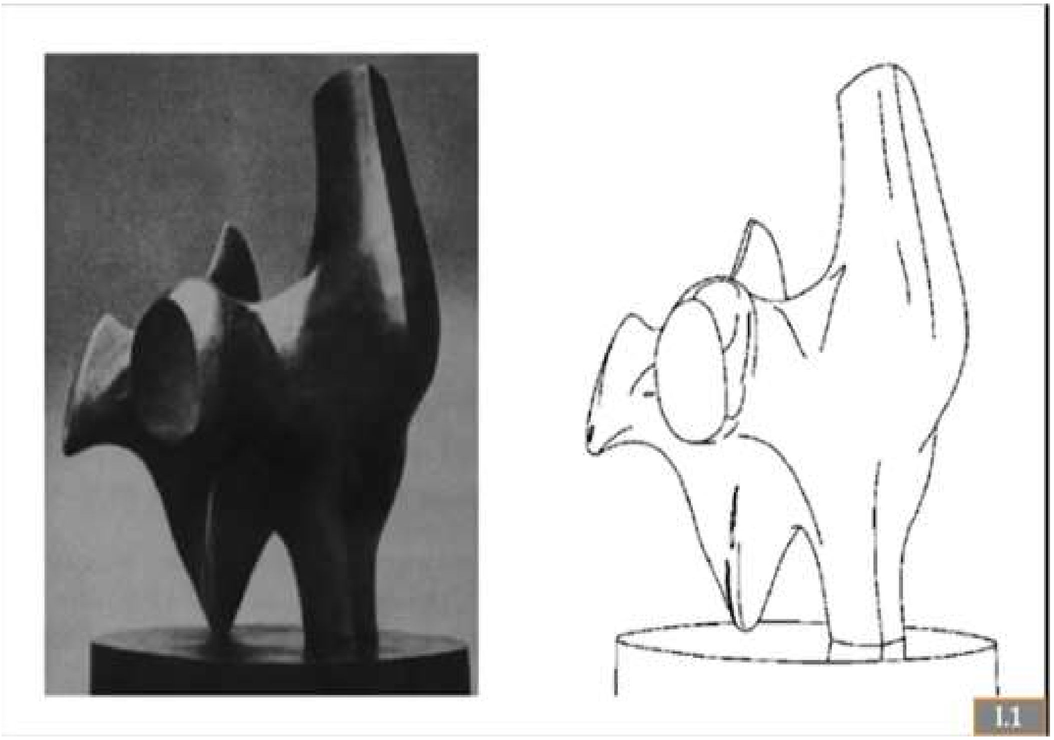
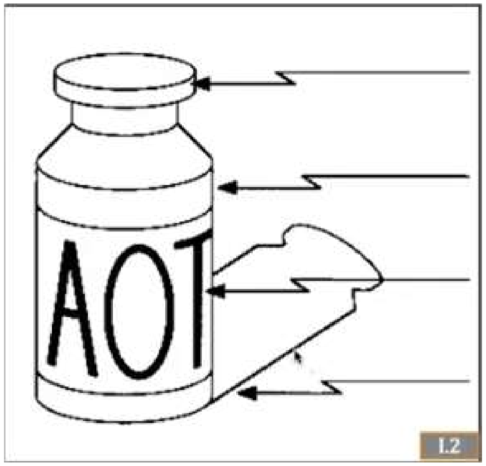
Edge Detection

- Convert the 2D image into the set of points where the image intensity changes rapidly
  - What is an edge.
  - Edge detection using gradient
  - Edge detection using Laplacian
  - Canny edge detectors
  - Corner detection

What is an Edge?

- Rapid change in image intensity within small region.
- Change causes by various physical phenomenon
  - Surface normal discontinuity
  - Depth discontinuity
  - Surface Reflectance discontinuity
  - Illumination discontinuity





- Surface normal discontinuity
- Depth discontinuity
- Surface Reflectance discontinuity
- Illumination discontinuity

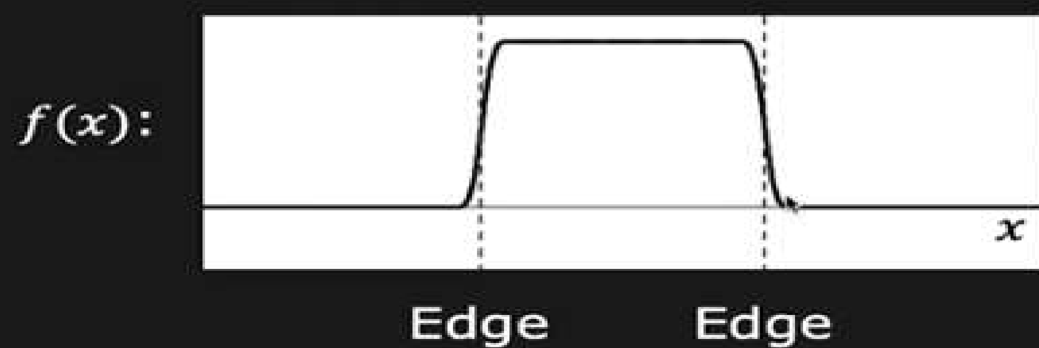
Edge Detector

- We need to design an Edge operator that produces:
  - Edge position
  - Edge Magnitude (Strength)
  - Edge Orientation (Direction)

- Performance requirement
  - High detection rate
  - Good localization
  - Low noise sensitivity

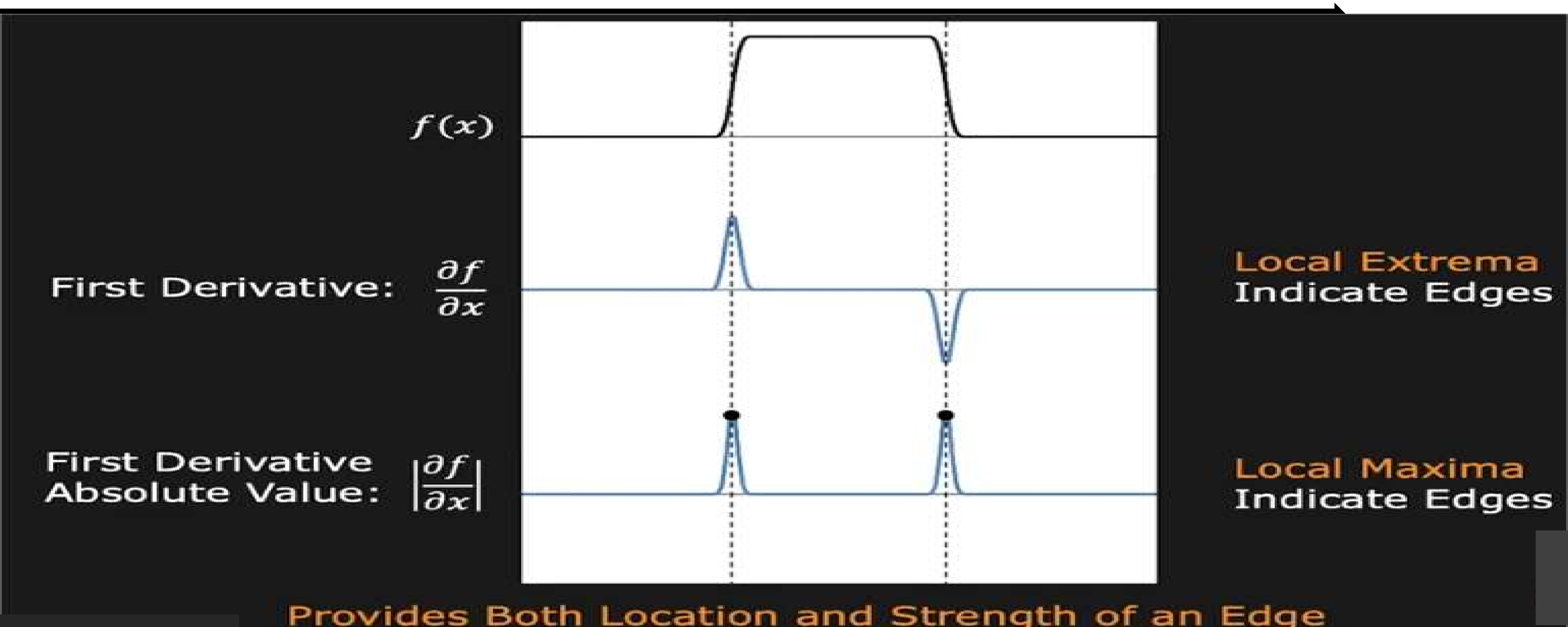
Edge Detection Using Gradients

Edge is a rapid change in image intensity in a small region.



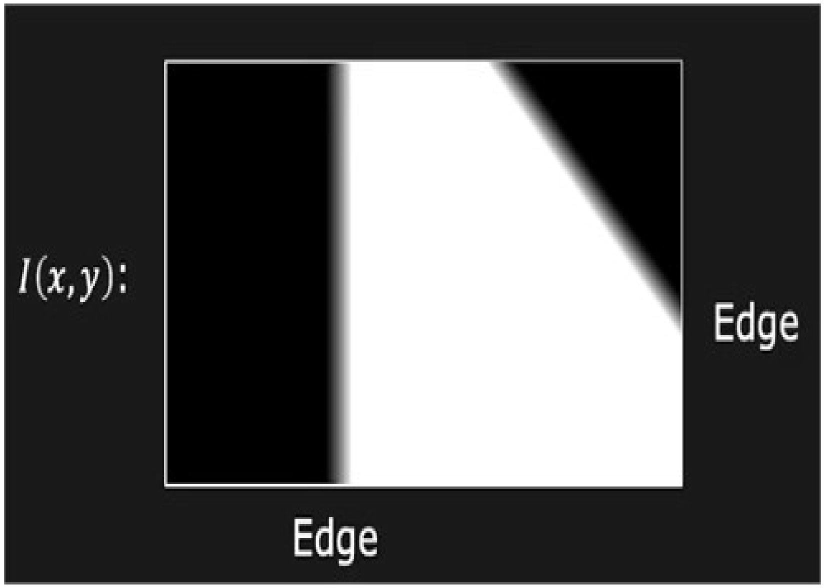
Basic Calculus: Derivative of a continuous function represents the amount of change in the function.

