

Concepts in Edge Detection

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

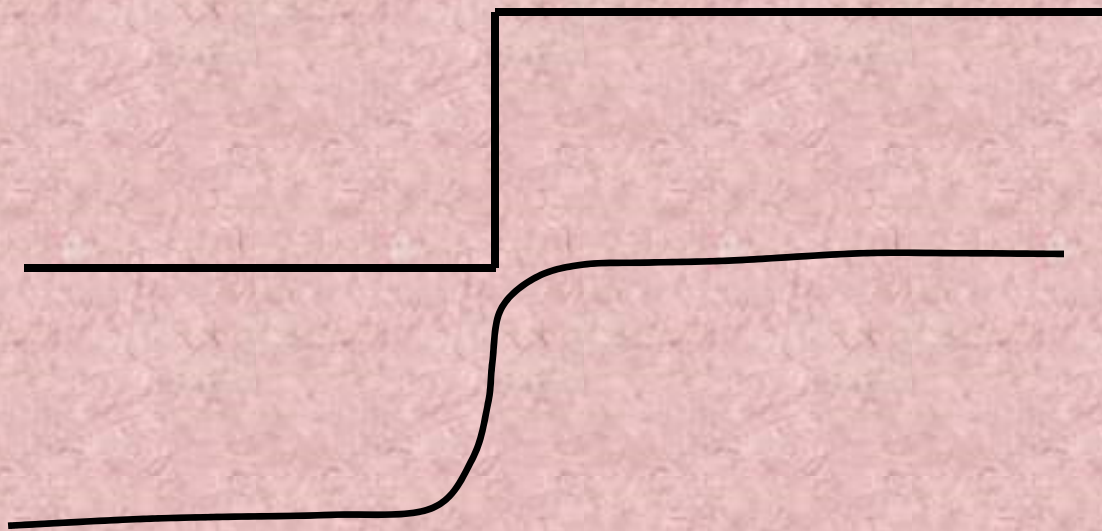
<http://www.cs.iitm.ernet.in/~sdas>
Email: sdas@iitm.ac.in

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

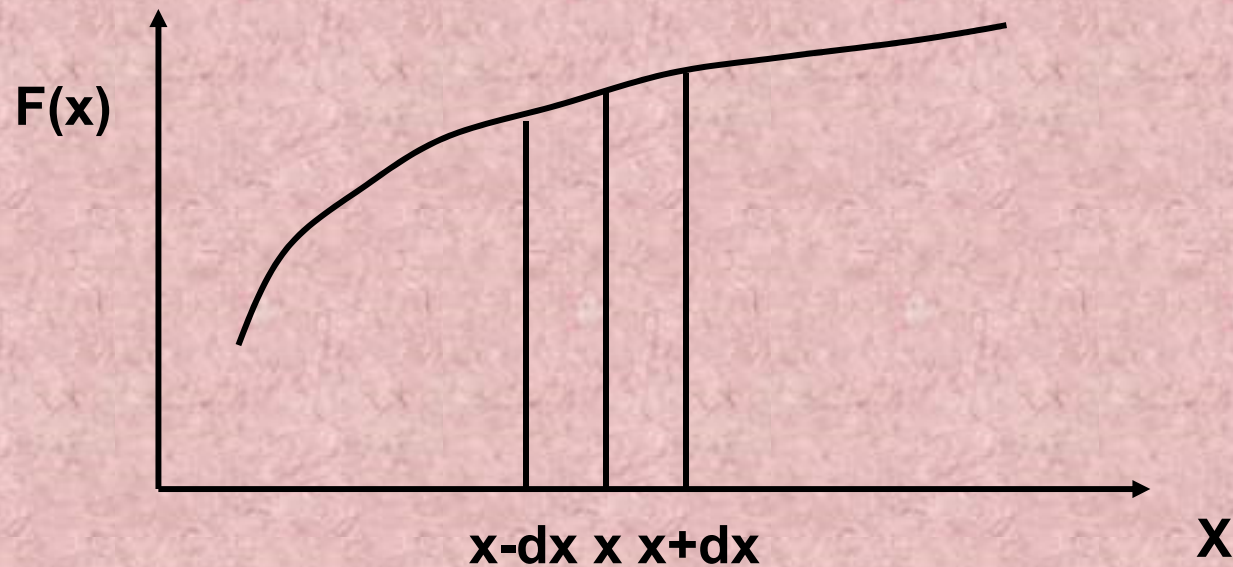


Step edge in 2-D

Computing the derivative: Finite difference in 1-D

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

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



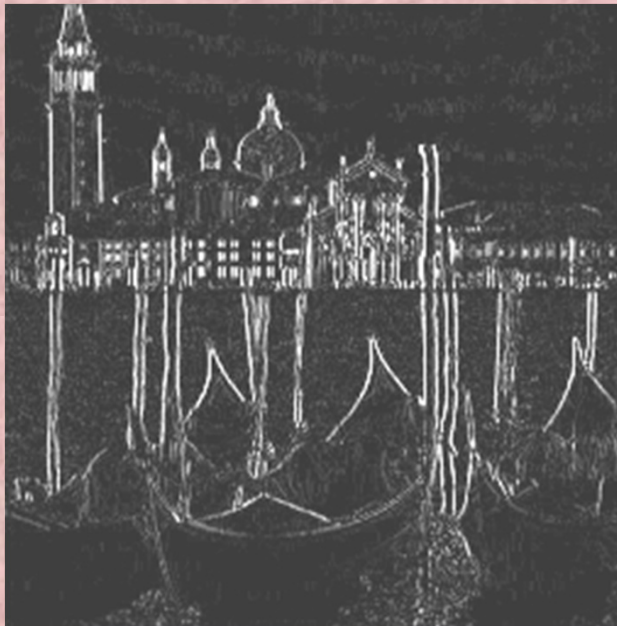
Computing the derivative: Finite differences in 2-D

$$\frac{\partial f}{\partial x} \approx \frac{f(x + dx, y) - f(x, y)}{dx} \approx \frac{f(x + dx, dy) - f(x - dx, dy)}{2dx}$$

$$\frac{\partial f}{\partial y} \approx \frac{f(x, y + dy) - f(x, y)}{dy} \approx \frac{f(x, y + dy) - f(x, y - dy)}{2dy}$$



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

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

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

Two components of the edge values computed are:

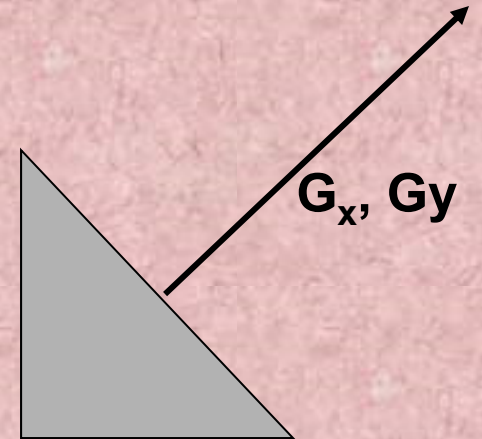
Gradient values: $G_x = \delta f / \delta x$; $G_y = \delta f / \delta y$.

The magnitude of the edge is calculated as:

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

and orientation as:

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

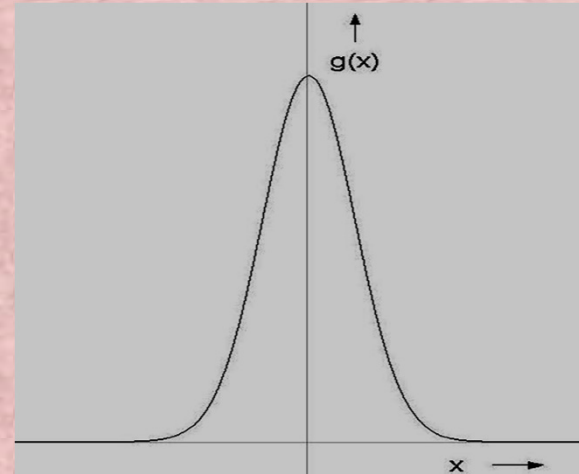


Most of these partial derivative operators are sensitive to noise. Use of these masks resulted in thick edges or boundaries, in addition to spurious edge pixels due to noise.

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:

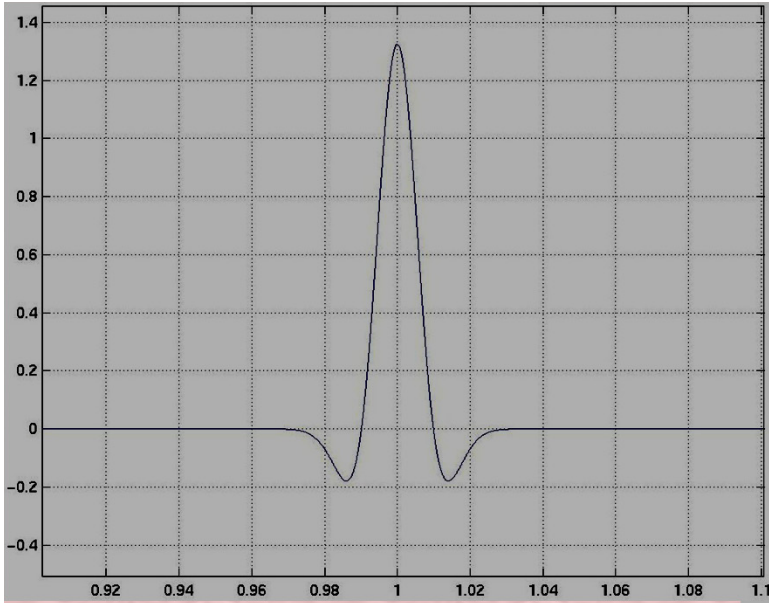
$$g(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$$



Gaussian Function

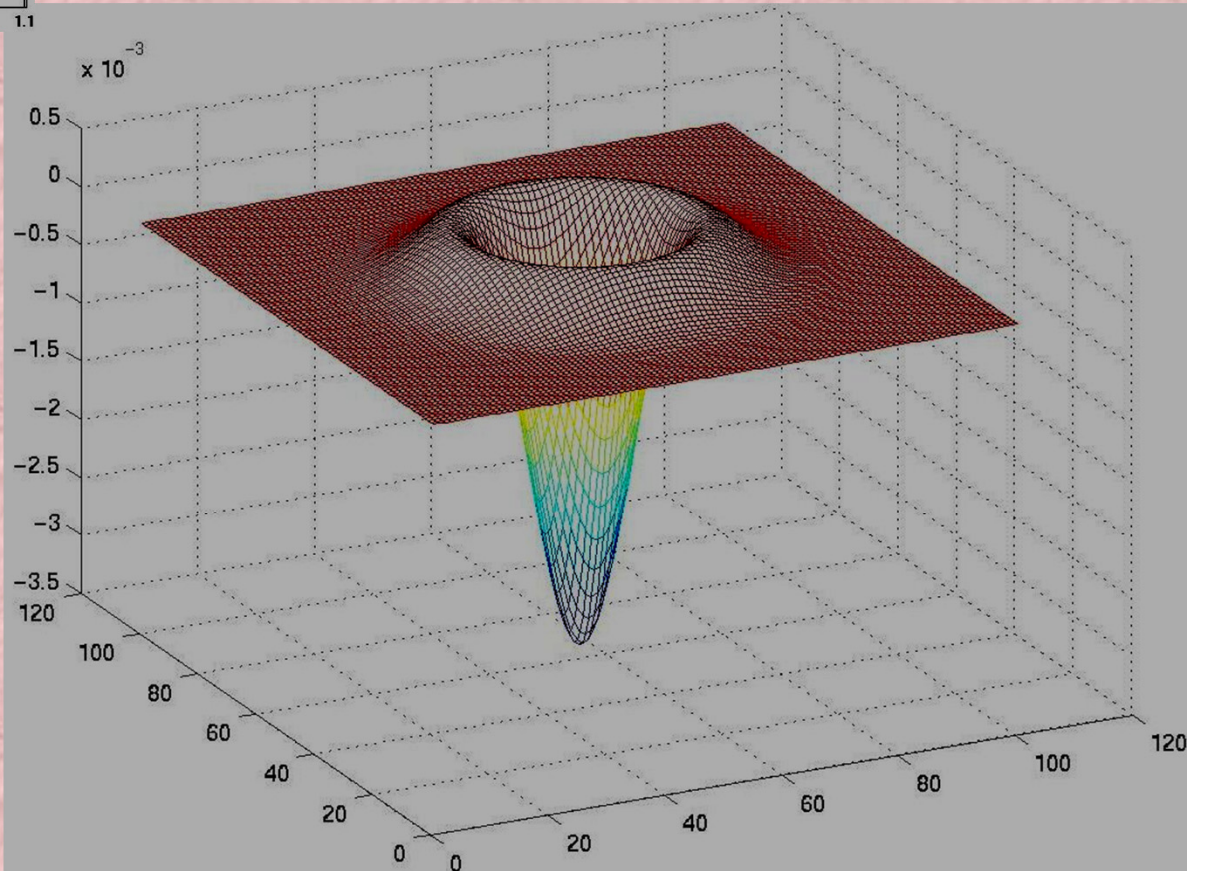
Marr and Hildreth (1980) suggested the use of the “Laplacian of the Gaussian” (LOG) operator to detect edges. This produced edges as Zero-Crossings (ZC’s) in the output function - why??

