Questions from Last Class?

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.

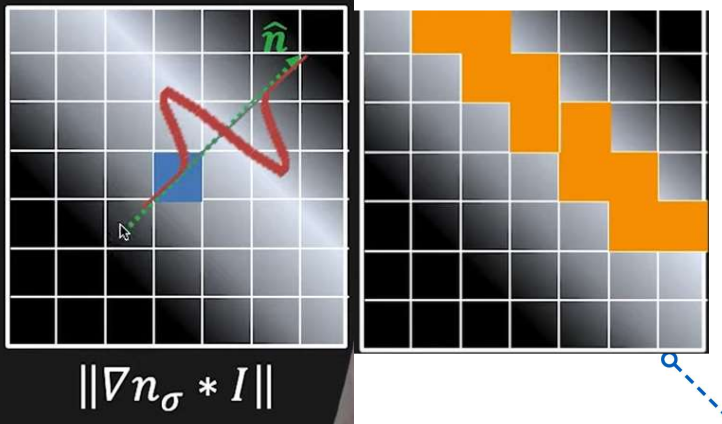
Canny Edge Detector

- 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:

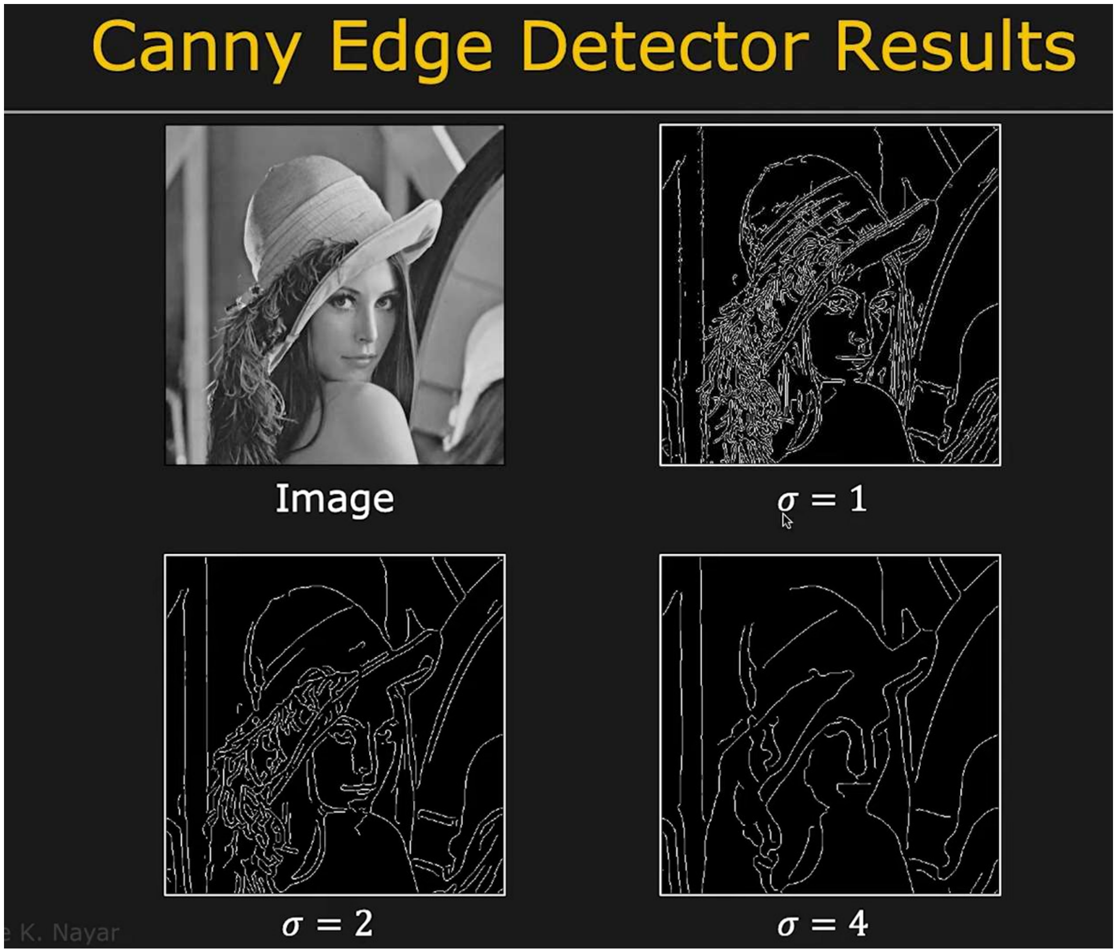
$$\hat{\boldsymbol{n}} = \frac{\nabla n_\sigma * I}{\|\nabla n_\sigma * I\|}$$

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

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



Find Zero Crossings in Laplacian to find the edge location


Canny Edge Detector Results
Image
$\sigma = 1$
$\sigma = 2$
$\sigma = 4$

Canny edge detector

1. Filter image with derivative of Gaussian
2. Find the magnitude and orientation of the gradient
3. **Non-maximum suppression**:
   - Thin multi-pixel wide "ridges" down to single-pixel width
4. Linking and thresholding (**hysteresis**):
   - Define two thresholds: low and high
   - Use the high threshold to start edge curves and the low threshold to continue them

Source: D. Lowe, L. Fei-Fei

71

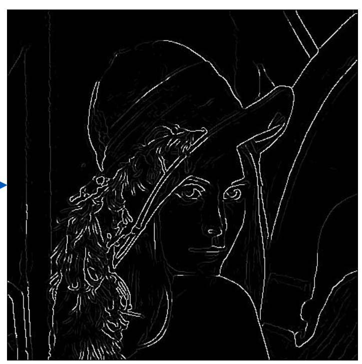
Grayscale image

Gx image

Gy image

Gradient mag

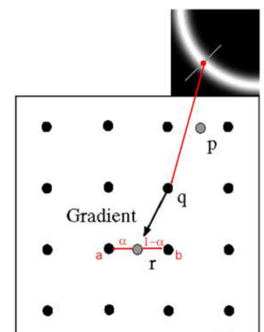
Non-maximal suppression



Thresholding

Hysteresis

Final edges after dropping weak edges



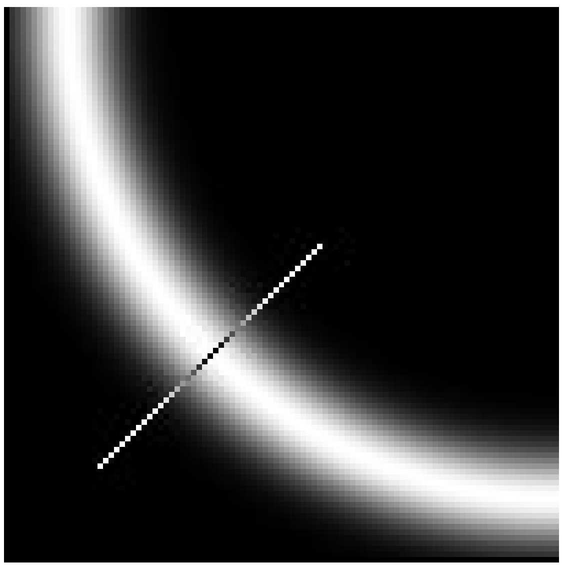
Non-maximal suppression

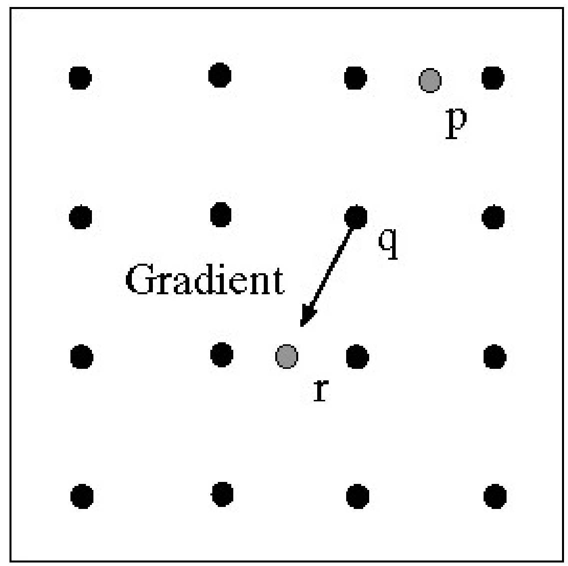
https://justin-liang.com/tutorials/canny/

Non-maximum suppression

Check if the pixel is local maximum along gradient direction, select single max across the width of the edge

