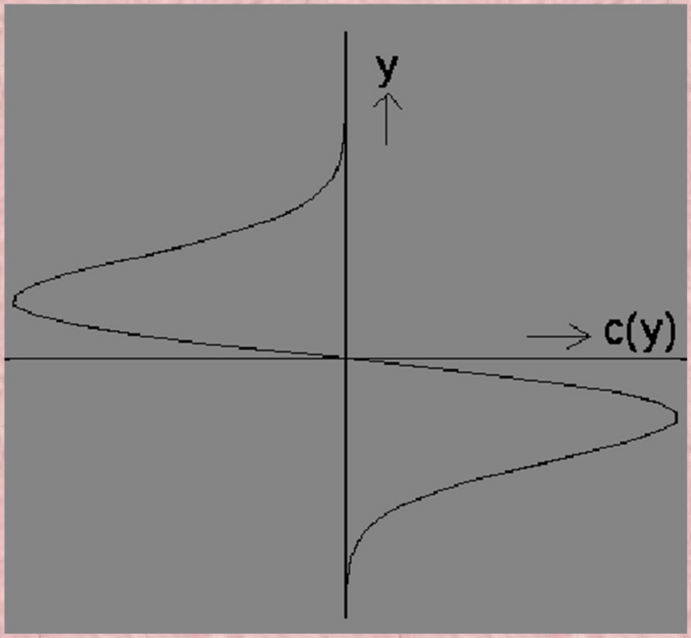
$\delta^2 F$


Edge
Noisy Edge
First Derivative
Second Derivative

Canny in 1986 suggested an optimal operator, which uses the Gaussian smoothing and the derivative function together. He proved that the first derivative of the Gaussian function, as shown below, is a good approximation to his optimal operator.

It combines both the derivative and smoothing properties in a nice way to obtain good edges. Canny also talks of a hysteresis based thresholding strategy for marking the edges from the gradient values.

Smoothing and derivative when applied separately, were not producing good results under noisy conditions. This is because, one opposes the other. Whereas, when combined together produces the desired output.



Expression of Canny (1-D operator is):

$$c(y) = g'(y) = \left(\frac{-y}{\sqrt{2\pi}\sigma^3}\right)\exp\left(\frac{-y^2}{2\sigma^2}\right)$$

Canny's algorithm for edge detection:

Detect an edge, where simultaneously the following conditions are satisfied:

$\nabla^2 G*f = 0$ and
$\nabla G*f$ reaches a maximum.

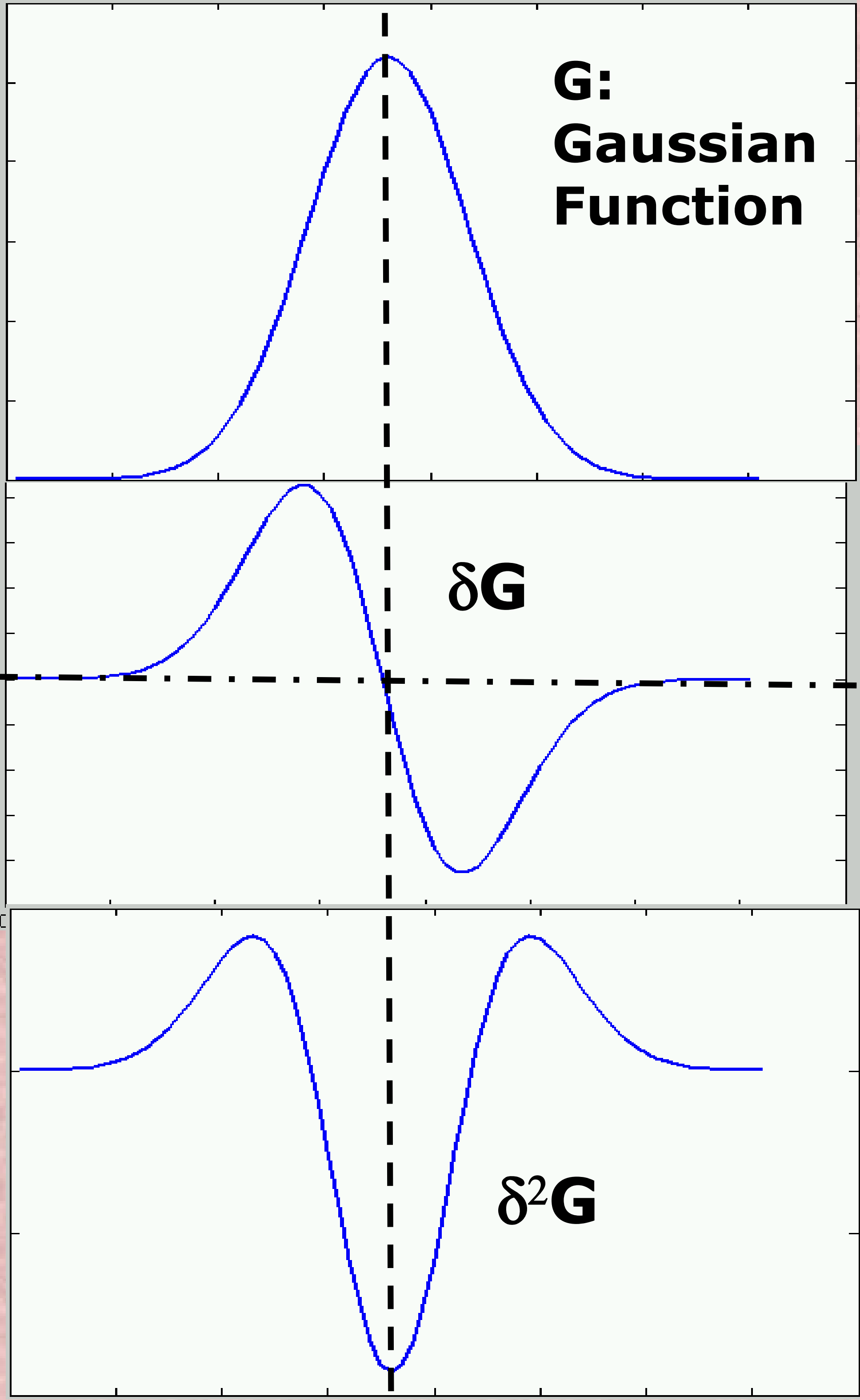
∇G is the first derivative of the Gausian defined (in 1-D) as:

$$\nabla G(x) = \frac{-x}{\sqrt{2\pi}\sigma_2^3} \exp(-\frac{x^2}{2\sigma_2^2})$$

and

$\nabla\, ^2G$ in two-dimension is given by (also known as the *"Laplacian of the Gaussian"* or *LOG operator*):

$$\nabla^2 G(r) = (1/\pi\sigma^4)(r^2/2\,\sigma^2 - 1)\exp(\frac{-r^2}{2\sigma^2})$$



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

$$\nabla G(x)=$$

$$\frac{-x}{\sqrt{2\pi}\sigma_2^3}\exp\left(-\frac{x^2}{2\sigma_2^2}\right)$$

$$\nabla^2 G(x)=$$

$$\frac{-[\left(x/\sigma\right)^2-1]}{\sqrt{2\pi}\sigma_2^3}\exp\left(-\frac{x^2}{2\sigma_2^2}\right)$$

1. Detection:

The probability of detecting real edge points should be maximized while the probability of falsely detecting non-edge points should be minimized. This corresponds to maximizing the signal-to-noise ratio (SNR).

(Detection of edge with low error rate, which means that the detection should accurately catch as many edges shown in the image as possible).

2. Localization:

The detected edges should be as close as possible to the real edges.

(The edge point detected from the operator should accurately localize on the center of the edge).

3. Number of responses:

Minimize the number of local maxima around the true edge;

One real edge should not result in more than one detected edge

(a given edge in the image should only be marked once, and where possible, image noise should not create false edges).

A Computational Approach to Edge Detection

JOHN CANNY, MEMBER, IEEE

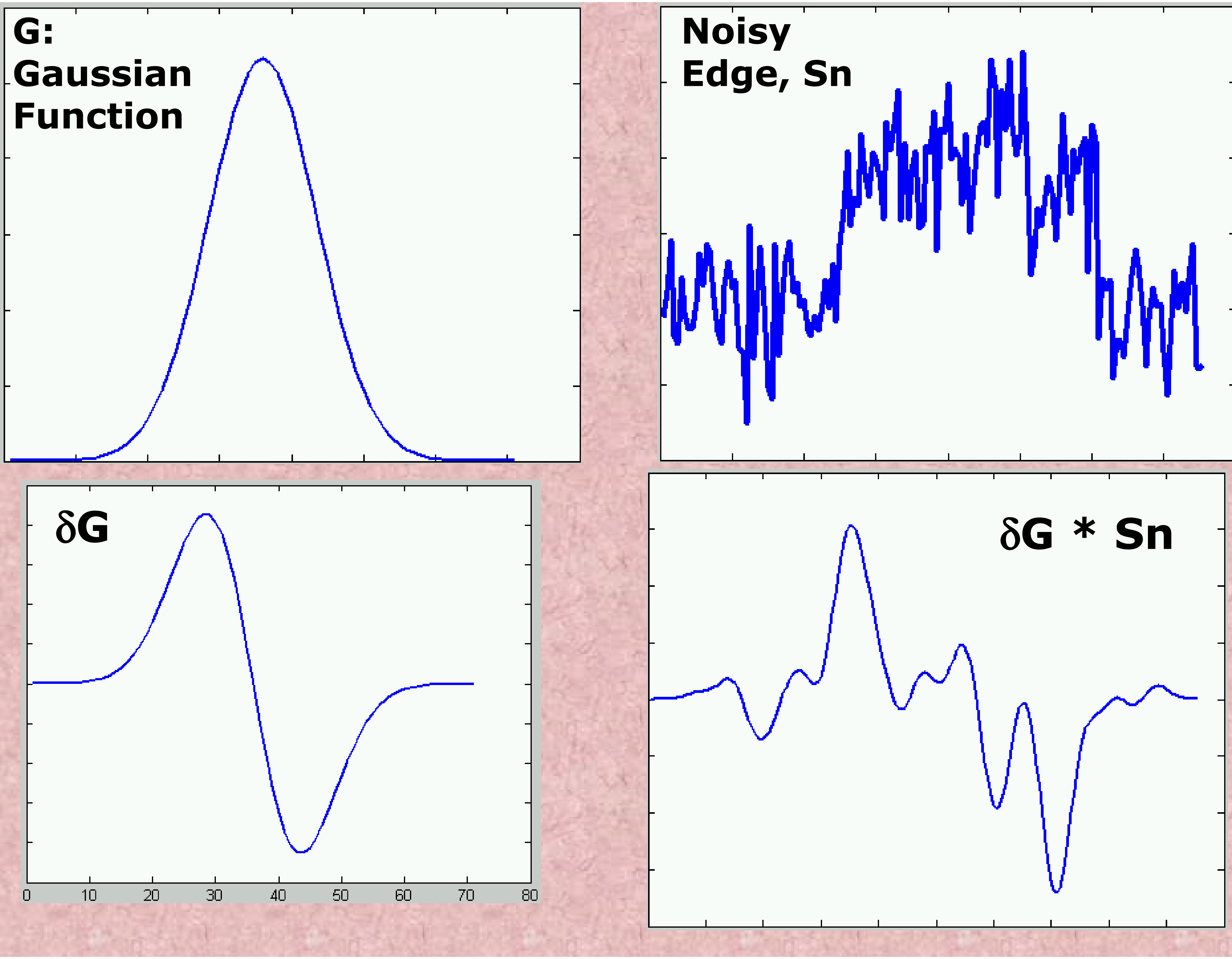
The general solution in the range $[-W, 0]$ may now be written

$$f(x) = a_1 e^{\alpha x} \sin \omega x + a_2 e^{\alpha x} \cos \omega x + a_3 e^{-\alpha x} \cdot \sin \omega x + a_4 e^{-\alpha x} \cos \omega x + c. \quad (35)$$

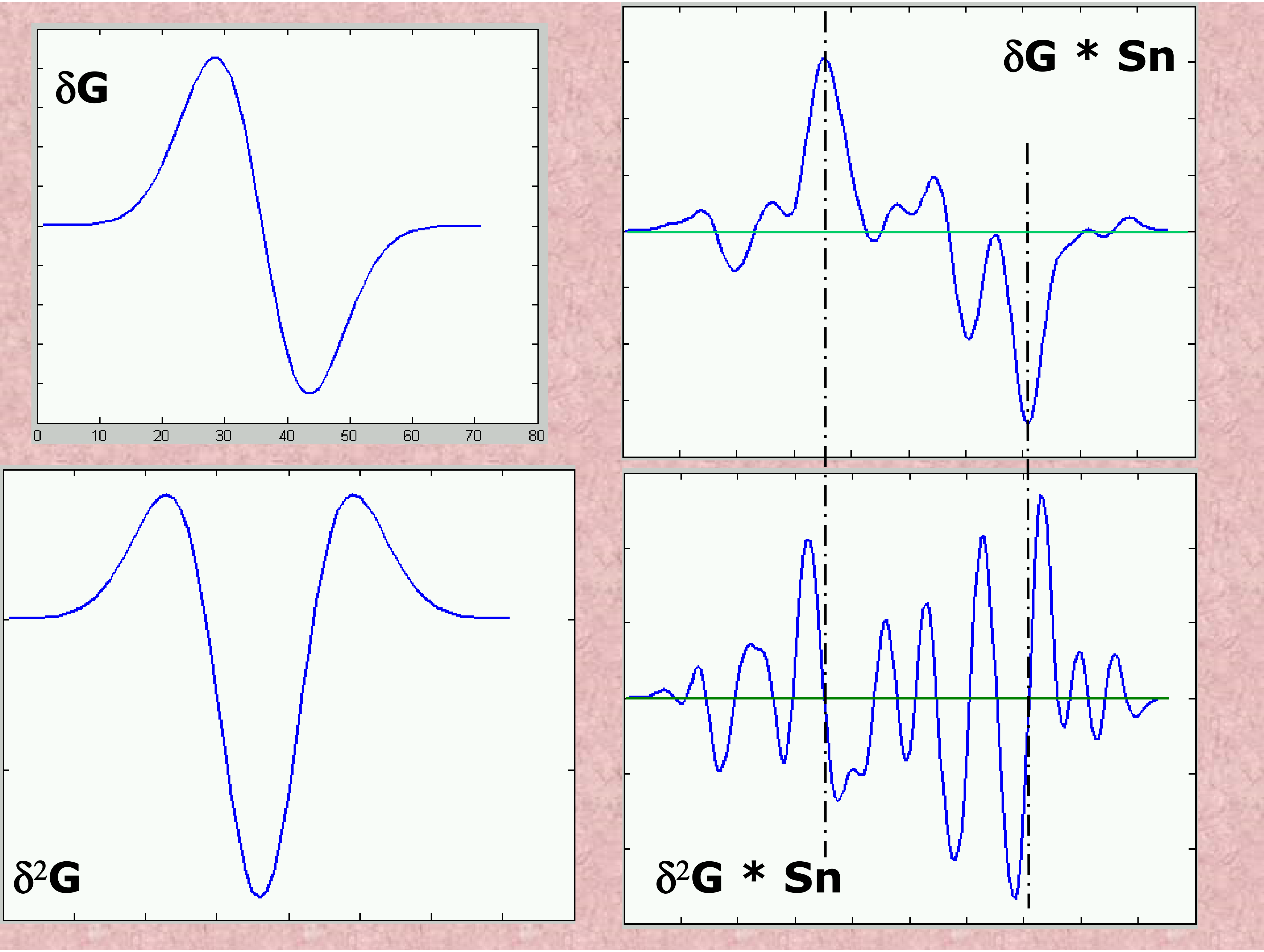
This function is subject to the boundary conditions

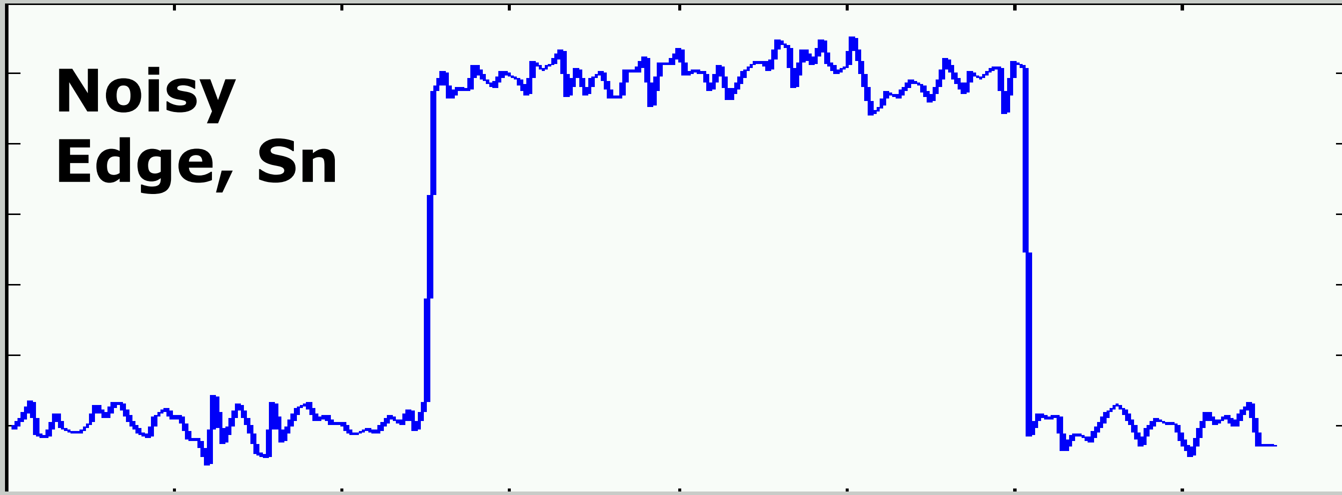
$$f(0) = 0 \qquad f(-W) = 0 \qquad f'(0) = s \qquad f'(-W) = 0$$

where s is an unknown constant equal to the slope of the function f at the origin. Since $f(x)$ is asymmetric, we can extend the above definition to the range $[-W, W]$ using $f(-x) = -f(x)$. The four boundary conditions enable us to solve for the quantities a_1 through a_4 in terms of the

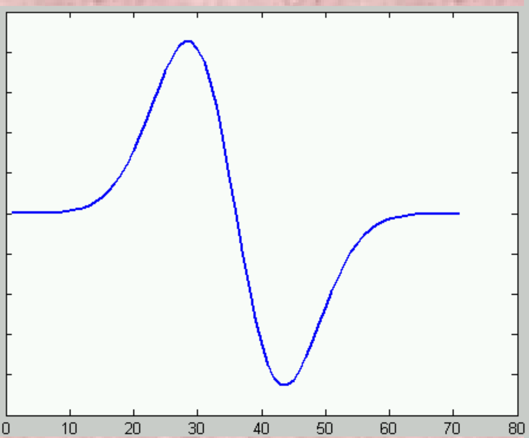


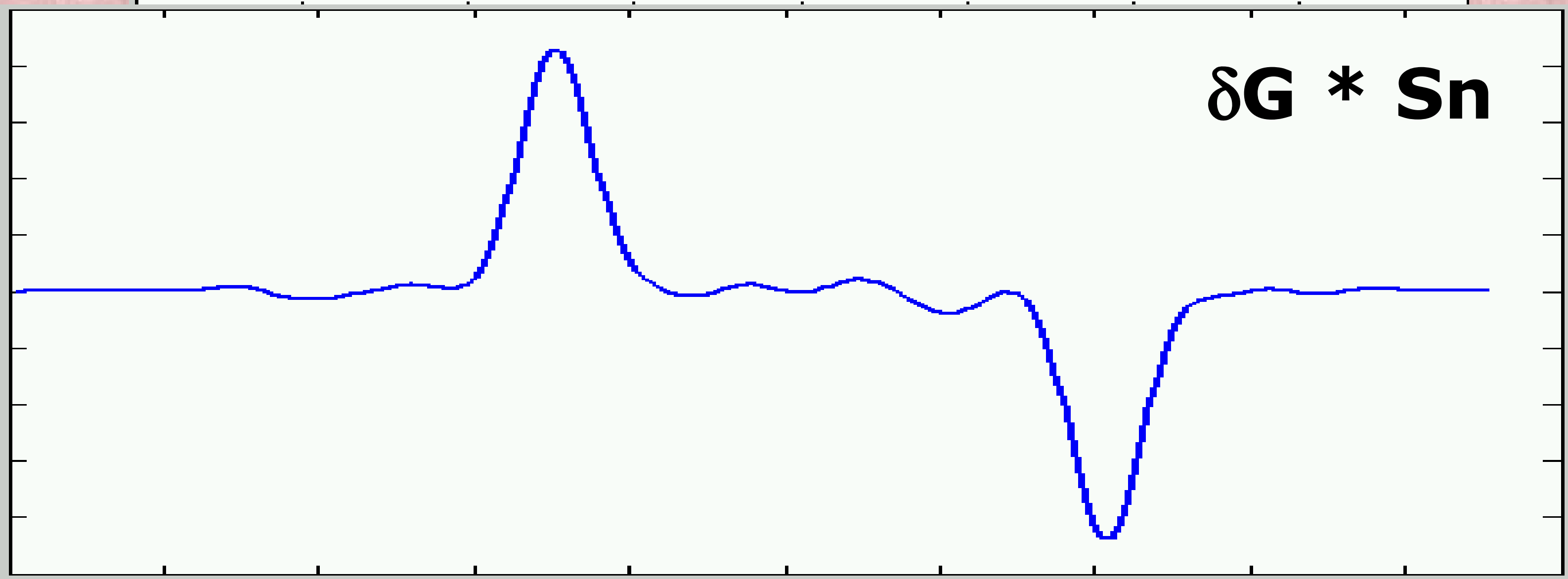
G:
Gaussian
Function
Noisy
Edge, Sn
δG
0
10
20
30
40
50
60
70
80
δG * Sn