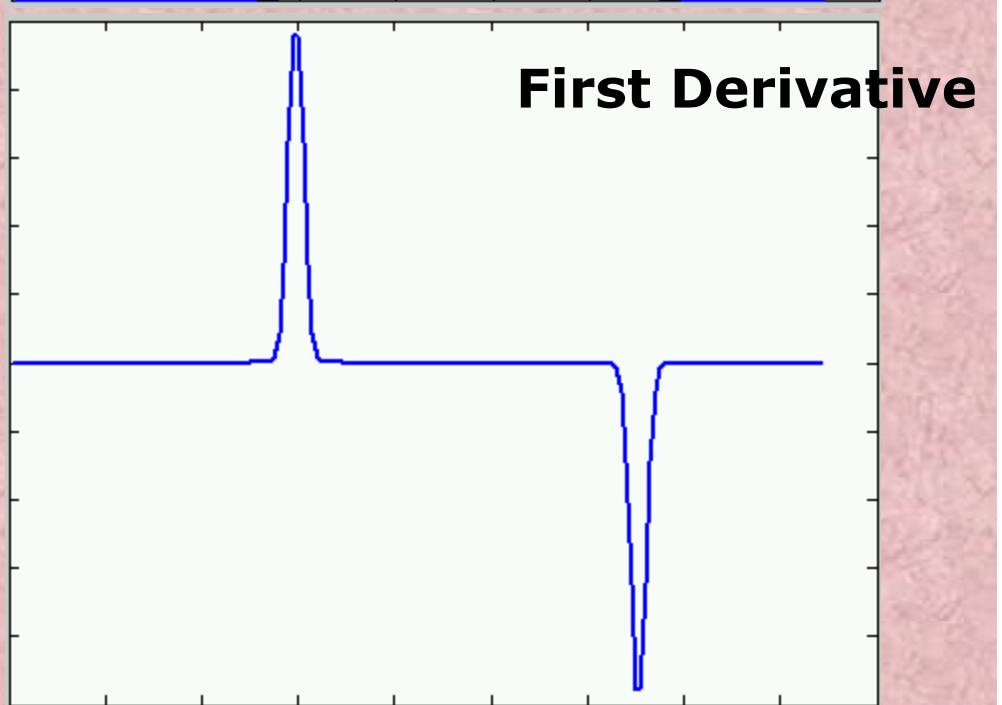
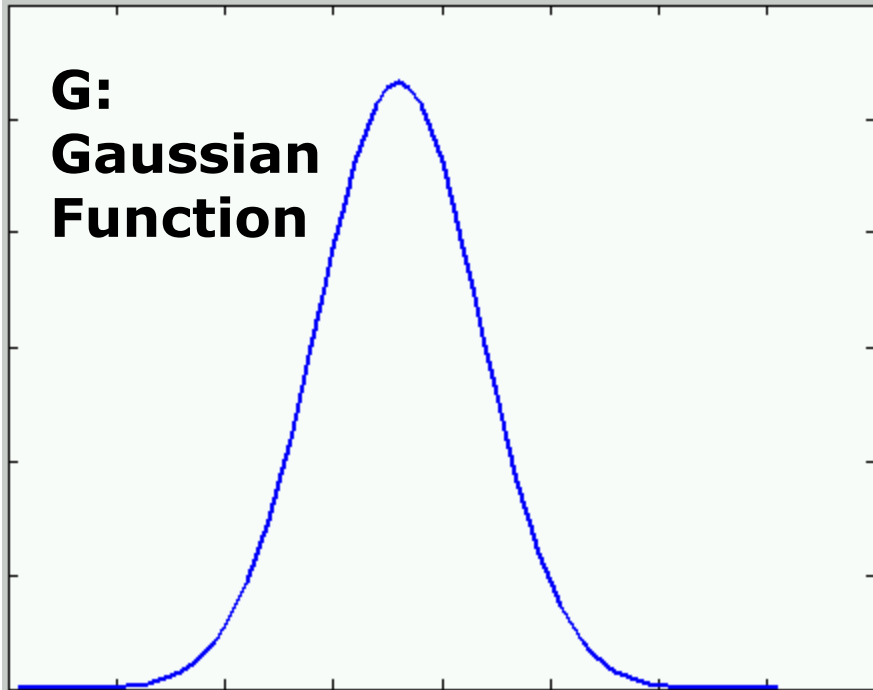
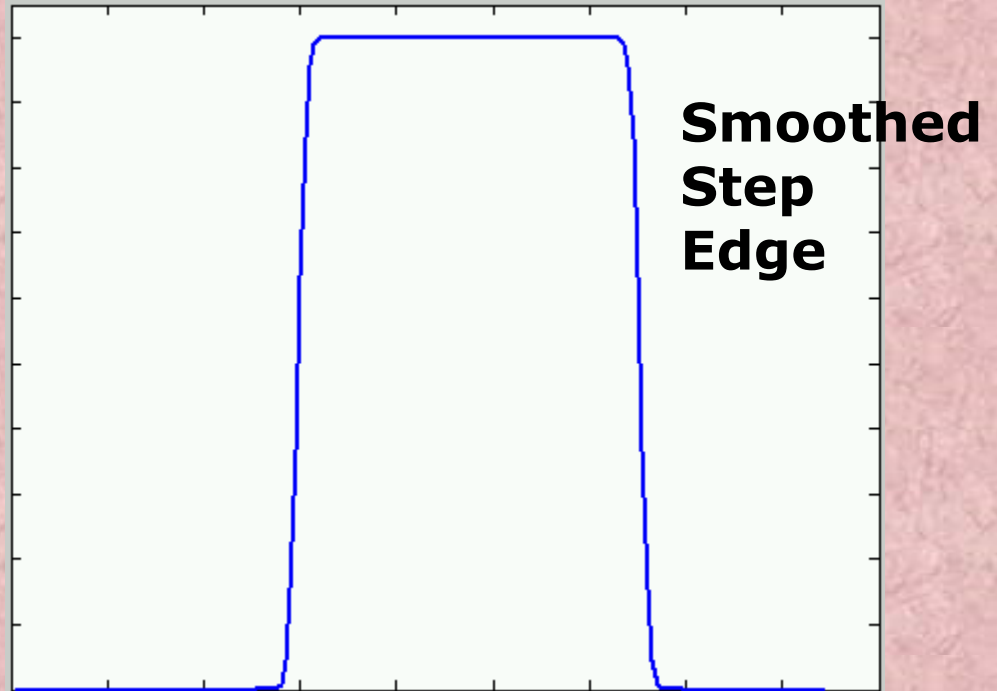
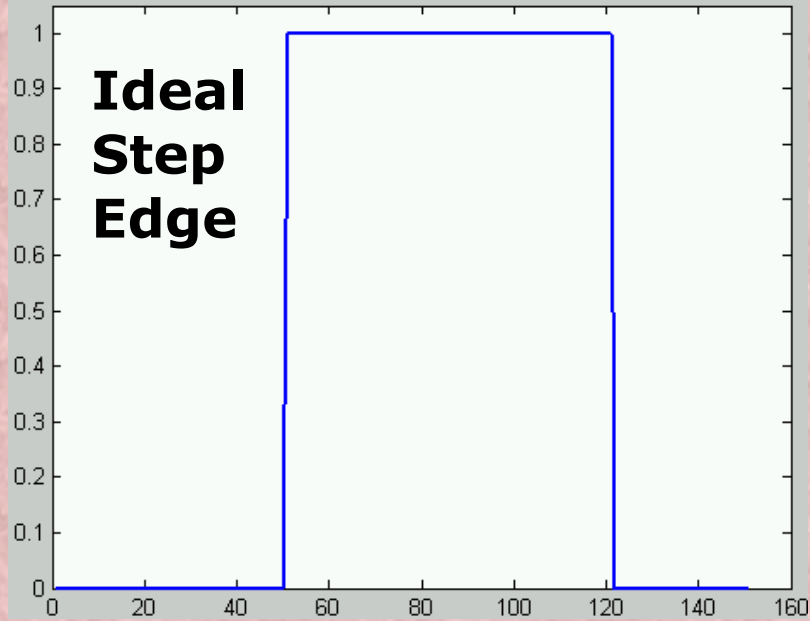
However, it did not give any idea of the gradient magnitude or orientation of the edges. But ZC’s were spread through-out an image. How do one detect true edges from ZC’s??