- requires checking interpolated pixels p and r

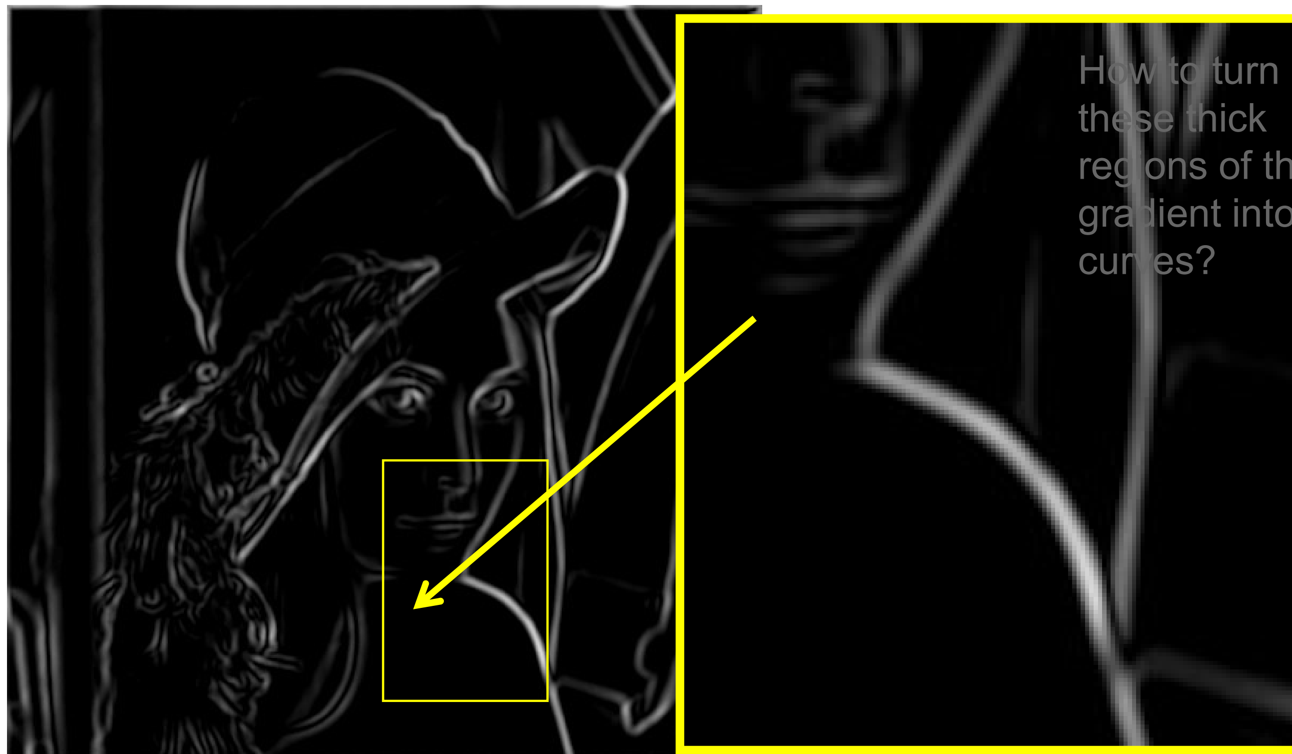




73

The Canny edge detector

Problem: pixels along this edge didn't survive the thresholding



thresholding

thinning
(non-maximum suppression)

74

The Canny edge detector

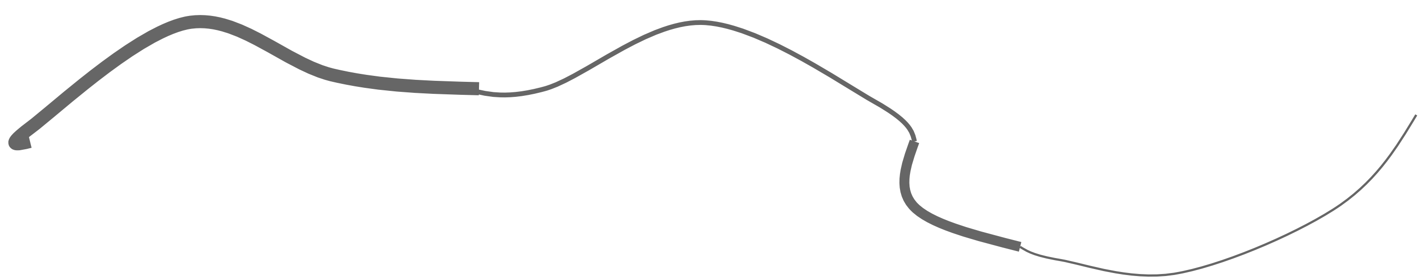
thinning
(non-maximum suppression)

Problem: pixels along this edge didn't survive the thresholding

75

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

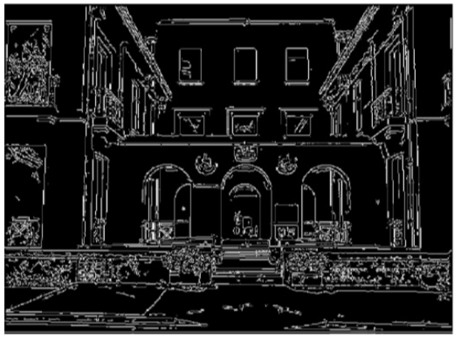
76

Hysteresis thresholding

original image

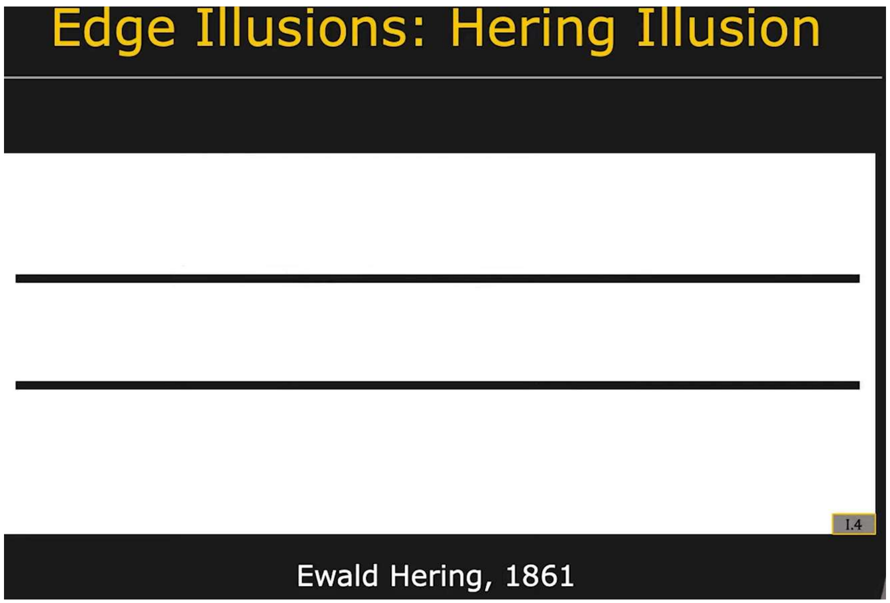
high threshold
(strong edges)

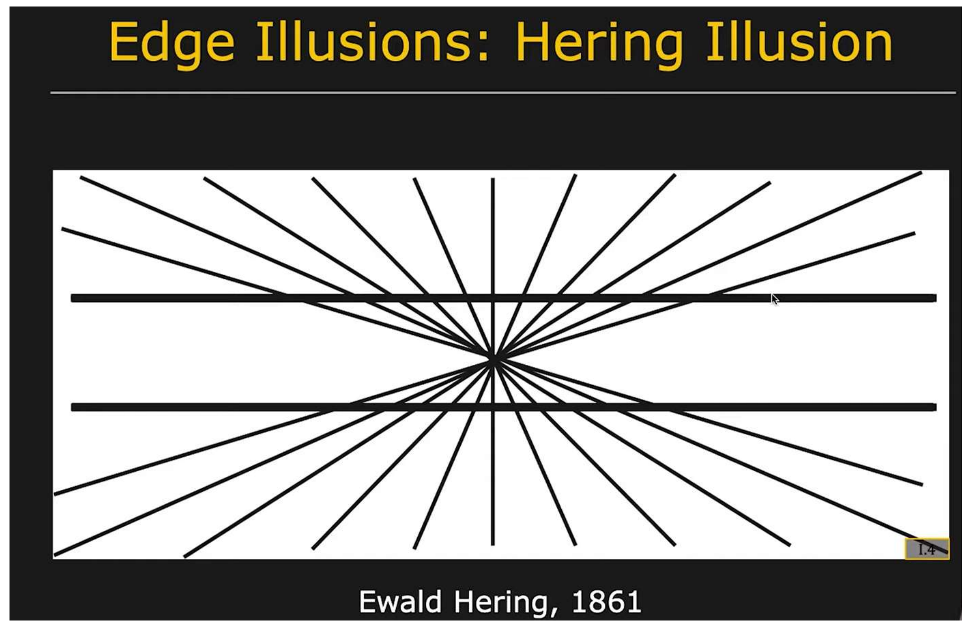
low threshold
(weak edges)