δG
0 10 20 30 40 50 60 70 80
δG * Sn
δ²G
δ²G * Sn

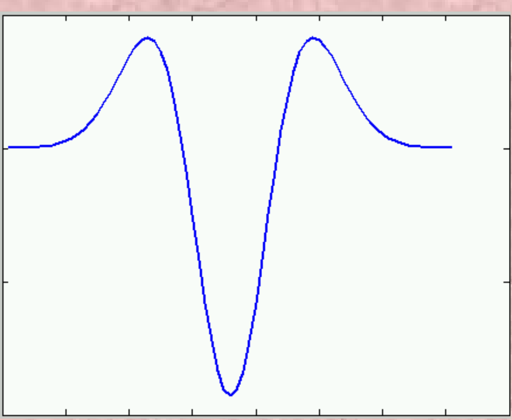


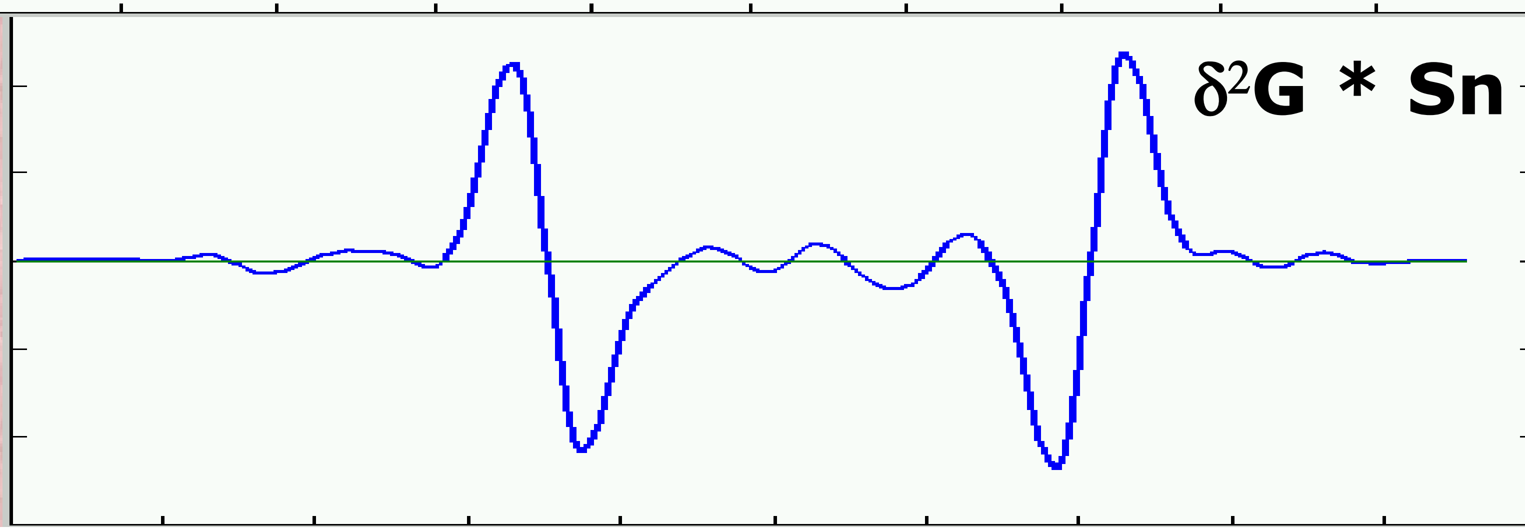
δG





$\delta^2 G$





Lena

Sobel

LOG

Canny

Venice

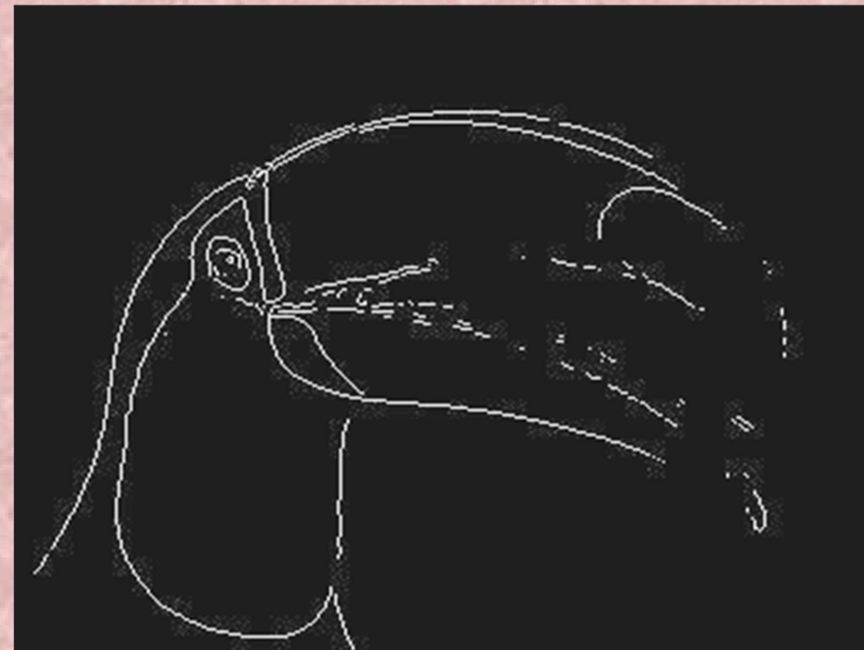
Sobel

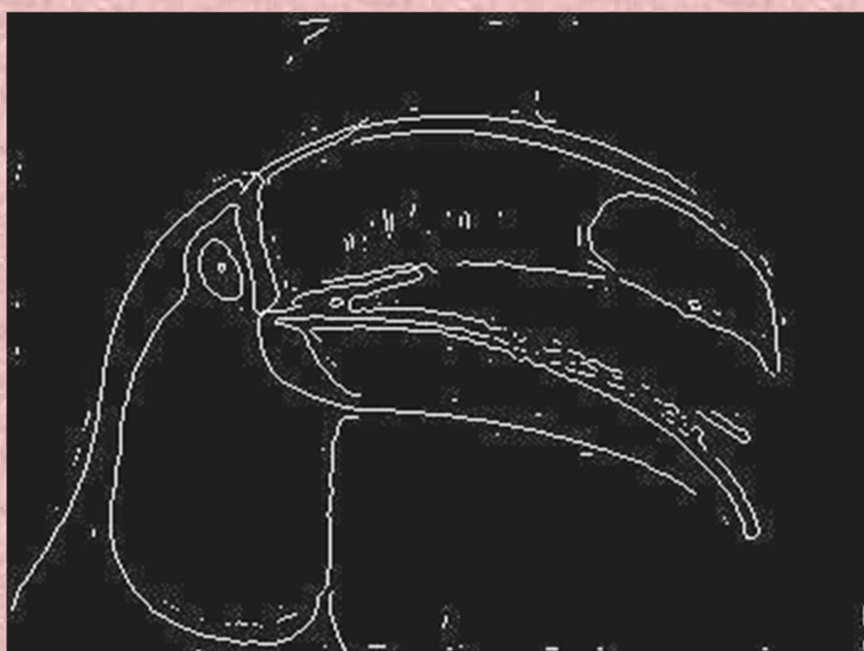
LOG

Canny

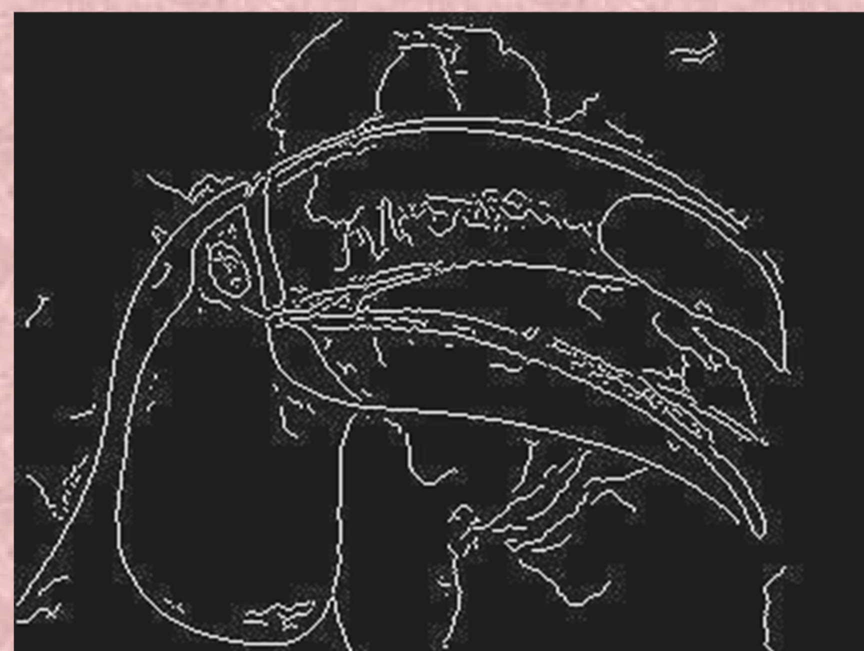
BIRD

SOBEL

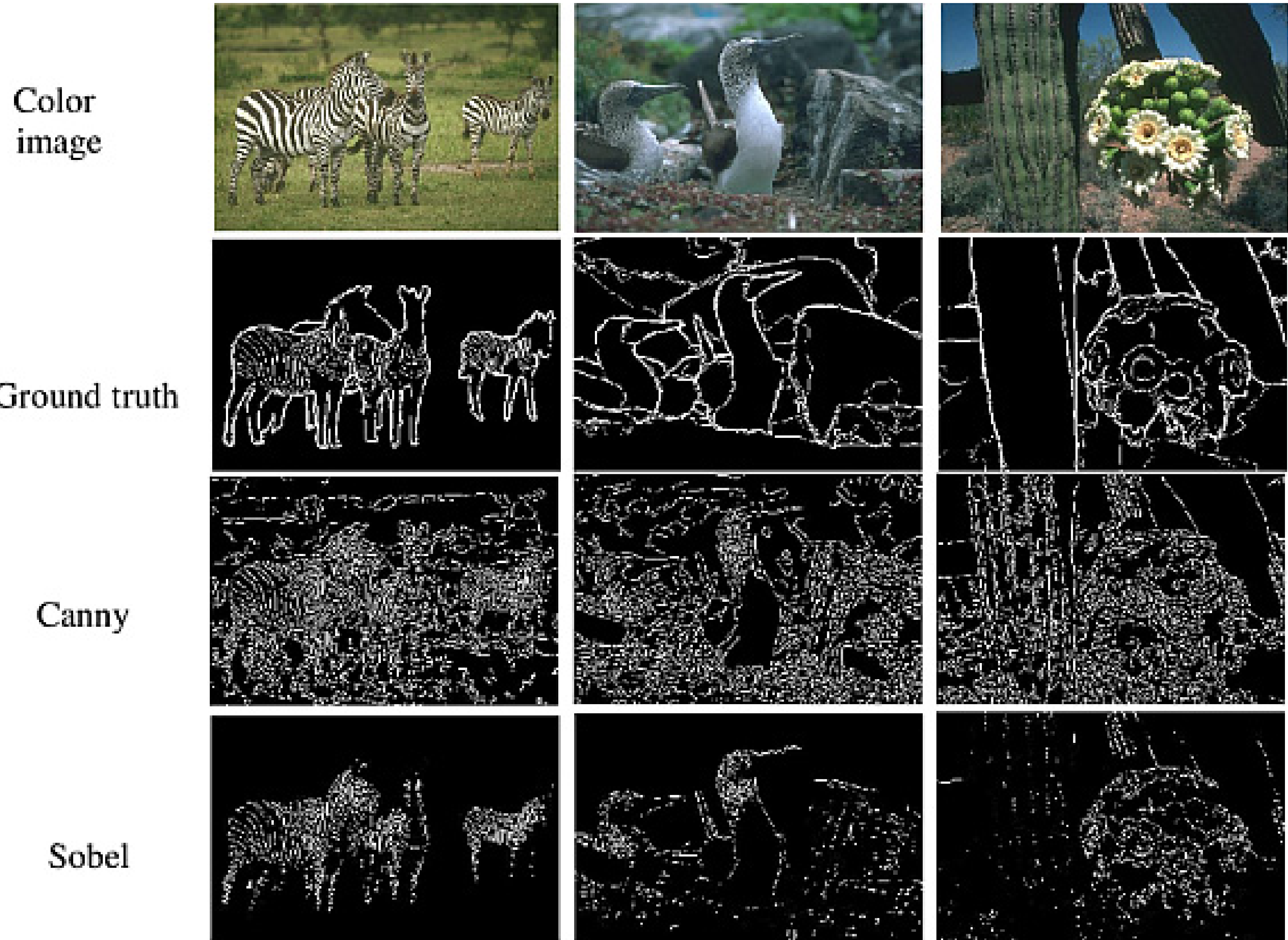




LOG



Canny


Color image
Ground truth
Canny
Sobel

Three criteria in the optimization function used by Canny, for deriving the operator:

- Localization, Detection and minimal response (SNR-based).

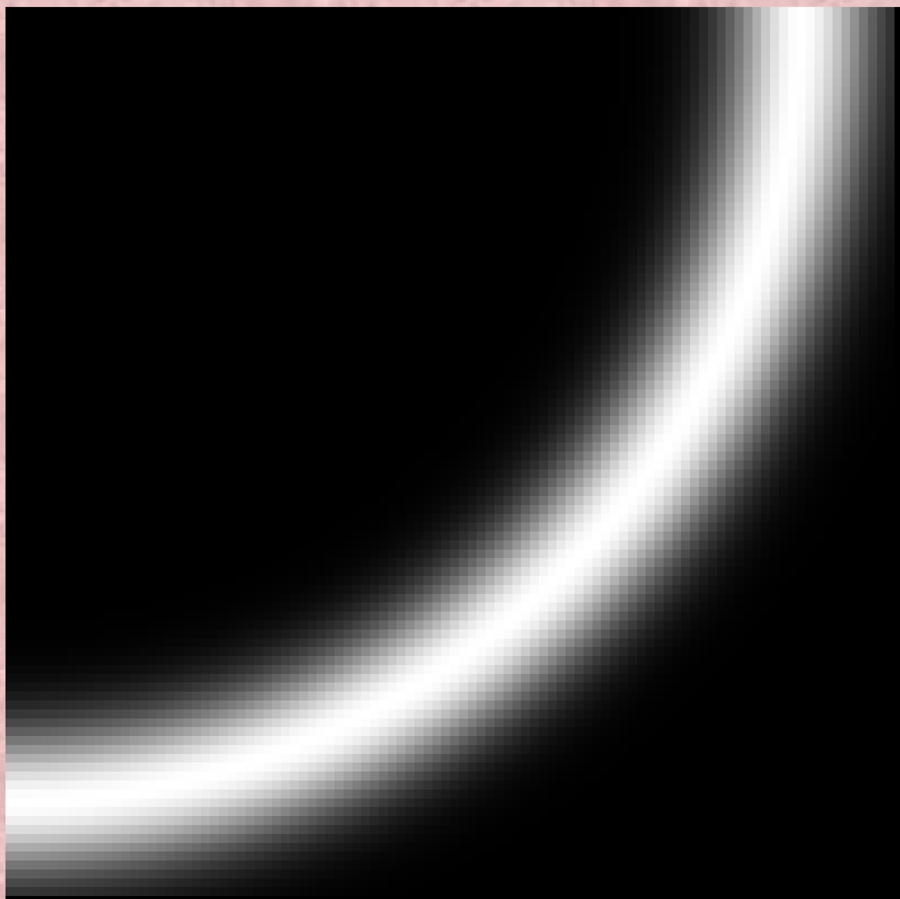
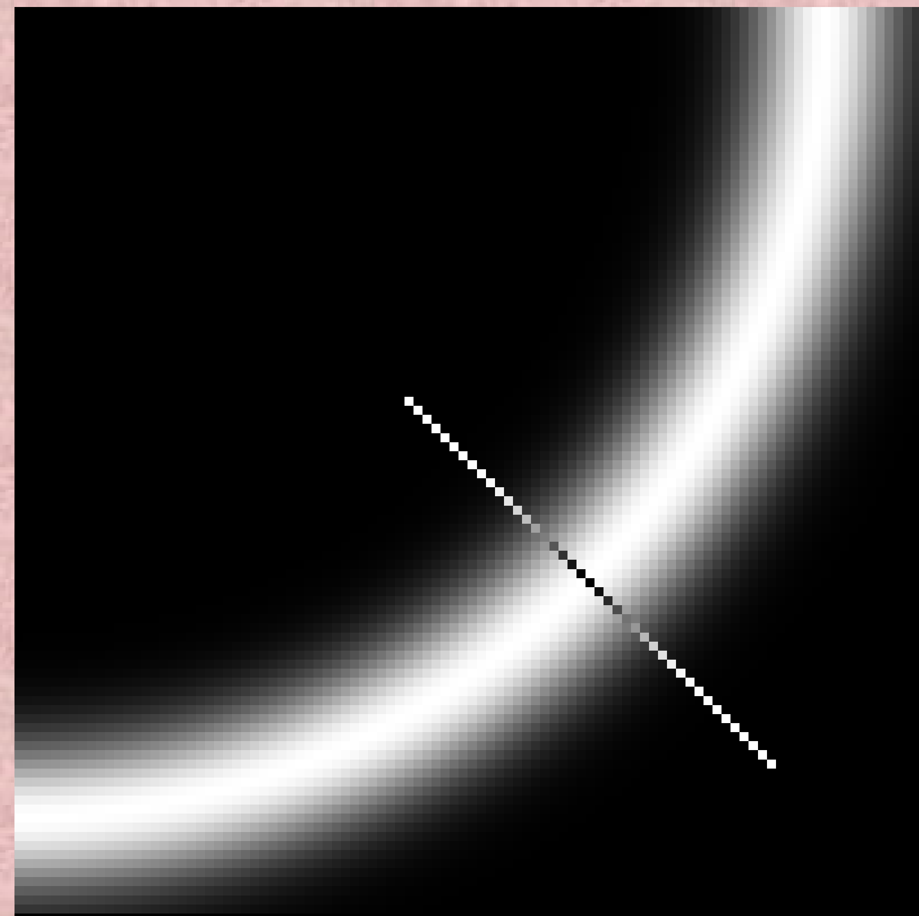
The three stages of Canny's algorithm:

- Apply Operator (often implemented as Smoothing then Derivative)

- Apply non-maximal suppression

- Apply hysteresis based (linking and) thresholding

Read about - DERICHE recursive filtering

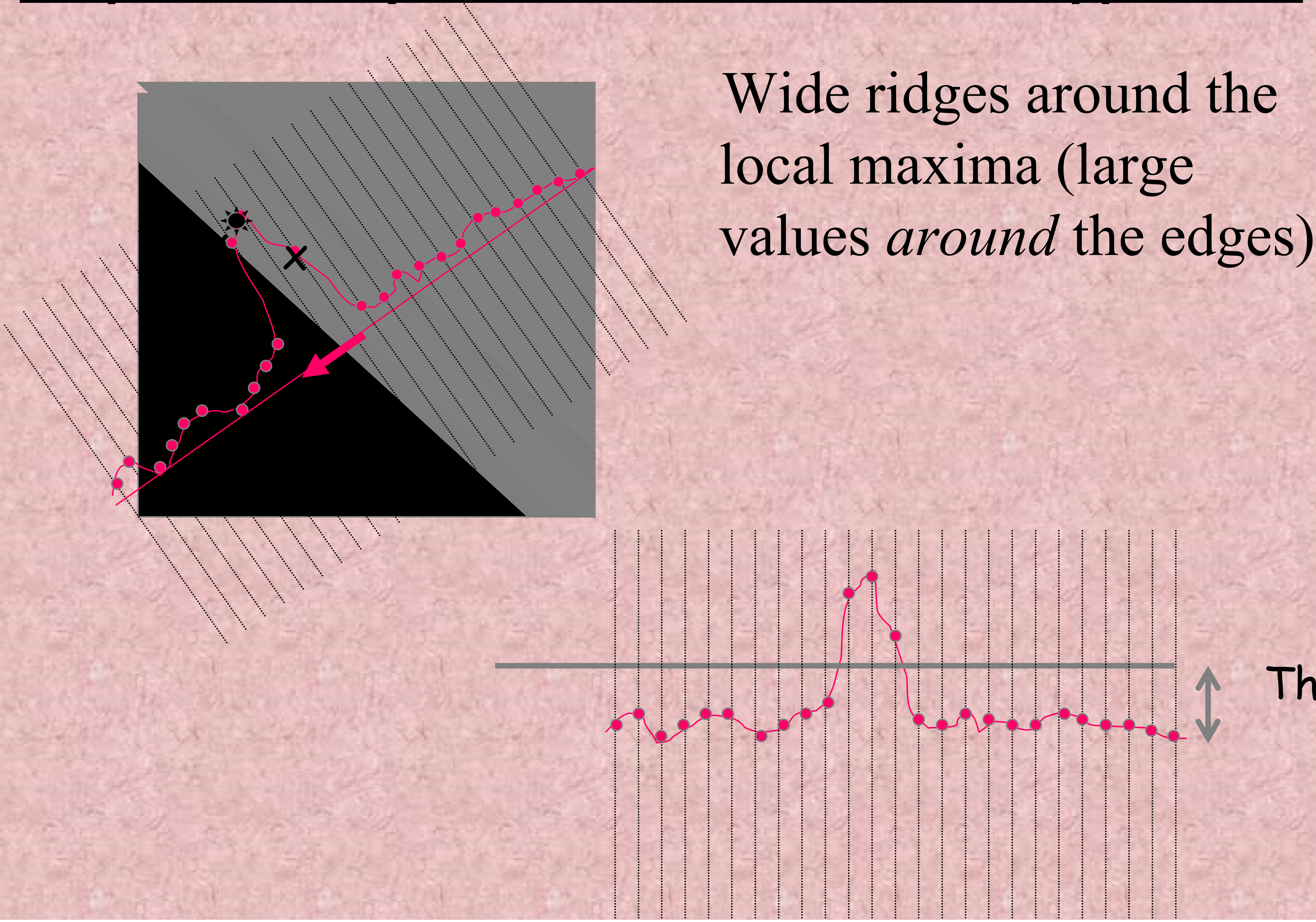
 

Non-maximum suppression:

Select the single maximum point across the width of an edge.

