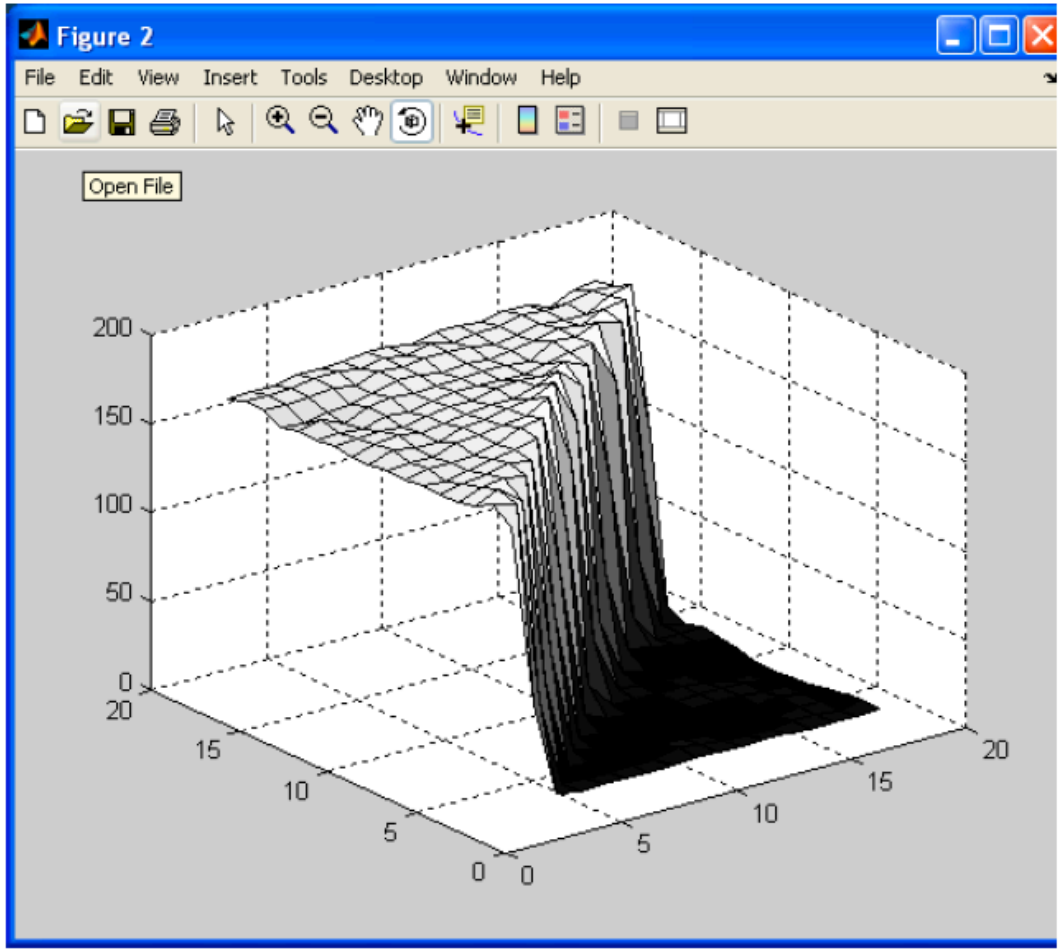
Edge and Line Detection

Chetan Arora

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

What are Edges?

- Mathematically: discontinuity in intensity or color.