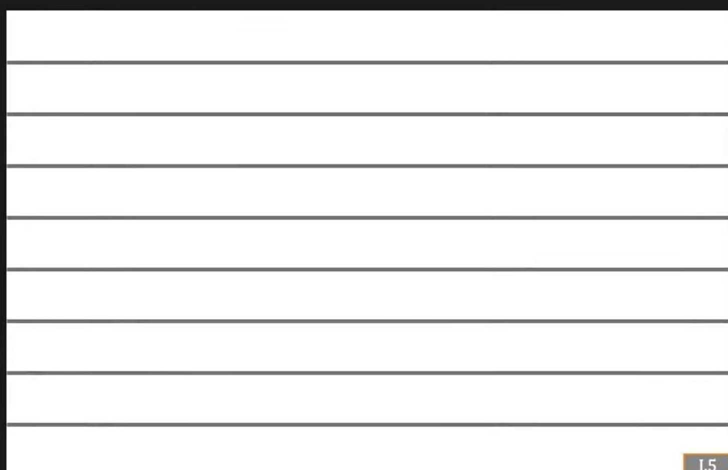
hysteresis threshold

Source: L. Fei-Fei


Edge Illusions: Hering Illusion
Ewald Hering, 1861

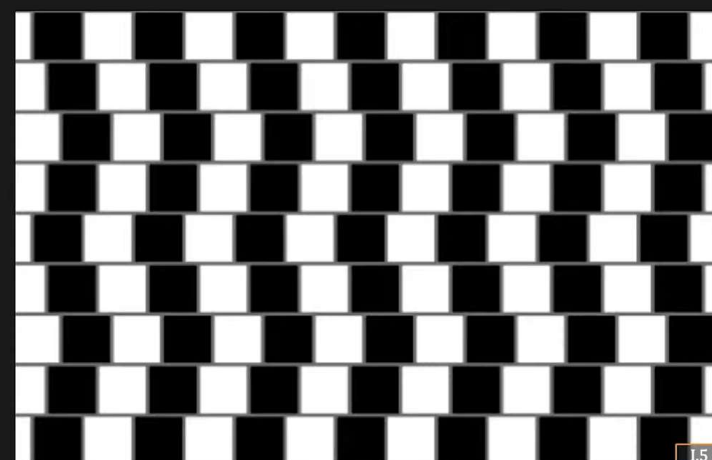

Edge Illusions: Hering Illusion
Ewald Hering, 1861

Edge Illusions: Café Wall Illusion



Gregory and Heard, 1979

Edge Illusions: Café Wall Illusion



Gregory and Heard, 1979

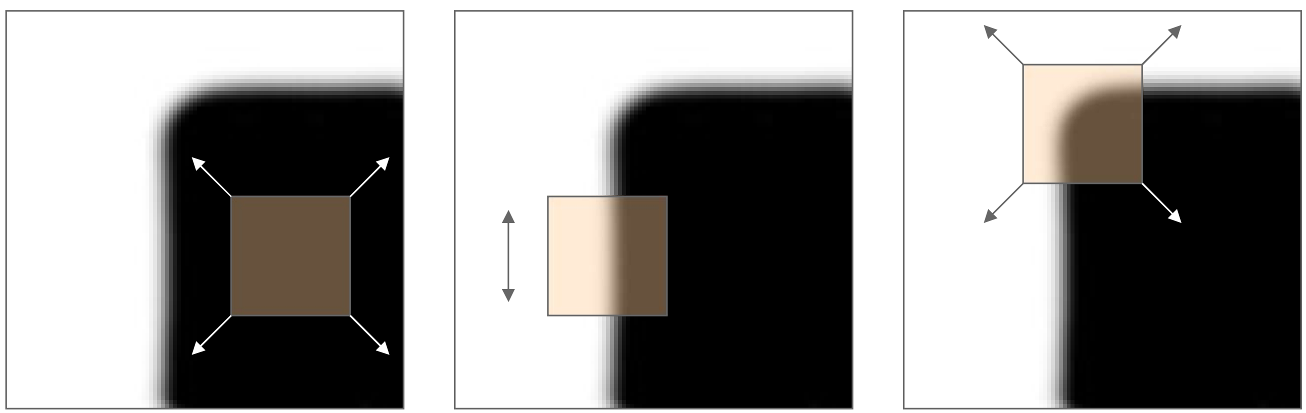
Corners

Corner: Point where Two Edges Meet. i.e., Rapid Changes of Image Intensity in Two Directions within a Small Regio