Graphical Interpretation of non-maximal suppression

Wide ridges around the local maxima (large values *around* the edges)



NONMAX_SUPRESSION (Mag, Dir)

- **Consider 4 directions** **$Del^+ = \{ (1, 0), (1, 1), (0, 1), (-1, 1)\}$**

  $Del^- = \{ (-1, 0), (-1, -1), (0, -1), (1, 1)\}$

- **For each pixel (i, j) do:**

  1. Find the direction of gradient (normal to the edge)
     d = (Dir(i, j)+π/8) *mod* π/4 <* or should be "*div*" ?? *>

  2. If **Mag(i,j)** is smaller than at least one of its neigh. along **d** then $I_N(i,j)=0$, otherwise, $I_N(i,j)= Mag(i,j)$

     If Mag(i, j) < Mag((i, j) + Del^+(d)) then I_N(i, j)=0

     Else If Mag(i, j) < Mag((i, j) + Del^-(d)) then I_N(i, j)=0

     Else I_N=Mag(i, j)

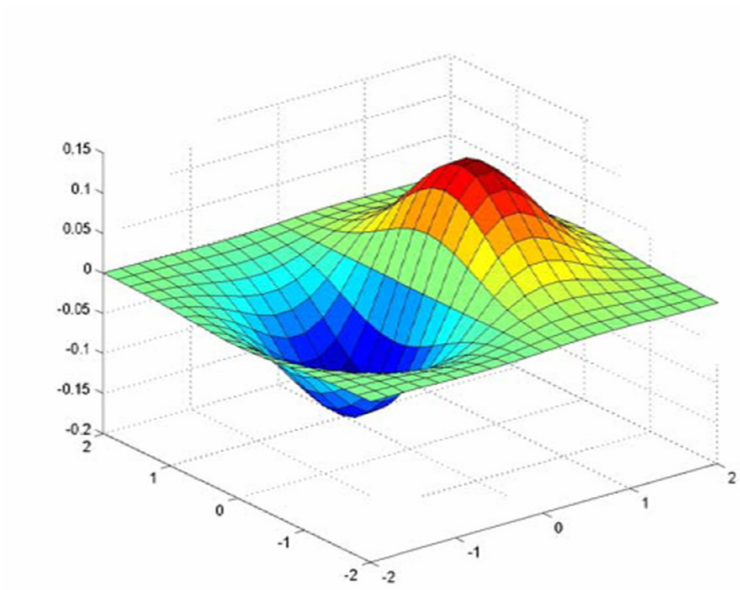
- **The output is the thinned edge image I_N**

Example

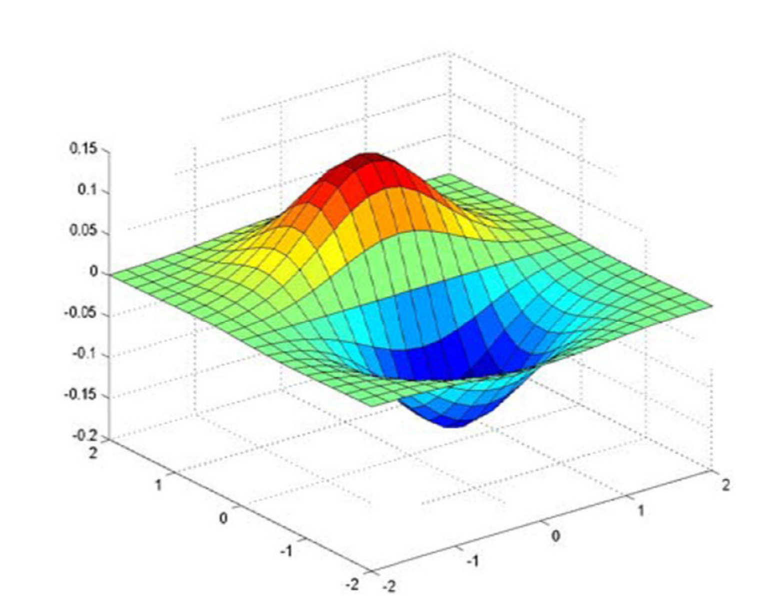
original image (Lena)

Derivative of Gaussian filter ??

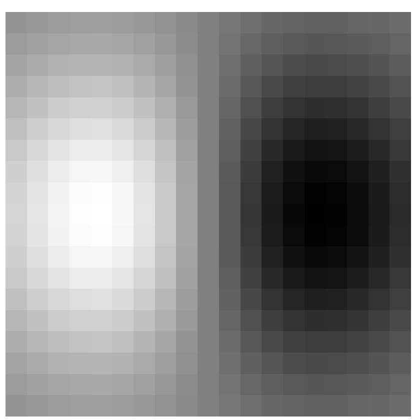
(say, Gaussian and LOG in 2-D given earlier)

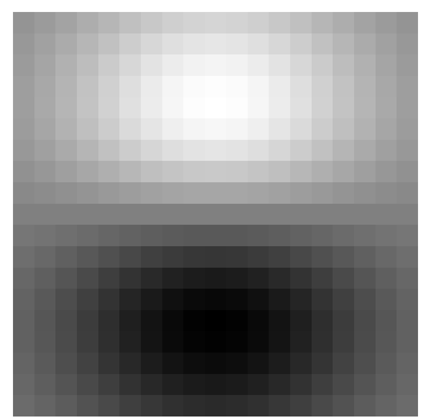


x-direction

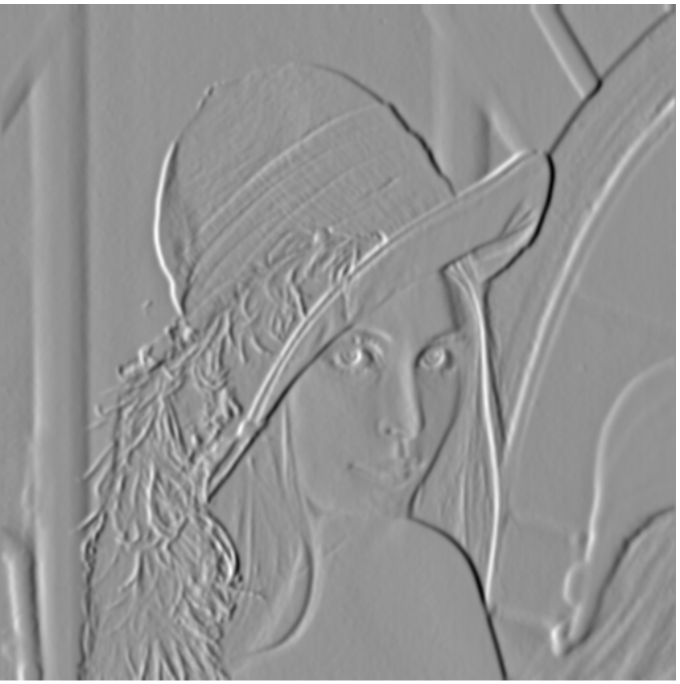


y-direction





Compute Gradients (DoG)



X-Derivative of Gaussian

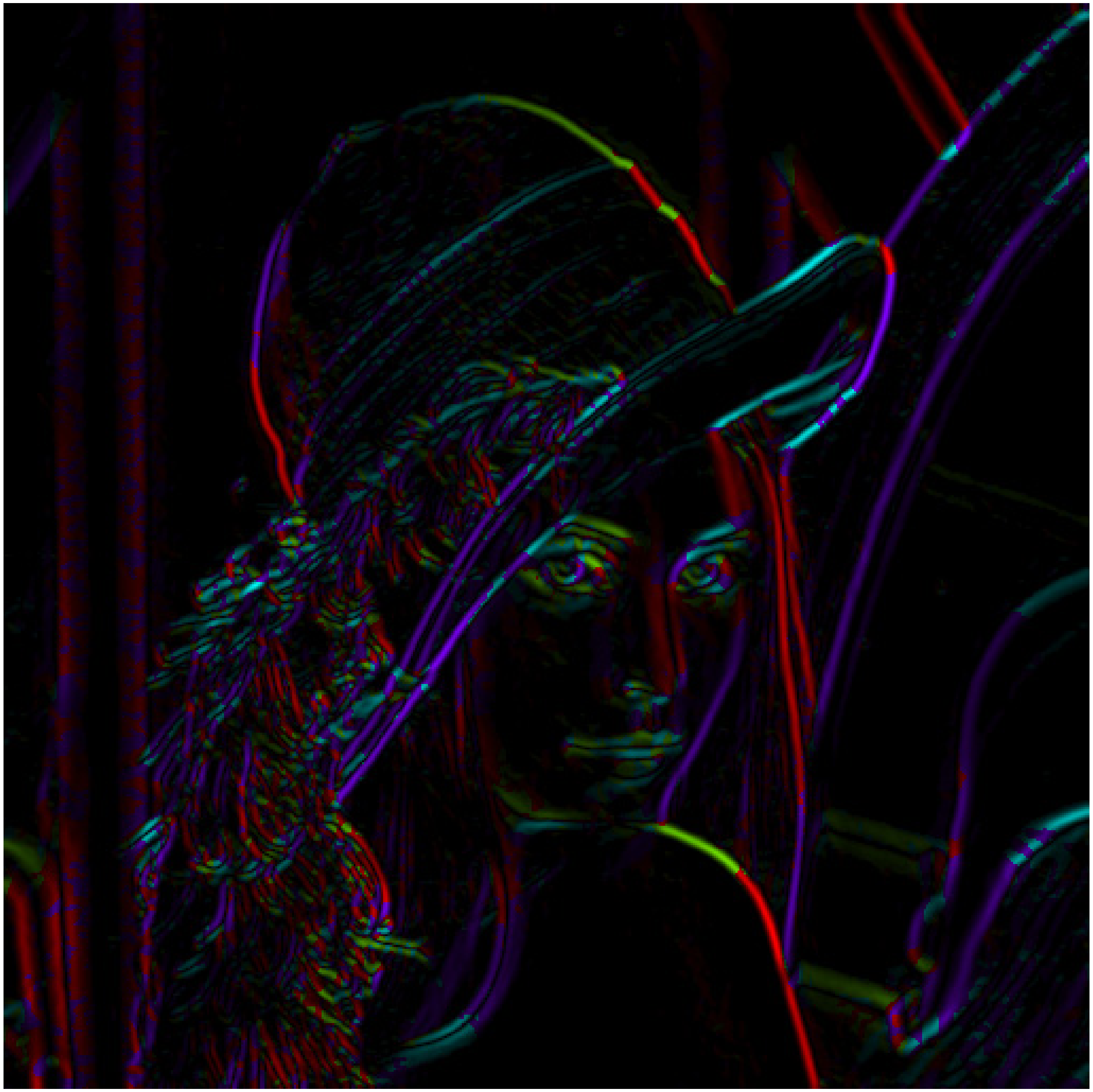
Y-Derivative of Gaussian

Gradient Magnitude

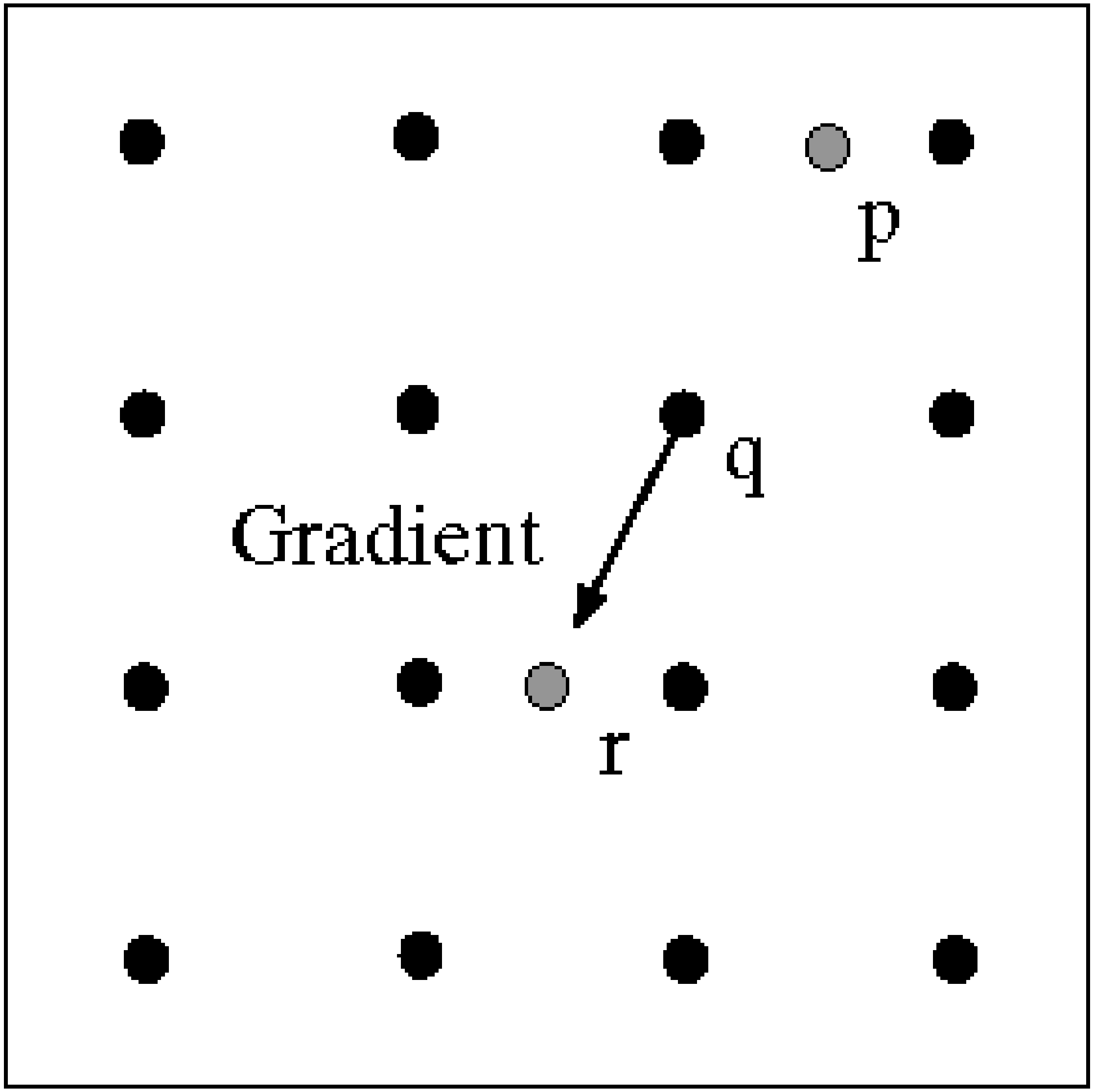
Get Orientation at Each Pixel

- Threshold at minimum level
- Get orientation

theta = atan2(gy, gx)



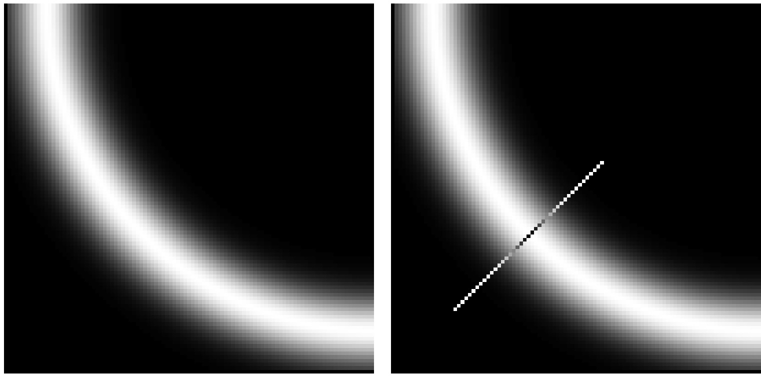
Non-maximum suppression for each orientation



At q, we have a maximum if the value is larger than those at both p and at r. Interpolate to get these values.

Check if pixel is local maximum along gradient direction.

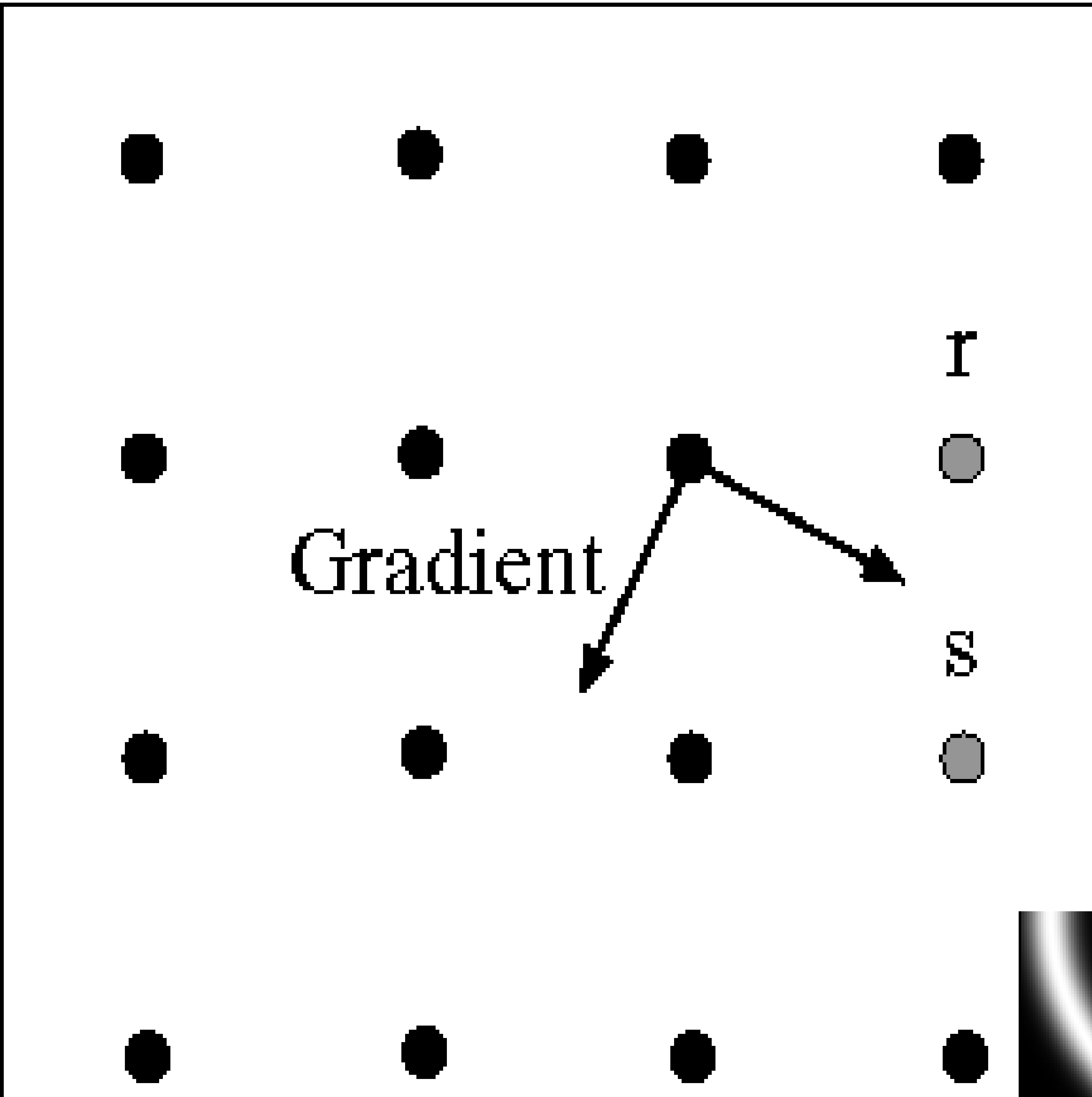
Requires checking interpolated pixels p and r



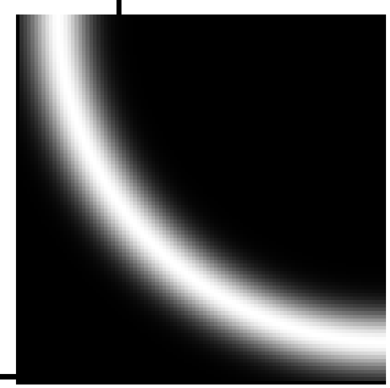
Source: D. Forsyth

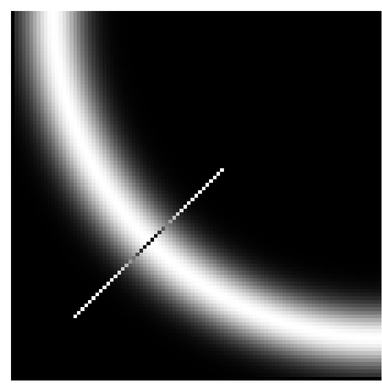
Predicting the next edge point

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).