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

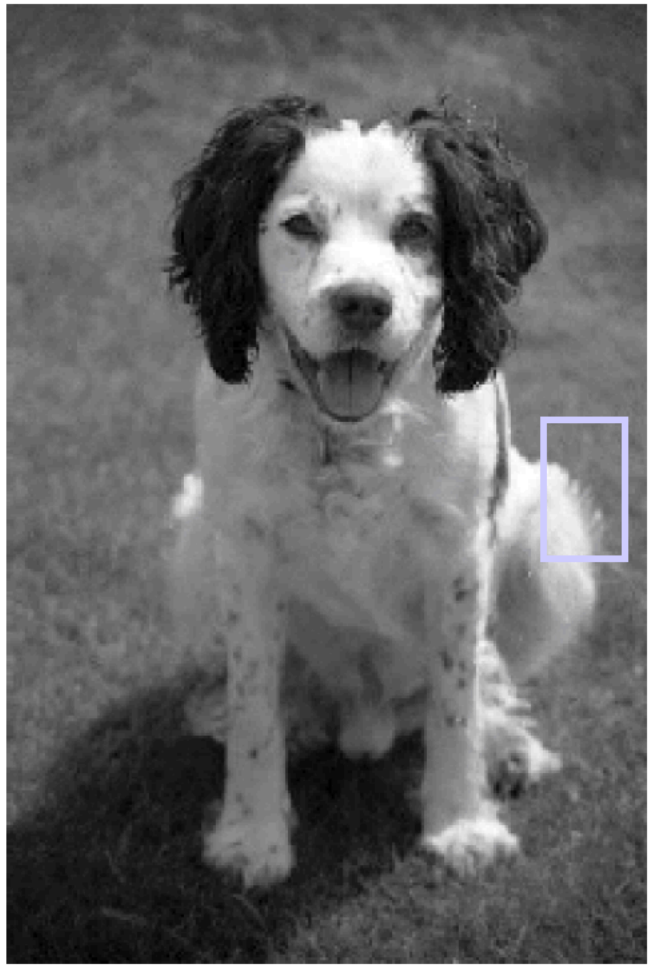
What are Edges?

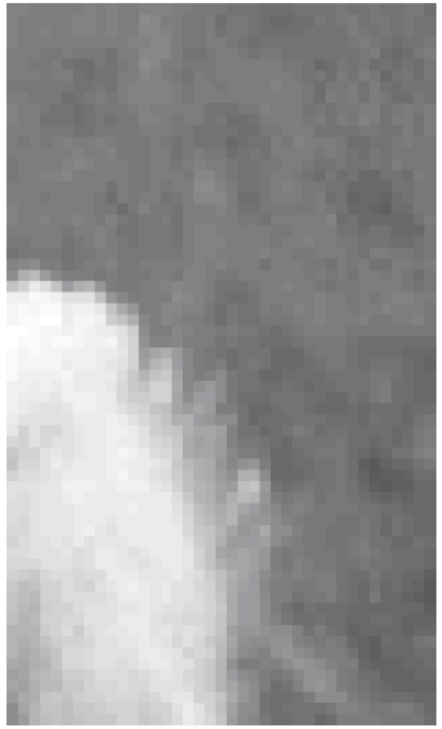
- Mathematically: discontinuity in intensity or color.

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

What are Edges?