"Flat" Region — "Edge" Region — "Corner" Region

Corner Detection: Basic Idea

- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



"flat" region:
no change in all directions

"edge":
no change along the edge direction

"corner":
significant change in all directions

Source: A. Efros

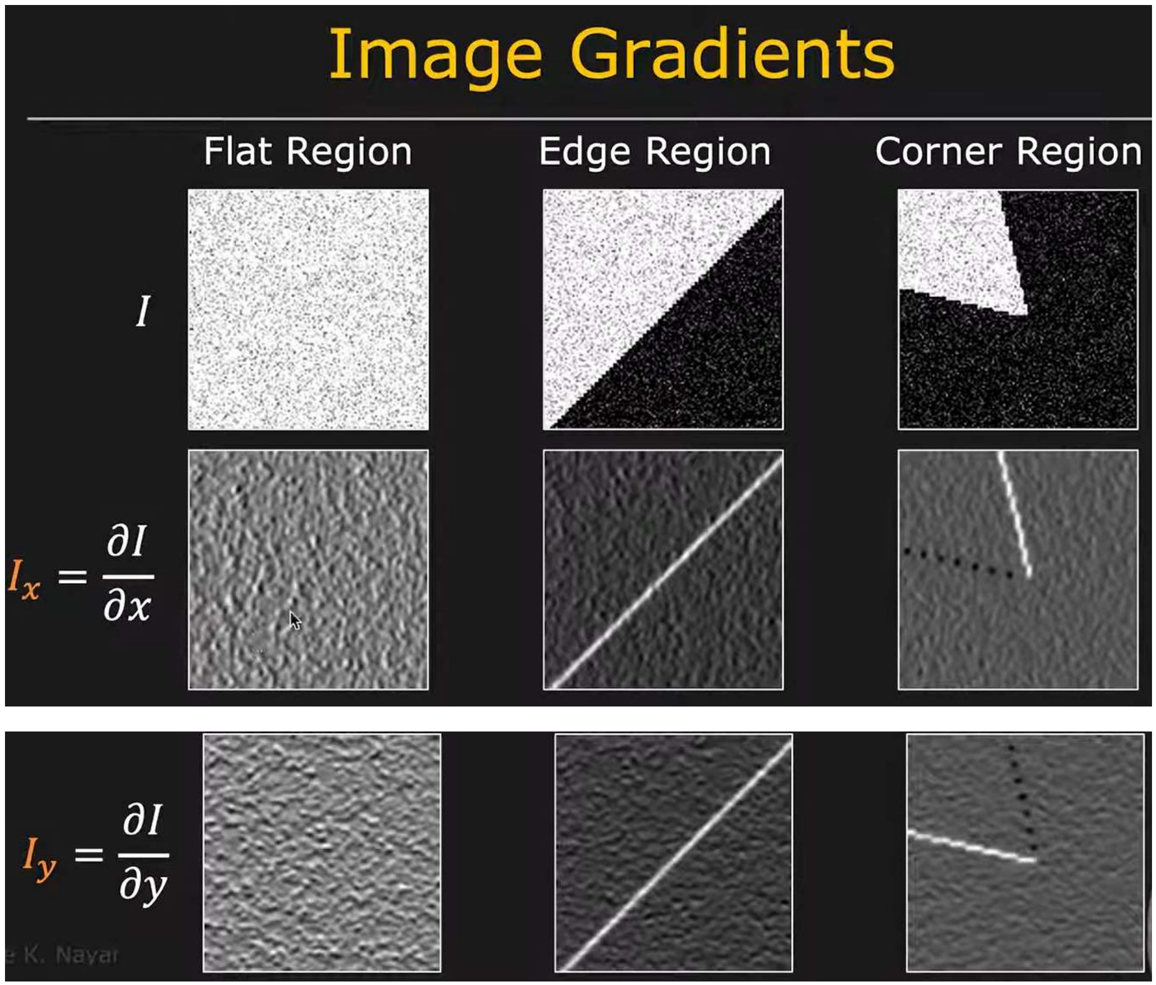
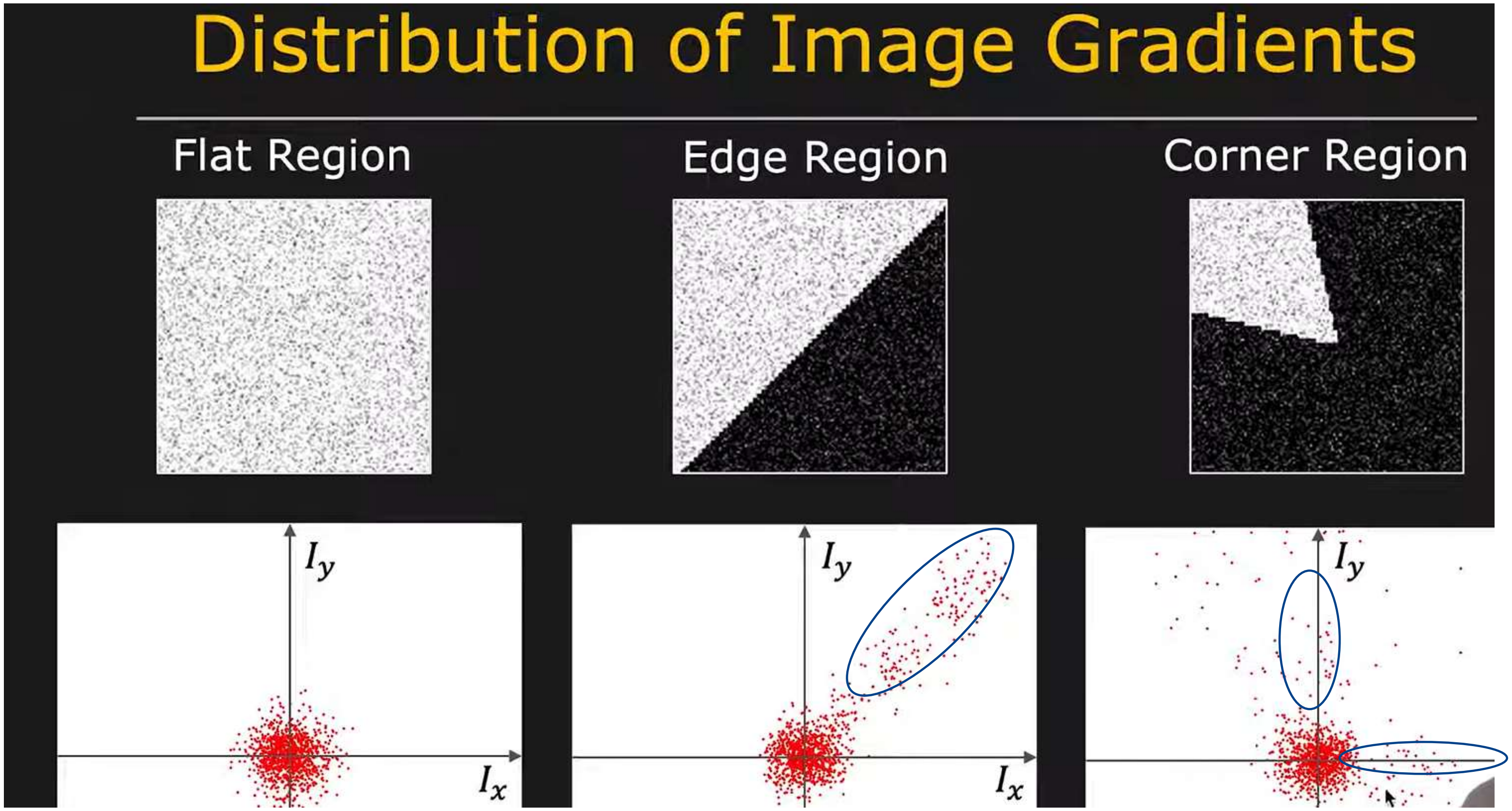
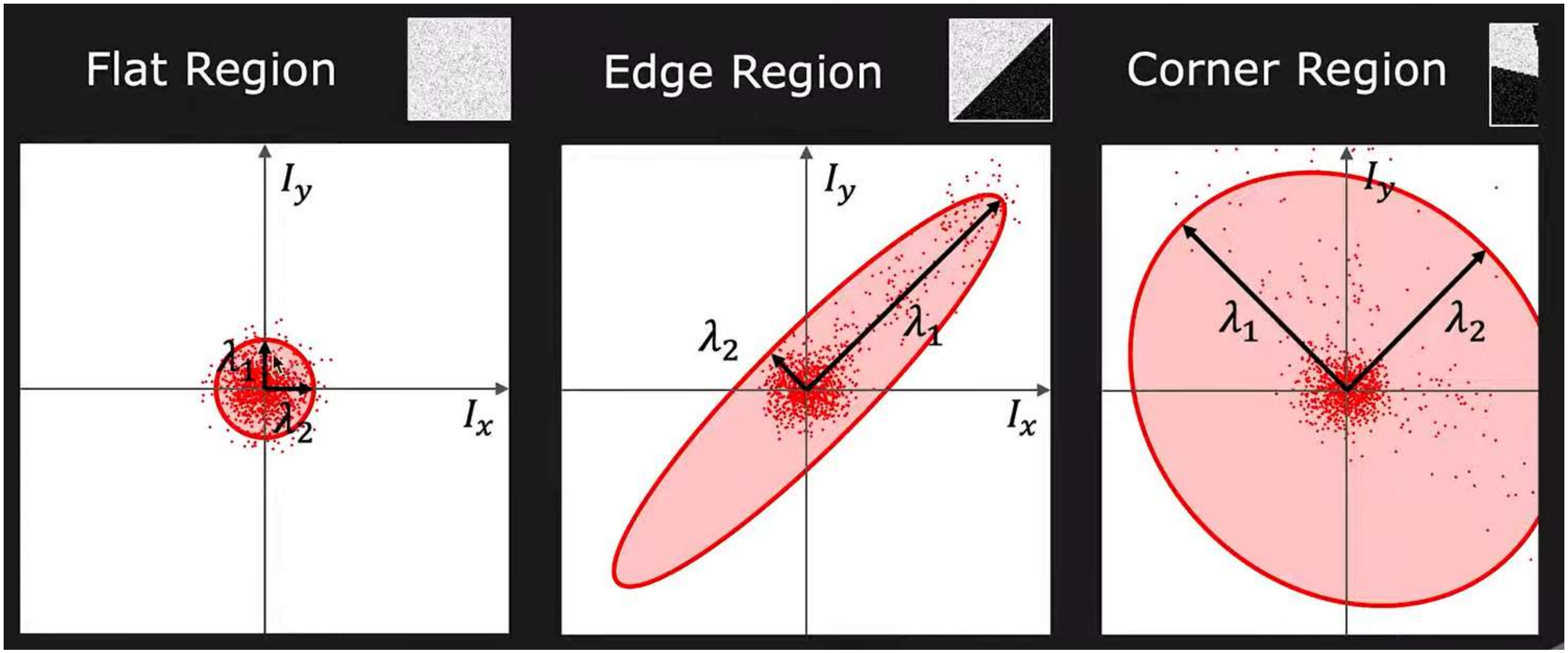


Image Gradients
Flat Region
Edge Region
Corner Region
I
$I_x = \frac{\partial I}{\partial x}$
$I_y = \frac{\partial I}{\partial y}$
K. Nayar

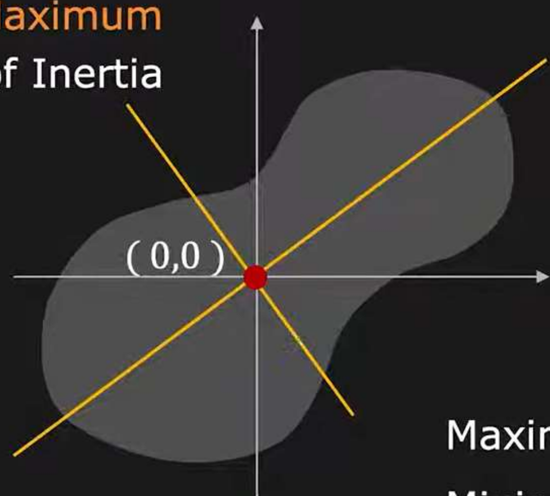

Distribution of Image Gradients
Flat Region
Edge Region
Corner Region
I_y
I_x
I_y
I_x
I_y