LOG operator in 2-D

LOG operator in 1-D

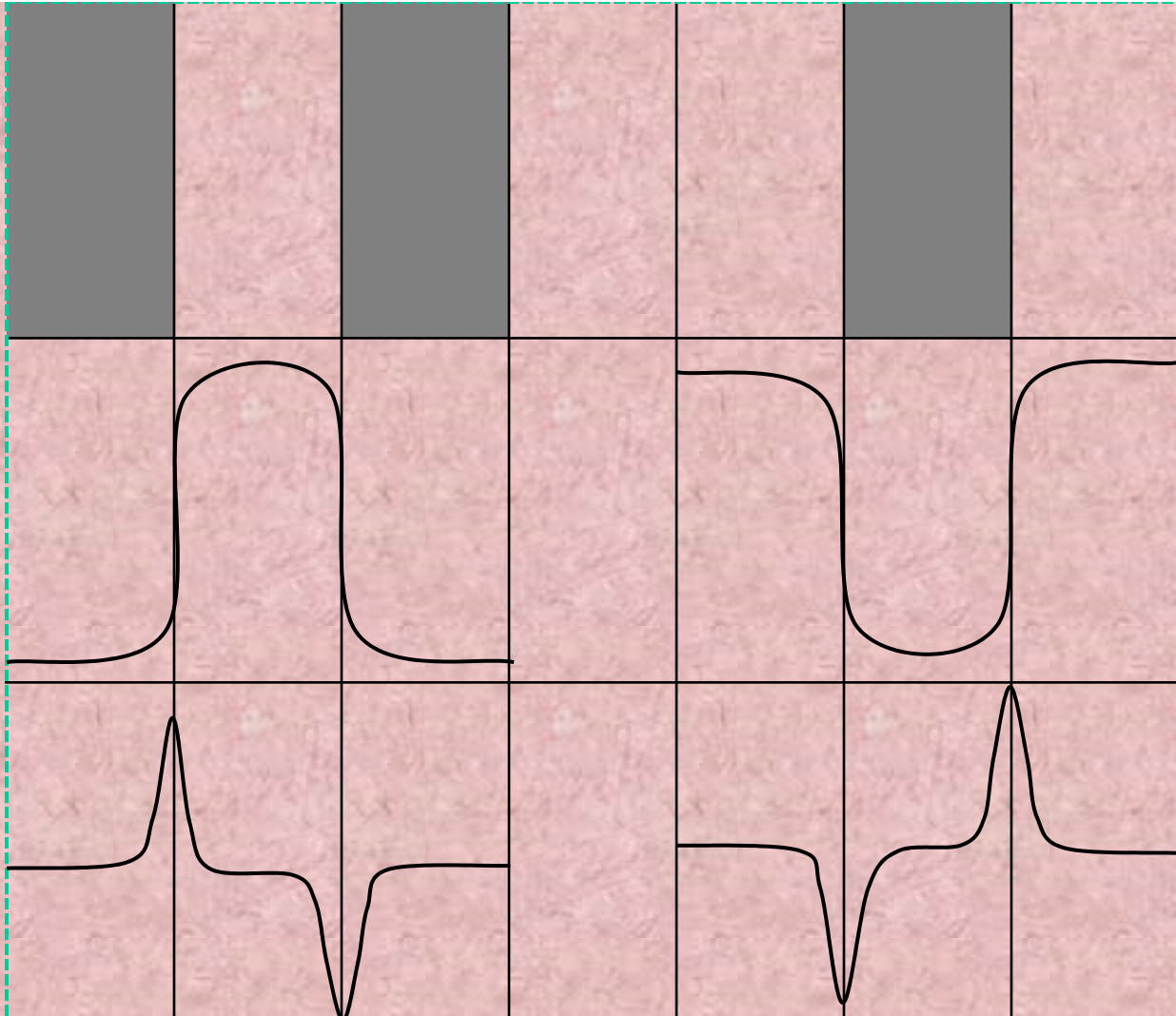




Images

**Horizontal
Intensity
Profiles**

**First
Derivative**

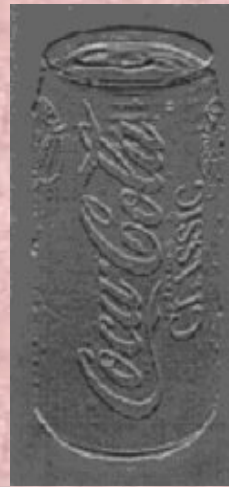




Original Gray
scale Image



X-gradient



Y-gradient



Total Gradient
Magnitude



Bi level
Thresholding



Original Grey
level Image



After Laplace Operator



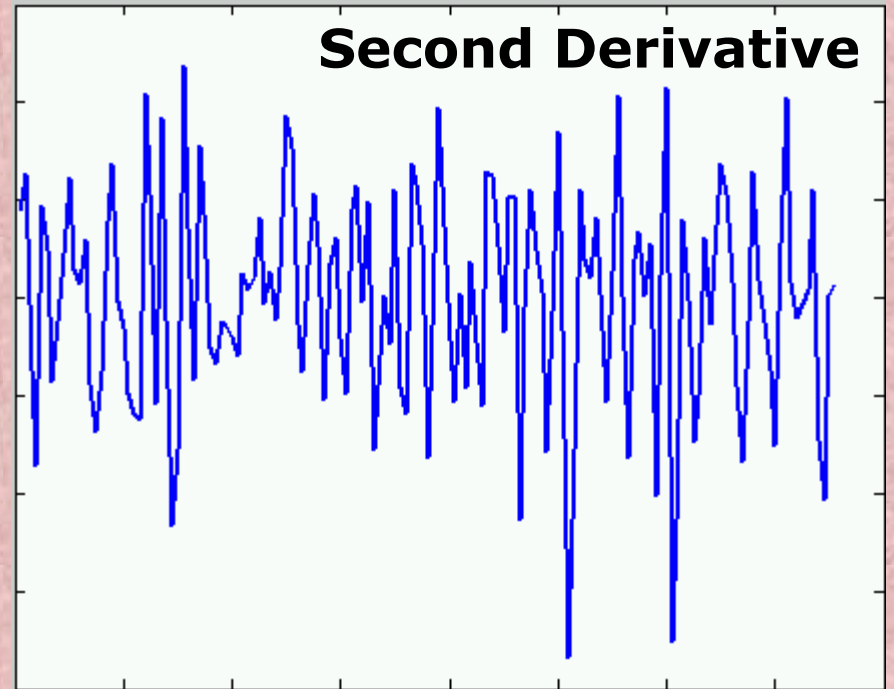
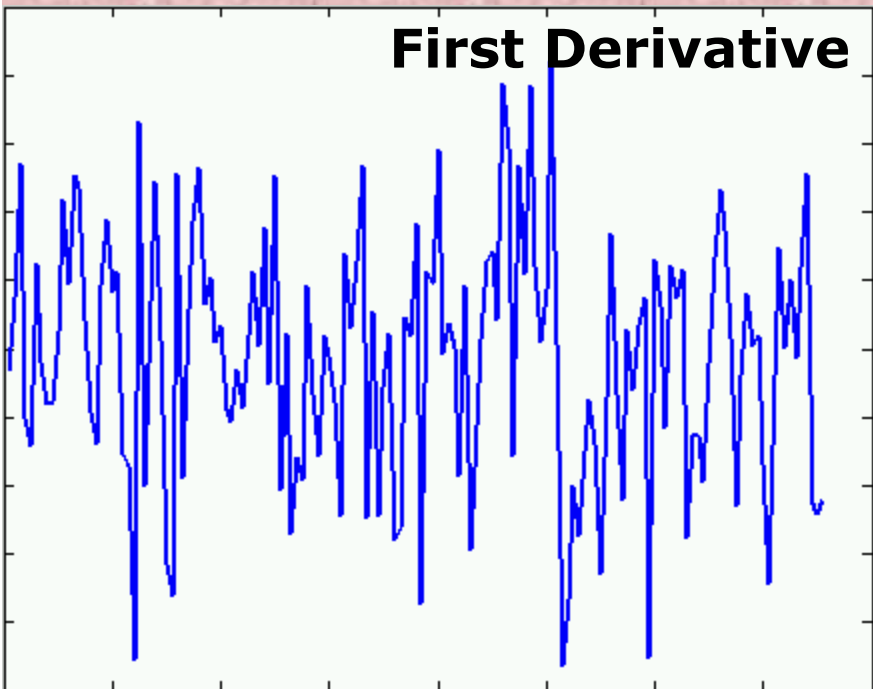
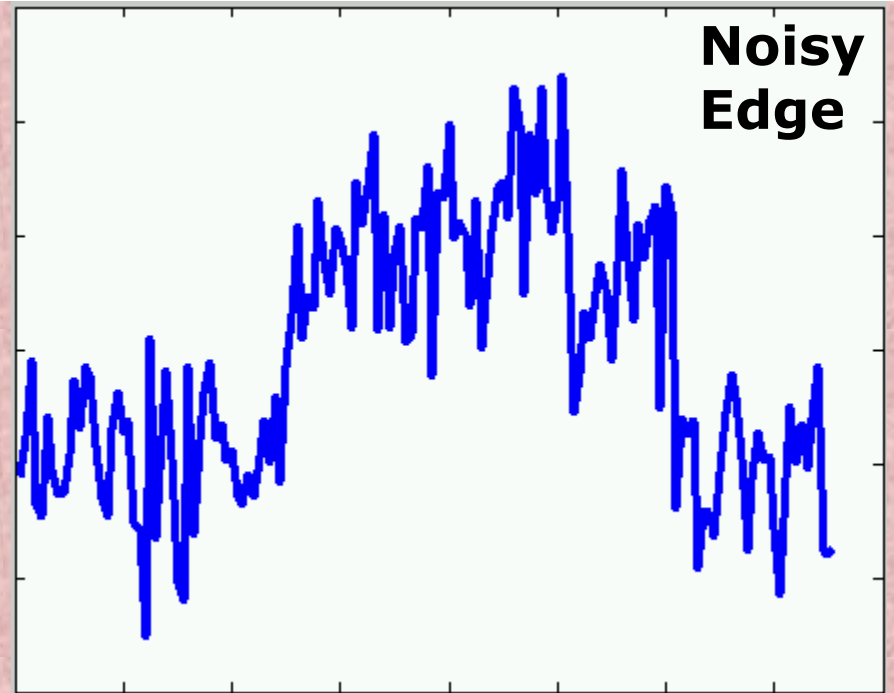
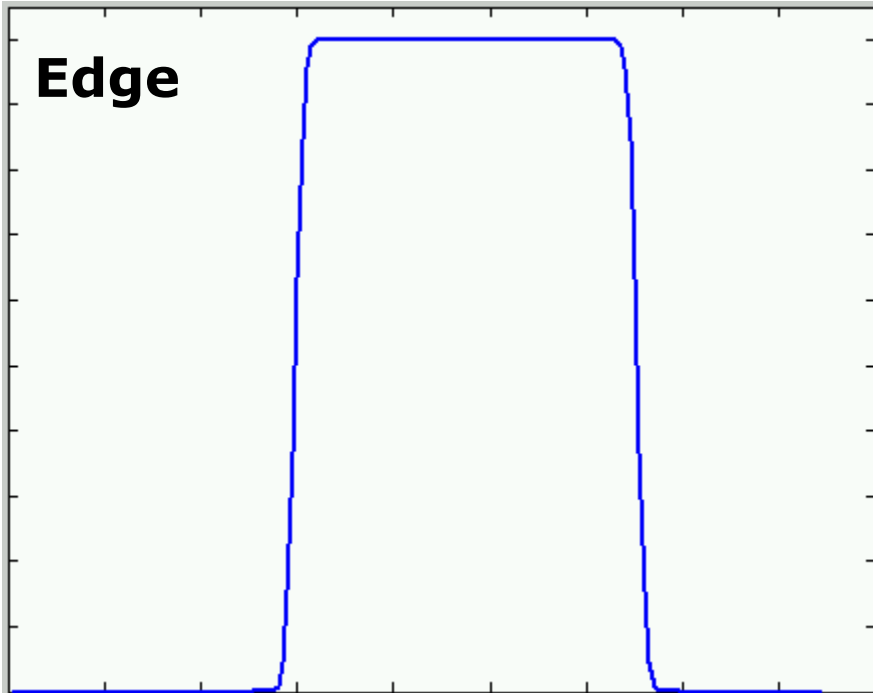
After Zero-crossing



δF



$\delta^2 F$



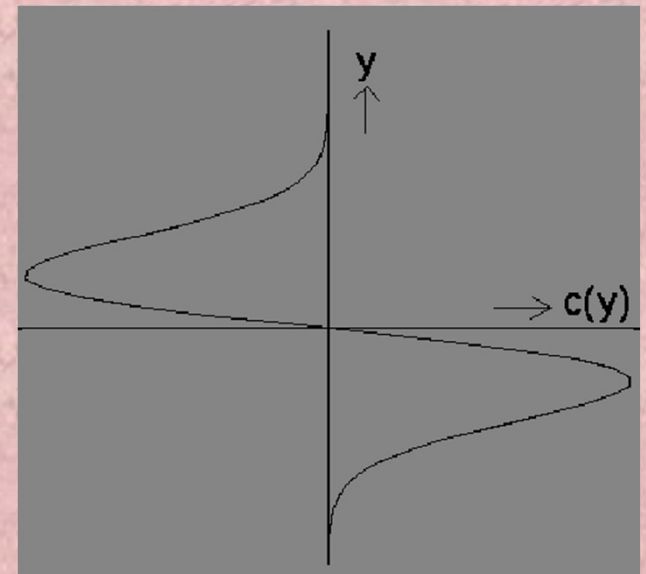
Canny in 1986 suggested an optimal operator, which uses the Gaussian smoothing and the derivative function together. He proved that the first derivative of the Gaussian function, as shown below, is a good approximation to his optimal operator.

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.

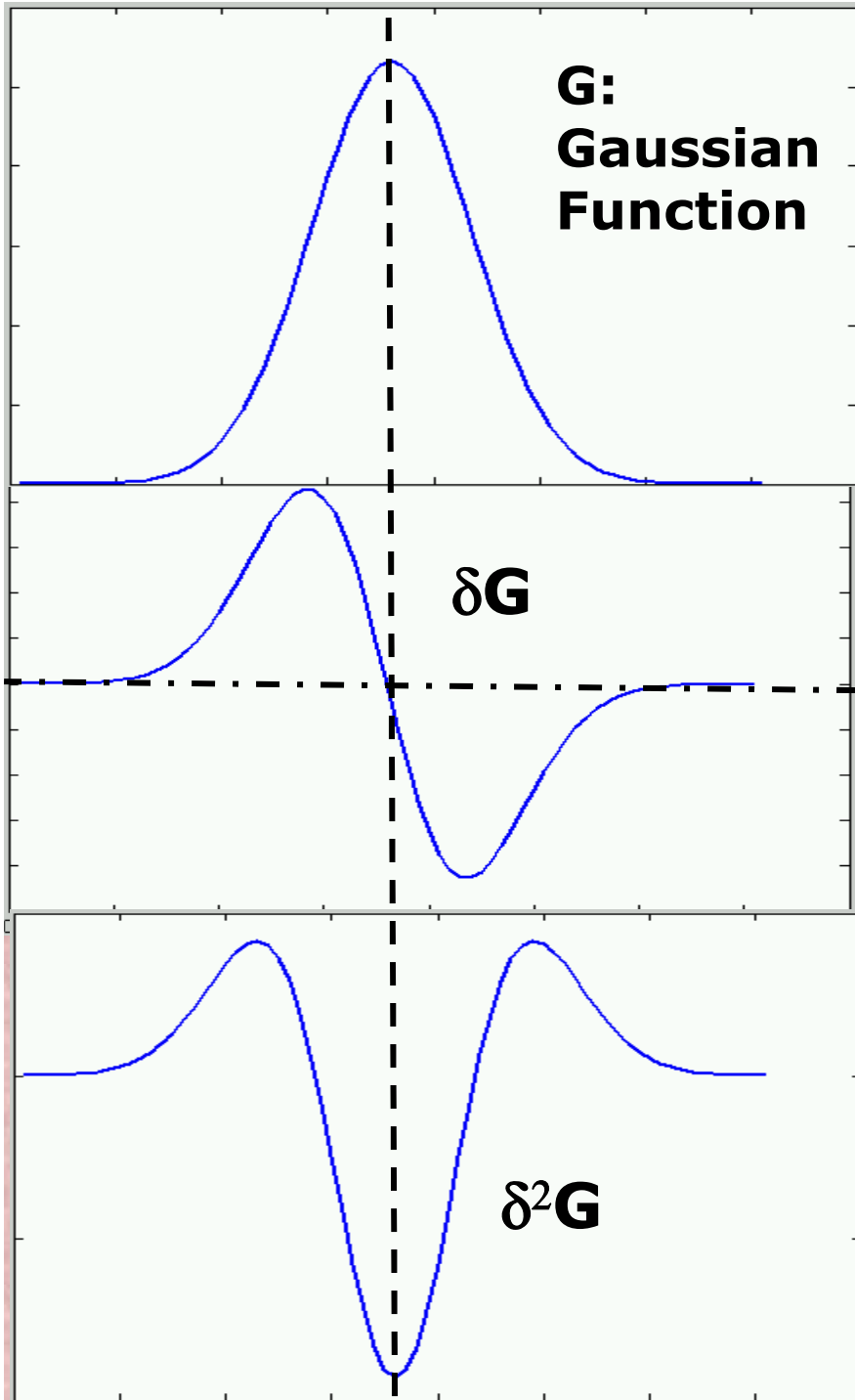
∇G is the first derivative of the Gaussian defined (in 1-D) as:

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

and

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

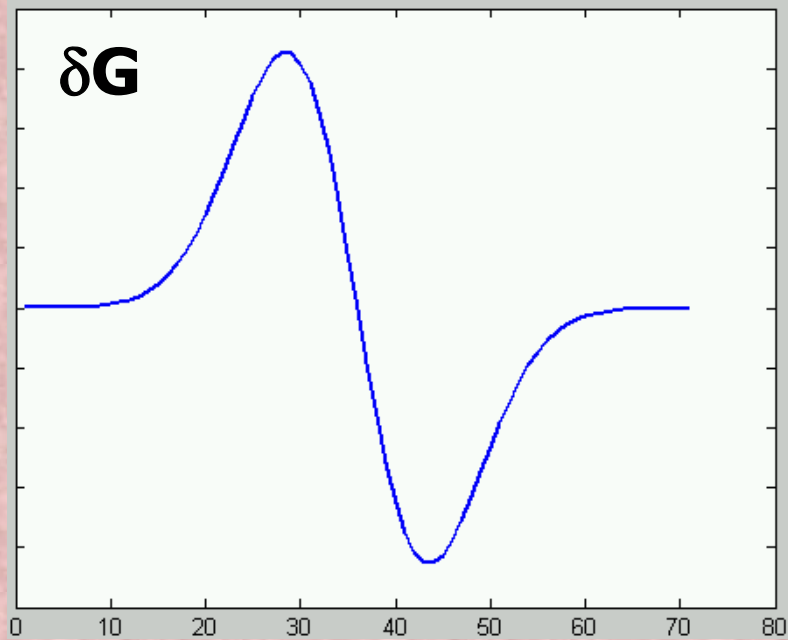
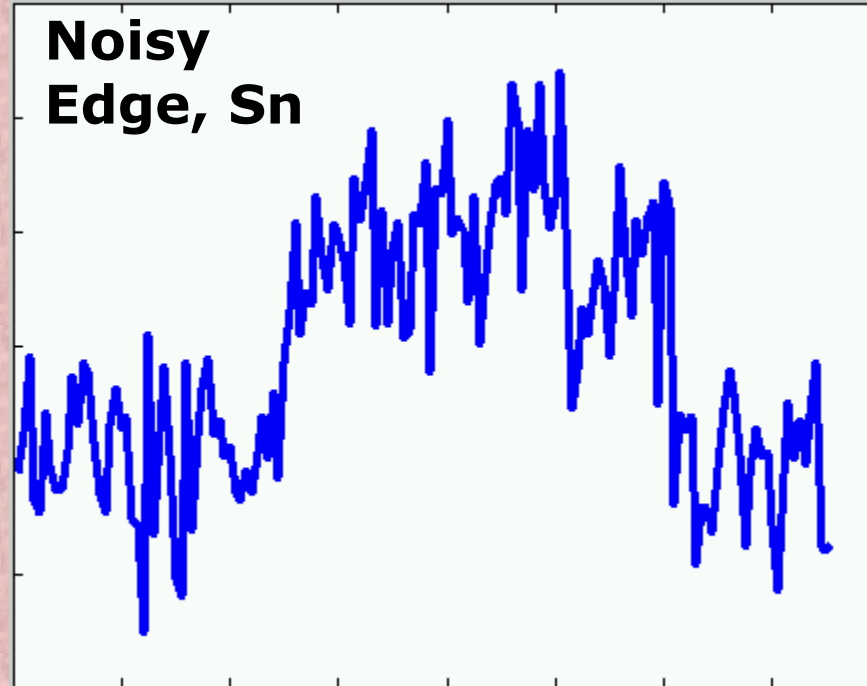
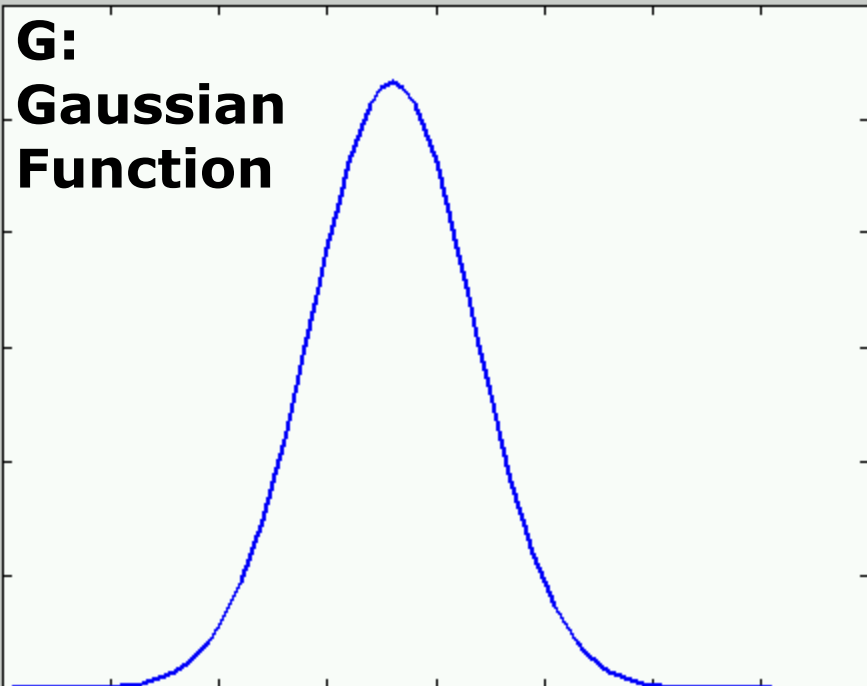
$$\nabla^2 G(r) = \left(\frac{1}{\pi\sigma^4}\right)(r^2/2\sigma^2 - 1) \exp\left(\frac{-r^2}{2\sigma^2}\right)$$

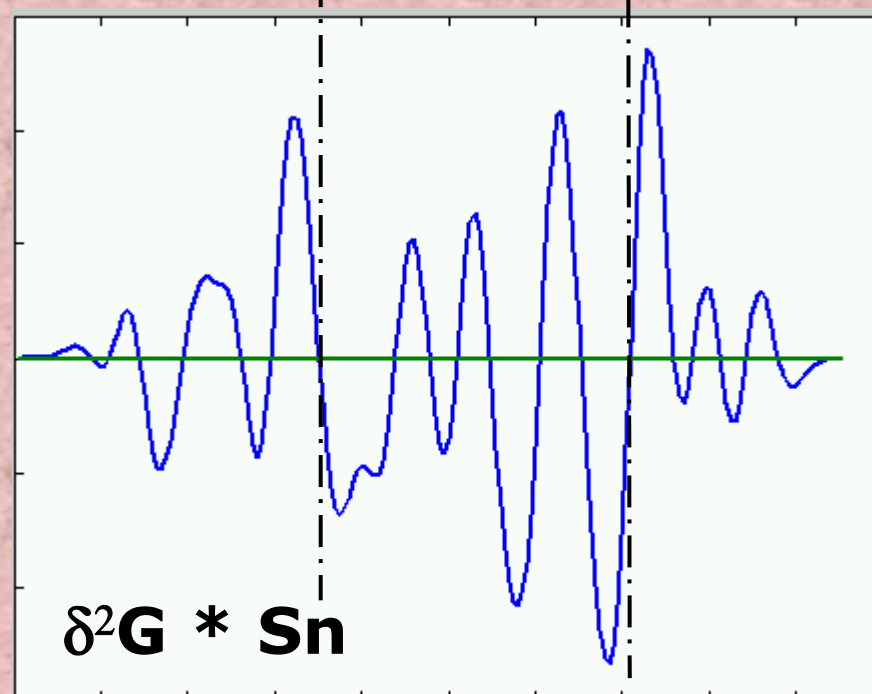
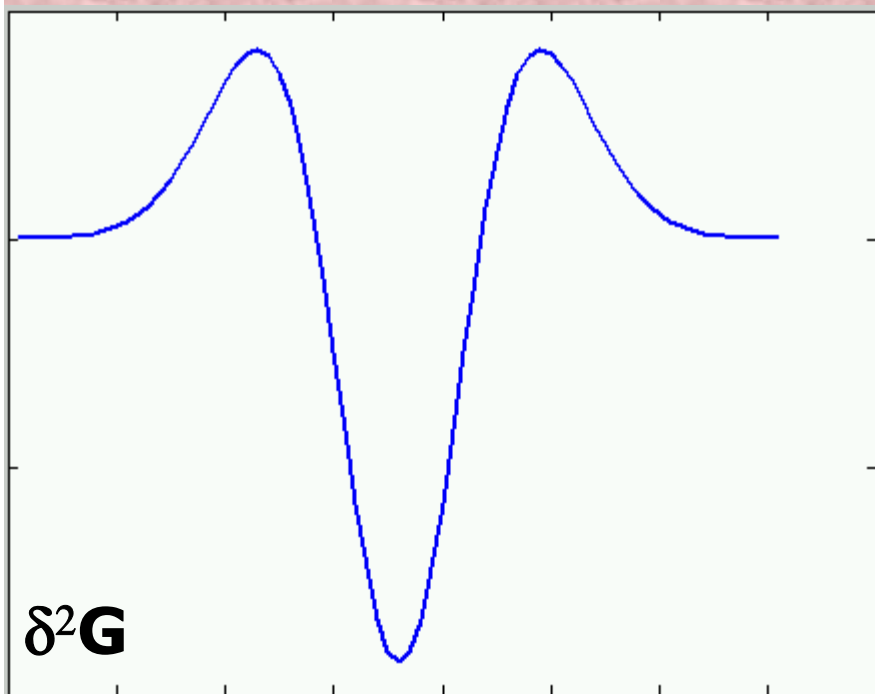
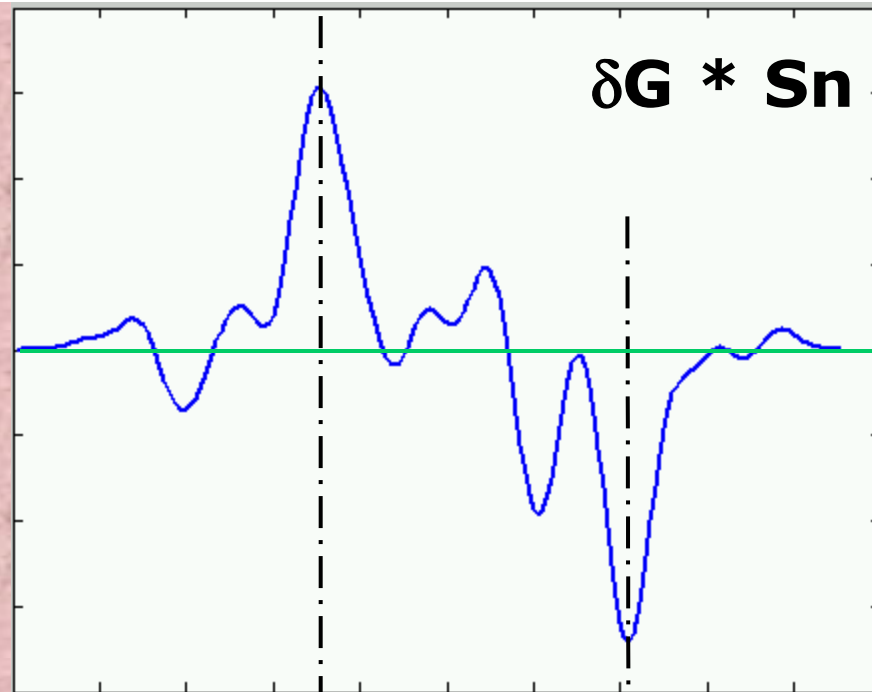
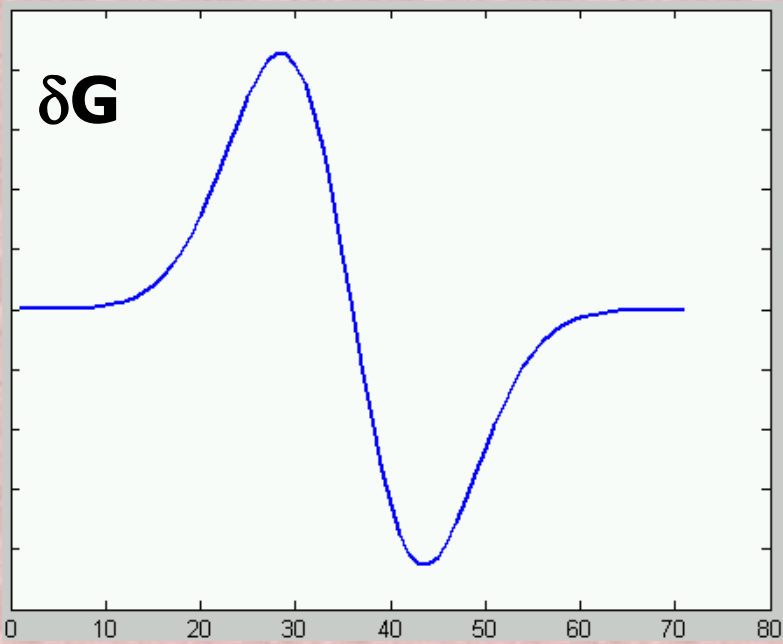