- Semantically: object boundaries

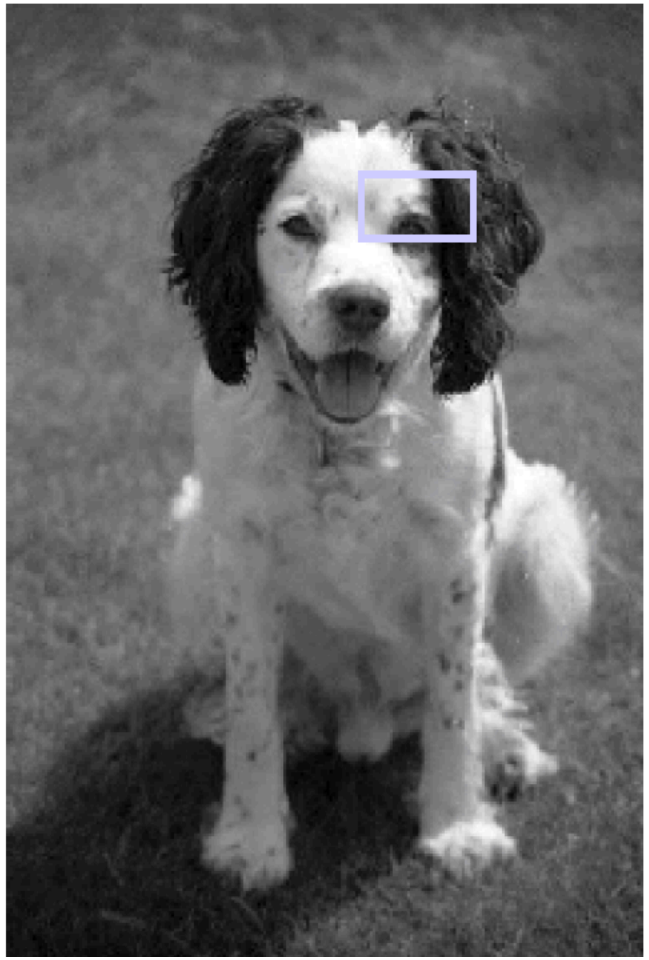


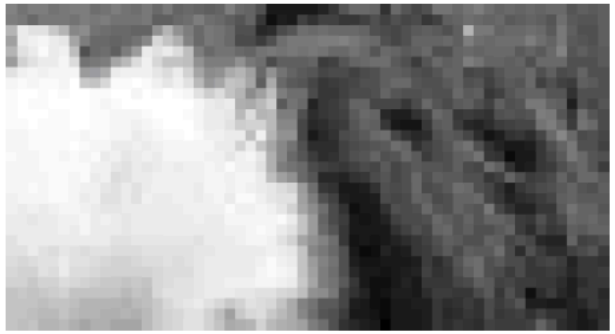


Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

What are Edges?

- Semantically: boundaries of material properties

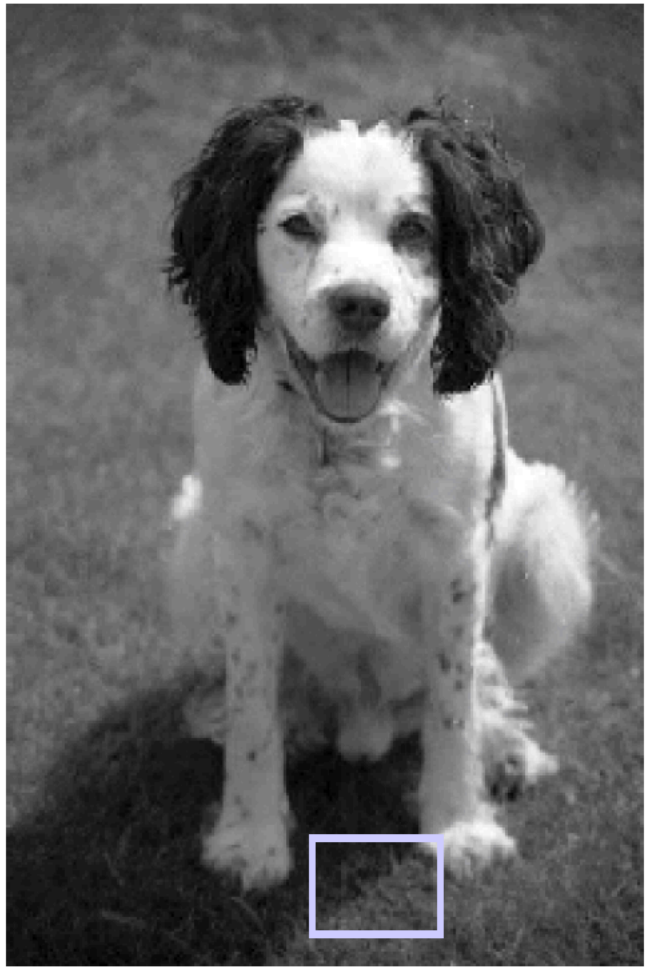




Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

What are Edges?