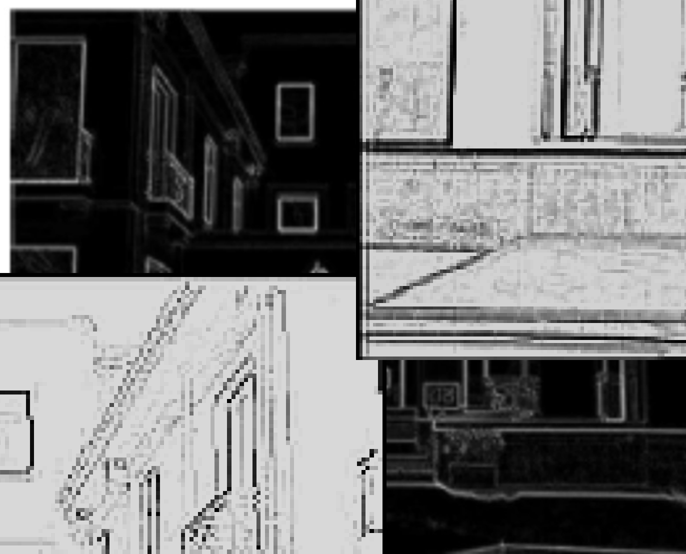


(Forsyth & Ponce)





Example:
Non-Maximum Suppressi



courtesy of G. Loy

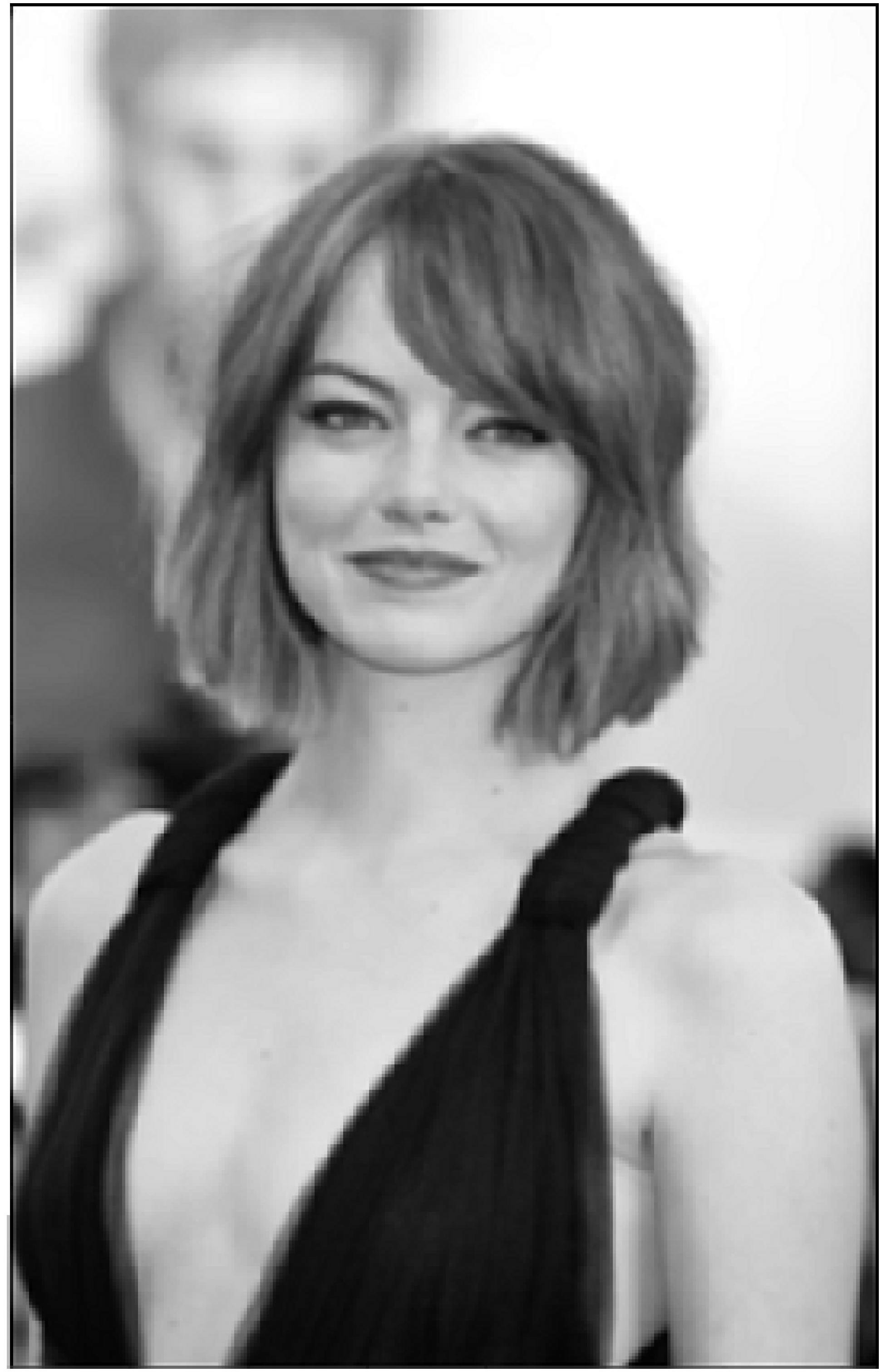
agnitude

Non-maxima suppressed

Before Non-max Suppression

After non-max suppression

https://towardsdatascience.com/canny-edge-detection-step-by-step-in-python-computer-vision-b49c3a2d8123



NMS Results

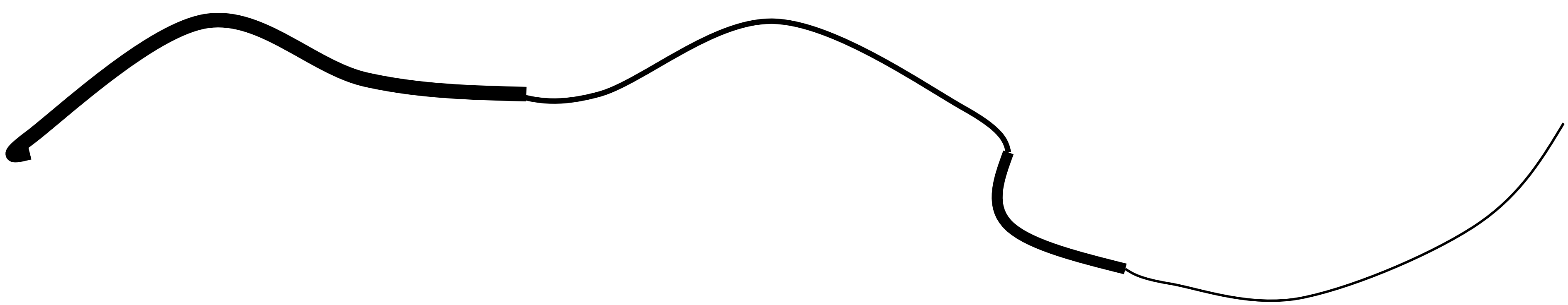
Non-Max Suppression image (left) — Bi-level Thresholding result (right): weak pixels in gray and strong ones in white.

Hysteresis thresholding

- Threshold at low/high levels to get weak/strong edge pixels
- Do connected components, starting from strong edge pixels

Hysteresis thresholding

- Check that maximum value of gradient value is sufficiently large
  – drop-outs? use **hysteresis**
    - use a high threshold to start edge curves and a low threshold to continue them.



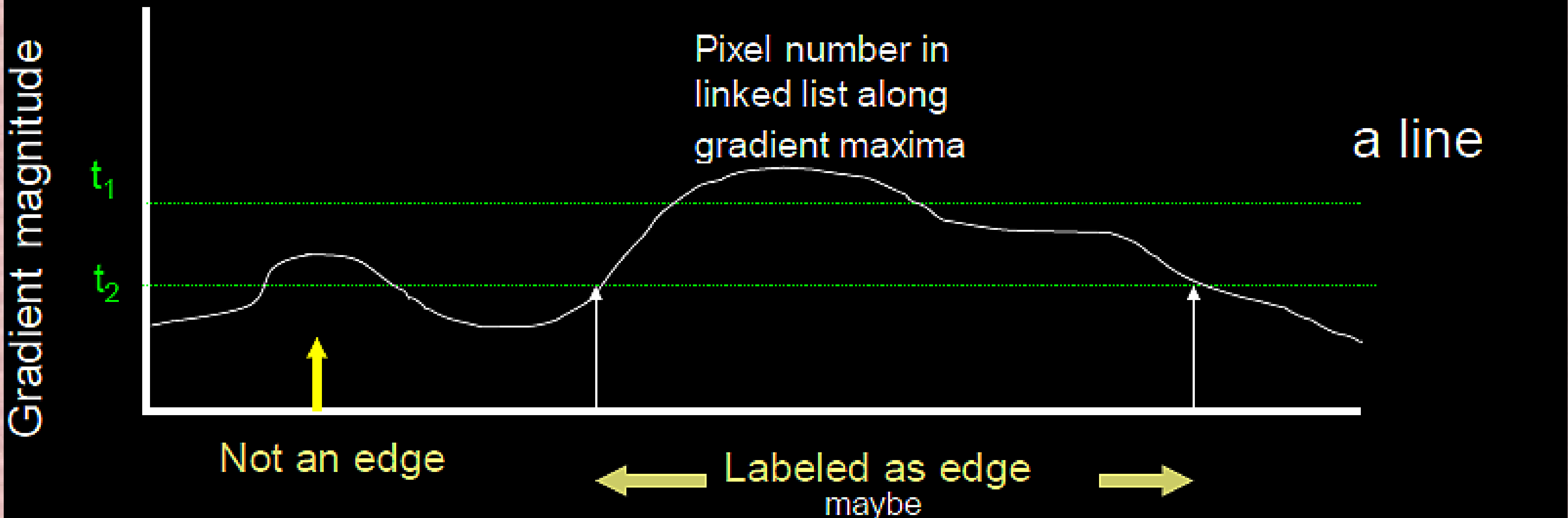
Source: S. Seitz

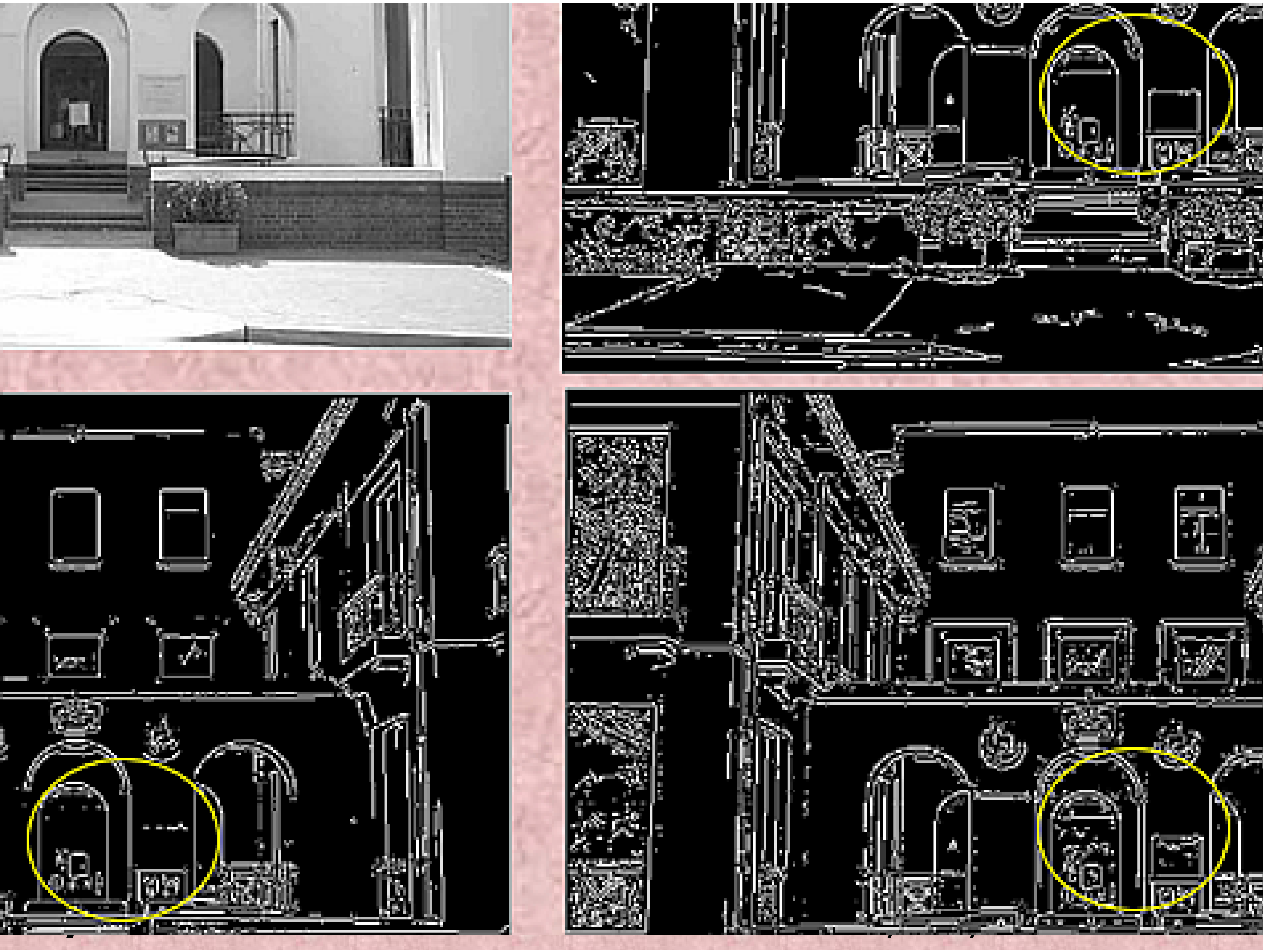
Closing edge gaps

- Check that maximum value of gradient value is sufficiently large and...

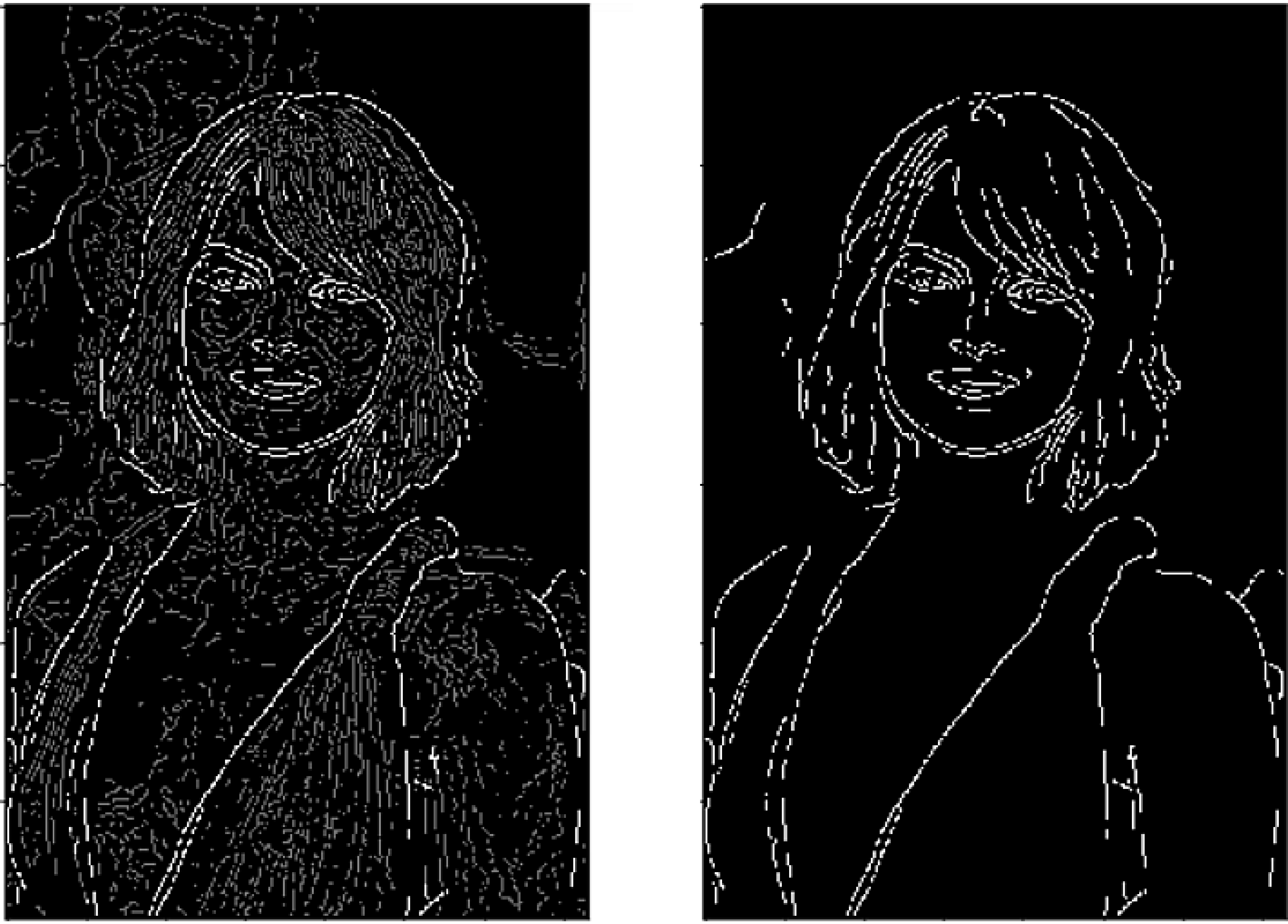
  ... use hysteresis.

  - use a high threshold to start edge curves and a low threshold to continue them.

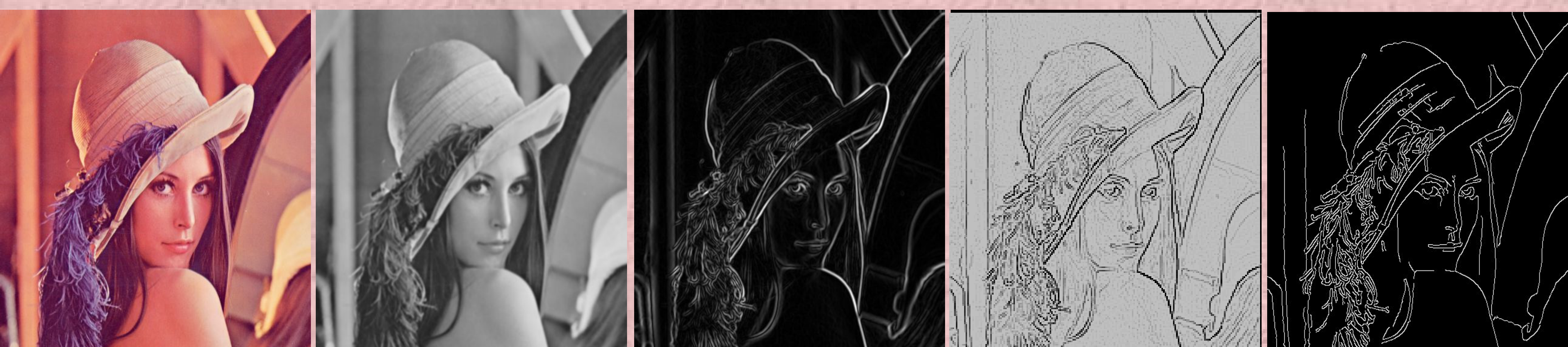


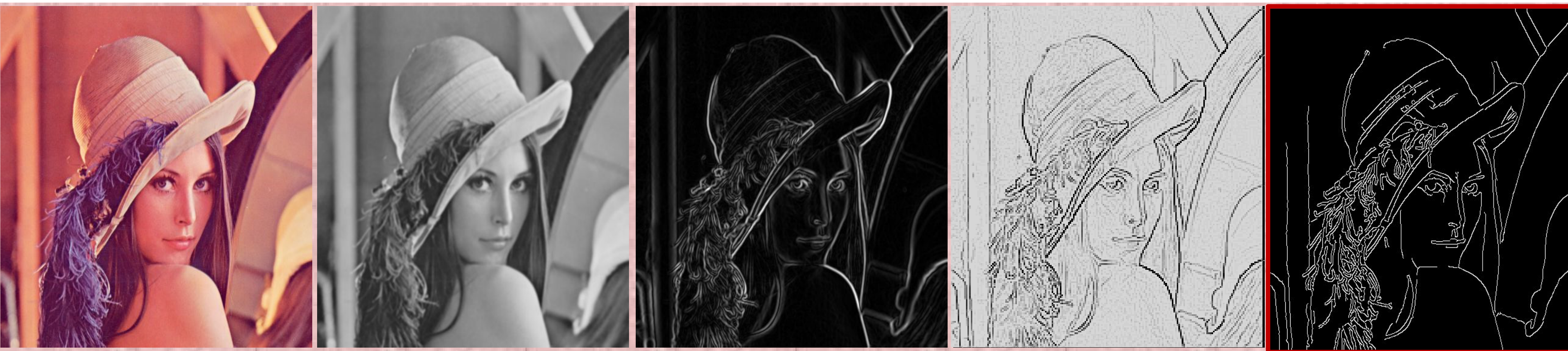


Final Canny Edges



Original Image, Presmoothed Image, Gradient Image, Non-maximum Suppressed Image, Final Result