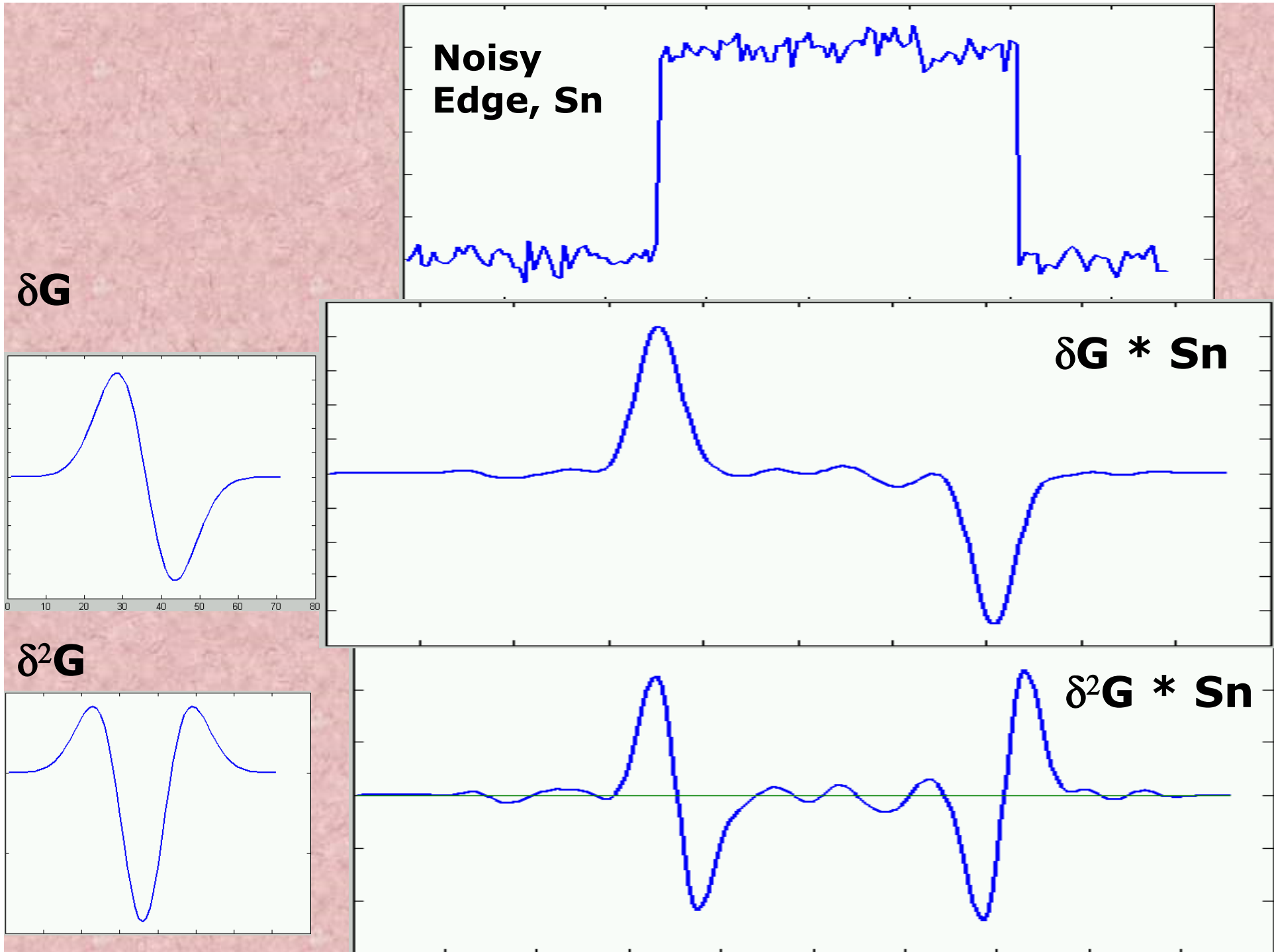
$$g(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2}{2\sigma^2}}$$

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

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







Lena



Sobel



LOG



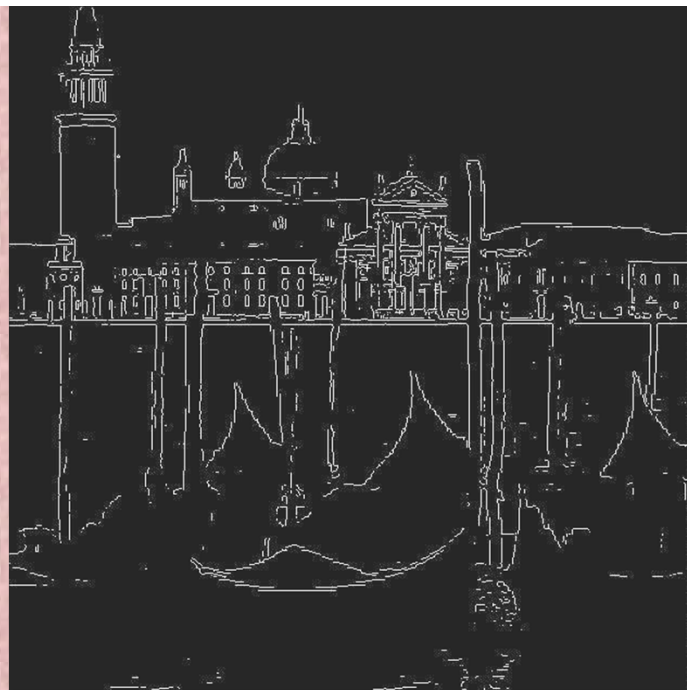
Canny



Venice



Sobel



LOG



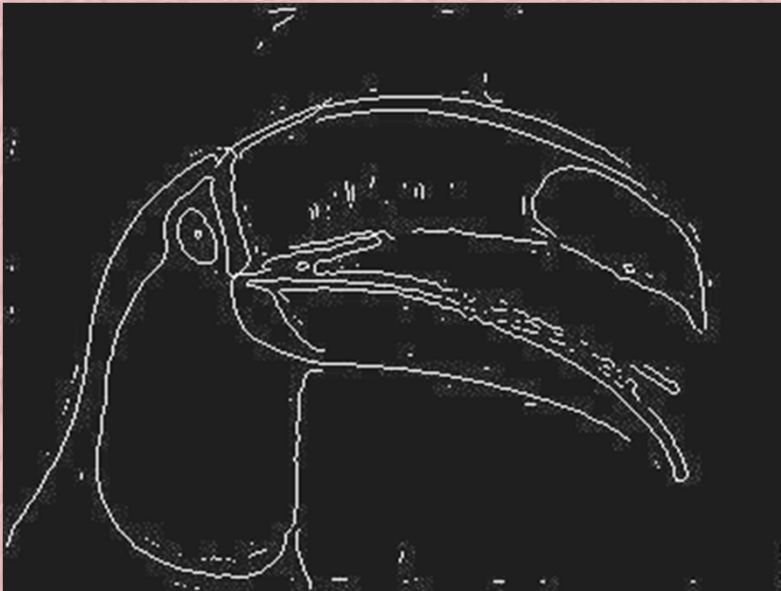
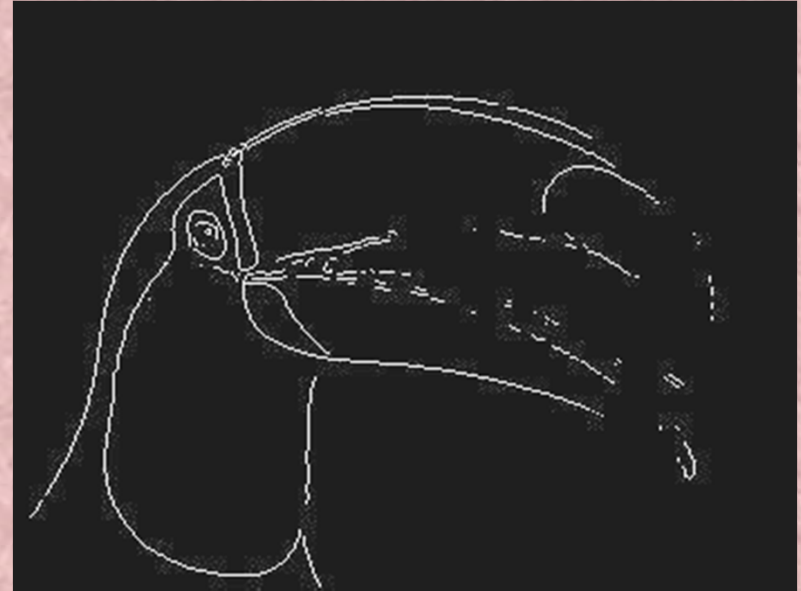
Canny



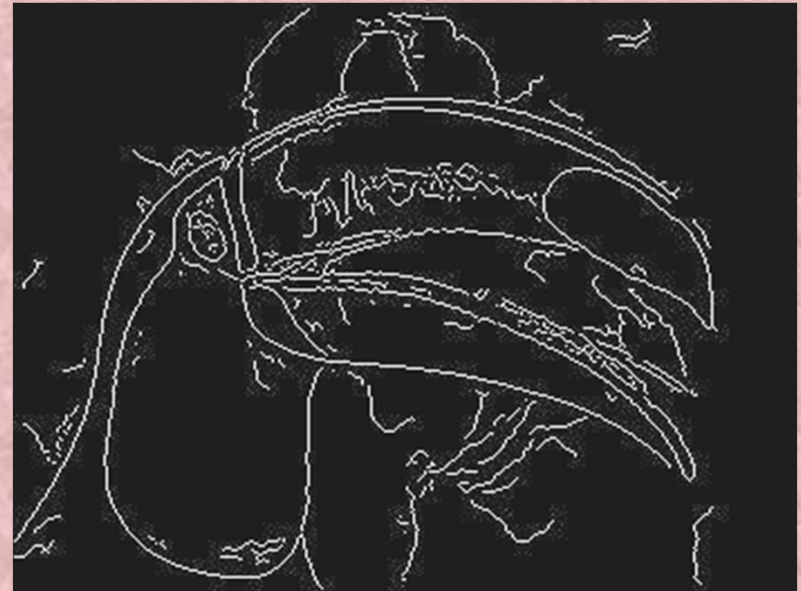
BIRD



SOBEL



LOG



Canny

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

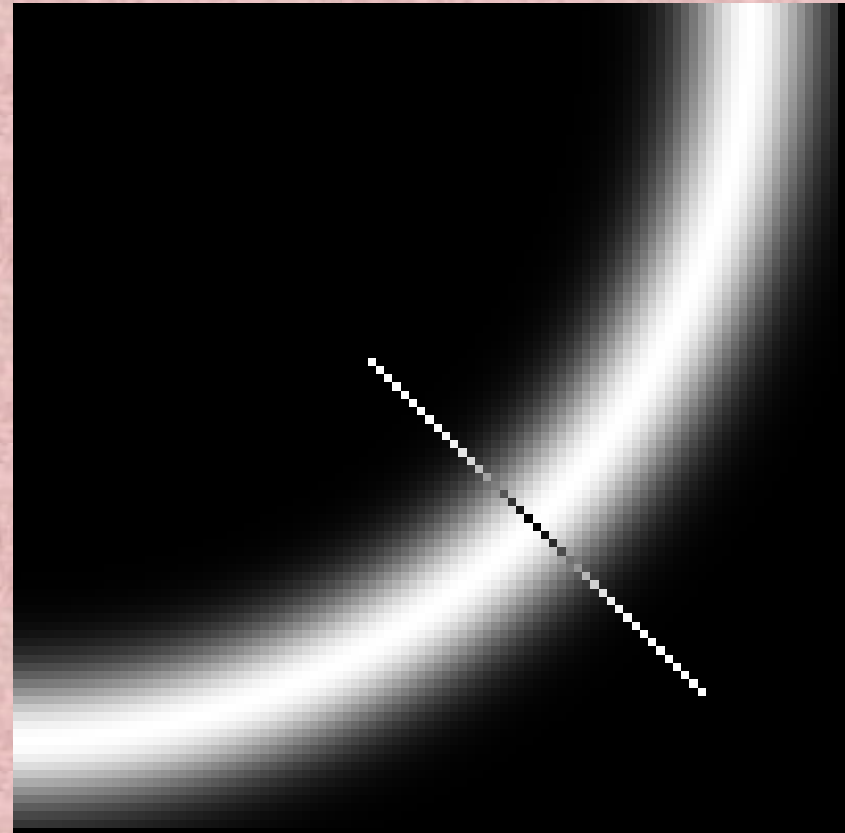
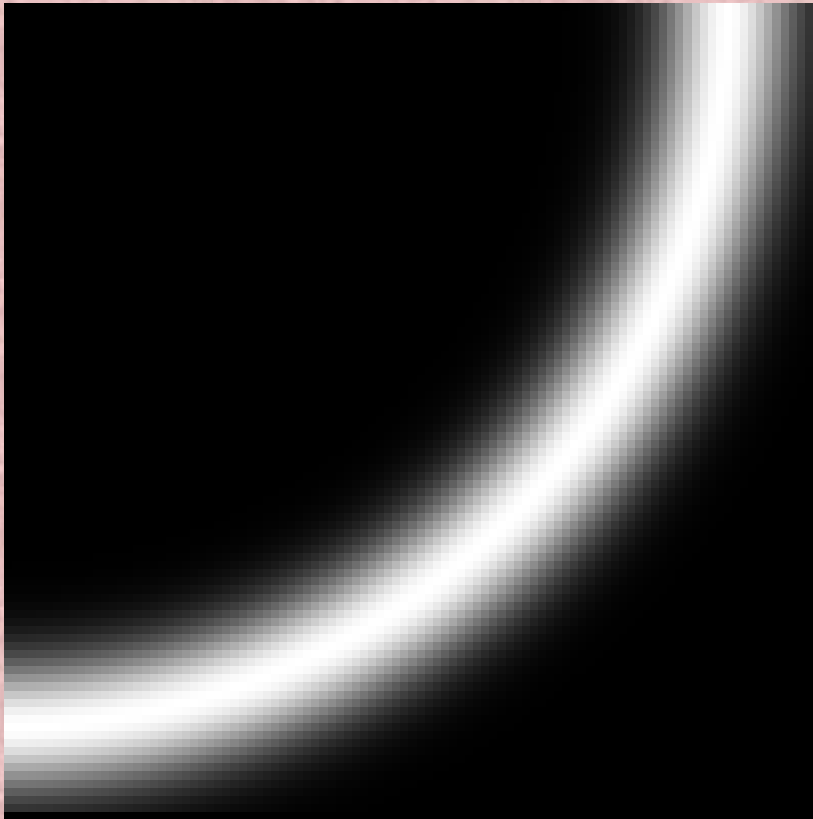
- **Localization, Detection and minimal response** (SNR-based).