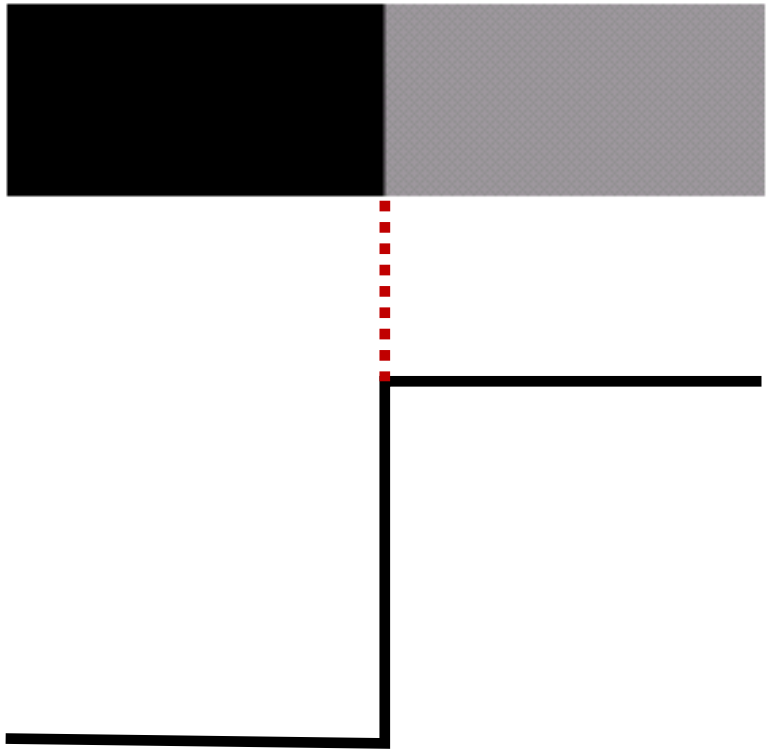
- Semantically: lighting boundary



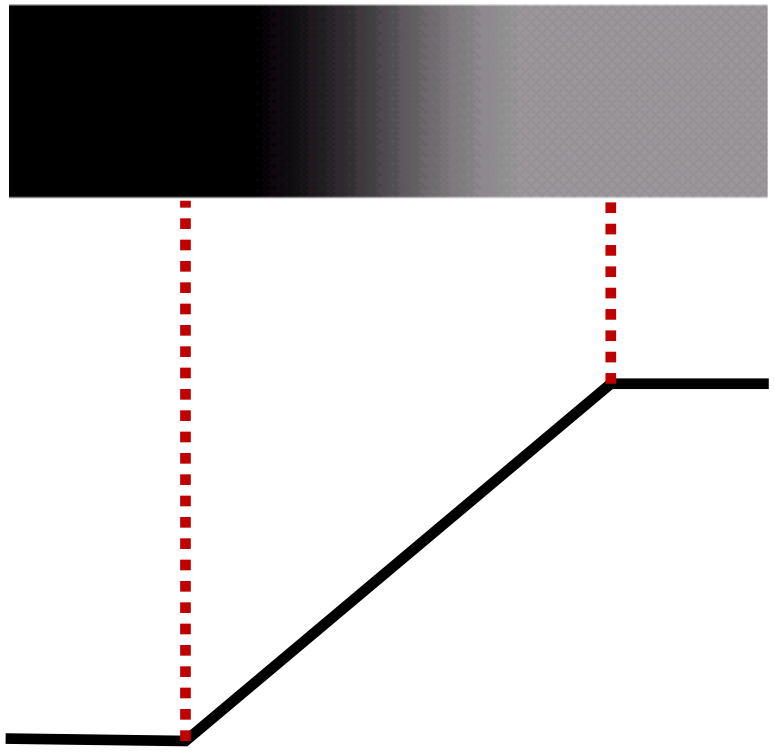
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Types of Edges