Original Image, Presmoothed Image, Gradient Image, Non-maximum Suppressed Image, Final Result

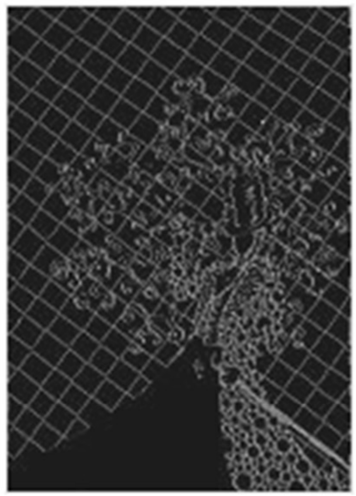
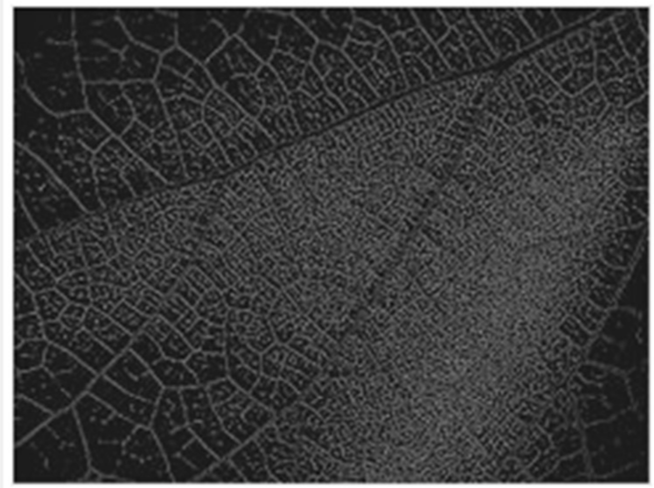
$\frac{1}{115}$

2	4	5	4	2
4	9	12	9	4
5	12	15	12	5
4	9	12	9	4
2	4	5	4	2

Figure 3 Discrete approximation to Gaussian function with σ=1.4

http://www-scf.usc.edu/~boqinggo/Canny.htm

Examples of using Deriche filter on various source images

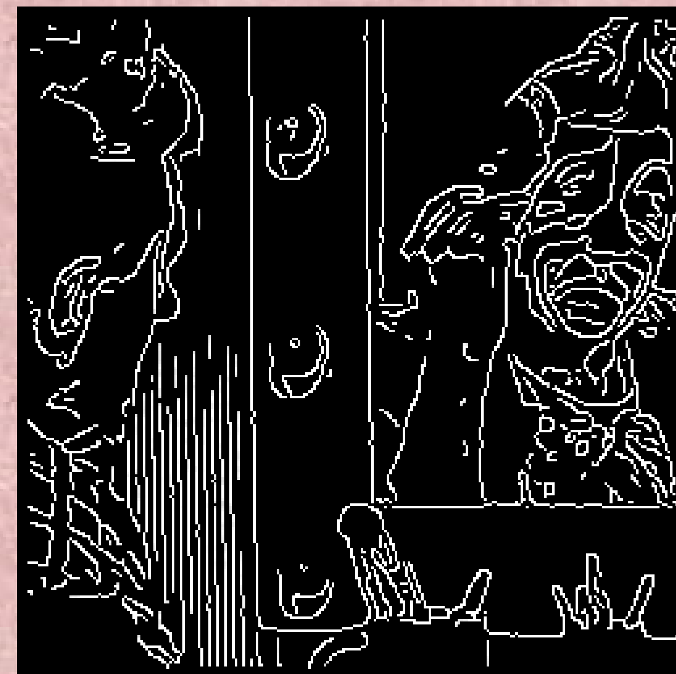
Source image				
Filtered image				
Filter parameters	$\alpha = 1.5$ low treshold = 20 high treshold = 40	$\alpha = 4.0$ low treshold = 50 high treshold = 90	$\alpha = 0.8$ low treshold = 26 high treshold = 41	$\alpha = 1.0$ low treshold = 15 high treshold = 35

Effect of threshold

original

$\sigma = 1$

$T_{high} = 255 \quad T_{low} = 1$

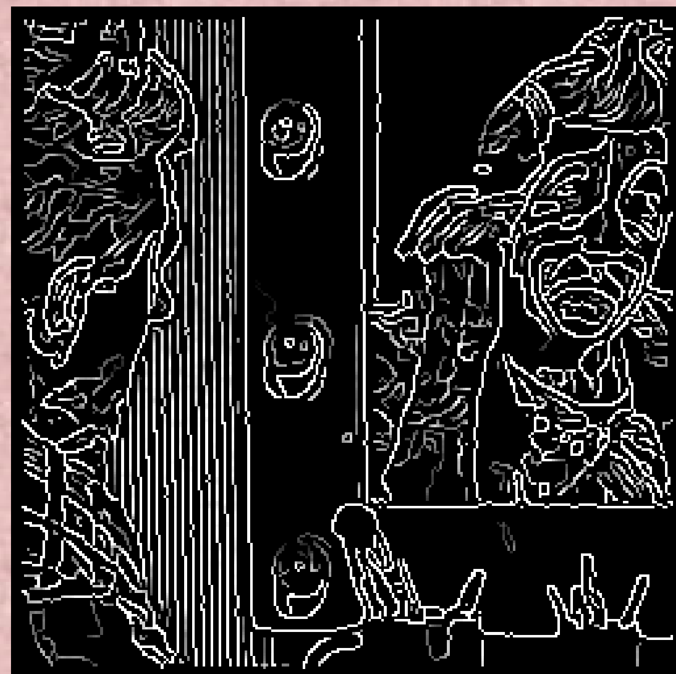


$\sigma = 1$

$T_{high} = 255 \quad T_{low} = 220$

Effect of threshold and of σ (Gaussian kernel size)

original



$\sigma = 1$

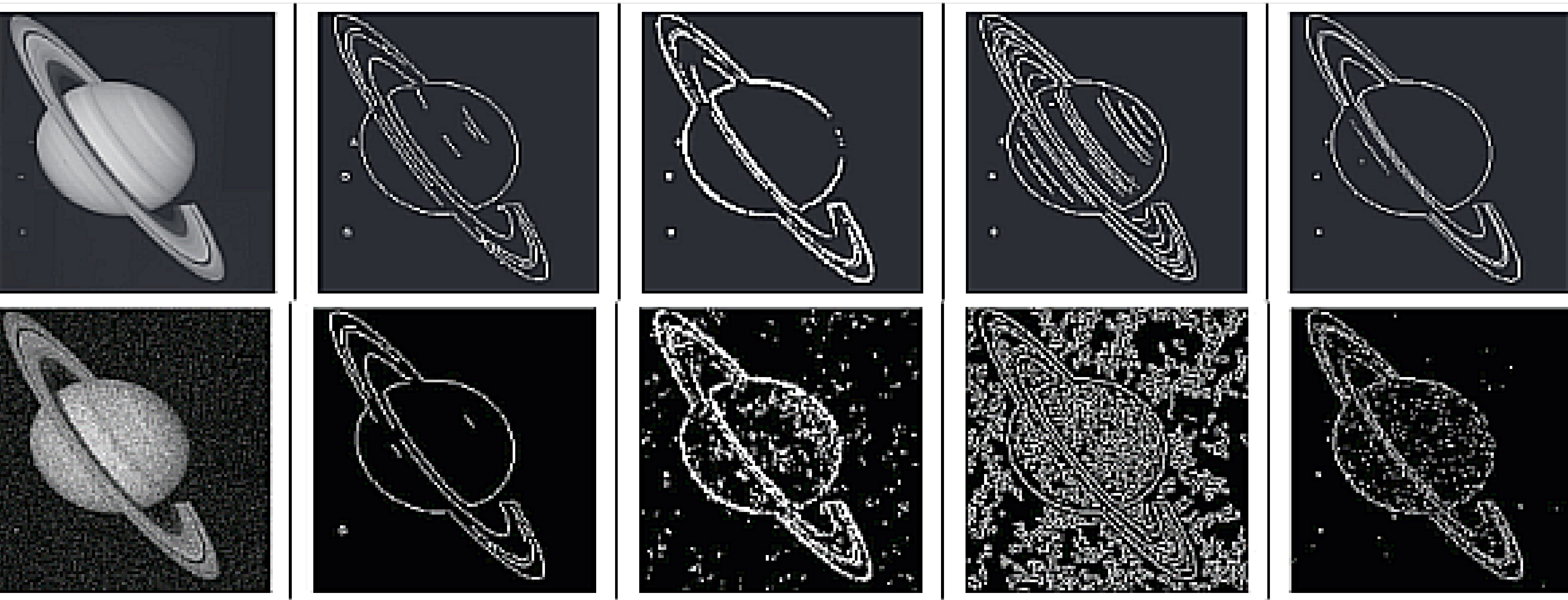
$T_{high} = 120$ $T_{low} = 1$

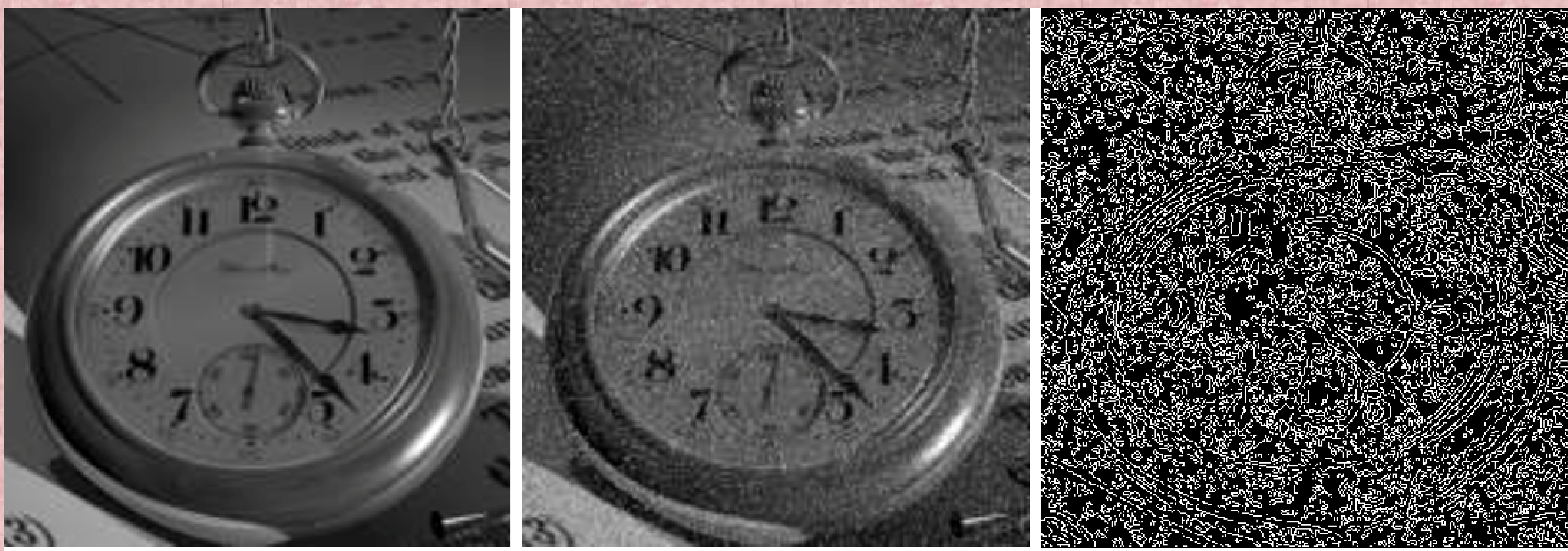
$\sigma = 2$

$T_{high} = 120$ $T_{low} = 1$

The choice of depends on desired behavior

- **large σ   detects large scale edges**
- **small σ   detects fine features**



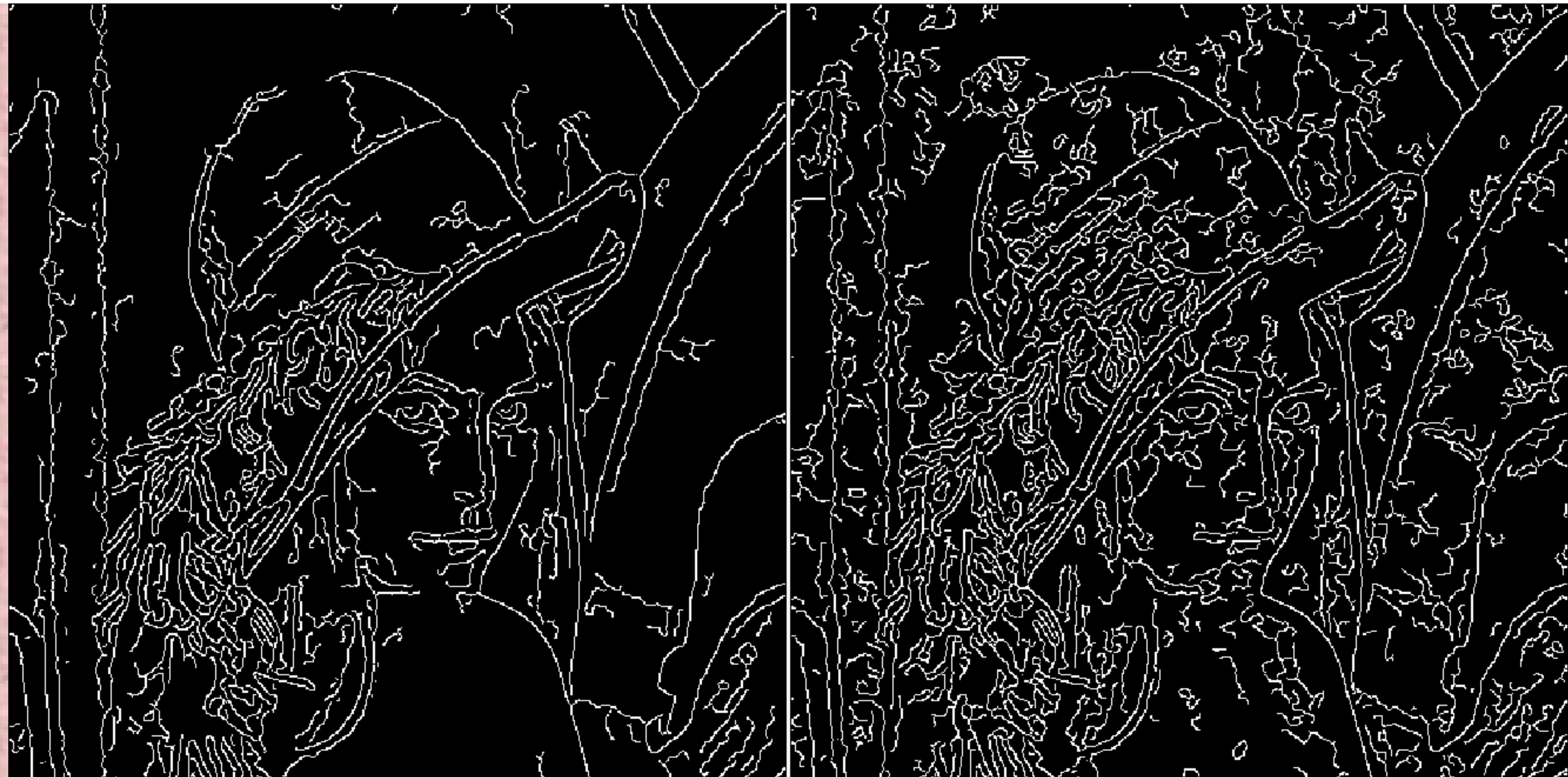


(a). Original Image (b). Noise Image (c). Traditional Canny Operator

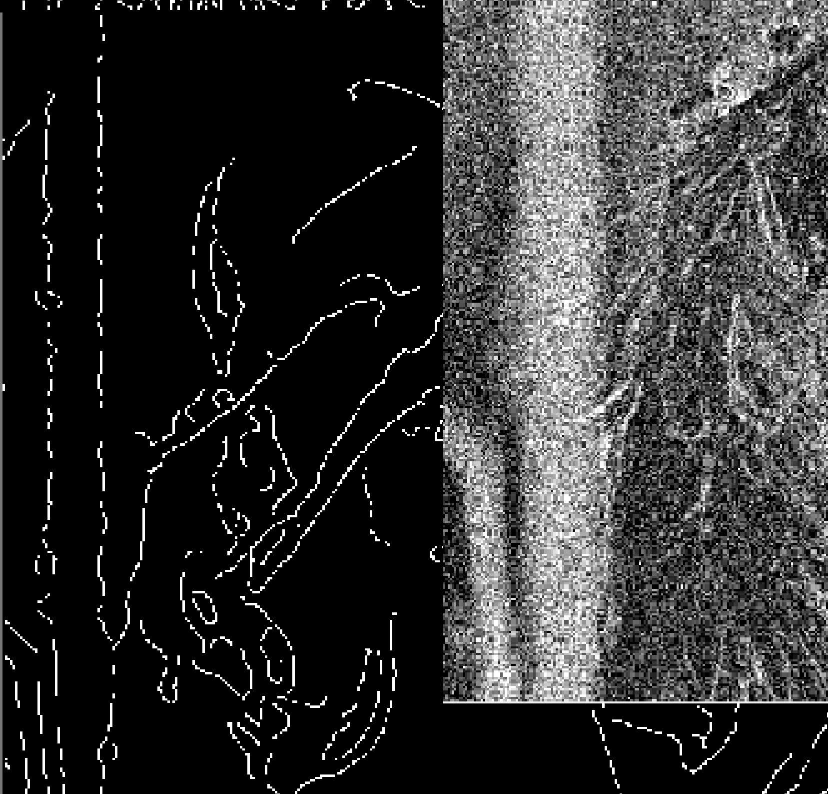
Effect of Noise On Canny Edge;

Thresh = 0.15; sigma = 2;

Effect of Noise On Canny Edge



Thresh = 0.25;
sigma = 3;



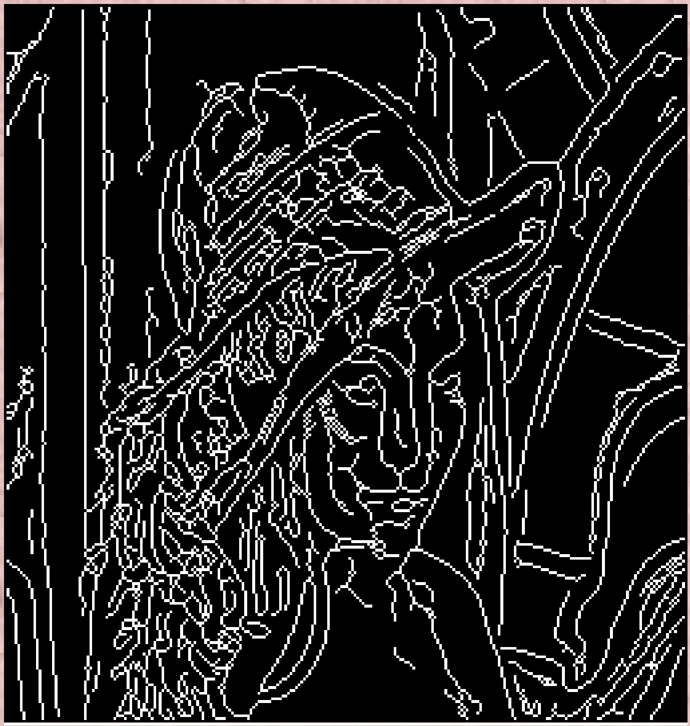
Multi-scale Edge detection

Problem definition

- Our goal is to simultaneously extract edges of all lengths
- Edges are well localized across the scale-space