The three-four stages of Canny's algorithm:

- **Apply Operator** (often implemented as Smoothing then Derivative)
- **Apply non-maximal suppression**
- **Apply hysteresis based (linking and) thresholding**



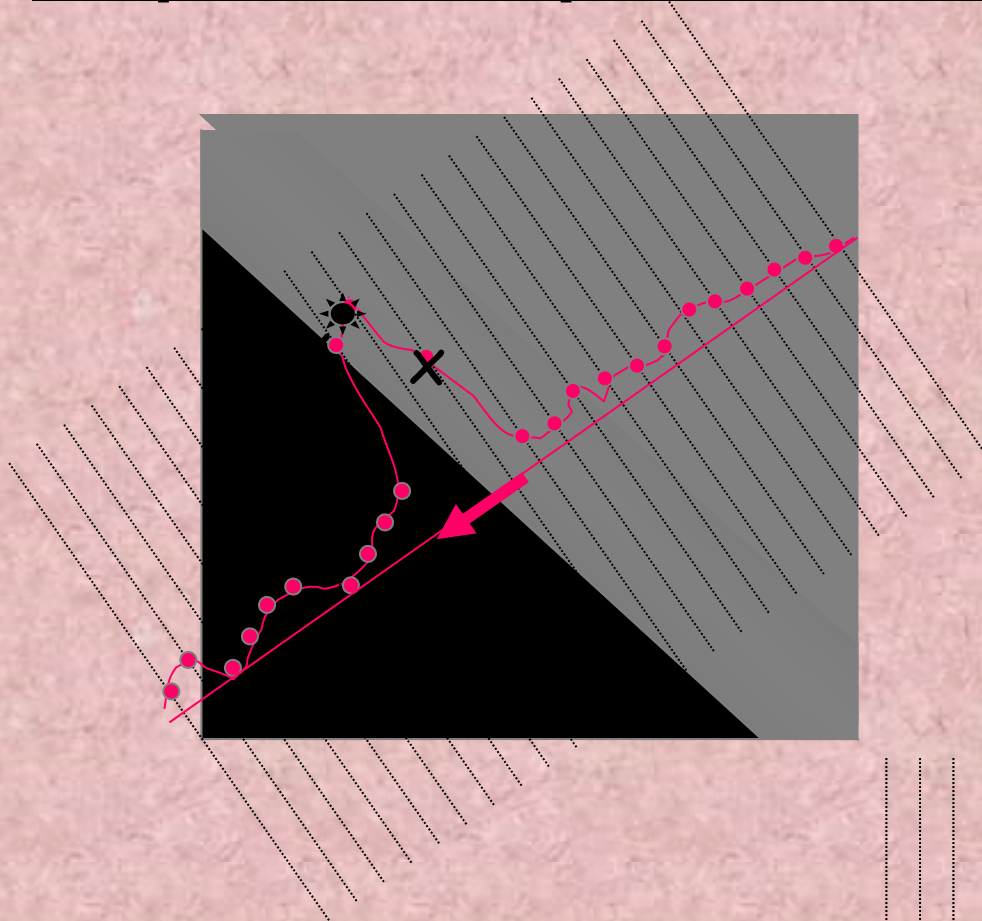
Read about - **DERICHE** recursive filtering



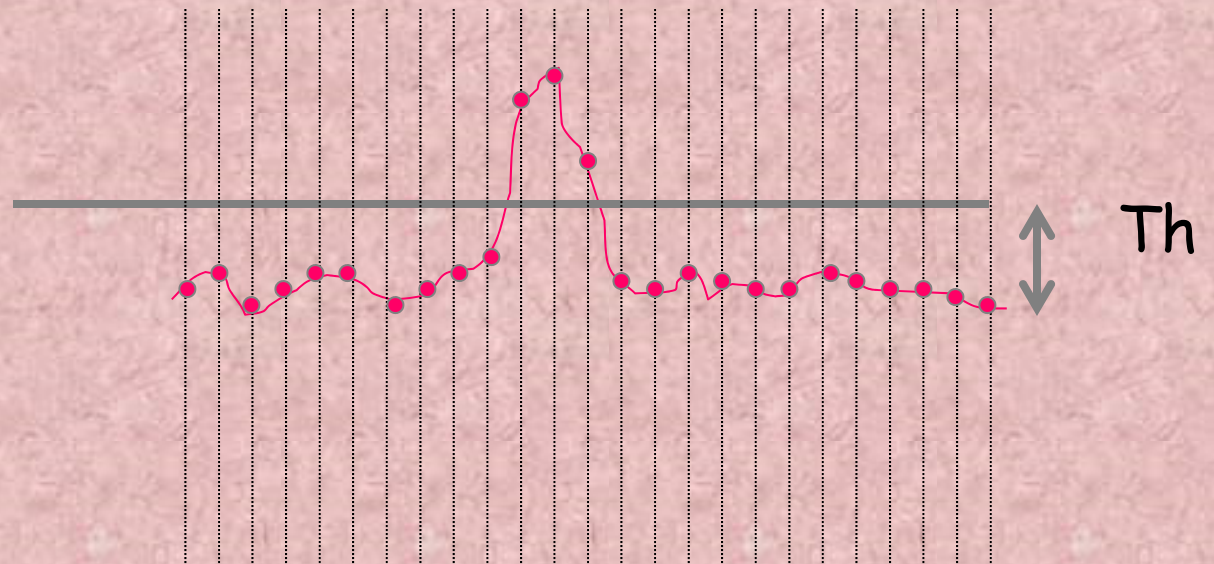
Non-maximum suppression:

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)

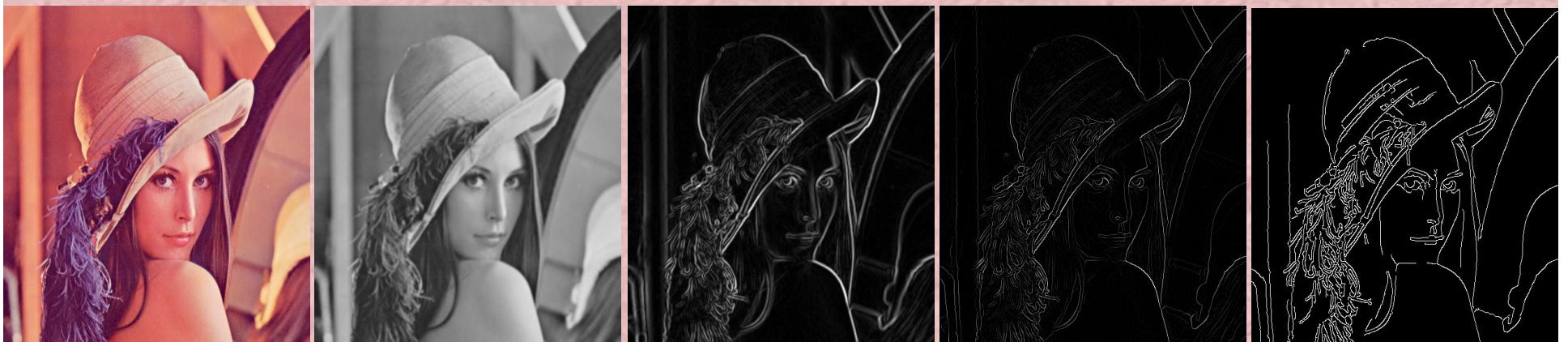


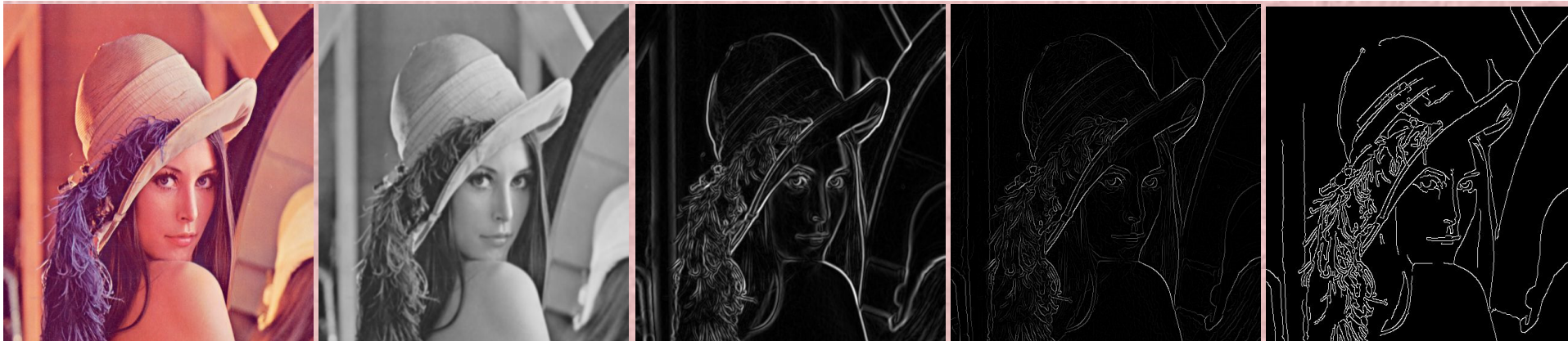
NONMAX_SUPPRESSION (Mag, Dir)

- Consider 4 directions $\text{Del}^+ = \{ (1,0),(1,1),(0,1),(-1,1) \}$
 $\text{Del}^- = \{ (-1,0),(-1,-1),(0,-1),(1,-1) \}$
- For each pixel (i,j) do:
 1. Find the direction of gradient (normal to the edge)
 $\mathbf{d} = (\text{Dir}(i,j) + \pi/8) \bmod \pi/4$
 2. If $\text{Mag}(i,j)$ is smaller than at least one of its neigh. along \mathbf{d} then $I_N(i,j)=0$, otherwise, $I_N(i,j)=\text{Mag}(i,j)$
If $\text{Mag}(i,j) < \text{Mag}((i,j) + \text{Del}^+(\mathbf{d}))$ then $I_N(i,j)=0$
Else If $\text{Mag}(i,j) < \text{Mag}((i,j) + \text{Del}^-(\mathbf{d}))$ then $I_N(i,j)=0$
Else $I_N = \text{Mag}(i,j)$
- The output is the thinned edge image I_N



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 |

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

Figure 3 Discrete approximation to Gaussian function with $\sigma=1.4$

Canny Edge Detection (Example)

**Original
image**



**Strong +
connected
weak edges**



**Strong
edges
only**






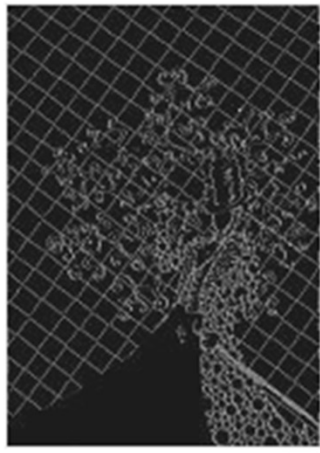

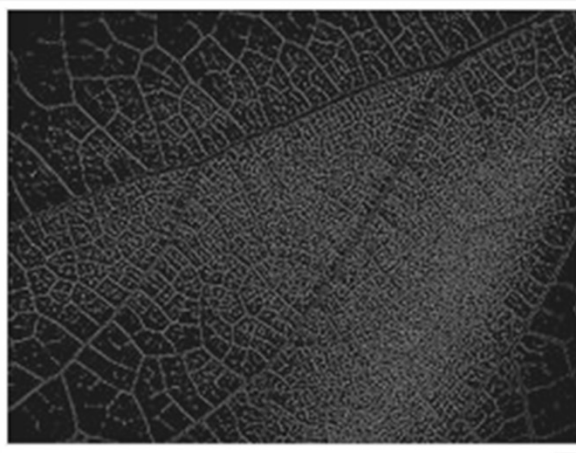


**Weak
edges**



courtesy of G. Loy

Examples of using Deriche filter on various source images

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

Effect of threshold



original



$$\sigma = 1$$

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



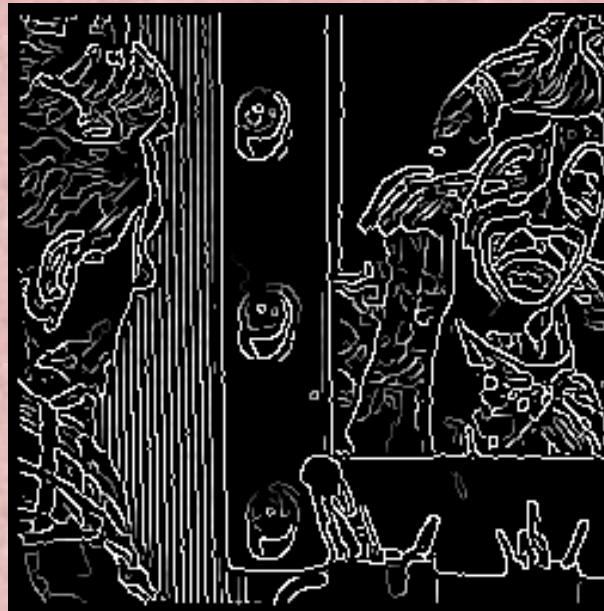
$$\sigma = 1$$

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

Effect of threshold and of σ (Gaussian kernel size)