Step edge



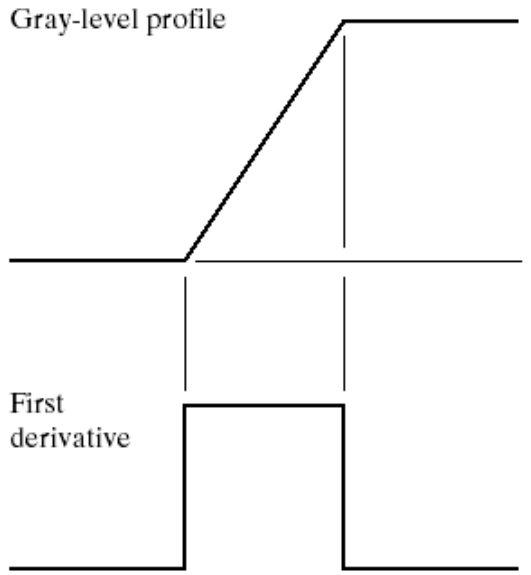
Ramp edge

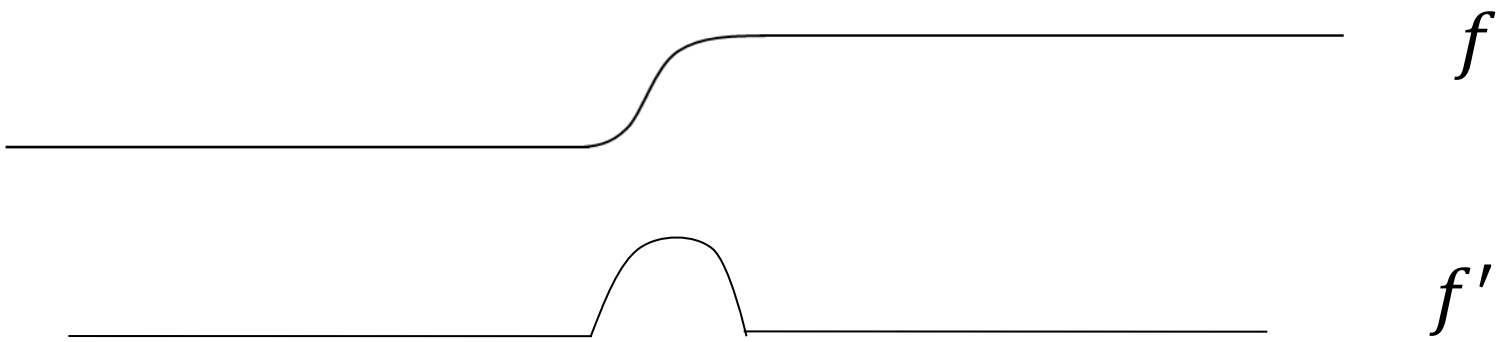


Gray-level profile of the edge

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Derivate Response to a Ramp Edge



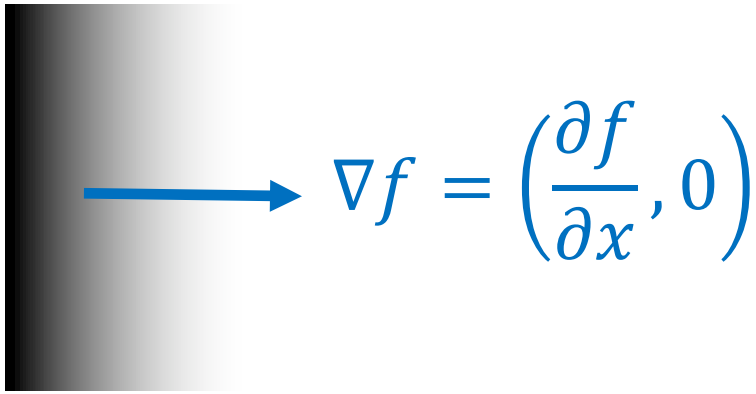


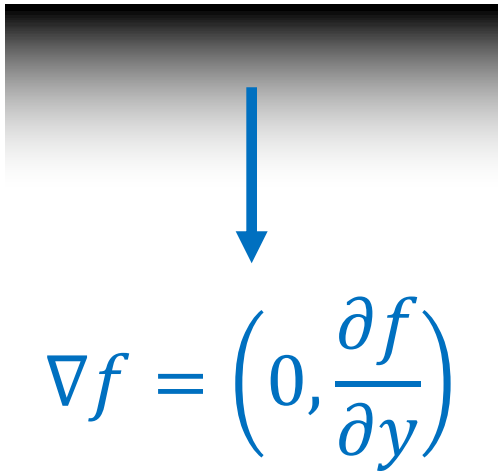
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

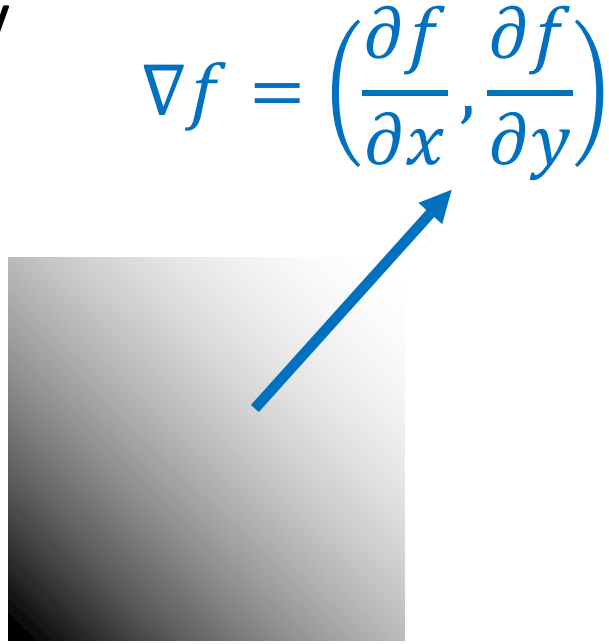
Image Gradient

- Gradient: The vector of derivatives $\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right) = (I_x, I_y)$