Input image

Edge map generated by single scale

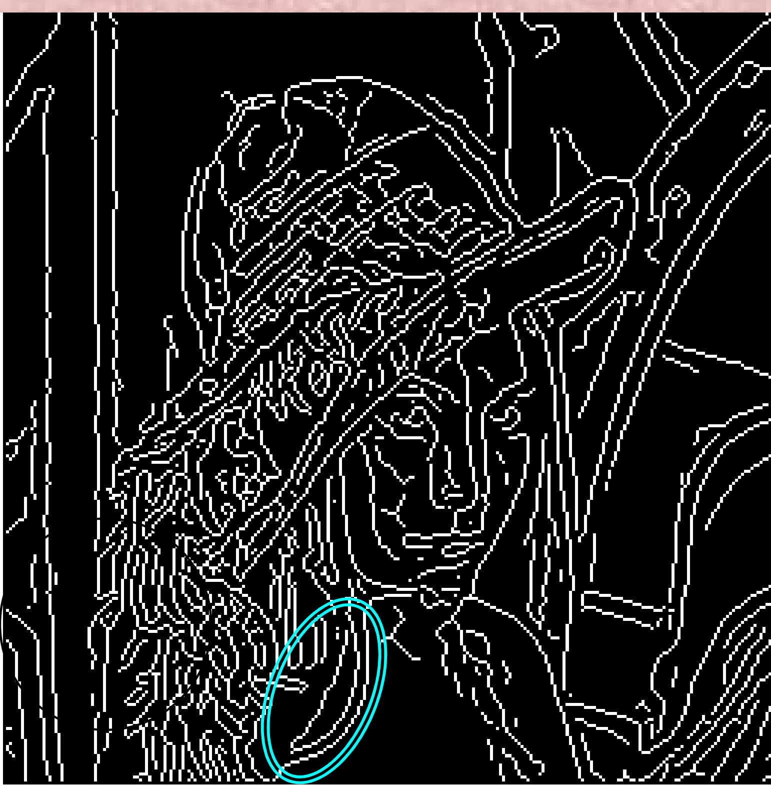


Edge map generated by scale space Combination

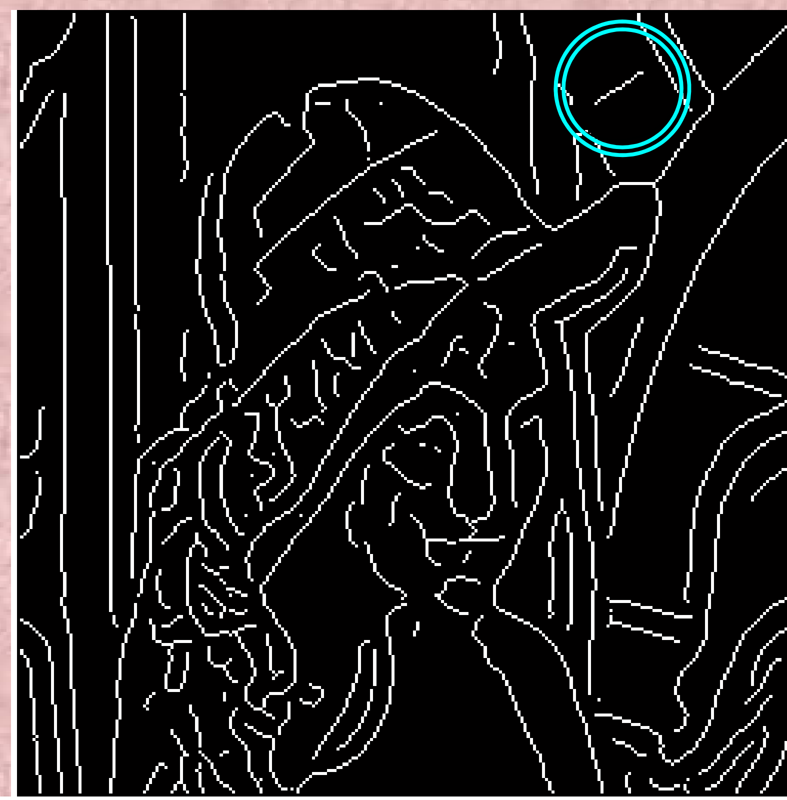
2-D Canny edge maps

$\sigma=0.5$

$\sigma=1.5$

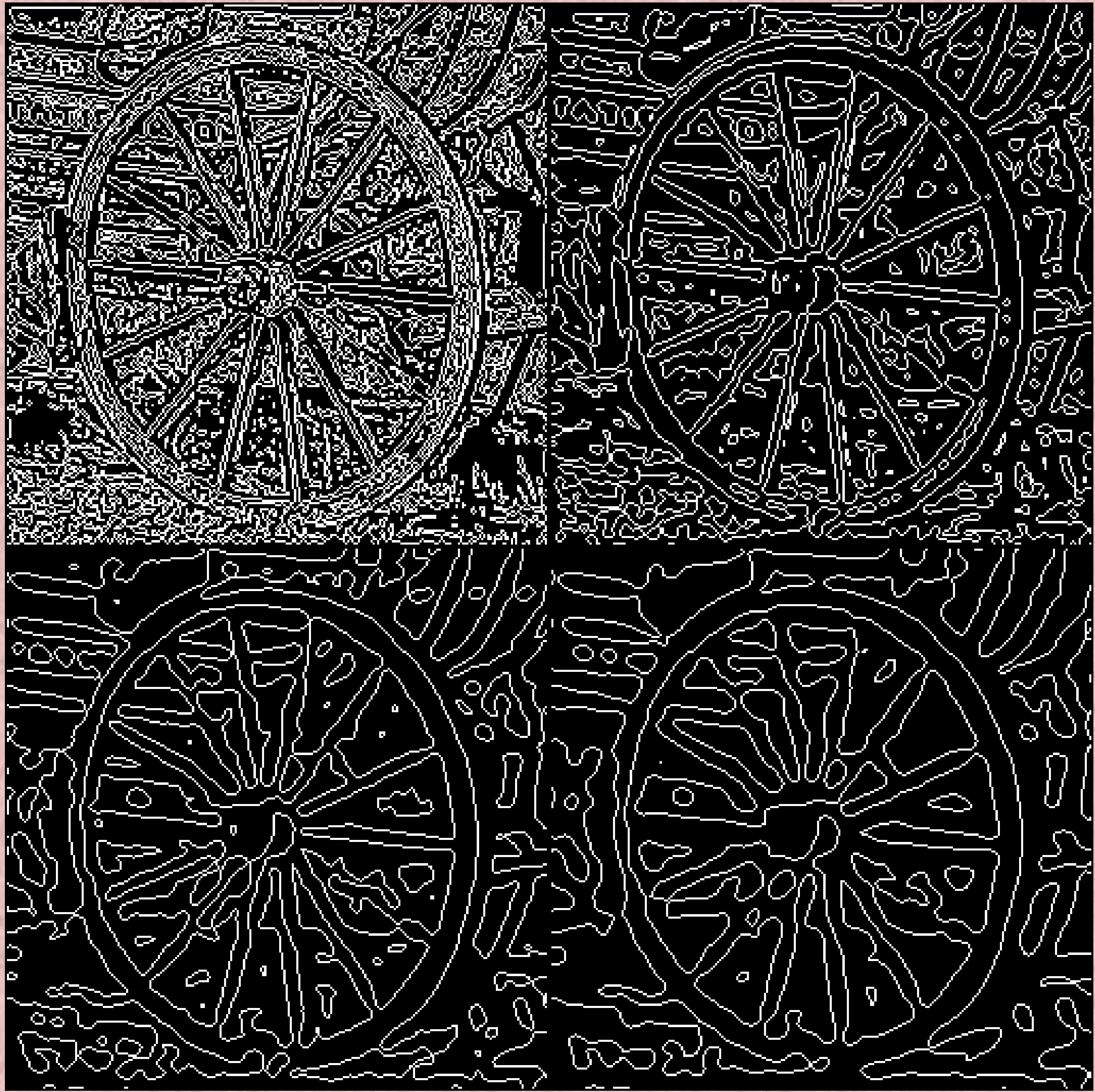


$\sigma=1$



$\sigma=2$

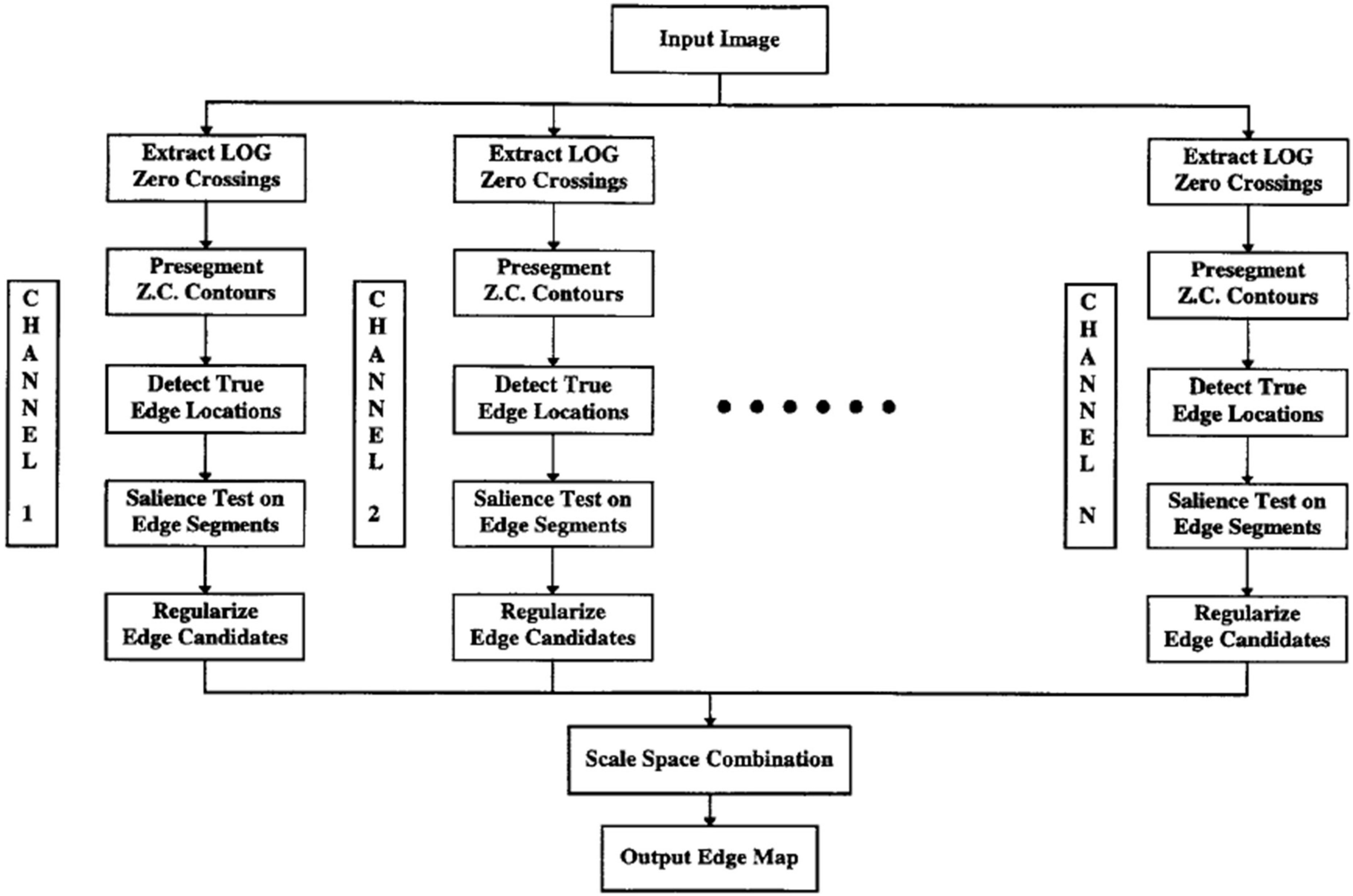
LOG with increasing SIGMA



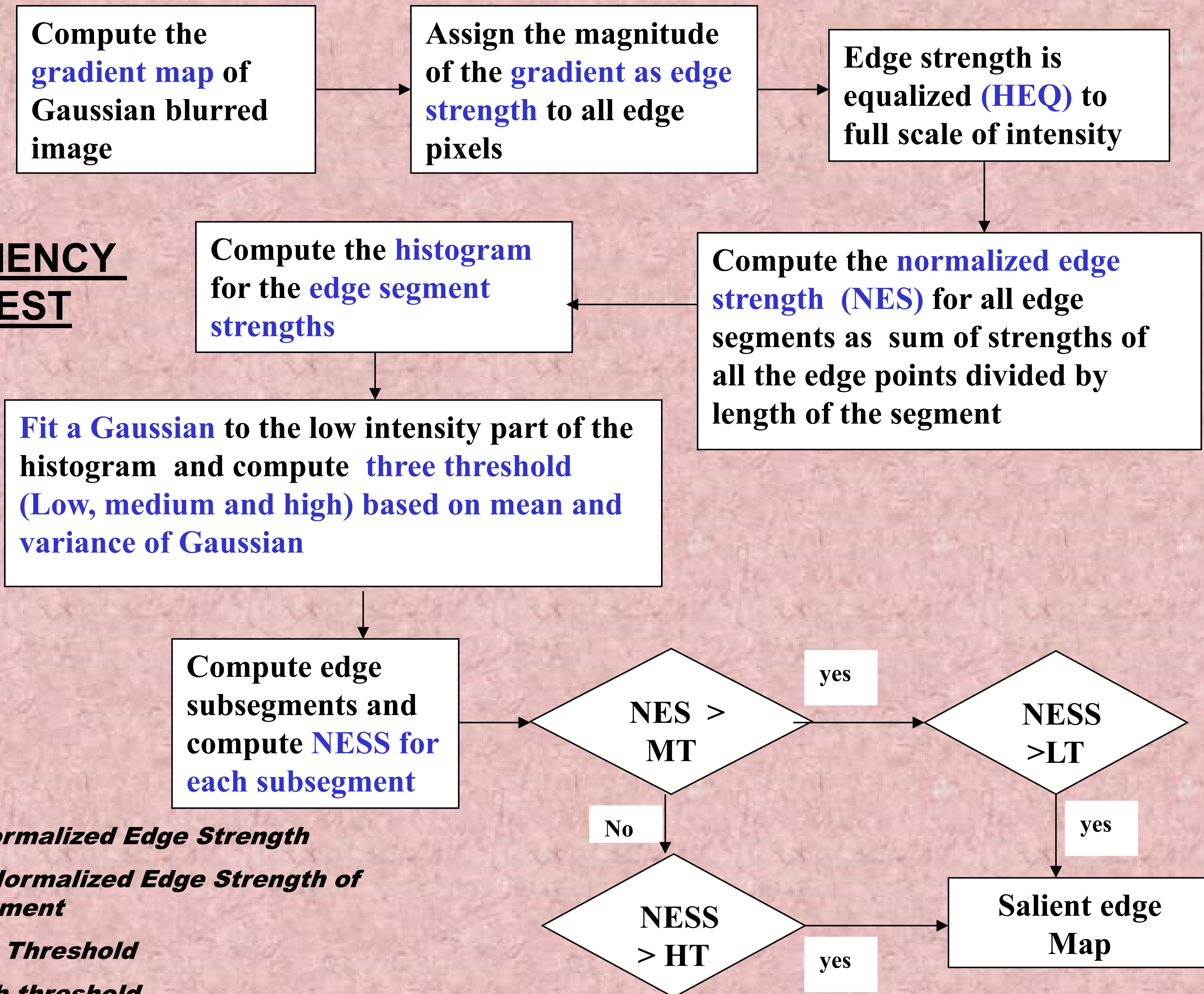
Motivation

- A step edge is sensed at various points by cells of the retinal array
- Real-world objects are composed of different structures at different scales
- Connectivity of an object depends on the scale at which it is observed
- In real-world images the edges may not be ideal
- Variation of the response over different scales is important

Optimal Edge Detection in Two-Dimensional Images



SALIENCY TEST



NES: Normalized Edge Strength

NESS: Normalized Edge Strength of sub-segment

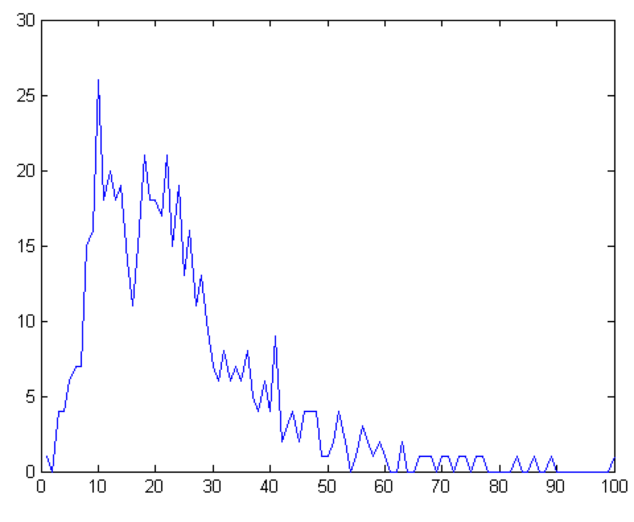
LT: Low Threshold

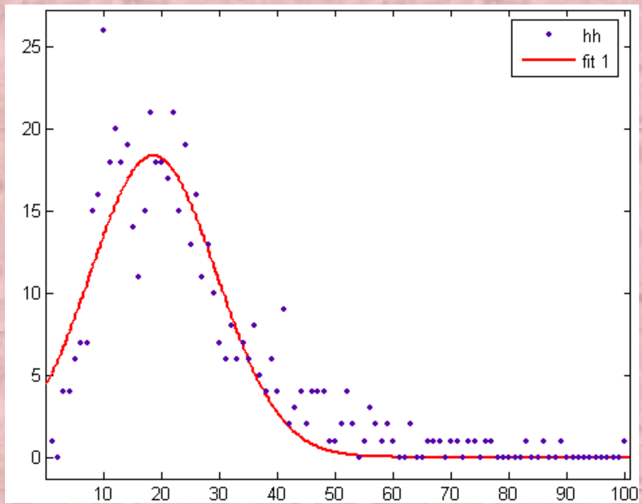
HT: High threshold

MT: Medium Threshold

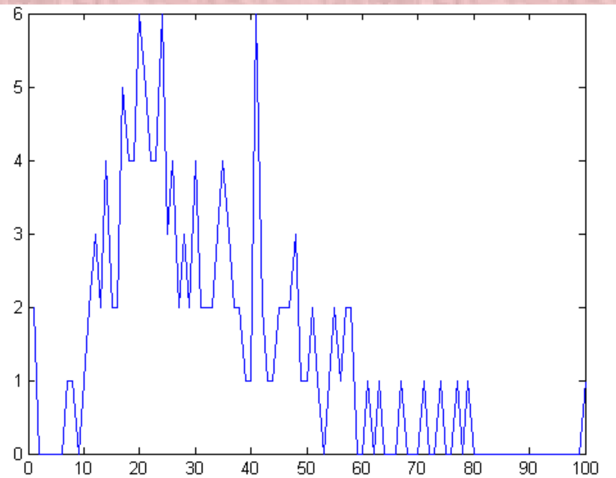
Histogram of the normalised edge strengths and fitted Gaussian distribution

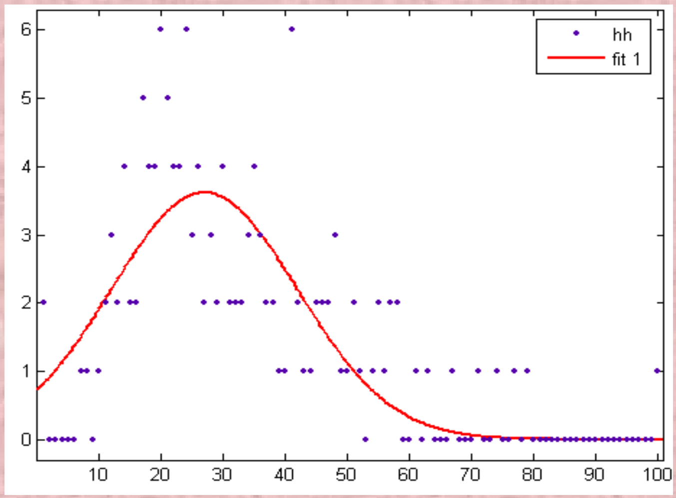
σ=0.5





σ=1.5





σ=0.5 σ=1 σ=1.5 σ=2

2-D Canny edge map

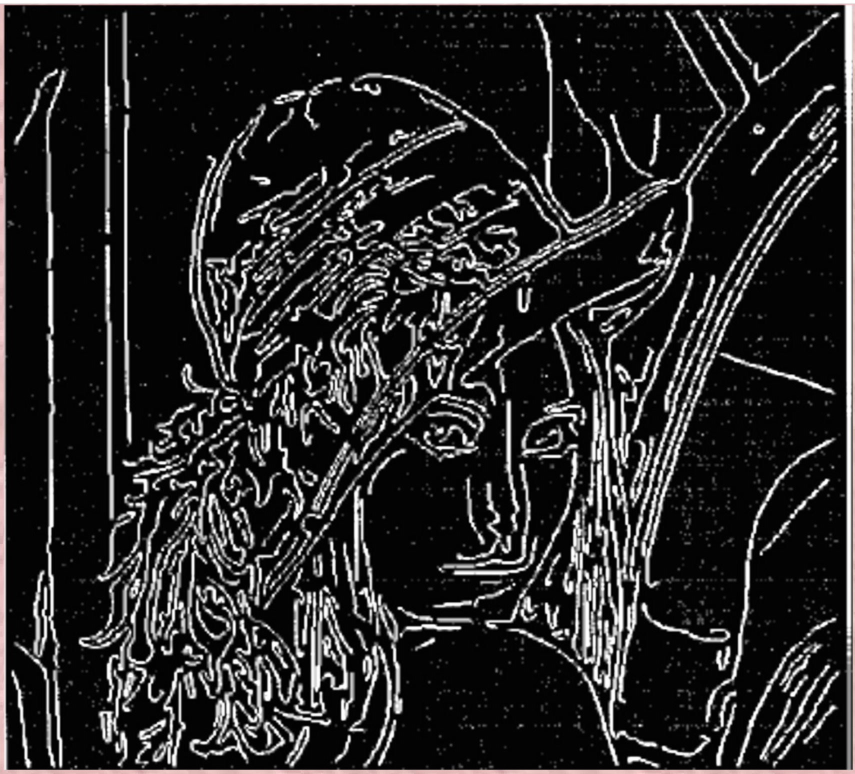
Salient Edge maps

Combining different scales

• The combination procedure checks if there are new salient edges in the detection results from larger scales

Algorithm

1. **Minmap, maxmap= edge map of smallest scale**
2. **Compare maxmap with second smallest scale edgemap**
3. **If an edge segment of minimum length from second smallest scale does not appear in maxmap, add that particular segment to minmap**
4. **Repeat step 2 and step 3 with various scales**
5. **Minmap is the final combined scale output**



Scale space combination
2-D Canny edge model

Scale space combination
of Qian & Huang
edge model

Lena
256x256

REFERENCES

"Digital Image Processing and Computer Vision"; Robert J. Schallkoff; John Wiley and Sons; 1989+.