original



$\sigma = 1$

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



$\sigma = 2$

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

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**

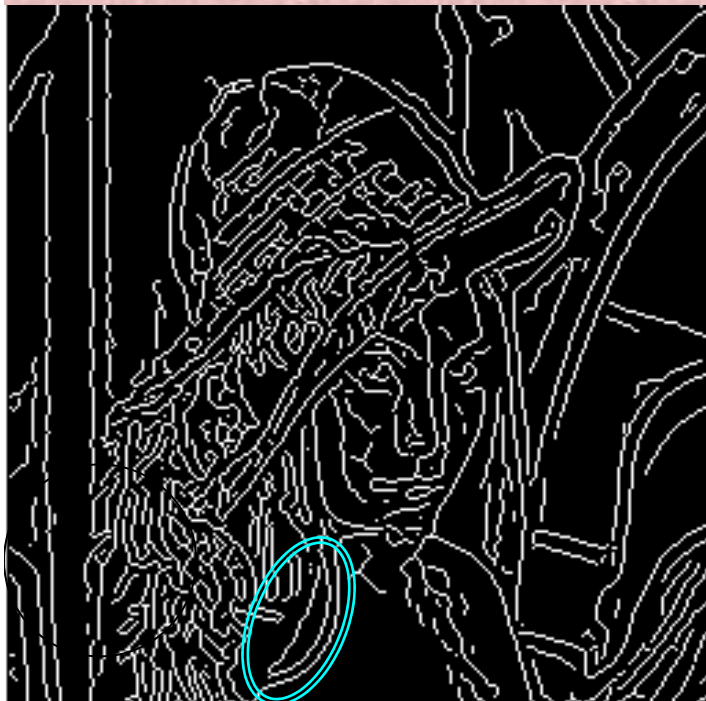


$\sigma=0.5$

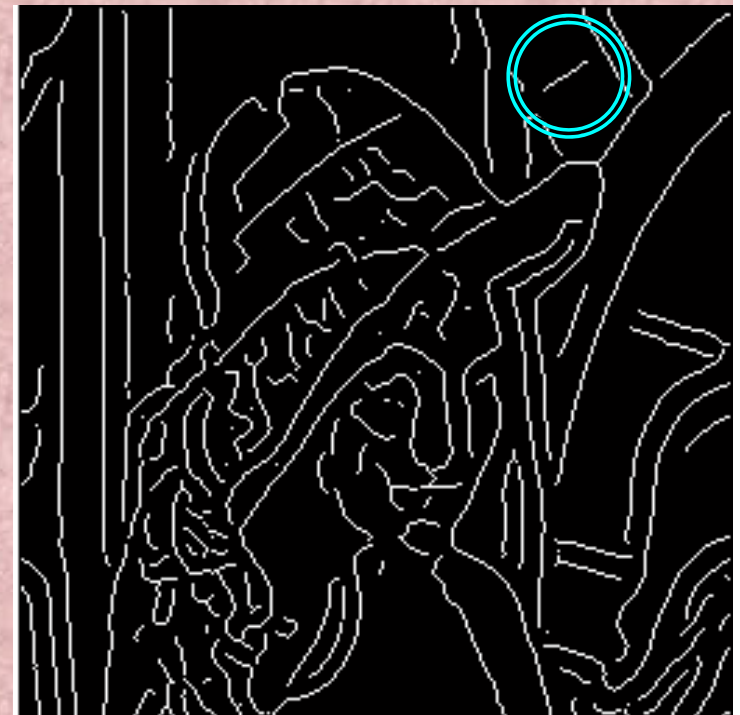


$\sigma=1.5$

**2-D Canny
edge maps**

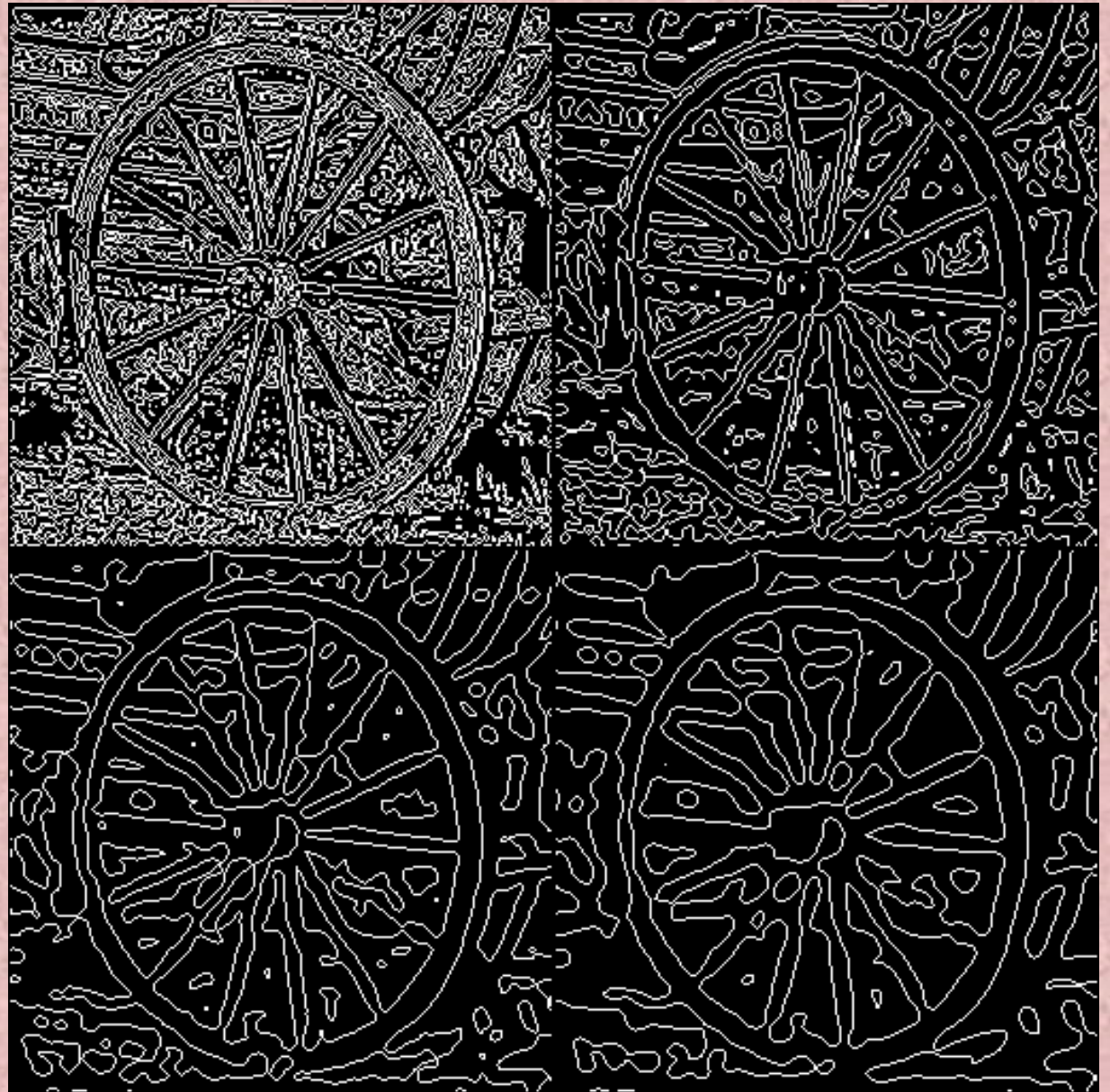


$\sigma=1$



$\sigma=2$

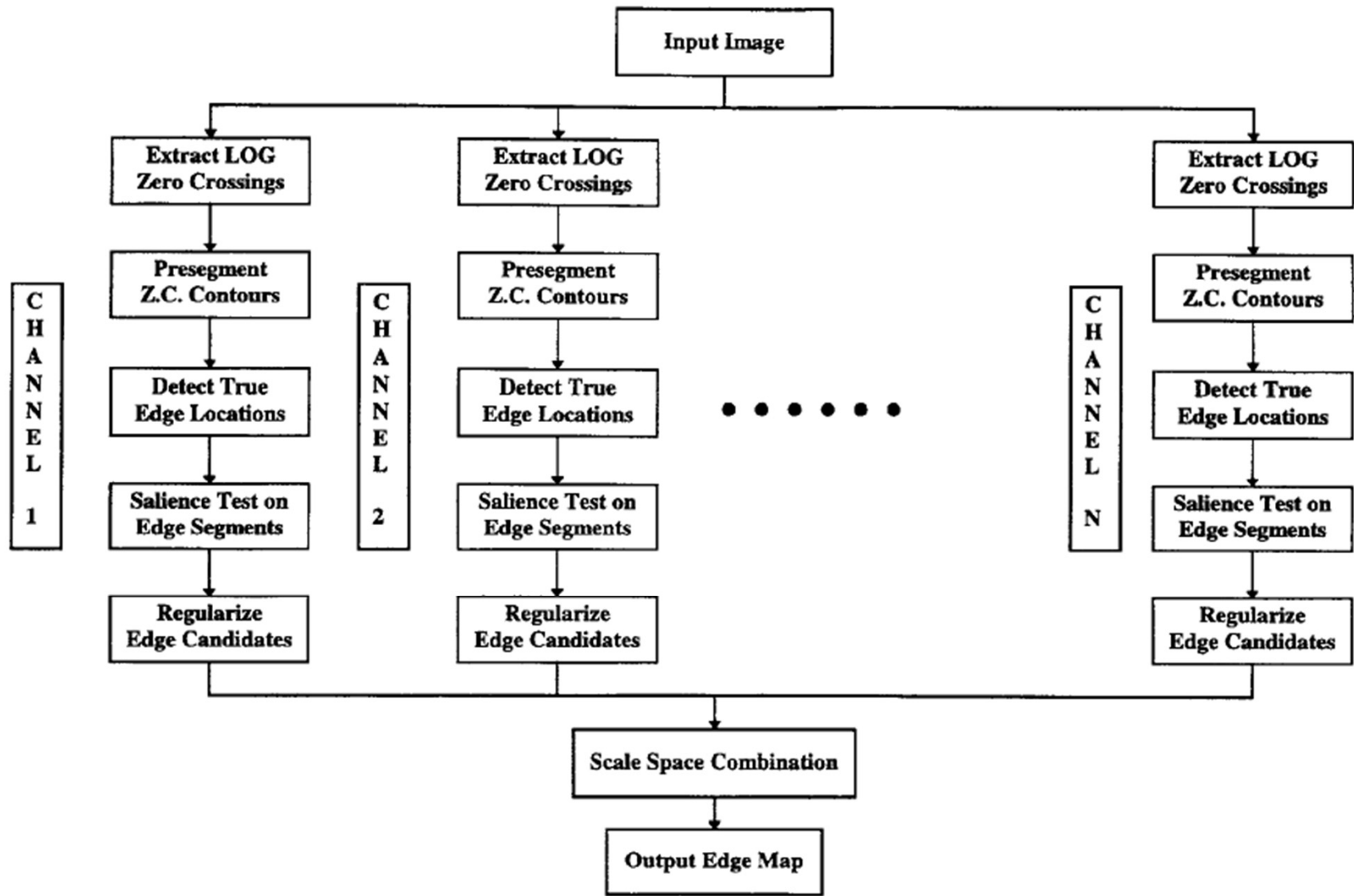
LOG with increasing SIGMA



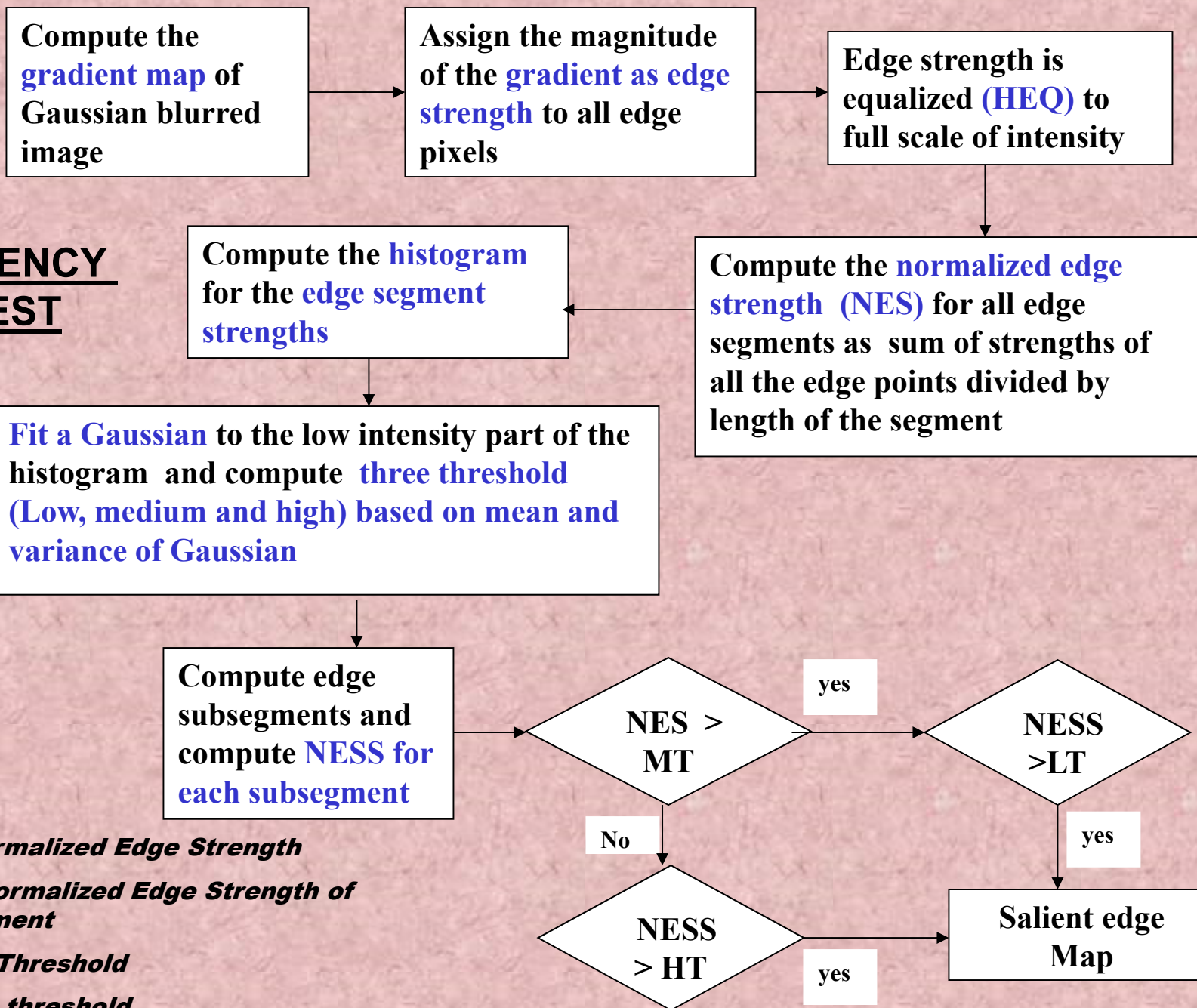
Motivation

- **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

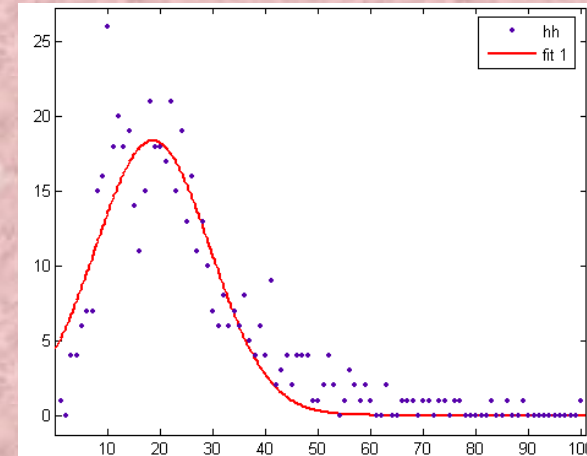
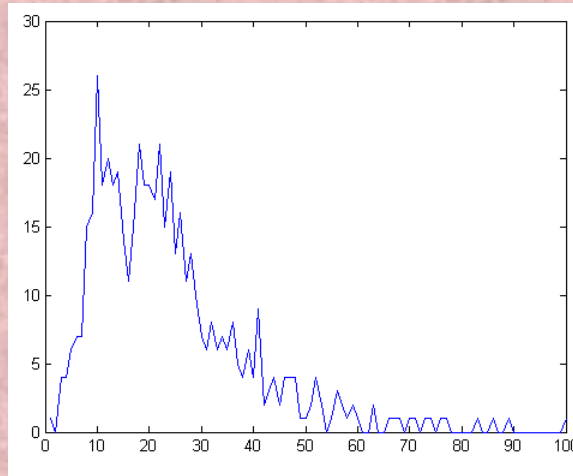
LT: Low Threshold

HT: High threshold

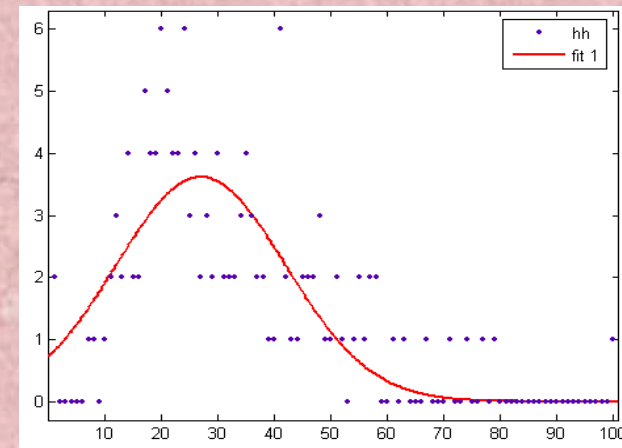
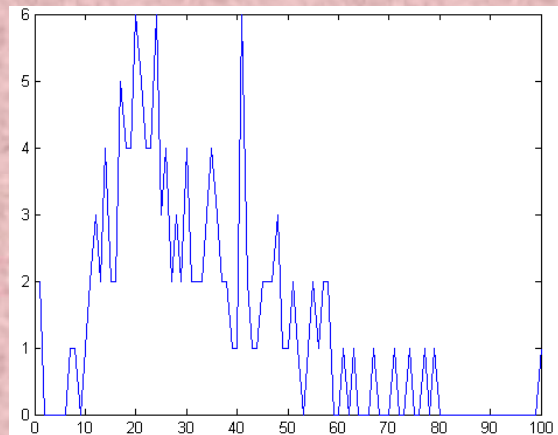
MT: Medium Threshold

Histogram of the normalised edge strengths and fitted Gaussian distribution

$\sigma=0.5$



$\sigma=1.5$

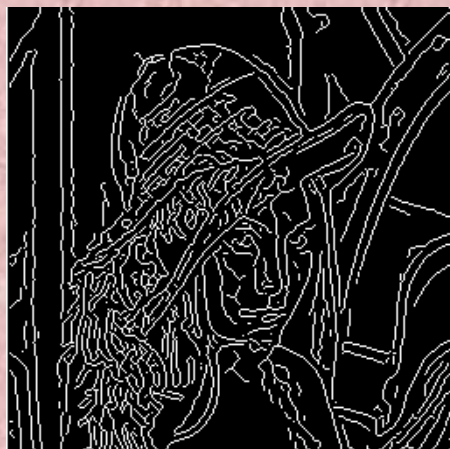
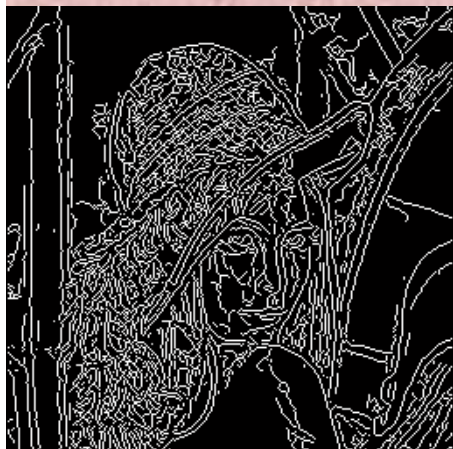


$\sigma=0.5$

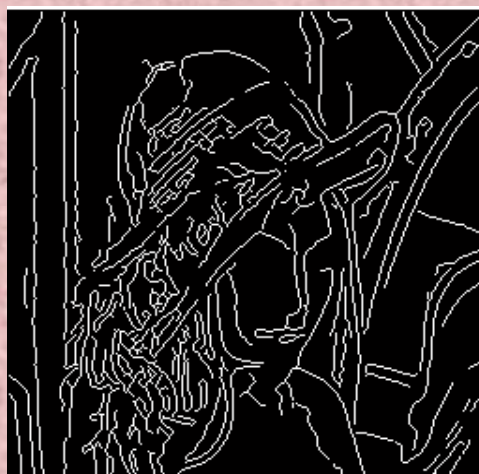
$\sigma=1$

$\sigma=1.5$

$\sigma=2$



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**

**Lena
256x256**



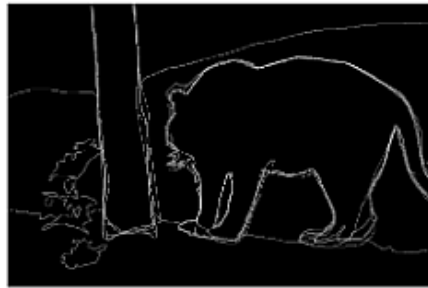
**Scale space combination
of Qian & Huang
edge model**

Read about:

- **Hysteresis based Thresholding**