Edge detection using 1st derivative



2D edge detection

- Partial derivation of the 2D continuous function represents the amount of change along the each direction.
-

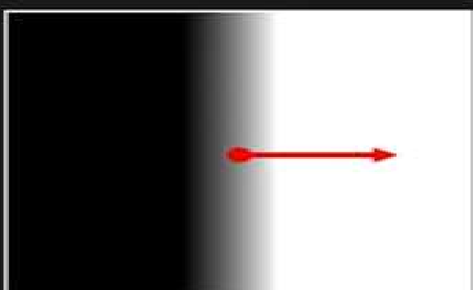


Gradient

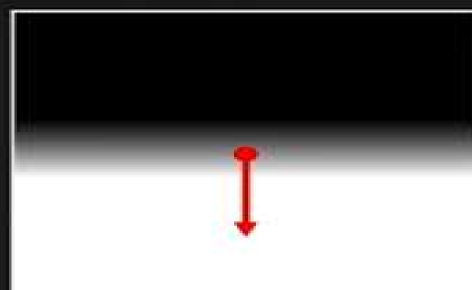
Gradient (Partial Derivatives) represents the direction of most rapid change in intensity

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

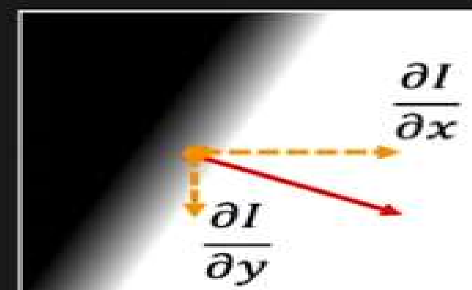
Pronounced as "Del I"



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



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



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

Gradient as edge detector

Gradient Magnitude $S = \|\nabla I\| = \sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2}$

Gradient Orientation $\theta = \tan^{-1}\left(\frac{\partial I}{\partial y} \Big/ \frac{\partial I}{\partial x}\right)$

Finite difference approximations:

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

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

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

ε

Can be implemented as Convolution!

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

Gradient	Roberts	Prewitt	Sobel (3x3)	Sobel (5x5)
$\frac{\partial I}{\partial x}$	$\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -1 & -2 & 0 & 2 & 1 \\ -2 & -3 & 0 & 3 & 2 \\ -3 & -5 & 0 & 5 & 3 \\ -2 & -3 & 0 & 3 & 2 \\ -1 & -2 & 0 & 2 & 1 \end{bmatrix}$
$\frac{\partial I}{\partial y}$	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 & 3 & 2 & 1 \\ 2 & 3 & 5 & 3 & 2 \\ 0 & 0 & 0 & 0 & 0 \\ -2 & -3 & -5 & -3 & -2 \\ -1 & -2 & -3 & -2 & -1 \end{bmatrix}$

- Good localization
- Noise sensitive
- Poor detection

- Poor localization
- Less Noise sensitive
- good detection

Gradient using the sobel operator

Edge Thresholding

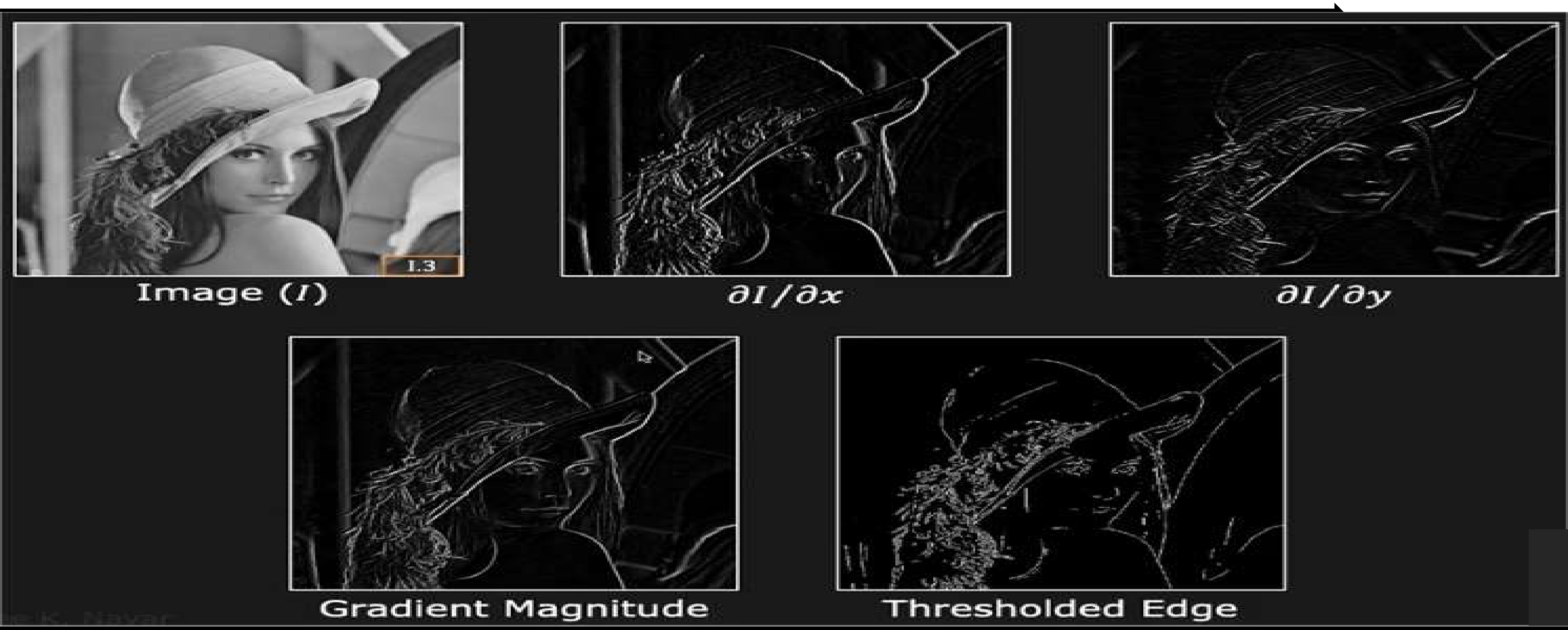
Standard: (Single Threshold T)

$\|\|\nabla I(x,y)\|\| < T$	Definitely Not an Edge
$\|\|\nabla I(x,y)\|\| \geq T$	Definitely an Edge

Hysteresis Based: (Two Thresholds $T_0 < T_1$)

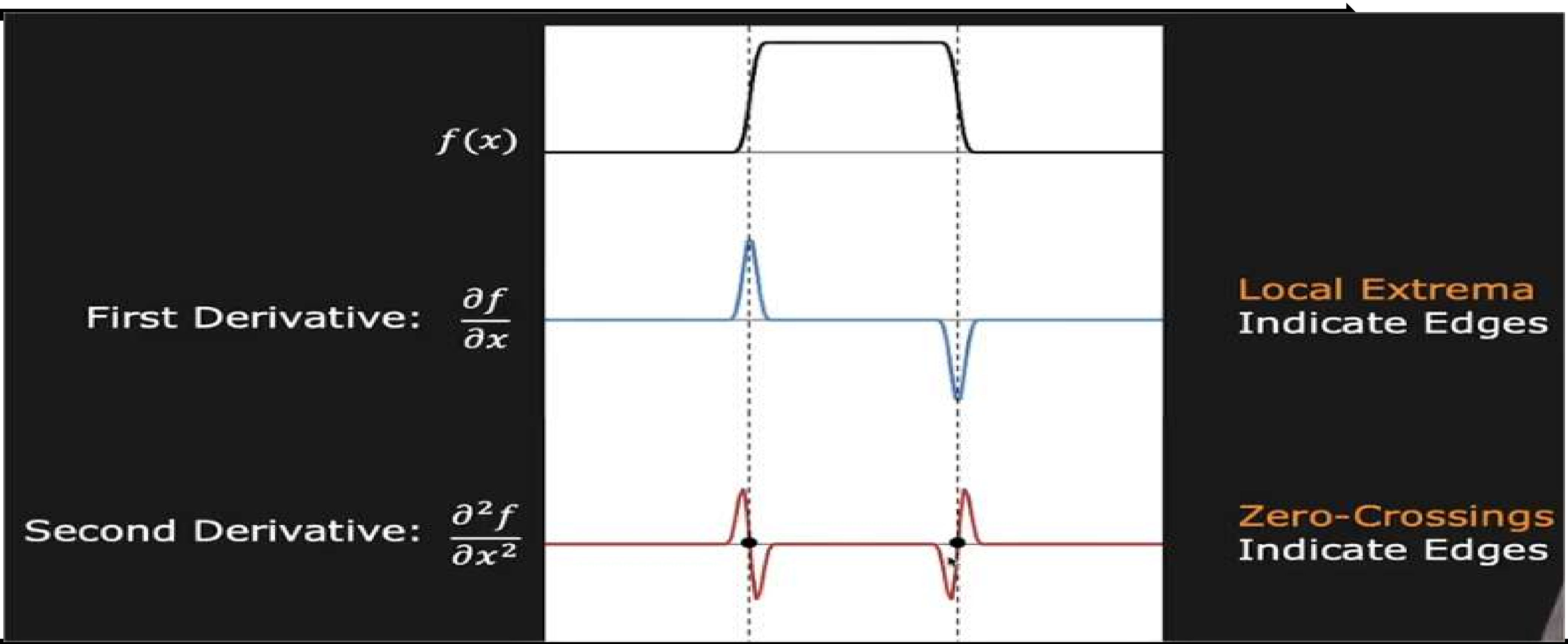
$\|\|\nabla I(x,y)\|\| < T_0$	Definitely Not an Edge
$\|\|\nabla I(x,y)\|\| \geq T_1$	Definitely an Edge
$T_0 \leq \|\|\nabla I(x,y)\|\| < T_1$	Is an Edge if a Neighboring Pixel is Definitely an Edge

Sobel



I.3
Image (I)
$\partial I/\partial x$
$\partial I/\partial y$
Gradient Magnitude
Thresholded Edge

Edges - LOG



$f(x)$
First Derivative: $\frac{\partial f}{\partial x}$
Second Derivative: $\frac{\partial^2 f}{\partial x^2}$
Local Extrema
Indicate Edges
Zero-Crossings
Indicate Edges

Laplacian: Sum of Pure Second Derivatives

$$\boxed{\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}}$$

Pronounced as "Del Square I"

- Edges are "zero-crossings" in Laplacian of image
- Laplacian does not provide directions of edges

For the discrete image

Finite difference approximations:

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

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

$$\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

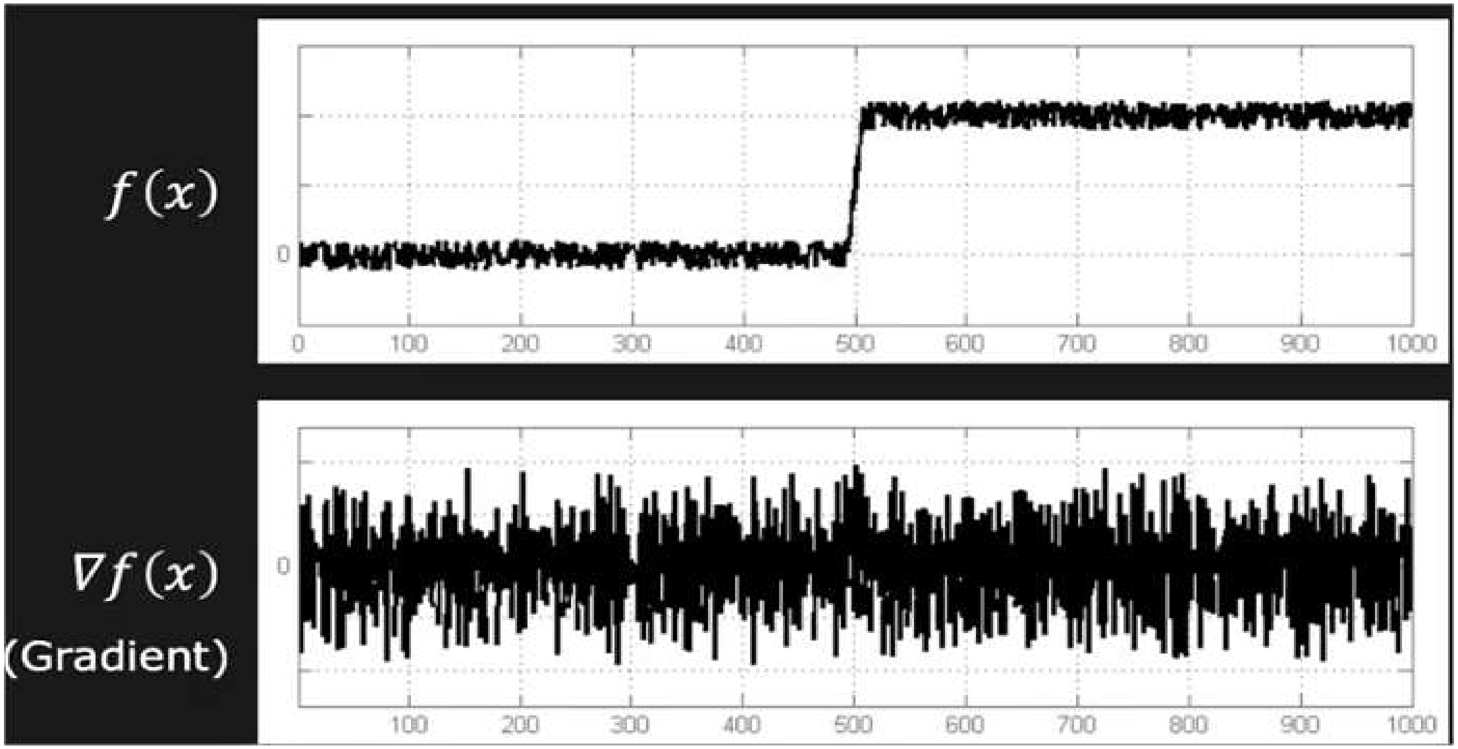
$I_{i-1,j+1}$	$I_{i,j+1}$	$I_{i+1,j+1}$	$\updownarrow \varepsilon$
$I_{i-1,j}$	$I_{i,j}$	$I_{i+1,j}$	
$I_{i-1,j-1}$	$I_{i,j-1}$	$I_{i+1,j-1}$	

Convolution Mask:

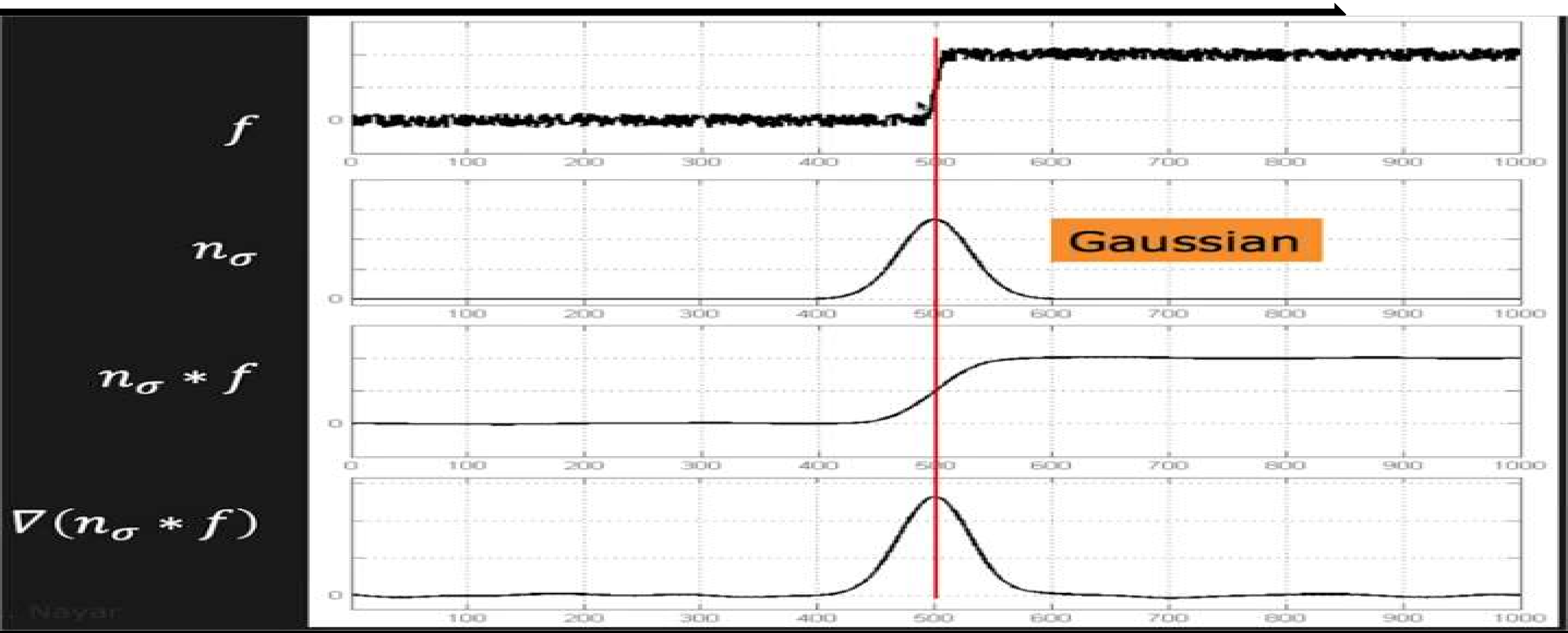
$$\nabla^2 \approx \frac{1}{\varepsilon^2}\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \text{OR} \quad \nabla^2 \approx \frac{1}{6\varepsilon^2}\begin{bmatrix} 1 & 4 & 1 \\ 4 & -20 & 4 \\ 1 & 4 & 1 \end{bmatrix}$$

(More Accurate)