- Gradient: The direction of most rapid change in intensity







Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Image Gradient

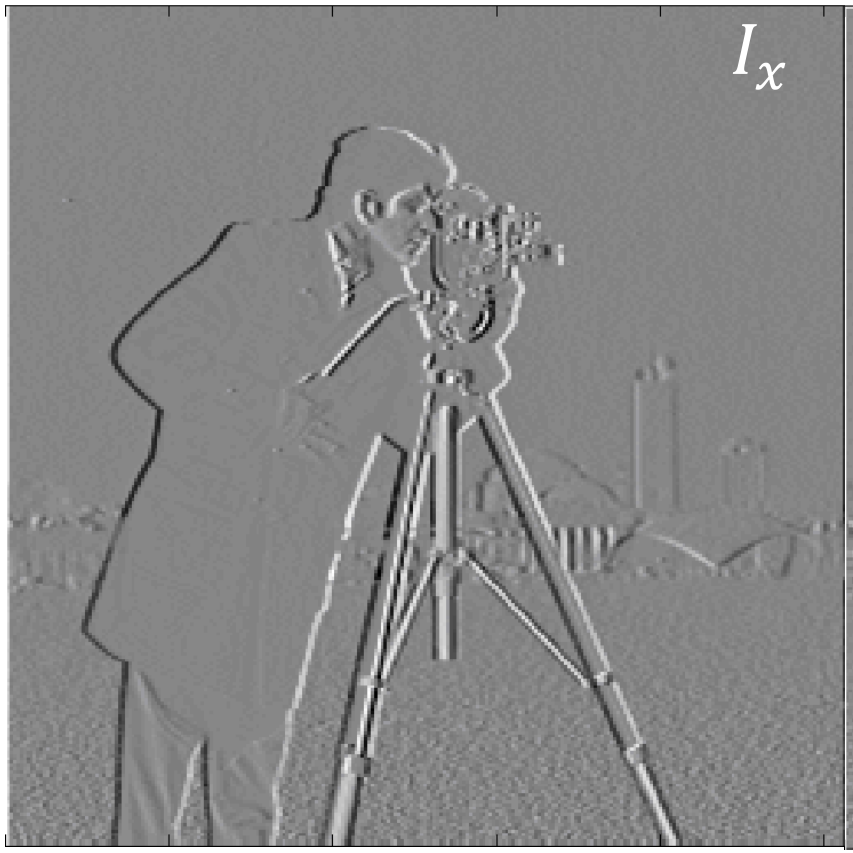
$$\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$$

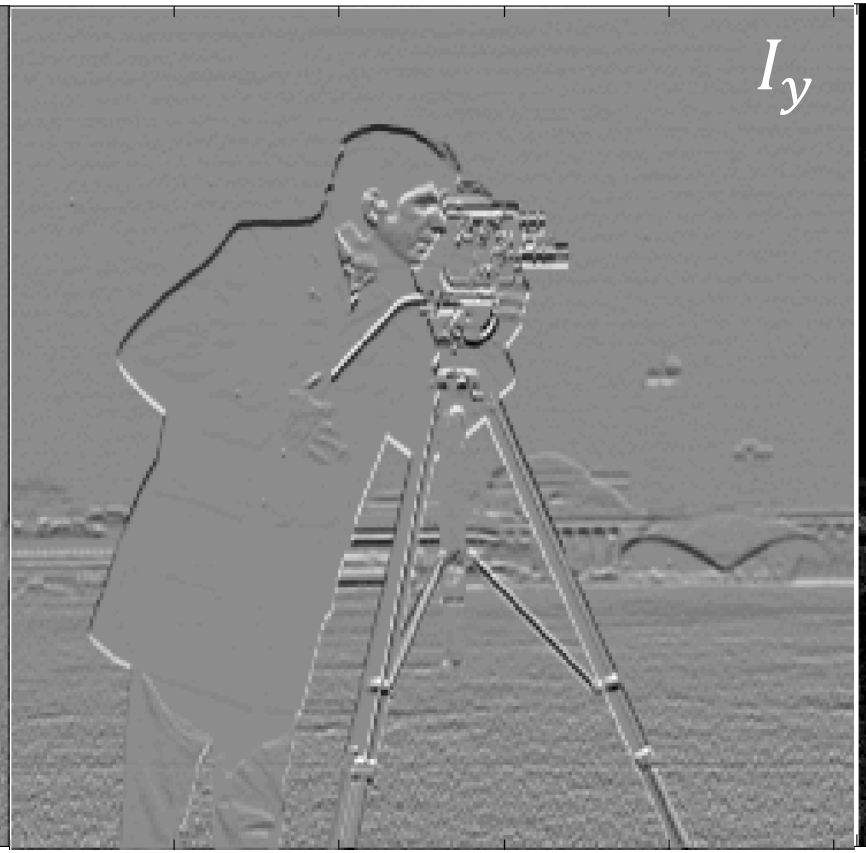
- Edge (Gradient) Magnitude: $|\nabla f| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$
- Edge (Gradient) Direction: $\alpha = \tan^{-1}\left(\frac{\partial f}{\partial y} \Big/ \frac{\partial f}{\partial x}\right)$
- Directional Derivative in direction α: $\cos(\alpha)\frac{\partial f}{\partial x} + \sin(\alpha)\frac{\partial f}{\partial y}$