"Digital Image Processing"; R. C. Gonzalez and R. E. Woods; Addison Wesley; 1992+.

Optimal Edge Detection in Two-Dimensional Images, Richard J. Qian and Thomas S. Huang, IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. *5,* NO. 7, JULY 1996, 1215-1220.

A Two-Dimensional Edge Detection Scheme for General Visual Processing, Qian, R.J. and Huang, T.S, ICPR-94, YEAR = "1994", "595-598".

R. Deriche, *Using Canny's criteria to derive a recursively implemented optimal edge detector*, Int. J. Computer Vision, Vol. 1, pp. 167–187, April 1987.

J. Canny, *A Computational Approach To Edge Detection*, IEEE Trans. Pattern Analysis and Machine Intelligence, 8(6):679–698, 1986.

R. Sirdey, *A Gentle Introduction to the Deriche Optimal Edge Detector*, Éditions des Nik's news, 1998.

Read about:

- Hysteresis based Thresholding
- Non-maximal suppression
- Edge Linking & Thinning
- Multi-channel edge detection
- Berkeley Edge Detection/segmentation
- Structured Forests
- Edge preserving enhancement or super-resolution
- Contour Tracing
- Level set based or differential geometry based analysis
- Edge detection with sub-pixel accuracy
- Neuro-fuzzy models for optimal edge detection
- Phase Congruency model (Peter Kovesi) for edge detection
- Deriche model for optimal/recursive filtering
- Neural model for supervised edge detection
- Physics based processing
- Optimization based (MRF, HMM) edge detection, in presence of noise and blur

Few Modern / State-of-the-Art Edge Detectors

- -Berkeley, Spatial Clustering;
- SCG
- ST
- SF

References

- Martin, David R., Charless C. Fowlkes, and Jitendra Malik. "Learning to detect natural image boundaries using brightness and texture." *Advances in Neural Information Processing Systems*. 2002.
- Martin, David R., Charless C. Fowlkes, and Jitendra Malik. "Learning to detect natural image boundaries using local brightness, color, and texture cues." *Pattern Analysis and Machine Intelligence, IEEE Transactions* (2004): 530-549.
- Arbelaez, P., Maire, M., Fowlkes, C., & Malik, J. (2011). Contour detection and hierarchical image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *33*(5), 898-916. (gPb+owt+ucm)
- Xiaofeng, Ren, and Liefeng Bo. "Discriminatively trained sparse code gradients for contour detection." *Advances in neural information processing systems*. 2012. (SCG)
- Lim, Joseph, C. Zitnick, and Piotr Dollár. "Sketch tokens: A learned mid-level representation for contour and object detection." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013 (Sketch Tokens)
- Dollár, Piotr, and C. Zitnick. "Structured forests for fast edge detection."*Proceedings of the IEEE International Conference on Computer Vision*. 2013. (SE)
- Dollár, Piotr, and C. Lawrence Zitnick. "Fast edge detection using structured forests." *Pattern Analysis and Machine Intelligence, IEEE Transactions on;* 37/8 (2015): 1558-1570.

Berkley Edge Detector - Key Ideas (2002, 2004)

- Based on cues
  - Brightness, Color and Texture
- Cues are Optimized and then Simple Linear model for Cue combination is proposed

Results:

BG - Brightness Gradient;

CG – Color Gradient;

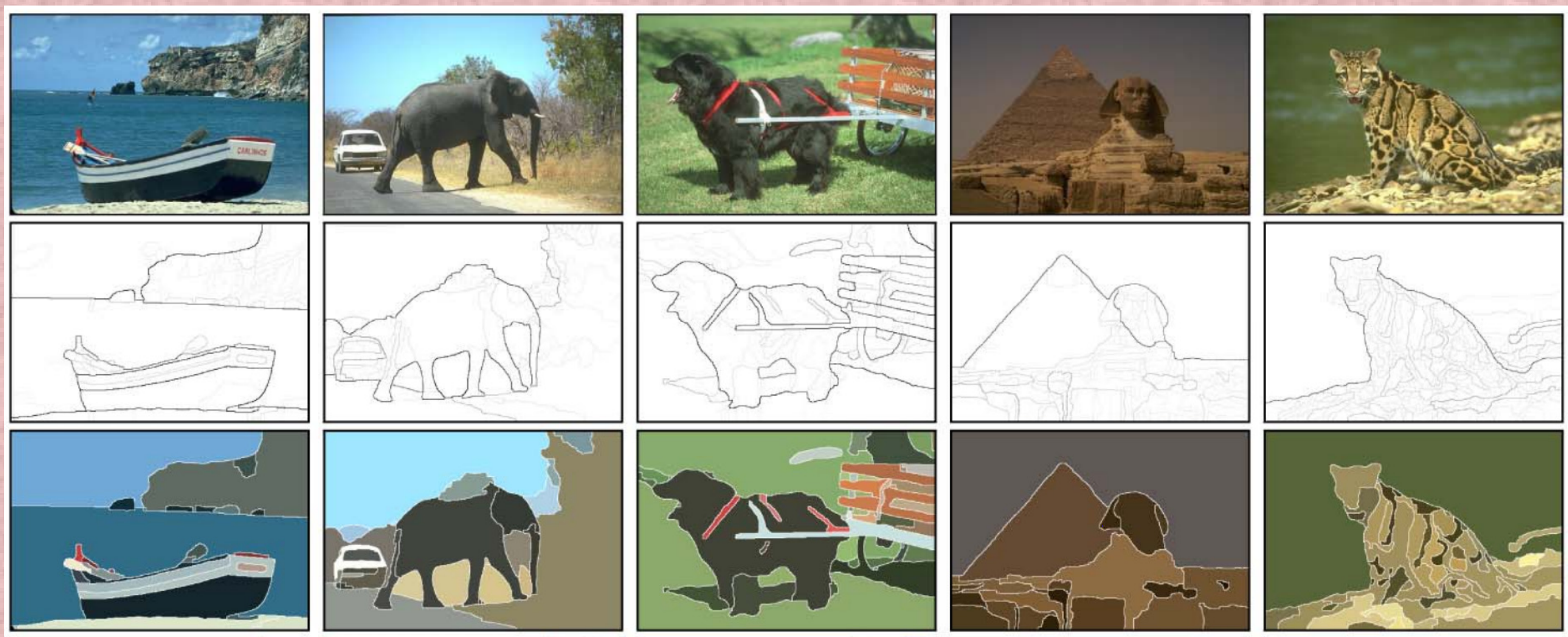
TG – Texture Gradient

Images are from BSDS500 Dataset

Improvement on the Previous Method (gPb+owt+ucm) (2011)

- Contour Detector - Combines Local cues into Globalization framework based on **Spectral Clustering**
- Also, gives Hierarchical Segmentation Output

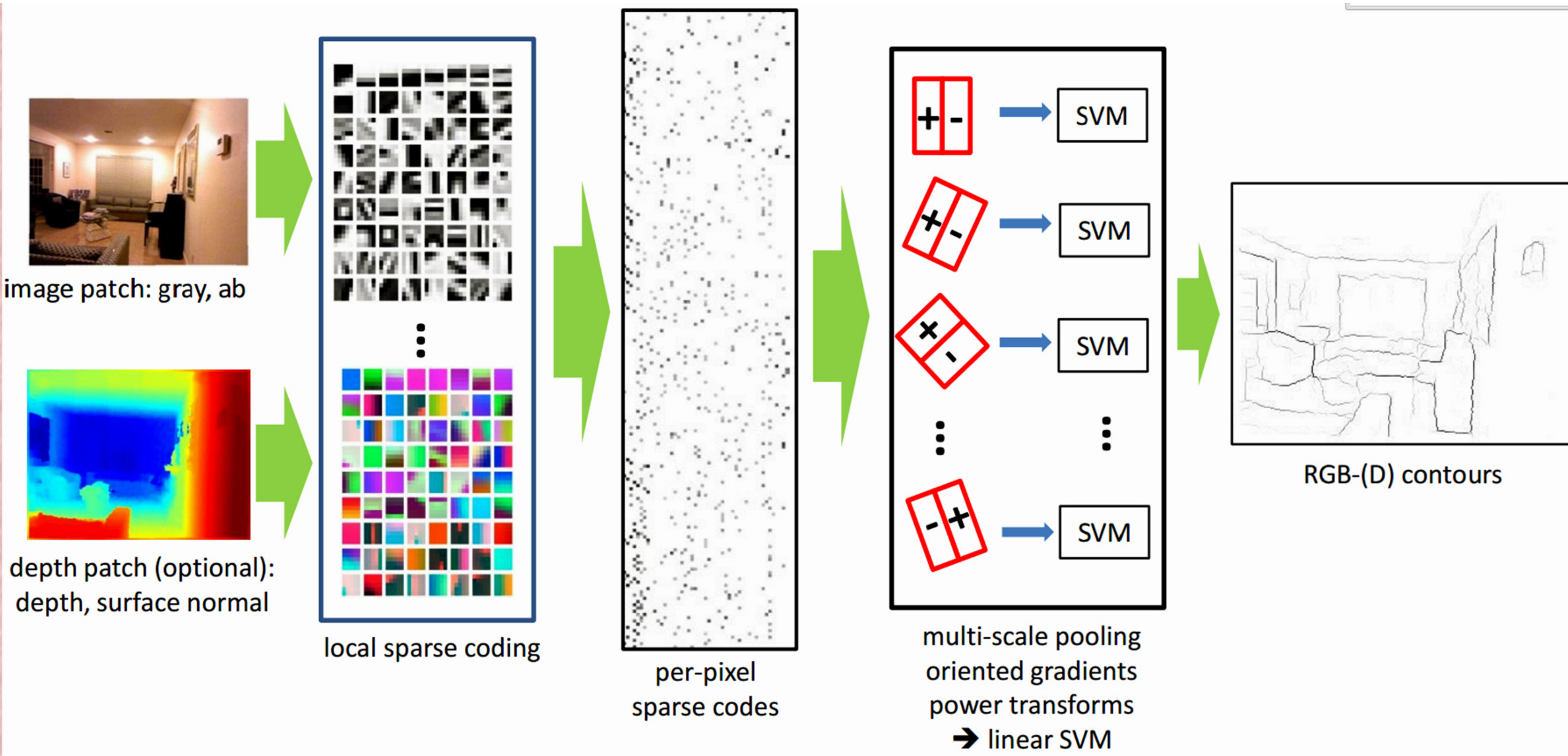
Results (gPb+owt+ucm):



Row 1 - Input Images from BSDS500 Dataset
Row 2 - global Probability of Boundary + Oriented Watershed Transform + Ultrametric Contour Map (gPb+owt+ucm)
Row 3 - Hierarchical segmentation results

Using Sparse Code Gradients (NIPS 2012)

- Sparse Code Gradients (SCG) - measure contrast using patch representations automatically learned through sparse coding
- Steps:
  - use K-SVD for dictionary learning and Orthogonal Matching Pursuit for computing sparse codes on oriented local neighborhoods
  - apply multi-scale pooling and power transforms
  - classify with linear SVMs



Steps:

Efficiency Over Previous Method

- **preserve fine details**
- **higher precision on large-scale contours**

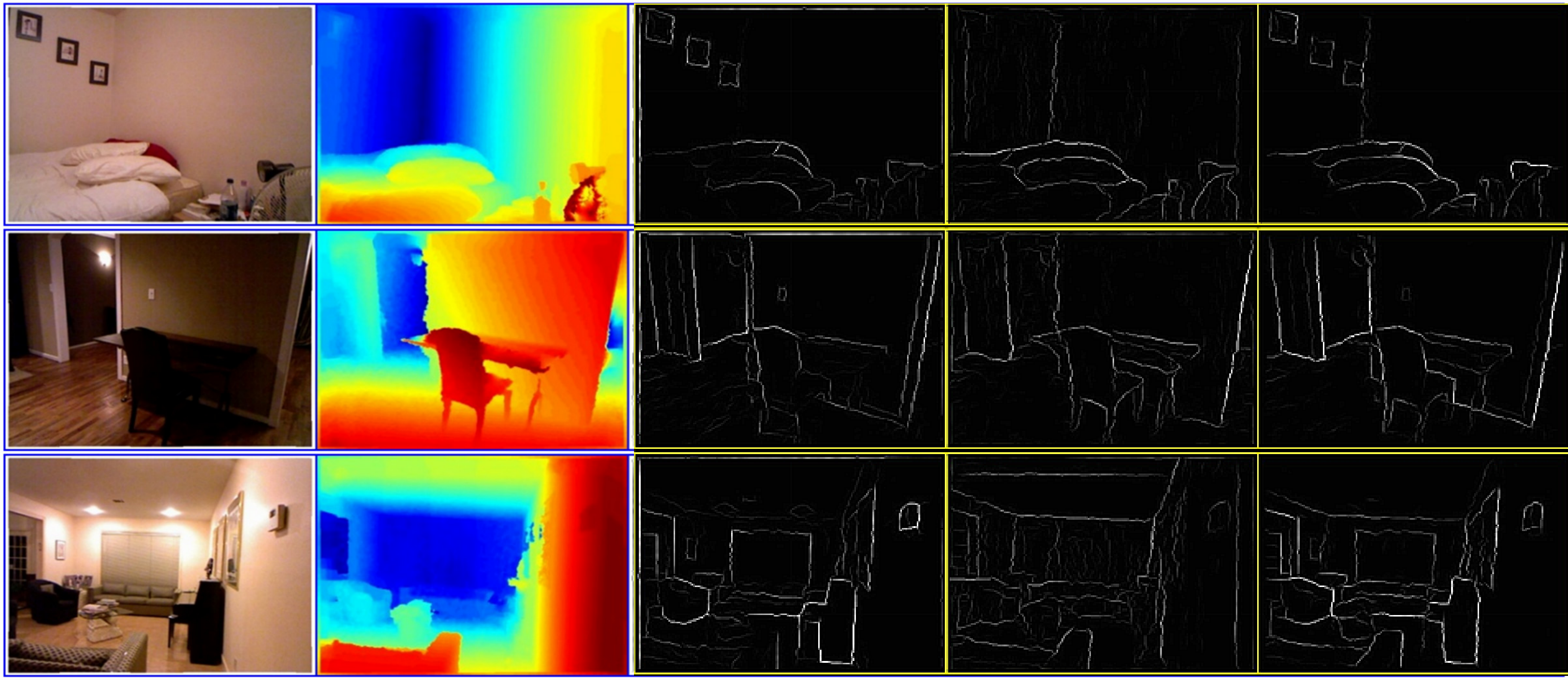
Results (SCG) (BSDS Dataset)

Row 1 - Input Images from BSDS500 Dataset;

Row 2 - gPb+owt+ucm (from the previous method);

Row 3 - SCG output (this work)

Results (SCG) (NYUD Dataset with depth maps)

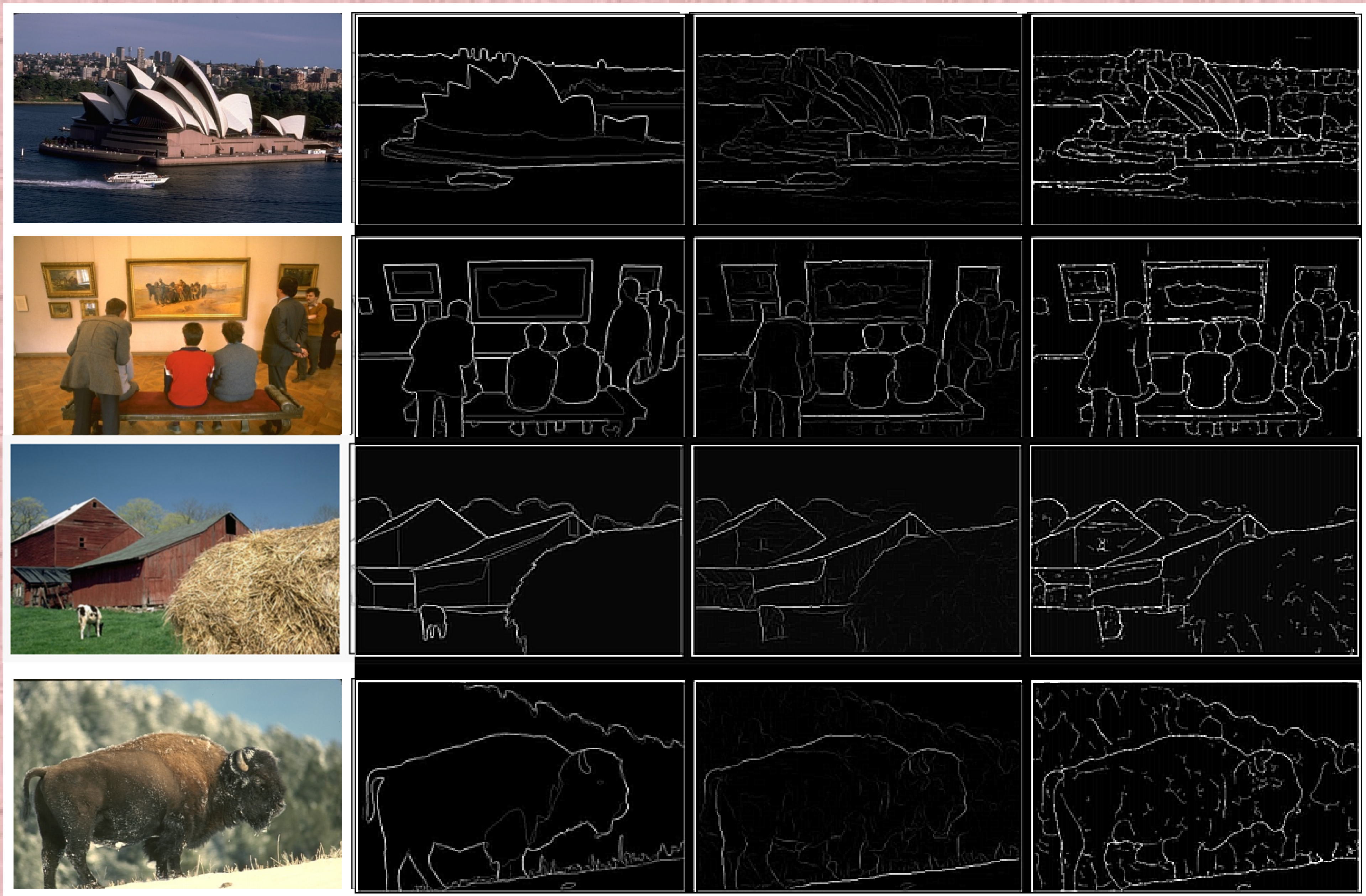


Column 1 - Input Images from NYUD Dataset
Column 2 - Input Depth
Column 3 - SCG contour output with image only
Column 4 - SCG contour output with depth only
Column 5 - SCG contour output with color + depth

Using Sketch Tokens (CVPR 2013)

- Sketch tokens - set of token classes that represent the wide variety of local edge structures that may exist in an image
- Discovered from human-generated image sketches
- Advantage - Captures even more fine details compared to SCG

Results (Sketch Token) (BSDS Dataset)



Col 1 - Input Images from BSDS500 Dataset; Col 2 - Ground Truth;
Col 3 - SCG output ; Col4 - Sketch Token output (this work)

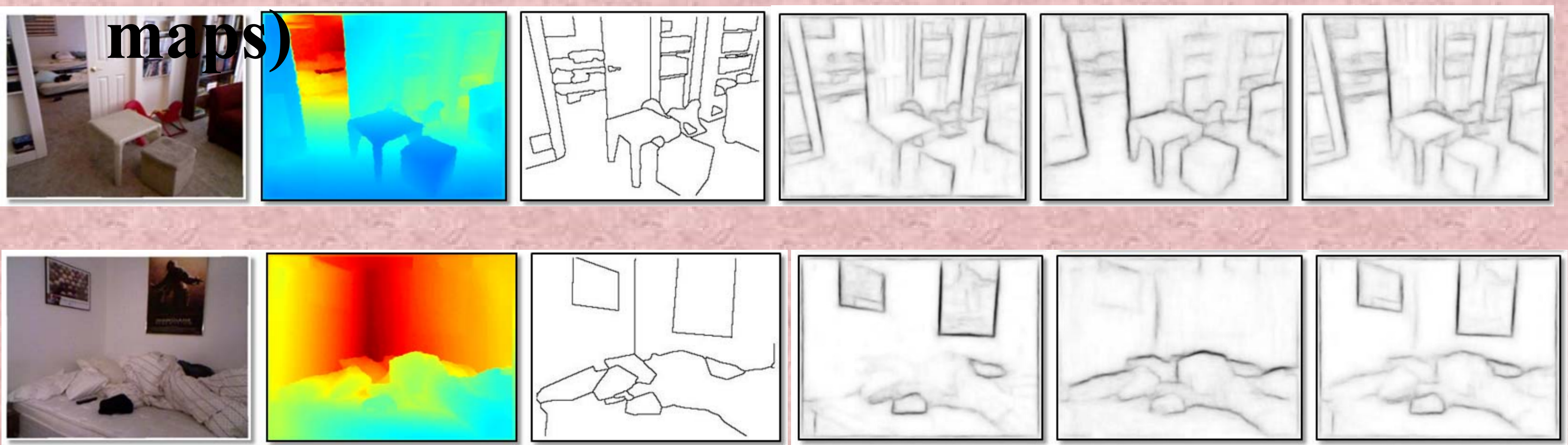
Structured Forest Approach (SF) (ICCV 2013)