Effects of noise

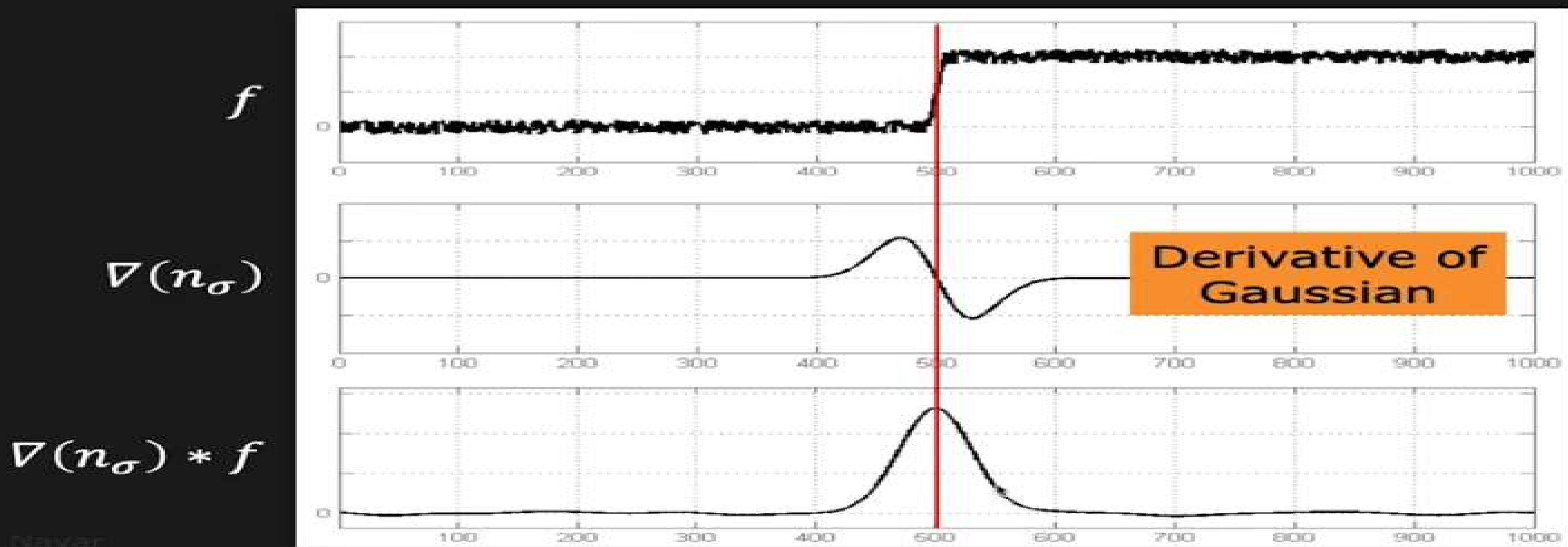


Solution : Gaussian smoothing

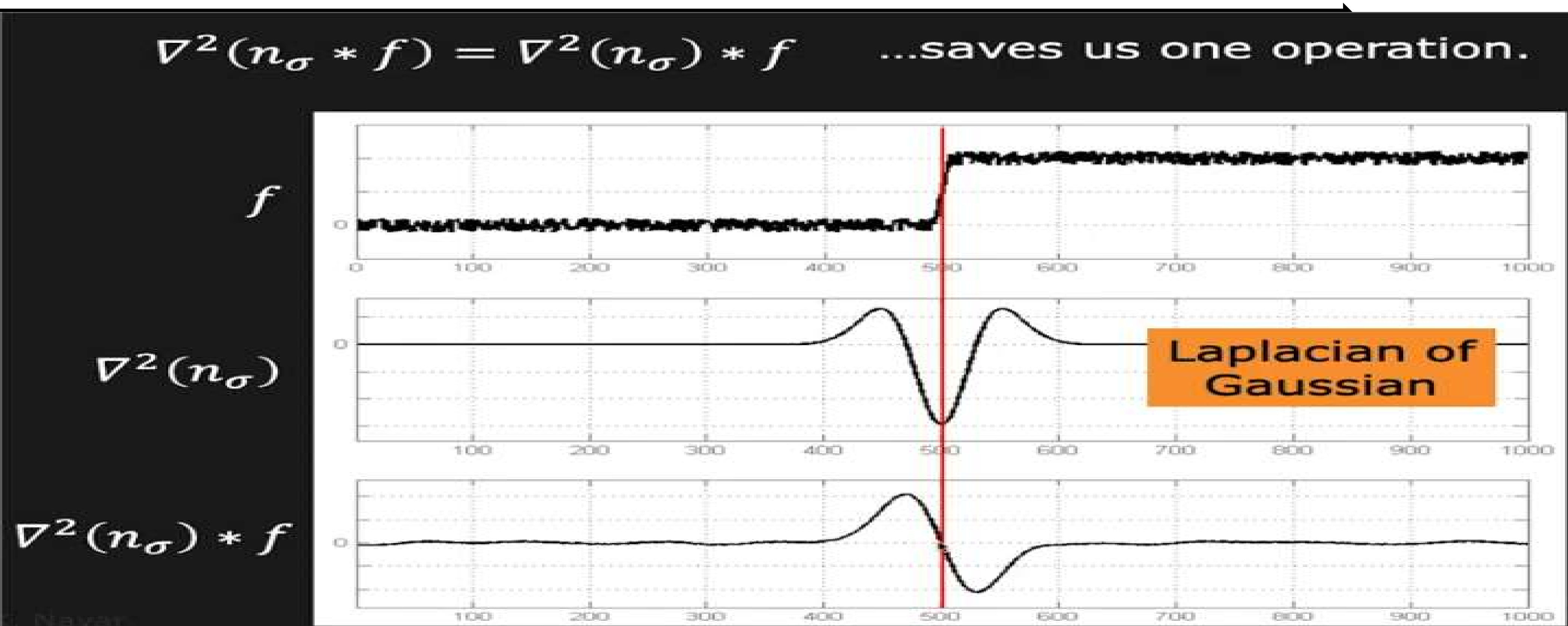


Derivative of Gaussian

$$\nabla(n_\sigma * f) = \nabla(n_\sigma) * f$$ ...saves us one operation.



Laplacian of Gaussian



Gradient vs. Laplacian

Gradient	Laplacian
Provides location, magnitude and direction of the edge.	Provides only location of the edge.
Detection using Maxima Thresholding.	Detection based on Zero-Crossing.
Non-linear operation. Requires two convolutions.	Linear Operation. Requires only one convolution.

Edges - Canny

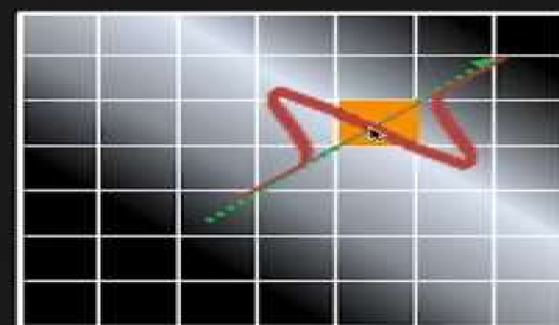
- Smooth Image with 2D Gaussian: $n_\sigma * I$
- Compute Image Gradient using Sobel Operator: $\nabla n_\sigma * I$
- Find Gradient Magnitude at each pixel: $\|\nabla n_\sigma * I\|$
- Find Gradient Orientation at each Pixel:

$$\hat{n} = \frac{\nabla n_\sigma * I}{\|\nabla n_\sigma * I\|}$$

- Compute Laplacian along the Gradient Direction $\hat{n}$ at each pixel

$$\frac{\partial^2 (n_\sigma * I)}{\partial \hat{n}^2}$$

- Find Zero Crossings in Laplacian to find the edge location



$\|\nabla n_\sigma * I\|$

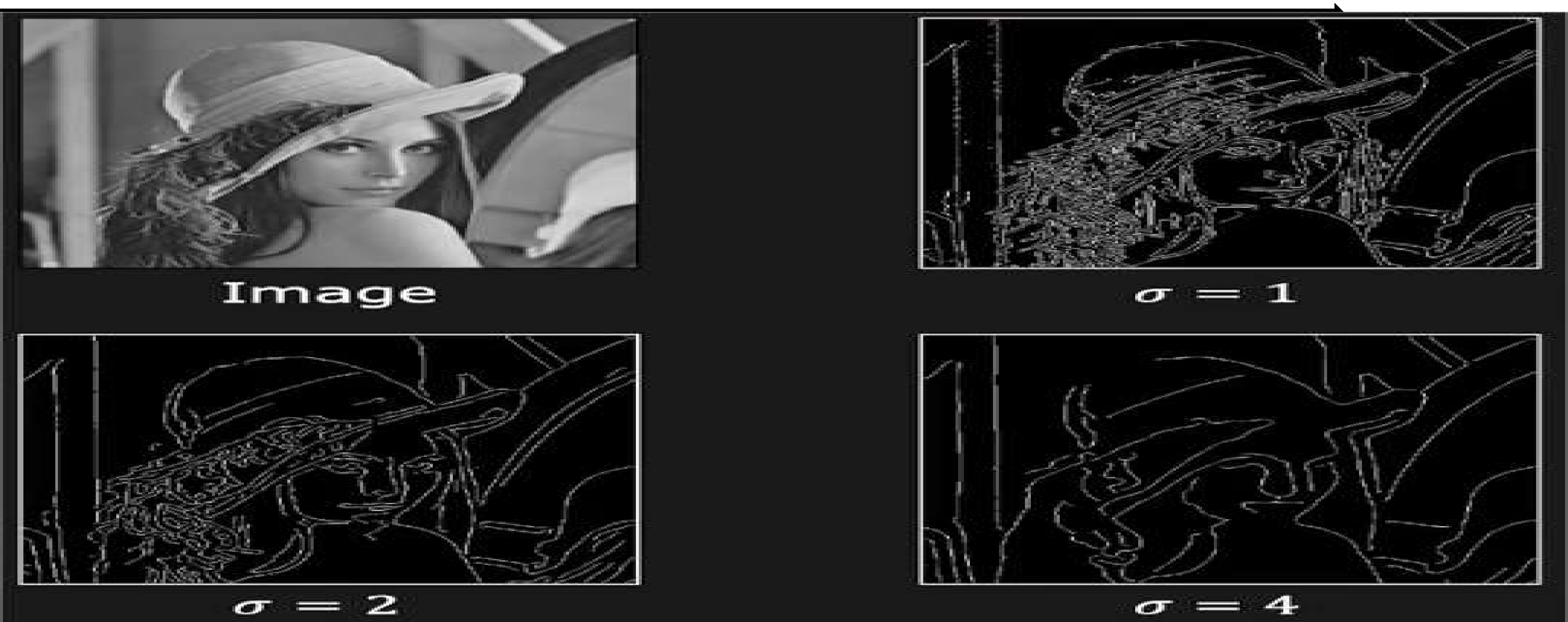


Image
$\sigma = 1$
$\sigma = 2$
$\sigma = 4$

Corner detection

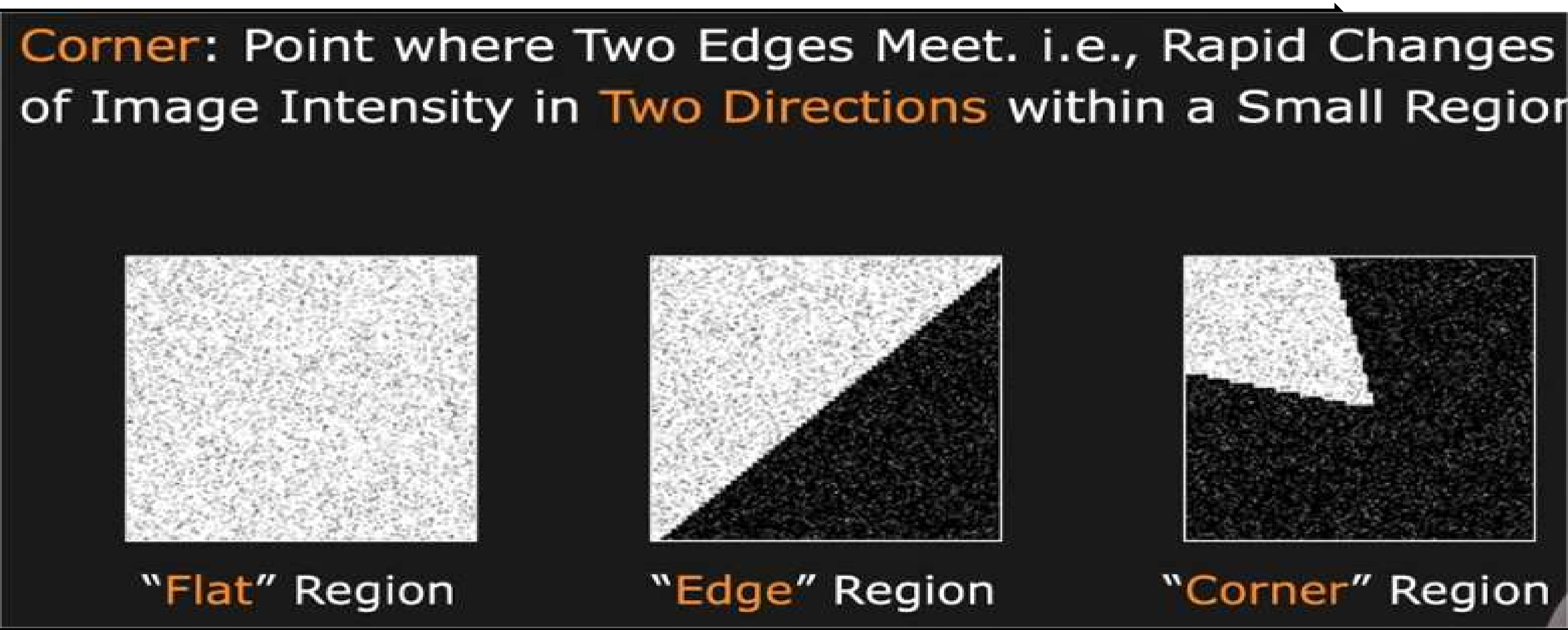
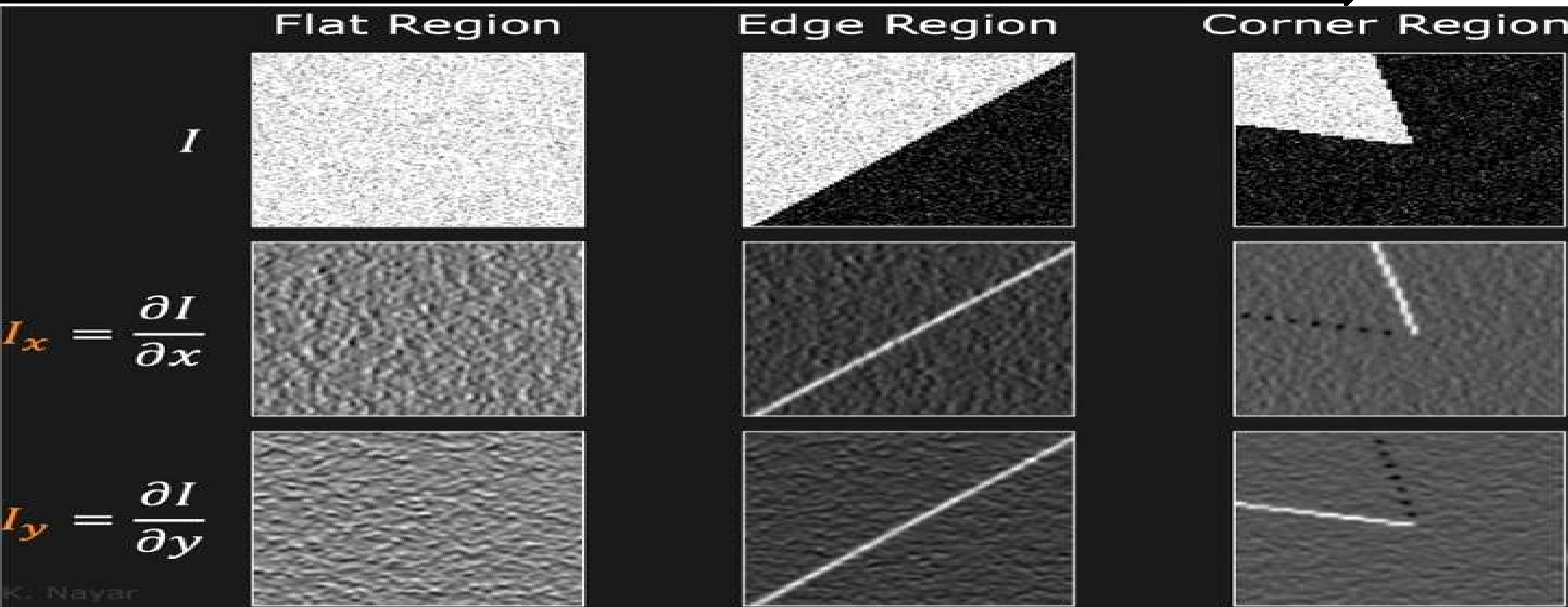
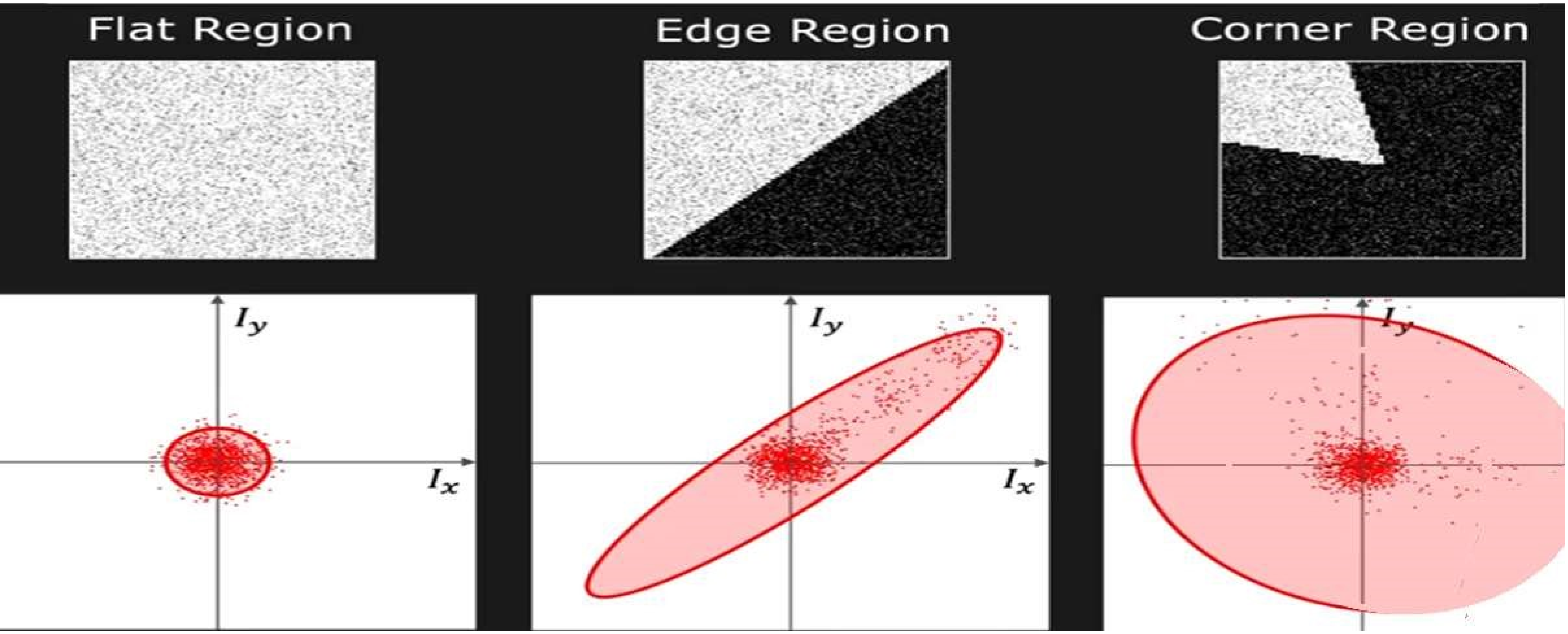


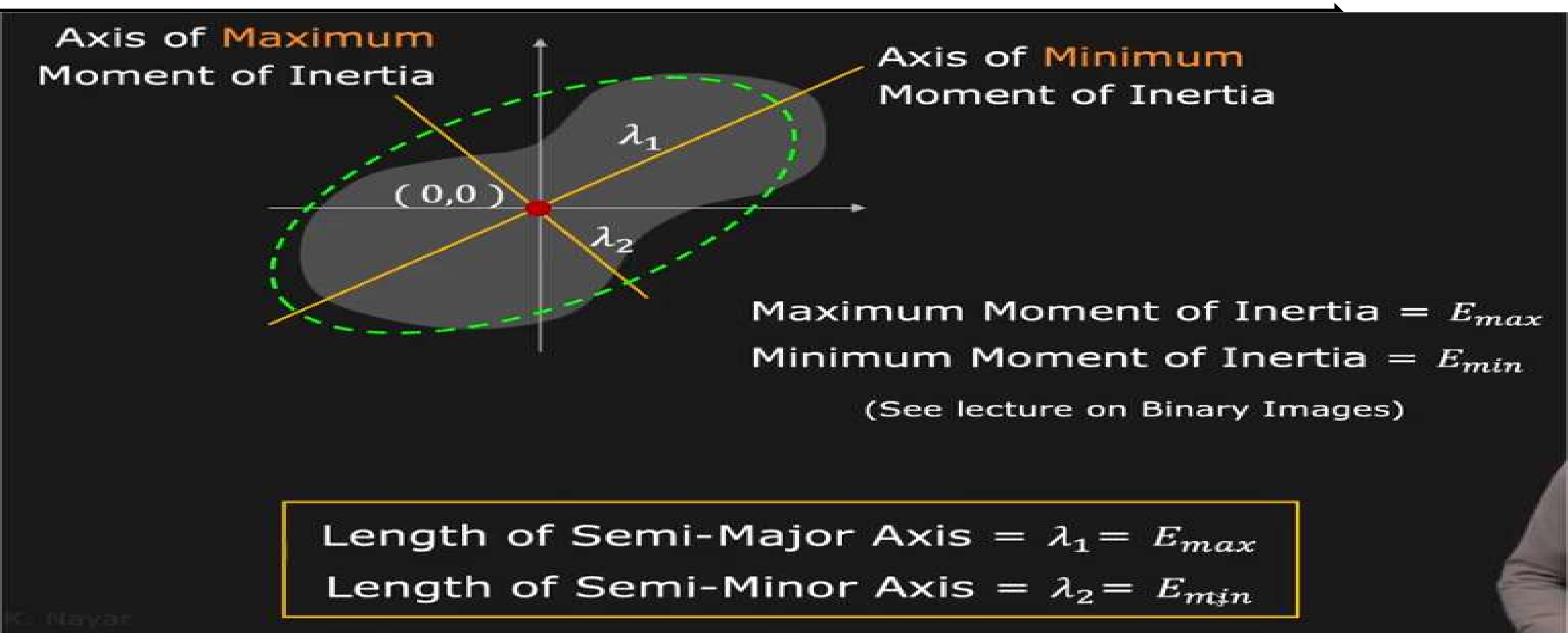
Image Gradients





Flat Region
Edge Region
Corner Region
I_y
I_x
I_y
I_x
I_y

Fitting an elliptical disk



Fitting an elliptical disk

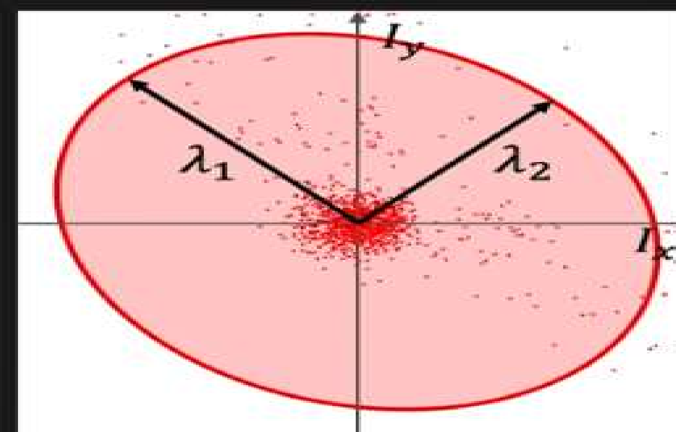
Second Moments for a Region:

$$a = \sum_{i \in W} \left({I_x}_i^{\,2} \right) \qquad b = 2 \sum_{i \in W} \left({I_x}_i {I_y}_i \right)$$

$$c = \sum_{i \in W} \left({I_y}_i^{\,2} \right)$$

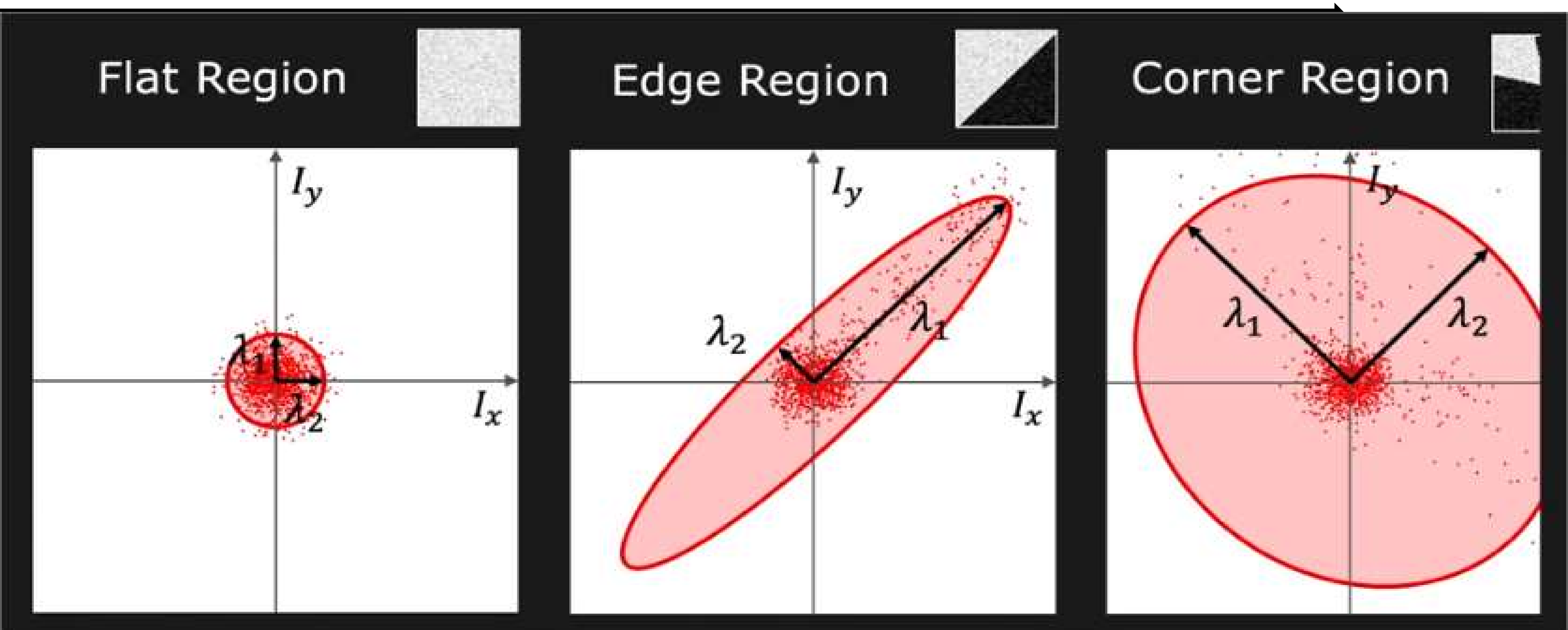
W: Window centered at pixel

(See lecture on Binary Images)

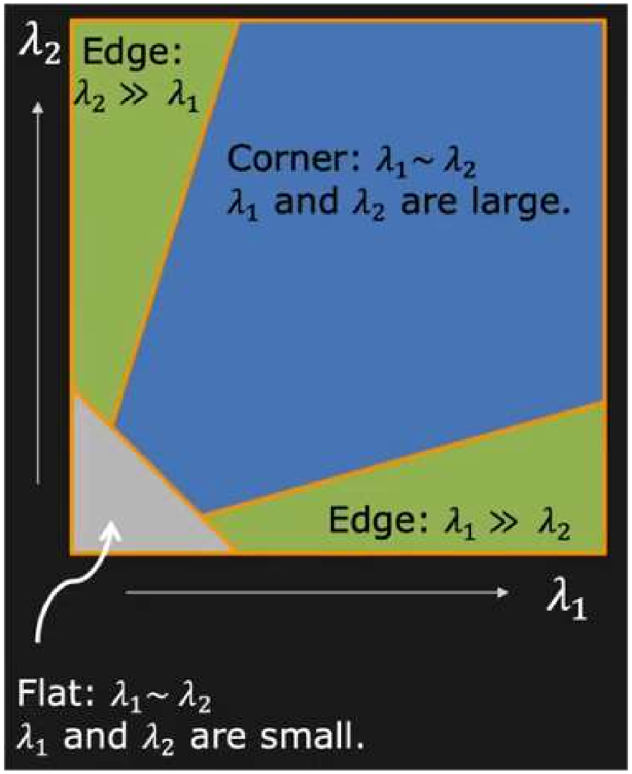


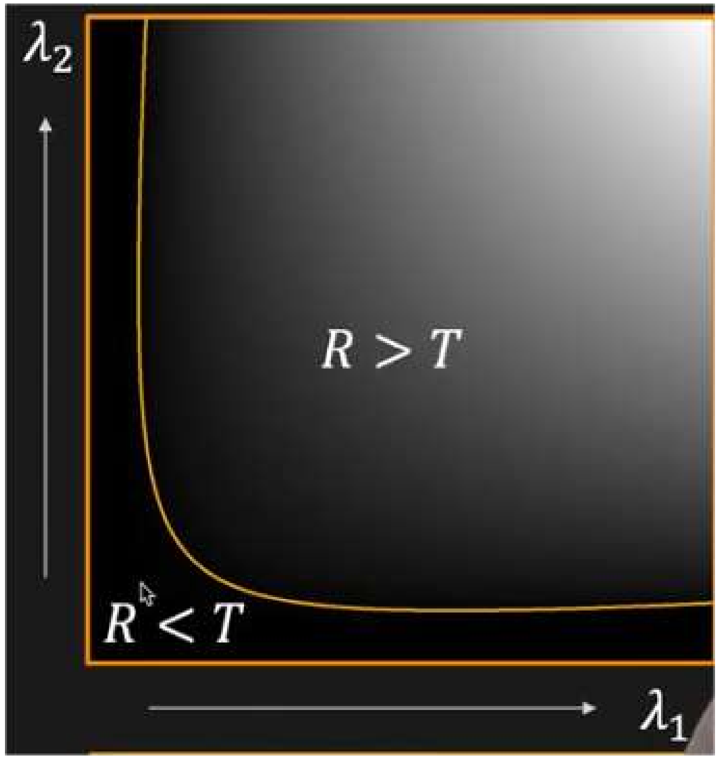
Ellipse Axes Lengths:

$$\boxed{\begin{aligned} \lambda_1 &= E_{max} = \frac{1}{2}\left[a + c + \sqrt{b^2 + (a-c)^2}\right] \\ \lambda_2 &= E_{min} = \frac{1}{2}\left[a + c - \sqrt{b^2 + (a-c)^2}\right] \end{aligned}}$$