Flat Region
Edge Region
Corner Region
I_y
I_x
λ_1
λ_2

Fitting an Elliptical Disk

Axis of Maximum Moment of Inertia

Axis of Minimum Moment of Inertia



Maximum Moment of Inertia = E_{max}

Minimum Moment of Inertia = E_{min}

Length of Semi-Major Axis = $\lambda_1 = E_{max}$

Length of Semi-Minor Axis = $\lambda_2 = E_{min}$

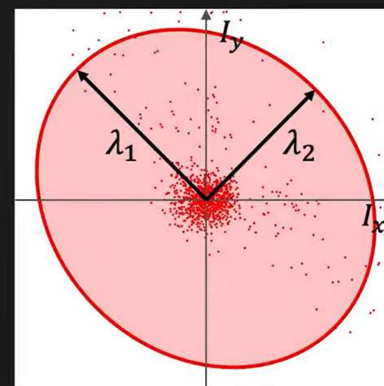
Fitting an Elliptical Disk

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

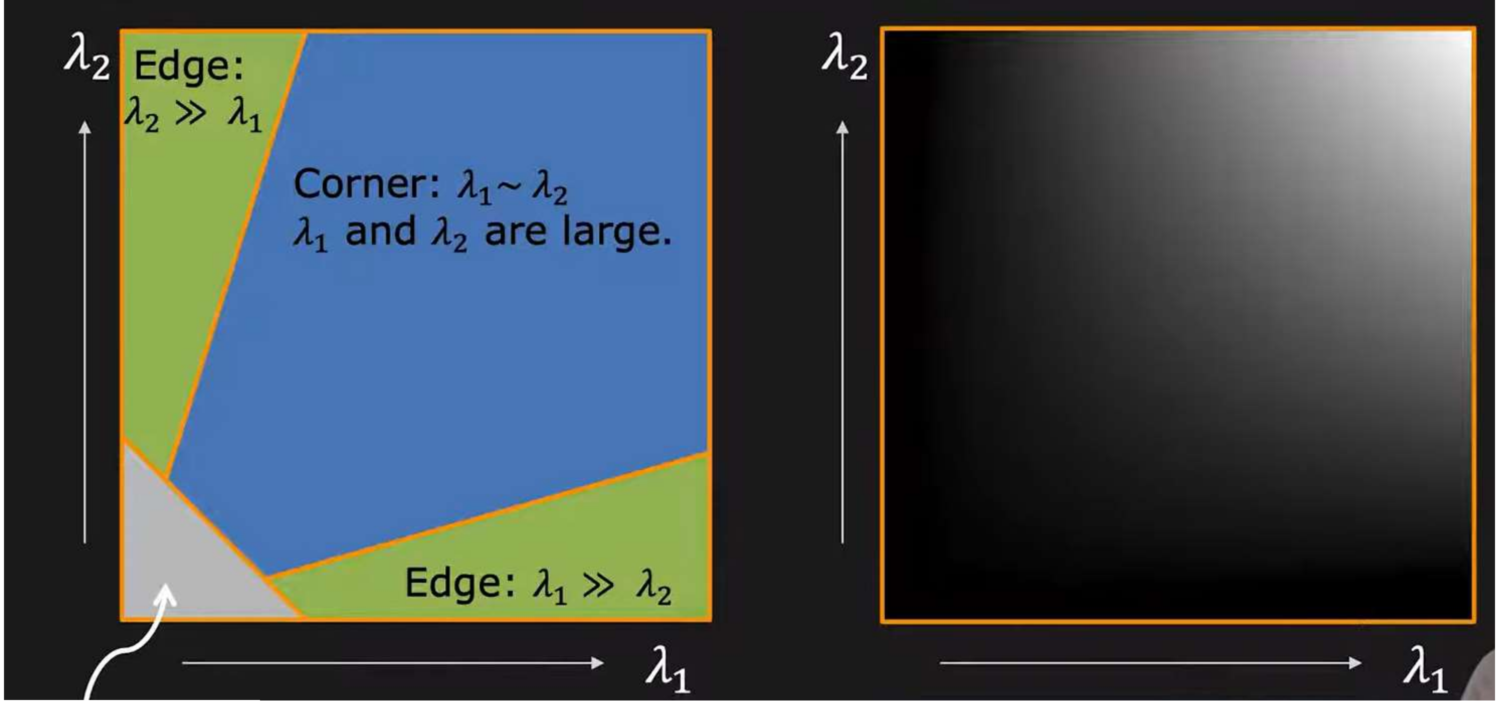


Ellipse Axes Lengths:

$$\lambda_1 = E_{max} = \frac{1}{2}\left[a + c + \sqrt{b^2 + (a-c)^2}\right]$$

$$\lambda_2 = E_{min} = \frac{1}{2}\left[a + c - \sqrt{b^2 + (a-c)^2}\right]$$

Harris Corner Response Function

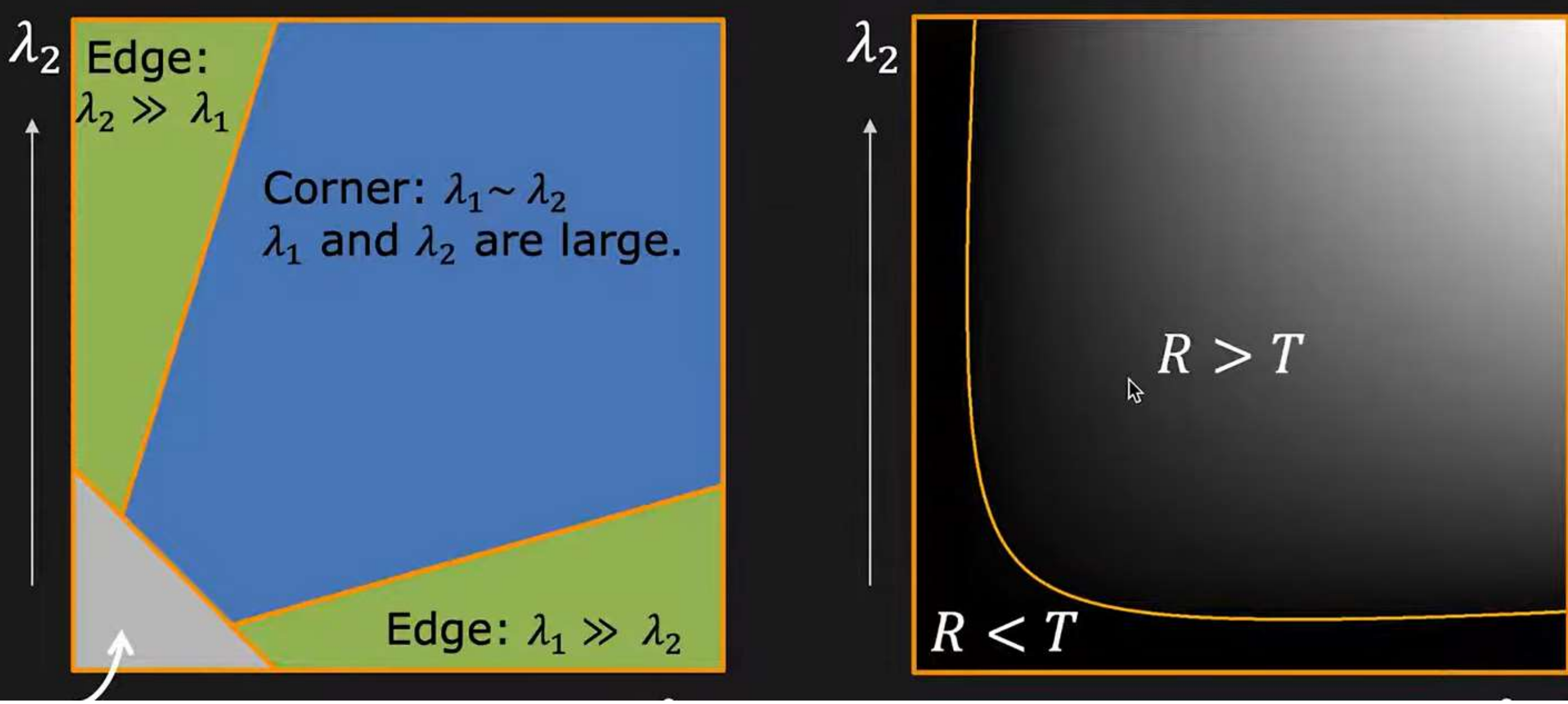


Flat: $\lambda_1 \sim \lambda_2$
λ_1 and λ_2 are small.