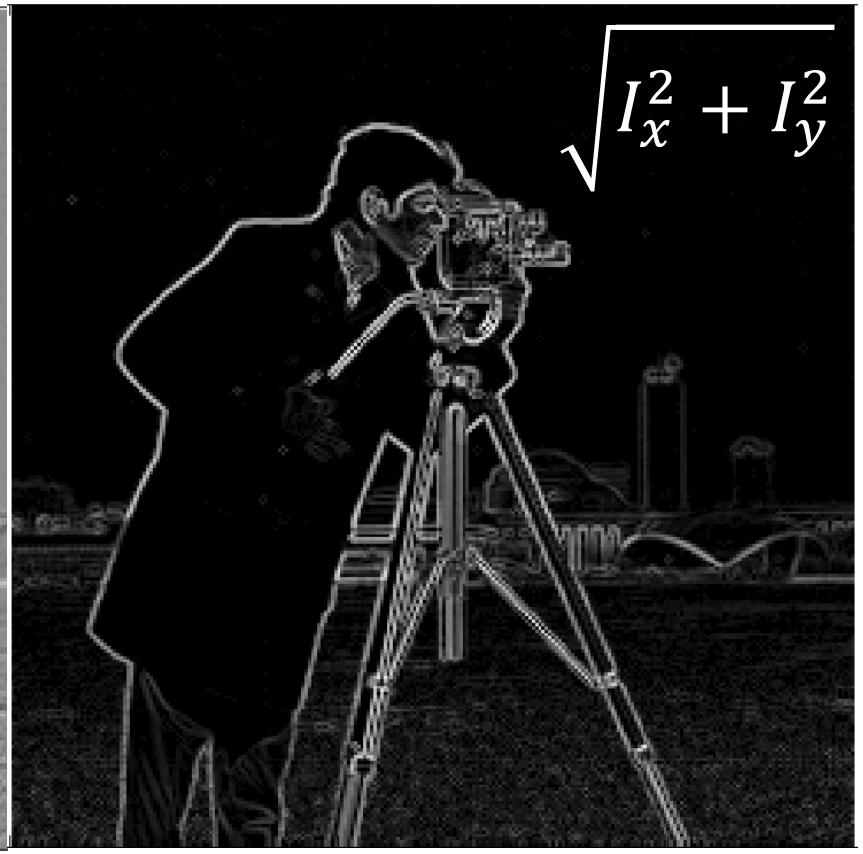
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Image Gradient