Flat Region
Edge Region
Corner Region
I_y
I_x
λ_1
λ_2

Corners - Harris and Hessian Affine

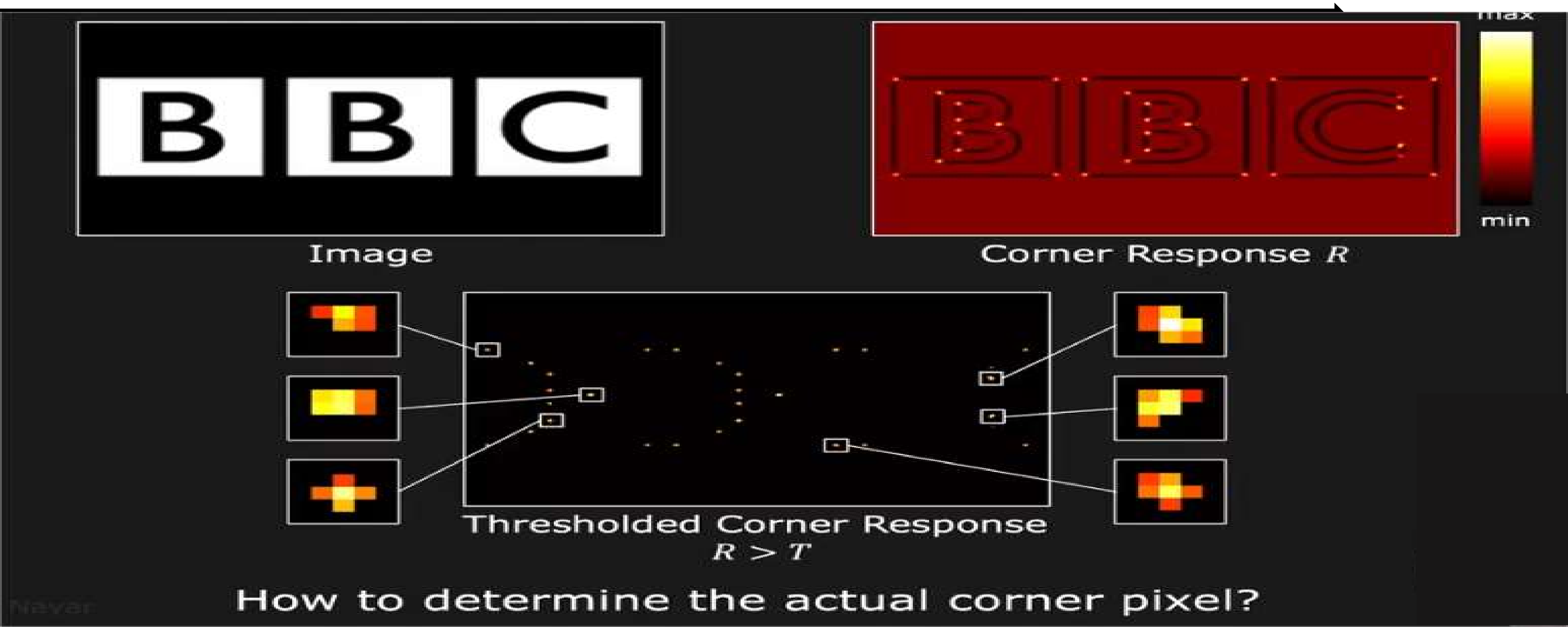




$$R = \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2$$

$0.04 < k < 0.06$

Corners - Harris and Hessian Affine

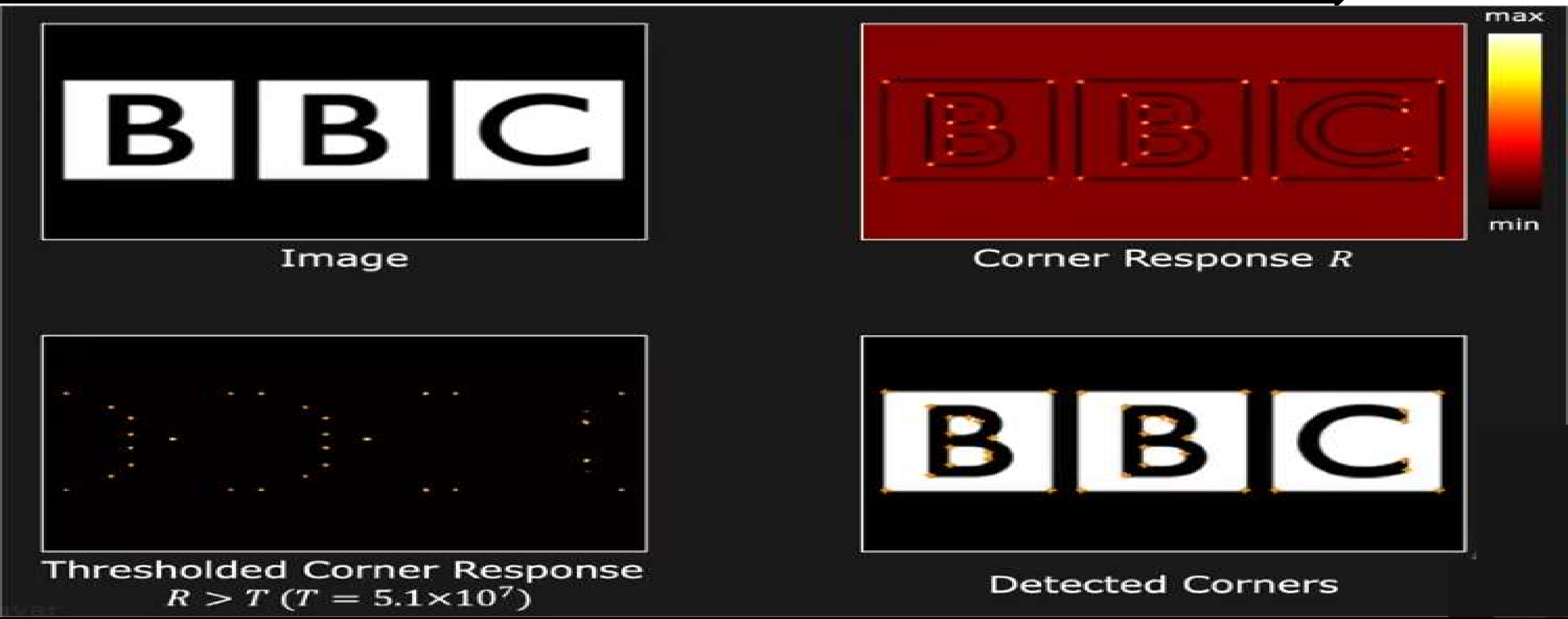


Non- Maximal suppression

1. Slide a window of size k over the image.
2. At each position, if the pixel at the center is the maximum value within the window, label it as positive (retain it). Else label it as negative (suppress it).

Suppress Suppress Retain

Used for finding Local Extrema (Maxima/Minima)

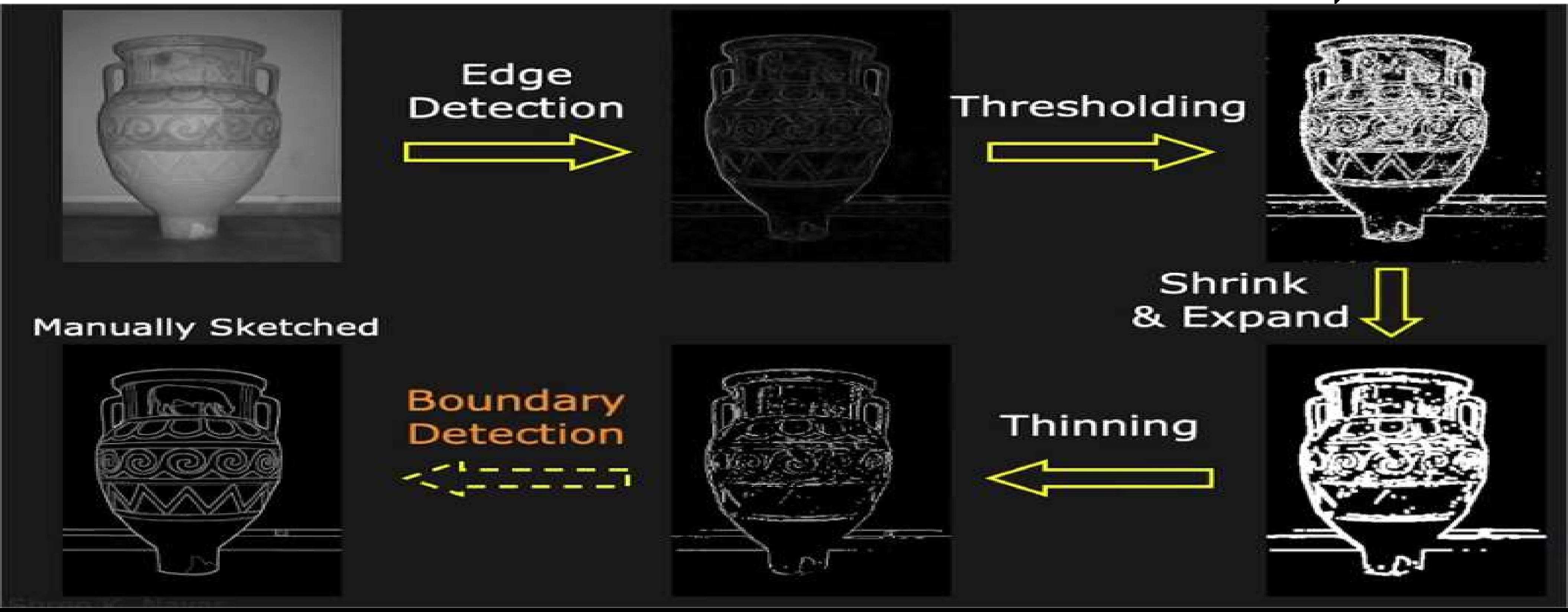


max
min
Image
Corner Response R
Thresholded Corner Response
$R > T$ ($T = 5.1\times10^7$)
Detected Corners

Boundary Detection

- We need to find object boundary from the edge pixels
  - Fitting lines and curves to edges
  - Active contours (Snakes)
  - The Hough Transform
  - The generalized Hough Transform

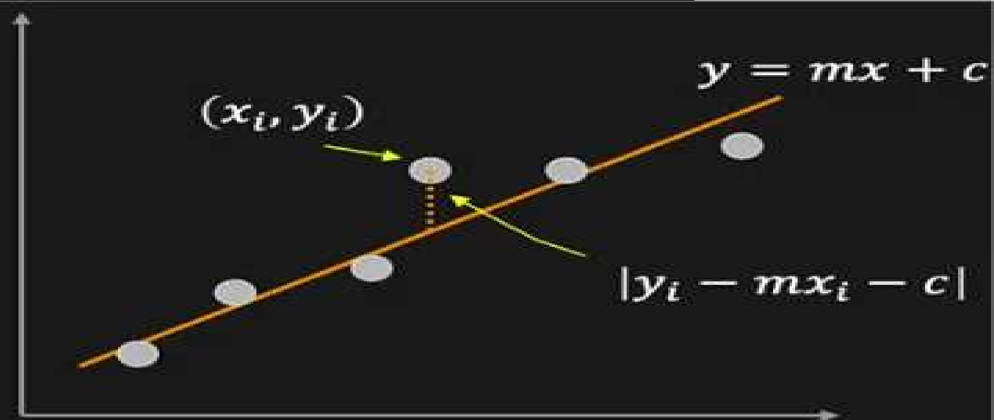
Fitting line and Curves: Preprocessing Edge Images



Line Fitting

Given: Edge Points (x_i, y_i)

Task: Find (m, c)