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

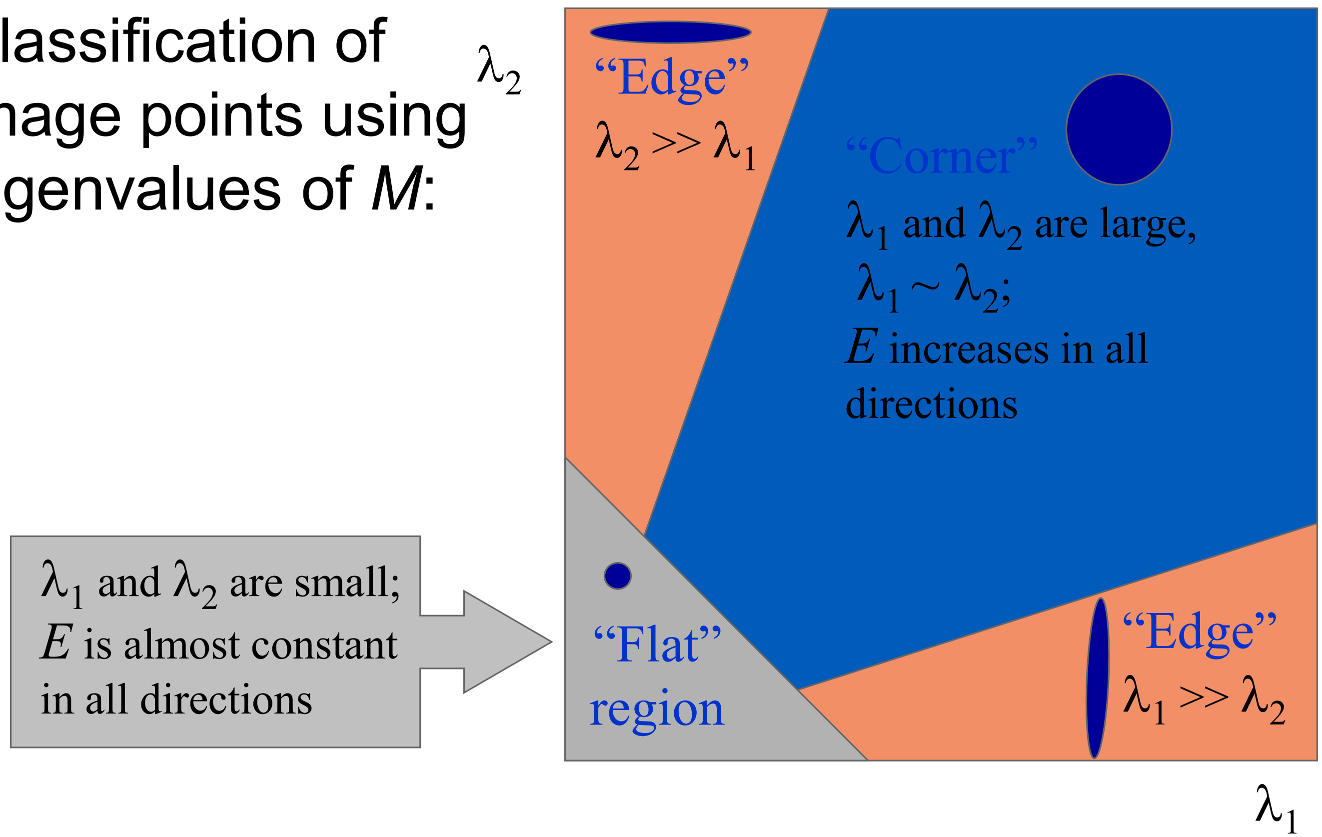
where: $0.04 \leq k \leq 0.06$
(Designed Empirically)

Harris Corner Response Function



Interpreting the eigenvalues

Classification of image points using eigenvalues of *M*:

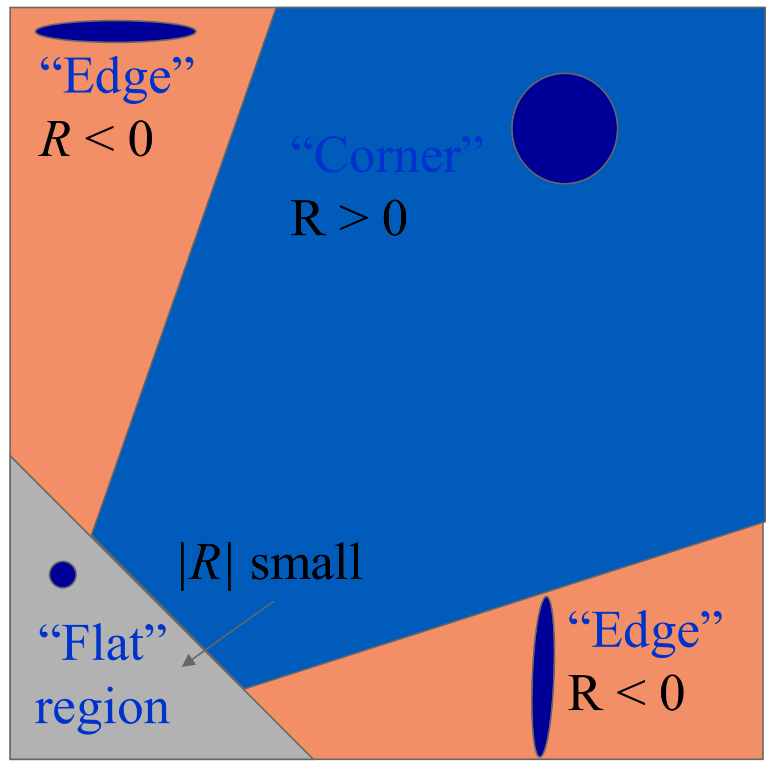


Corner response function

$$R = \det(M) - \alpha\,\mathrm{trace}(M)^2 = \lambda_1\lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$

$$M = \sum_{(x,y)\in W} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \sum_{(x,y)\in W} I_x^2 & \sum_{(x,y)\in W} I_x I_y \\ \sum_{(x,y)\in W} I_x I_y & \sum_{(x,y)\in W} I_y^2 \end{bmatrix}$$

α: constant (0.04 to 0.06)

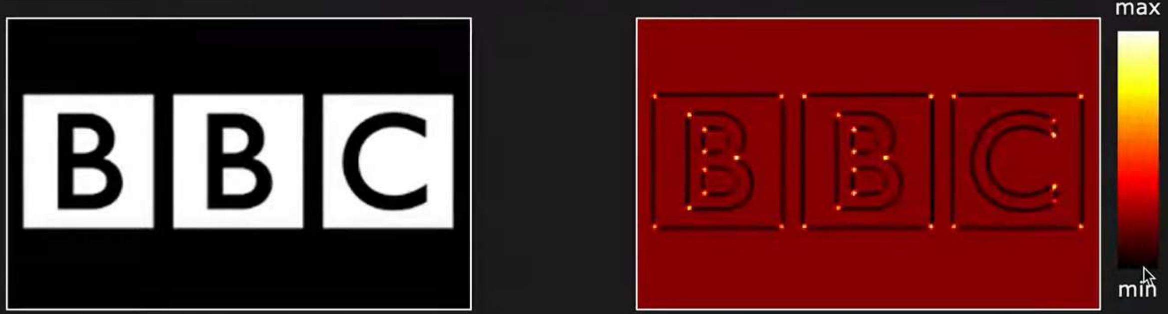


Harris corner detector

- Algorithm steps:
  - Compute M matrix within all image windows to get their R scores
  - Find points with large corner response
    (R > threshold)
  - Take the points of local maxima of R

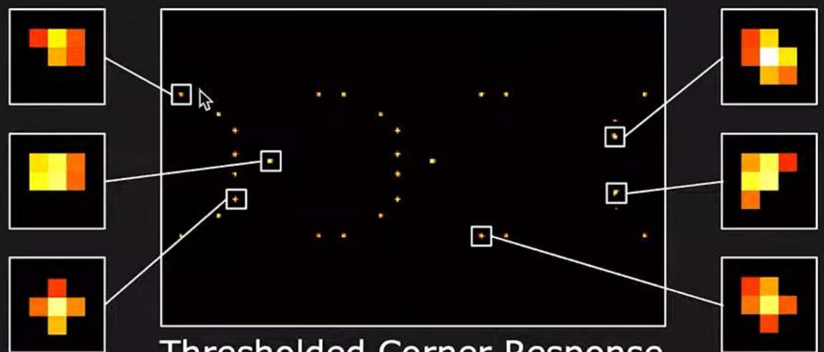
C.Harris and M.Stephens. "A Combined Corner and Edge Detector." *Proceedings of the 4th Alvey Vision Conference*: pages 147—151, 1988.