- **Non-maximal suppression**
- **Edge Linking & Thinning**
- **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 Kovesei) 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**
- **Multi-channel edge detection**
- **Berkeley Edge Detection/segmentation**
- **Structured Forests**



(a) Original Image



(b) Human



(a) Original Image



(b) Human



(c) Sobel



(d) Canny



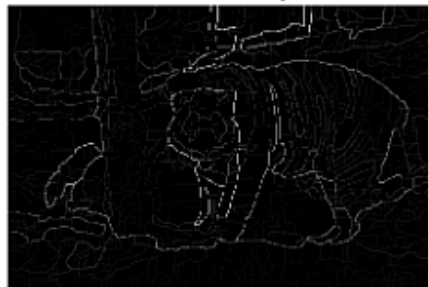
(c) Sobel



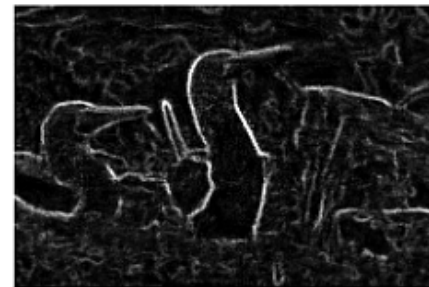
(d) Canny



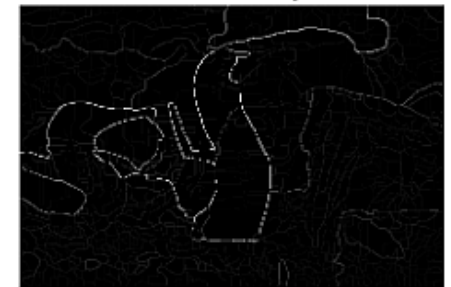
(e) BEL



(f) gPb



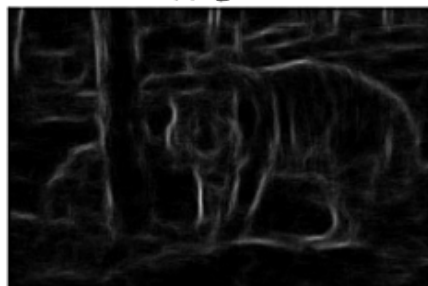
(e) BEL



(f) gPb



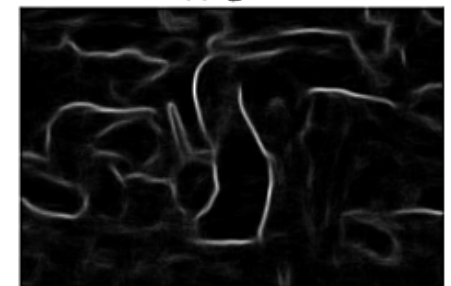
(g) Sketch Token



(h) Structured Forest



(g) Sketch Token



(h) Structured Forest



(a) Original Image



(b) Human



(c) Sobel



(d) Canny



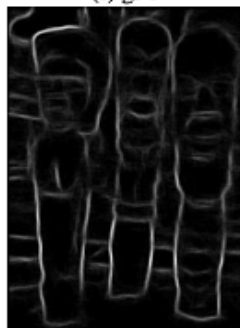
(e) BEL



(f) gPb



(g) Sketch Token



(h) Structured Forest



(a) Original Image



(b) Human



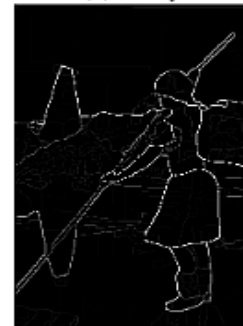
(c) Sobel



(d) Canny



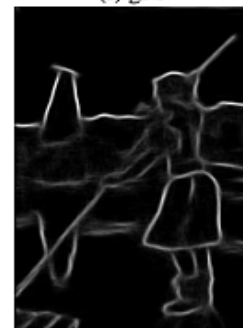
(e) BEL



(f) gPb



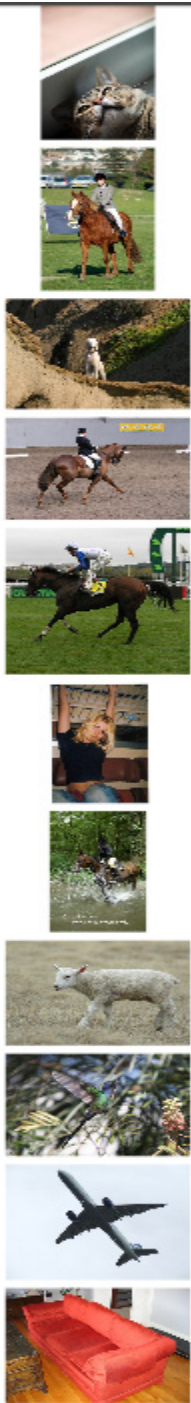
(g) Sketch Token



(h) Structured Forest



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



(a) Image



(c) SE



(e) EdgeNet

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.