Minimize: Average Squared Vertical Distance

$$E = \frac{1}{N}\sum_i (y_i - mx_i - c)^2$$

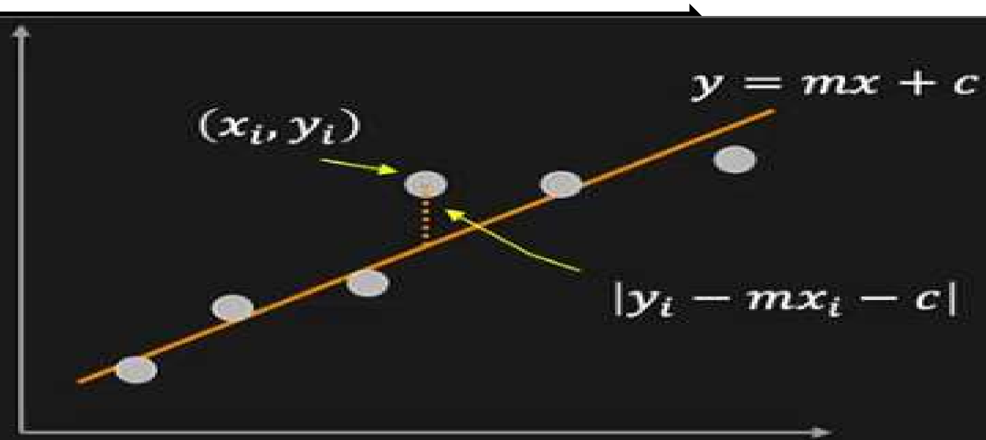
Least Squares Solution:

$$\frac{\partial E}{\partial m} = \frac{-2}{N}\sum_i x_i(y_i - mx_i - c) = 0 \qquad \frac{\partial E}{\partial c} = \frac{-2}{N}\sum_i (y_i - mx_i - c) = 0$$

Close form solution

Given: Edge Points (x_i, y_i)

Task: Find (m, c)



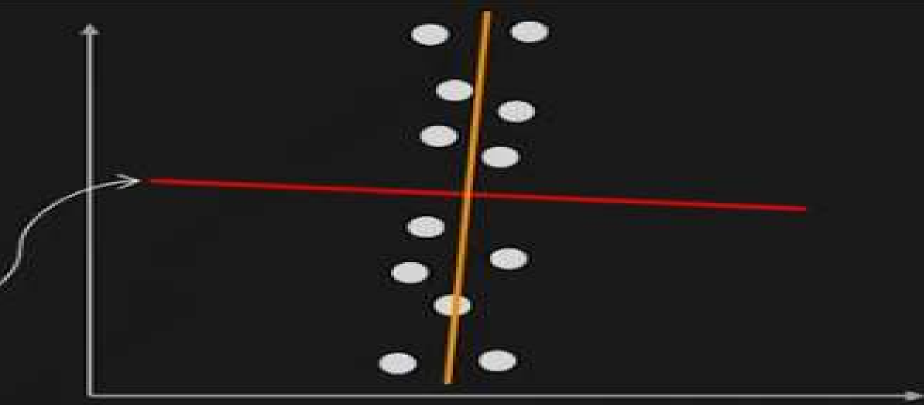
Solution:

$$m = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sum_i (x_i - \bar{x})^2} \qquad c = \bar{y} - m\bar{x}$$

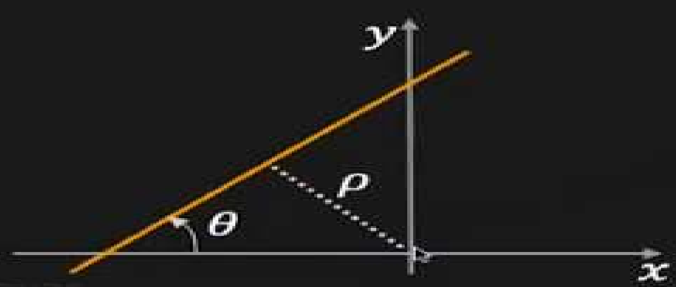
where: $\bar{x} = \frac{1}{N}\sum_i x_i \qquad \bar{y} = \frac{1}{N}\sum_i y_i$

Problem: When the points represent a vertical line.

Line that minimizes E!

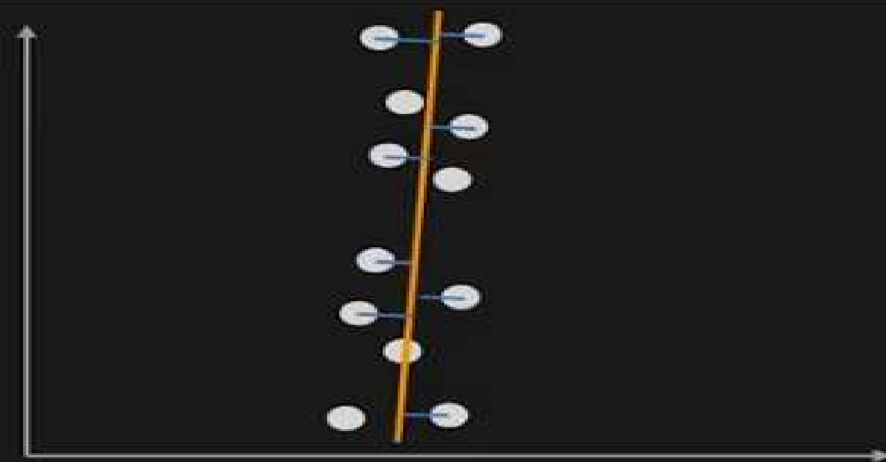


Solution: Use a different line equation



$$x \sin\theta - y \cos\theta + \rho = 0$$

Problem: When the points represent a vertical line.



Minimize: Average Squared **Perpendicular** Distance

$$E = \frac{1}{N}\sum_{i} \underbrace{(x_i \sin\theta - y_i \cos\theta + \rho)^2}_{\text{Perpendicular Distance}}$$

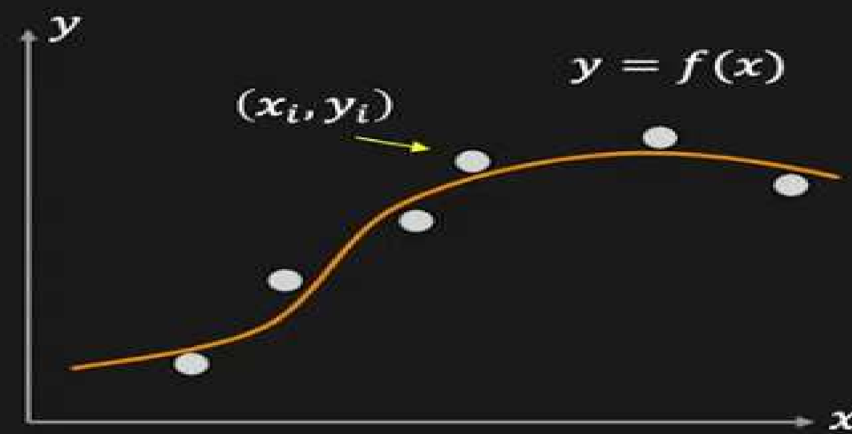
Fitting curves to edges

Given: Edge Points (x_i, y_i)

Task: Find polynomial

$y = f(x) = ax^3 + bx^2 + cx + d$

that best fits the points



Minimize:

$$E = \frac{1}{N}\sum_i (y_i - a{x_i}^3 - b{x_i}^2 - cx_i - d)^2$$

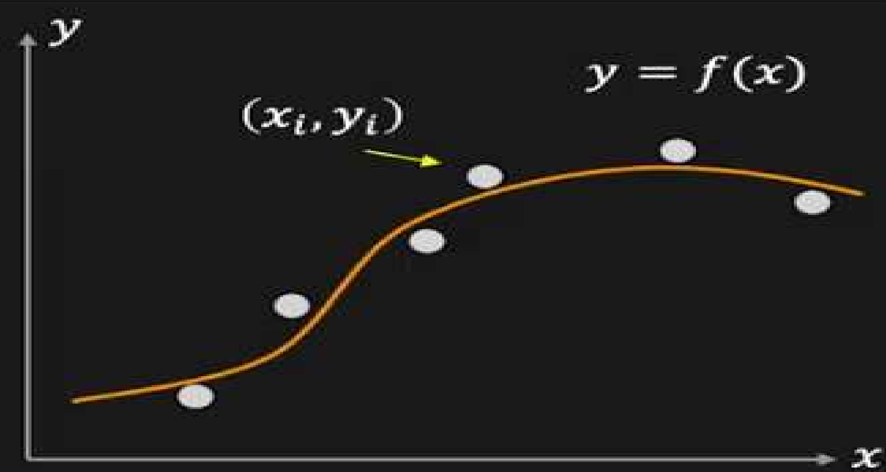
Solve the Linear System Using Least Squares Fit by:

$$\frac{\partial E}{\partial a} = 0 \qquad \frac{\partial E}{\partial b} = 0 \qquad \frac{\partial E}{\partial c} = 0 \qquad \frac{\partial E}{\partial c} = 0$$

Overdetermined problem

Solving as a Linear System:

$$y_0 = ax_0{}^3 + bx_0{}^2 + cx_0 + d$$
$$y_1 = ax_1{}^3 + bx_1{}^2 + cx_1 + d$$
$$\vdots$$
$$y_i = ax_i{}^3 + bx_i{}^2 + cx_i + d$$
$$\vdots$$
$$y_n = ax_n{}^3 + bx_n{}^2 + cx_n + d$$



Given many (x_i, y_i)'s, this is an over-determined linear system with four unknowns (a, b, c, d).

Solving Linear Equations

An over-determined linear system with m unknowns $\{a_j\}$ $(j = 0, \dots, m)$ and n observations $\{(x_{ij}, y_i)\}$ $(i = 0, \dots, n)$ $(n > m)$ can be written in a matrix form.

$$\underbrace{\begin{bmatrix} x_{00} & x_{01} & \cdots & x_{0m} \\ x_{10} & x_{11} & \cdots & x_{1m} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n0} & x_{n1} & \cdots & x_{nm} \end{bmatrix}}_{\substack{X_{n\times m} \\ \text{Known}}} \underbrace{\begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_m \end{bmatrix}}_{\substack{\mathbf{a}_{m\times 1} \\ \text{Unknown}}} = \underbrace{\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}}_{\substack{\mathbf{y}_{n\times 1} \\ \text{Known}}}$$

$$X\mathbf{a} = \mathbf{y}$$

$X_{n\times m}$ is not a square matrix and hence not invertible.

Least Squares Solution:

$$X^T X\mathbf{a} = X^T\mathbf{y} \quad \Rightarrow \quad \mathbf{a} = (X^T X)^{-1} X^T \mathbf{y}$$

$$X^+ = (X^T X)^{-1} X^T$$

(Pseudo Inverse)

$$\boxed{\mathbf{a} = X^+\mathbf{y}}$$

Line detectors (Hough Transform)

Orientation Histogram