Harris Corner Detection Example



Image

Corner Response R



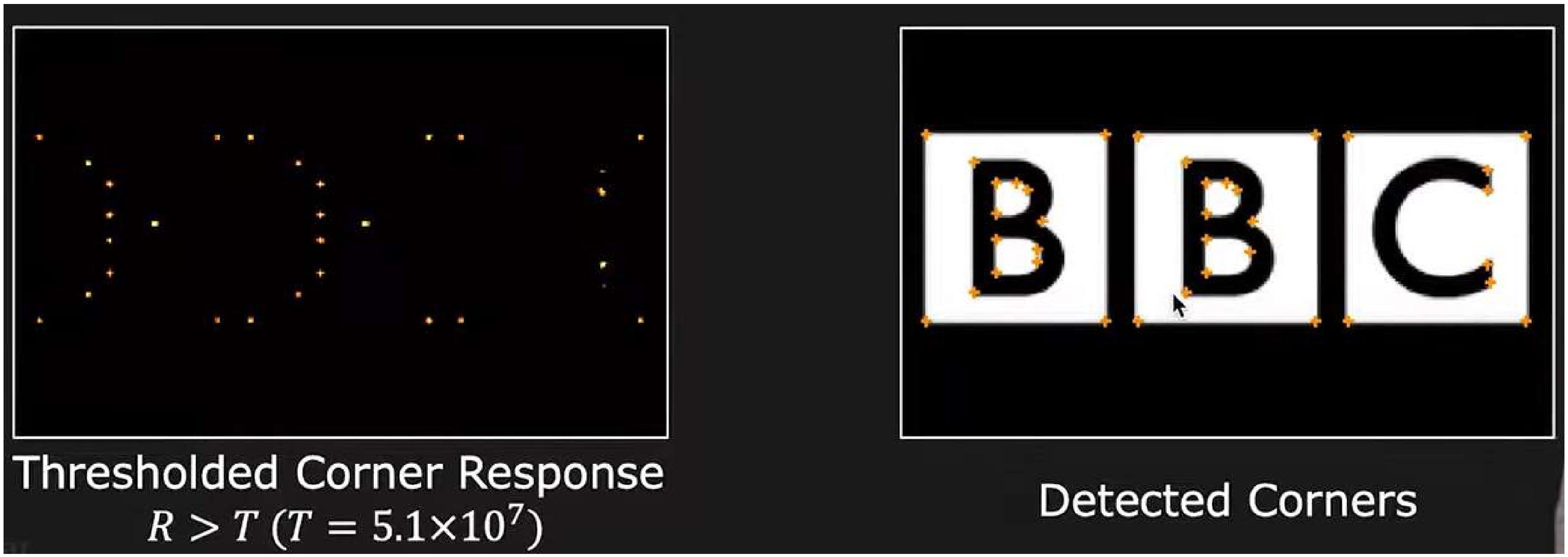
Thresholded Corner Response
$R > T$

Non-Maximal Suppression

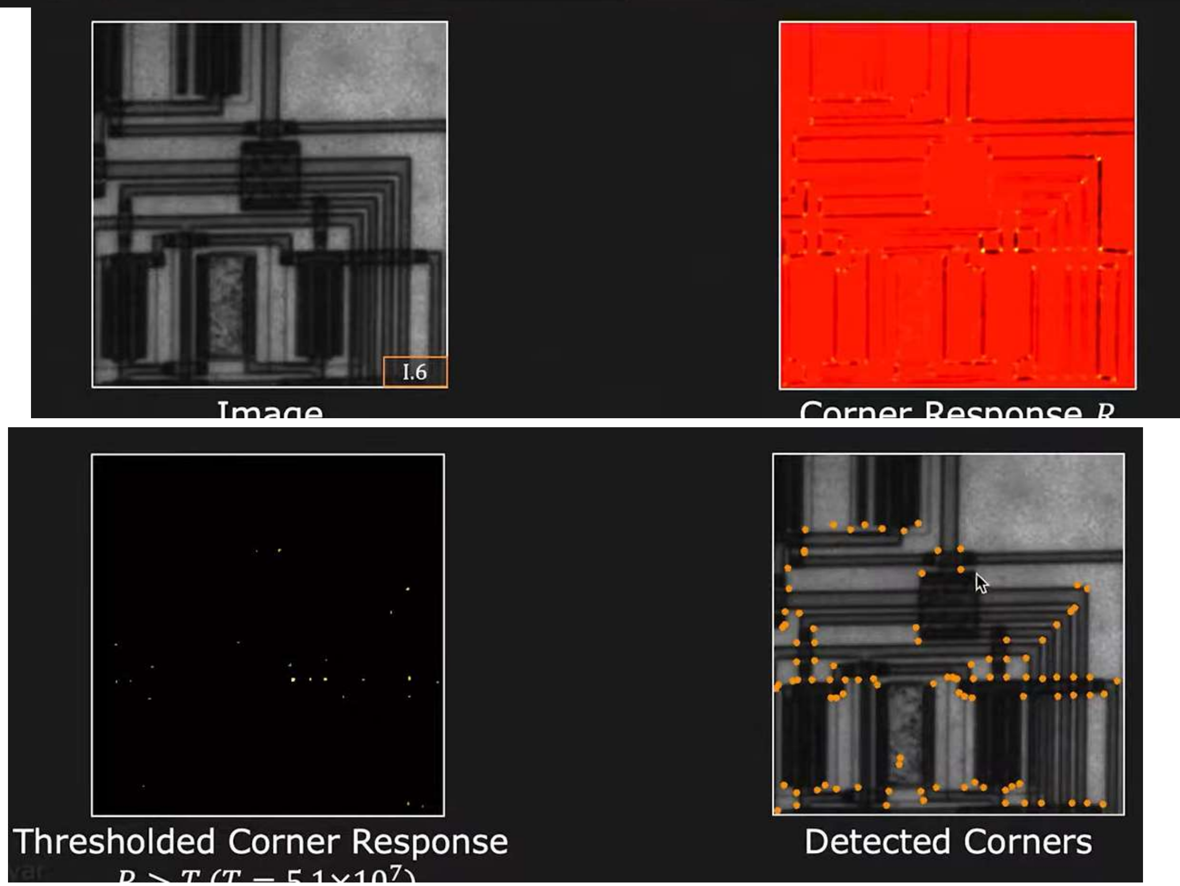
1. Slide a window of size k over the image.

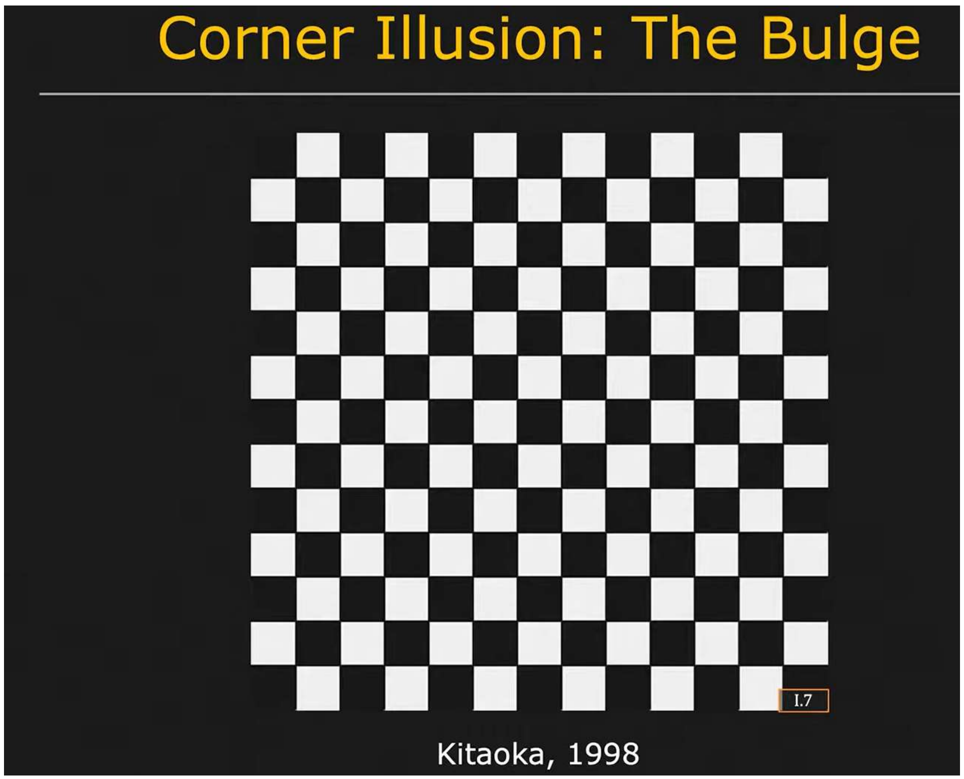
2. At each position, if the pixel at the center is the maximum value within the window, label it as positive (retain it). Else label it as negative (suppress it).