- Structured Learning - Edge Masks for the patches in the image
- Random Forest - Capture the structured information
- Outputs of the forests are aggregated across the image to compute our final edge map
- Advantage - Orders of Magnitude Faster than state-of-the-art approaches

Results (SF) - (BSDS Dataset)

Row 1 - Input Images from BSDS500 Dataset; Row 2 - Ground Truth
Row 3 - SCG output ; Row 4 - Structured Edges (SE) – (this work)

Results (SF - NYUD Dataset with depth) maps)

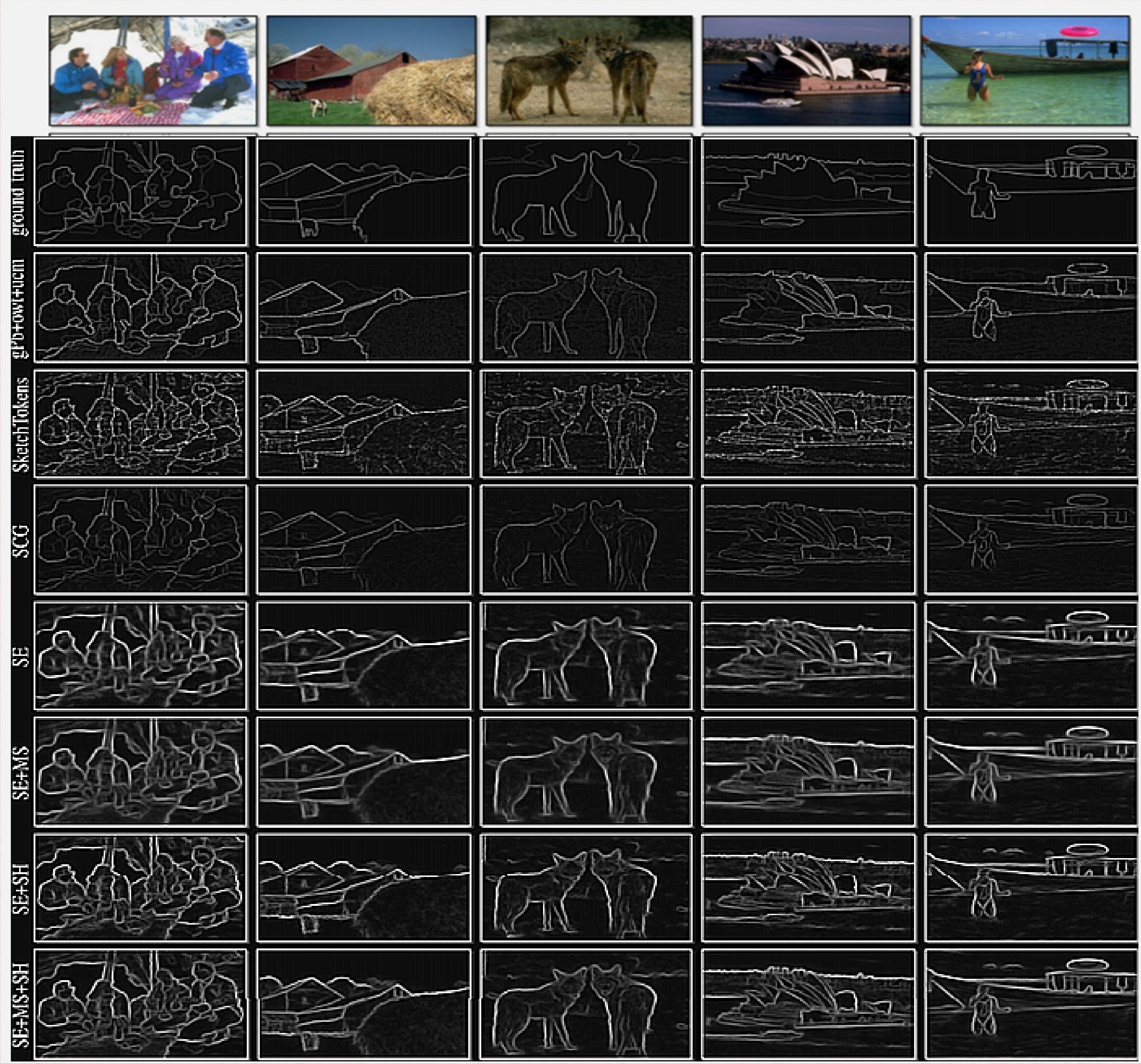


Column 1 - Input Images from NYUD Dataset
Column 2 - Input Depth
Column 3 - Ground Truth
Column 4 - SCG contour output with image only
Column 5 - SCG contour output with depth only
Column 6 - SCG contour output with color + depth

Improved SF (PAMI 2015)

- Edges are Sharpened

Results (BSDS Dataset)



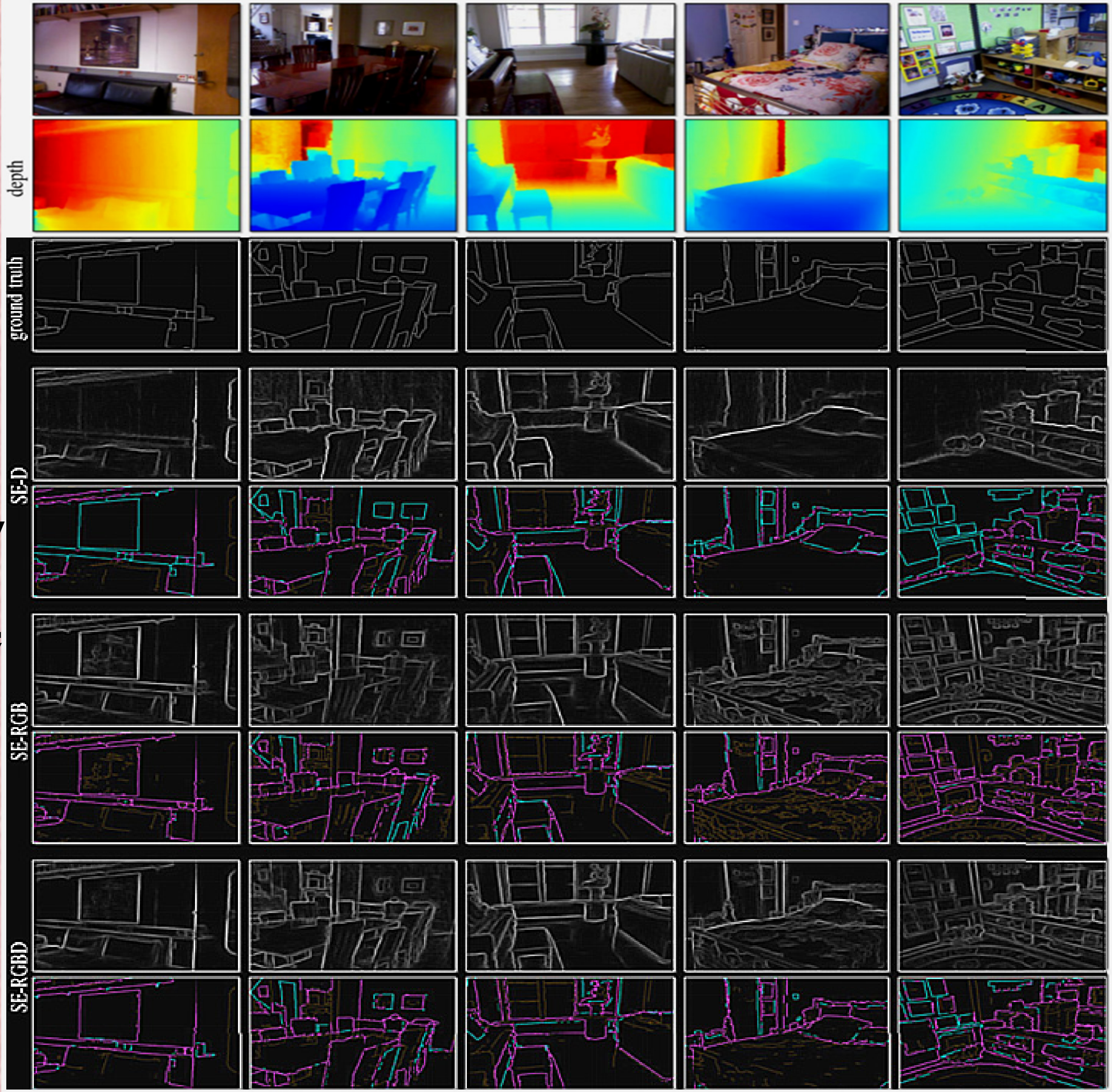
SCG - Sparse Code Gradient

SF - Structured Forest

MS - Multi Scale

SH - Sharpening

Results : (NYUD Dataset)



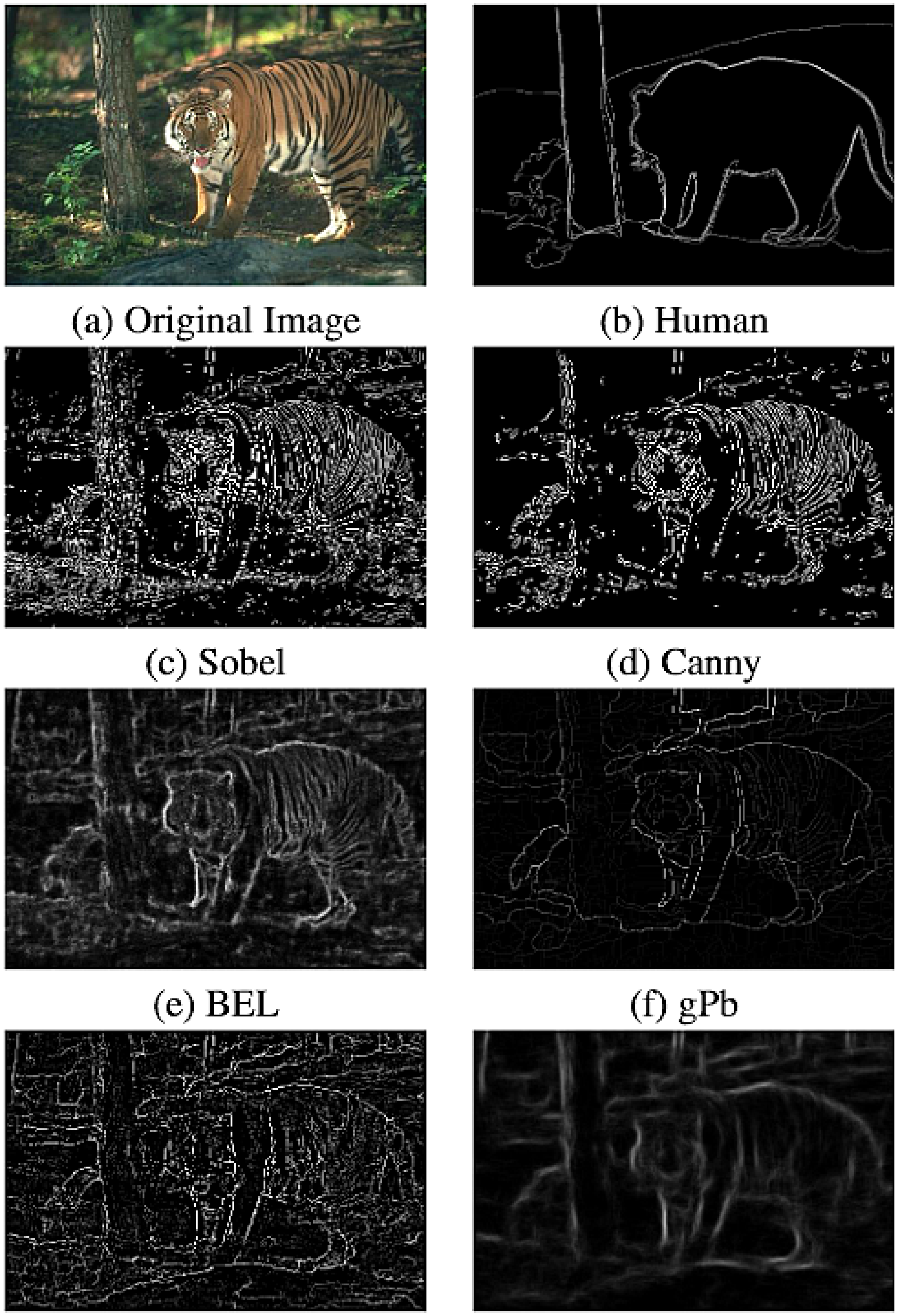
SF-D –
SF using Depth Only

SF-RGB –
SF using Color Only;

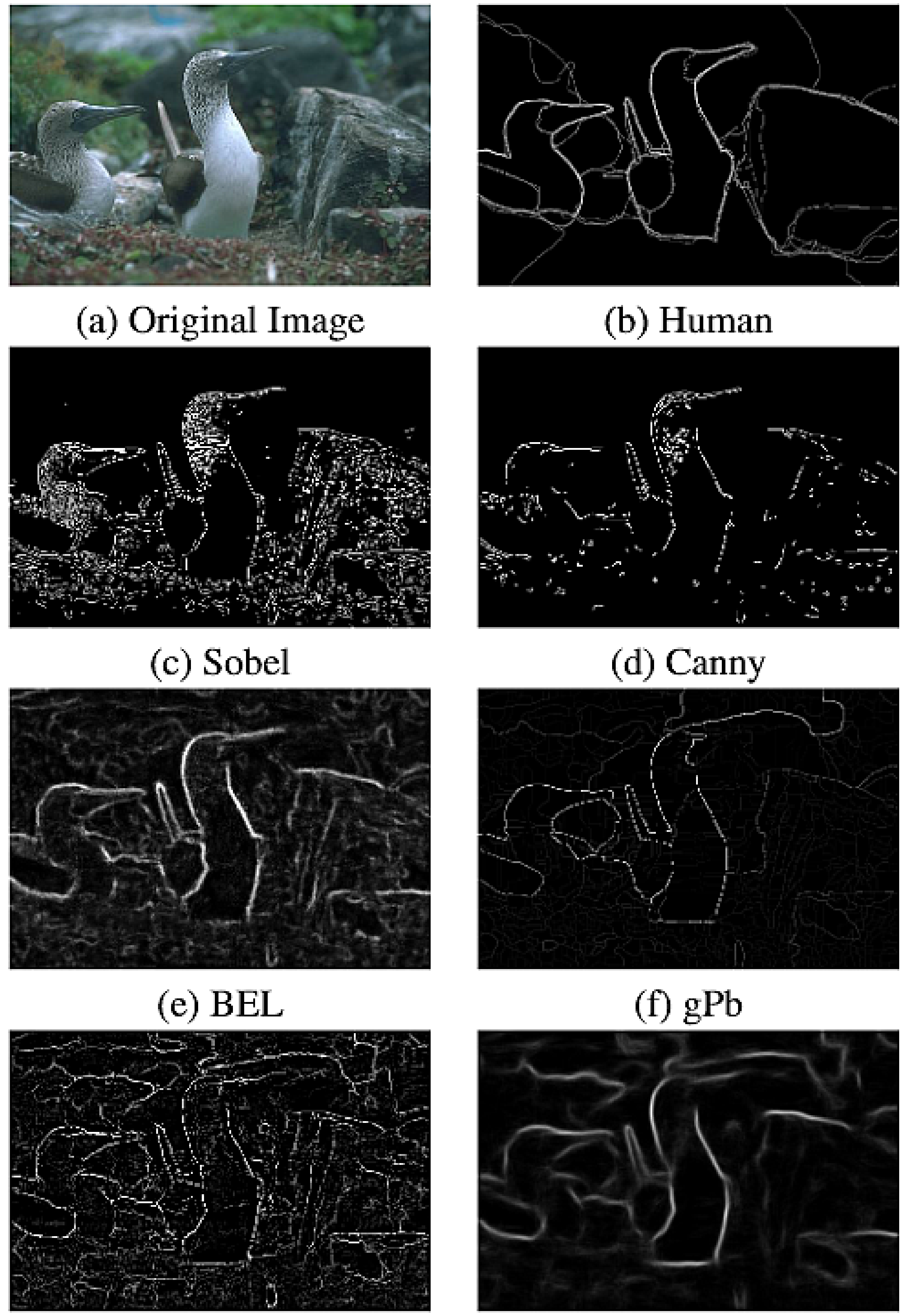
SF-RGBD –
SF using Depth and Color

References

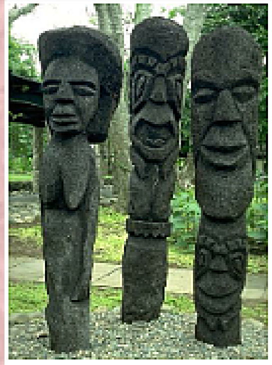
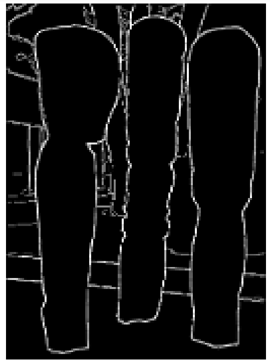
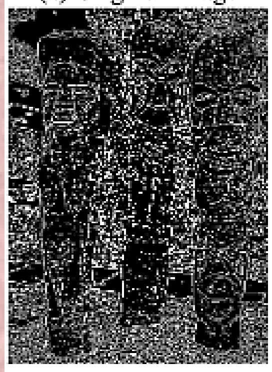
- Martin, David R., Charless C. Fowlkes, and Jitendra Malik. "Learning to detect natural image boundaries using brightness and texture." *Advances in Neural Information Processing Systems*. 2002.
- Martin, David R., Charless C. Fowlkes, and Jitendra Malik. "Learning to detect natural image boundaries using local brightness, color, and texture cues." *Pattern Analysis and Machine Intelligence, IEEE Transactions* (2004): 530-549.
- Arbelaez, P., Maire, M., Fowlkes, C., & Malik, J. (2011). Contour detection and hierarchical image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *33*(5), 898-916. (gPb+owt+ucm)
- Xiaofeng, Ren, and Liefeng Bo. "Discriminatively trained sparse code gradients for contour detection." *Advances in neural information processing systems*. 2012. (SCG)
- Lim, Joseph, C. Zitnick, and Piotr Dollár. "Sketch tokens: A learned mid-level representation for contour and object detection." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013 (Sketch Tokens)
- Dollár, Piotr, and C. Zitnick. "Structured forests for fast edge detection."*Proceedings of the IEEE International Conference on Computer Vision*. 2013. (SE)
- Dollár, Piotr, and C. Lawrence Zitnick. "Fast edge detection using structured forests." *Pattern Analysis and Machine Intelligence, IEEE Transactions on;* 37/8 (2015): 1558-1570.

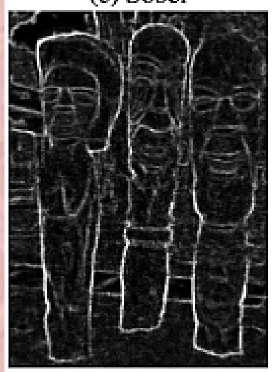
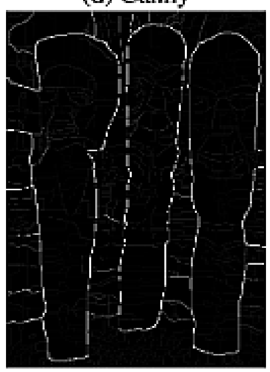
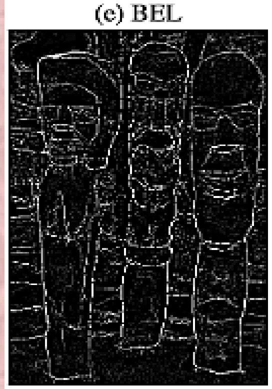
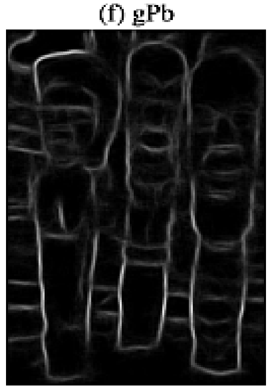


(a) Original Image
(b) Human
(c) Sobel
(d) Canny
(e) BEL
(f) gPb
(g) Sketch Token
(h) Structured Forest



(a) Original Image
(b) Human
(c) Sobel
(d) Canny
(e) BEL
(f) gPb
(g) Sketch Token
(h) Structured Forest


(a) Original Image

(b) Human

(c) Sobel

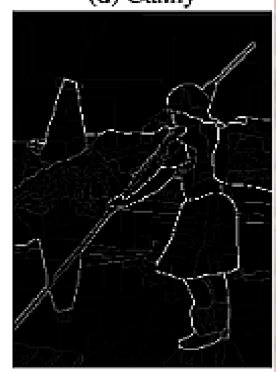
(d) Canny

(e) BEL

(f) gPb

(g) Sketch Token

(h) Structured Forest

(a) Original Image

(b) Human

(c) Sobel

(d) Canny

(e) BEL

(f) gPb

(g) Sketch Token

(h) Structured Forest

https://en.wikipedia.org/wiki/File:PST_edge_detector_saint_Paul.tif

(a) Image (c) SE (c) EdgeNet

OK;

Let’s get past the edge now.