Suppress | Suppress | Retain


Thresholded Corner Response
$R > T \ (T = 5.1 \times 10^7)$
Detected Corners

Harris Corner Detection Example




Corner Illusion: The Bulge
I.7
Kitaoka, 1998

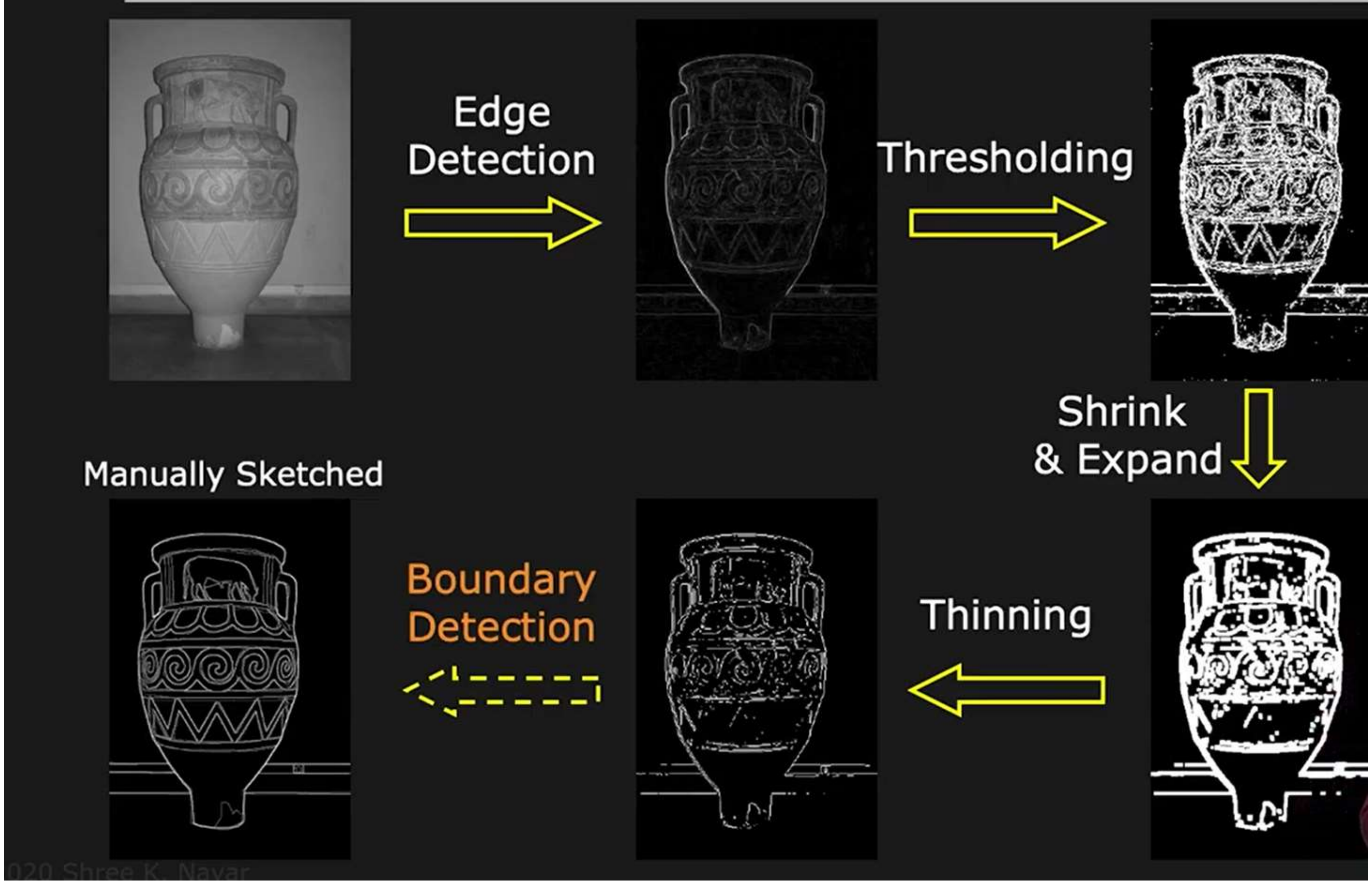

Corner Illusion: The Bulge
I.7
Kitaoka, 1998

Boundary Detection

We need to find Object Boundaries from Edge Pixels.

Preprocessing Edge Images

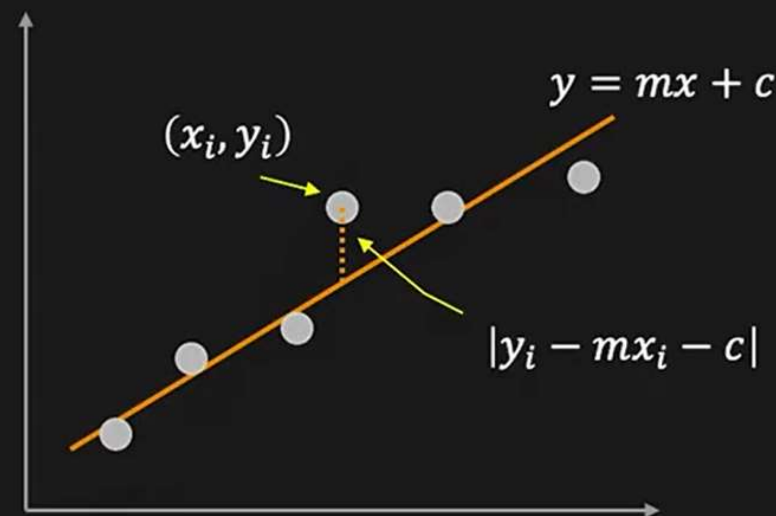
Preprocessing Edge Images



Fitting Lines to Edges

Given: Edge Points (x_i, y_i)

Task: Find (m, c)



Minimize: Average Squared Vertical Distance

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

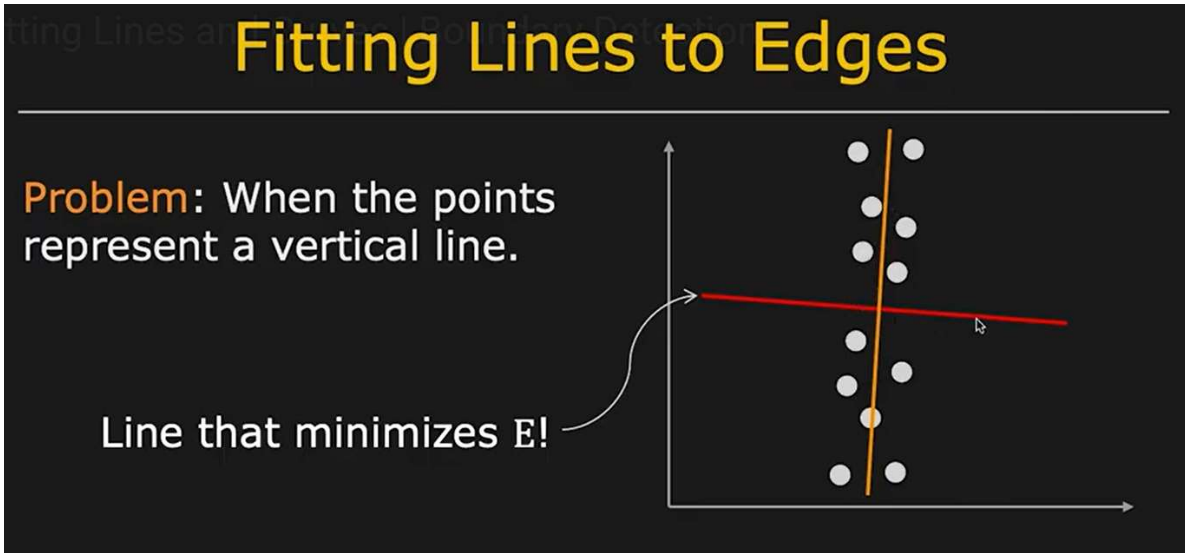
Least Squares Solution:

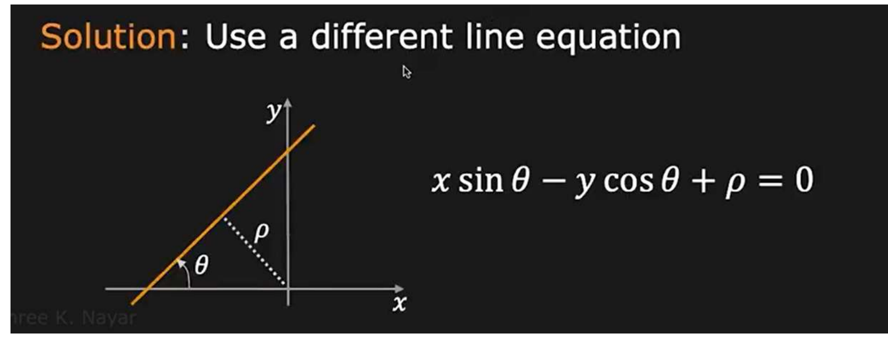
$$\frac{\partial E}{\partial m} = \frac{-2}{N} \sum_i x_i (y_i - m x_i - c) = 0 \qquad \frac{\partial E}{\partial c} = \frac{-2}{N} \sum_i (y_i - m x_i - c) = 0$$

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





Minimize: Average Squared Perpendicular Distance

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