Original





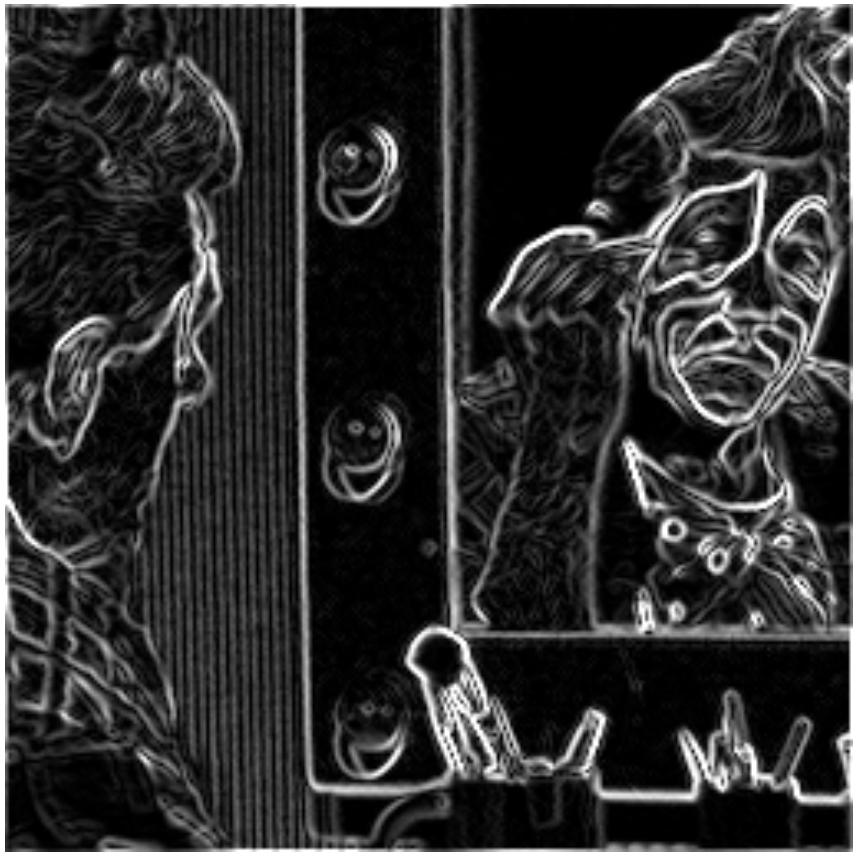


Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Edge Detection using Image Gradients

Original

|Gradient|



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Edge Detection using Image Gradients

Original

|Gradient|

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

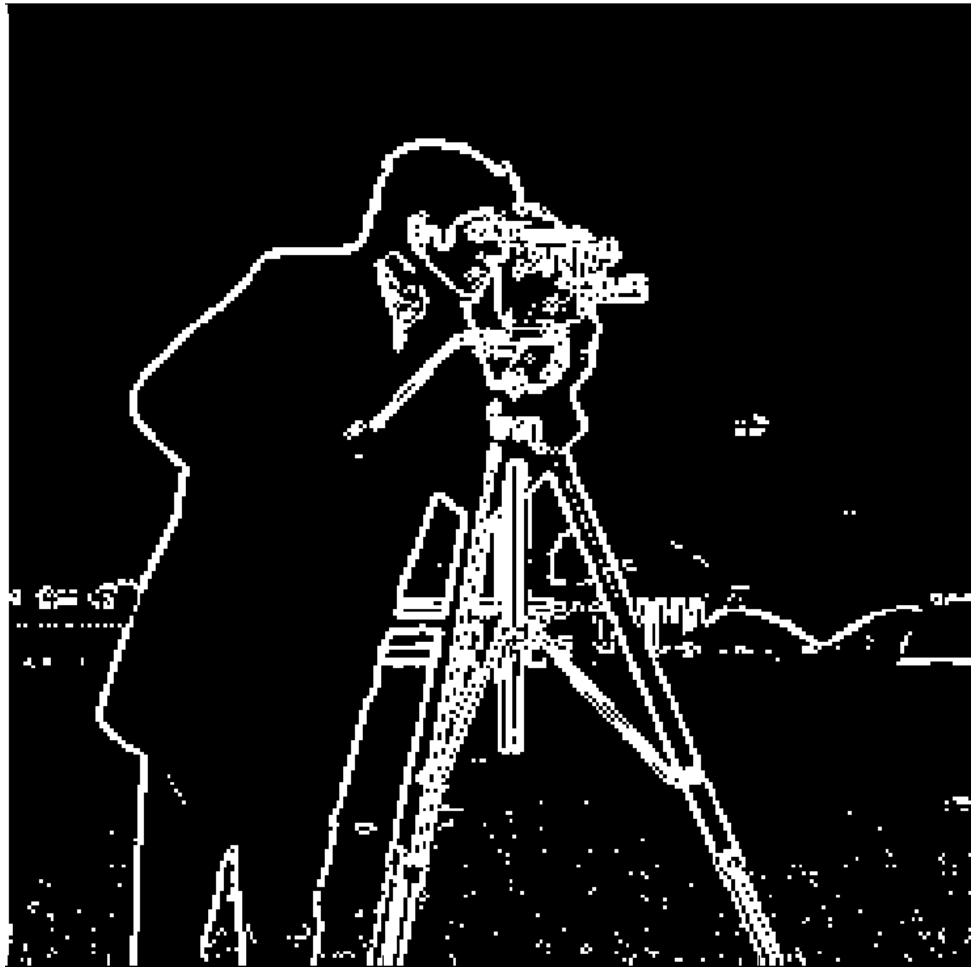
Image Gradients to Edges

|Gradient|

Cameraman

|Gradient| > 30

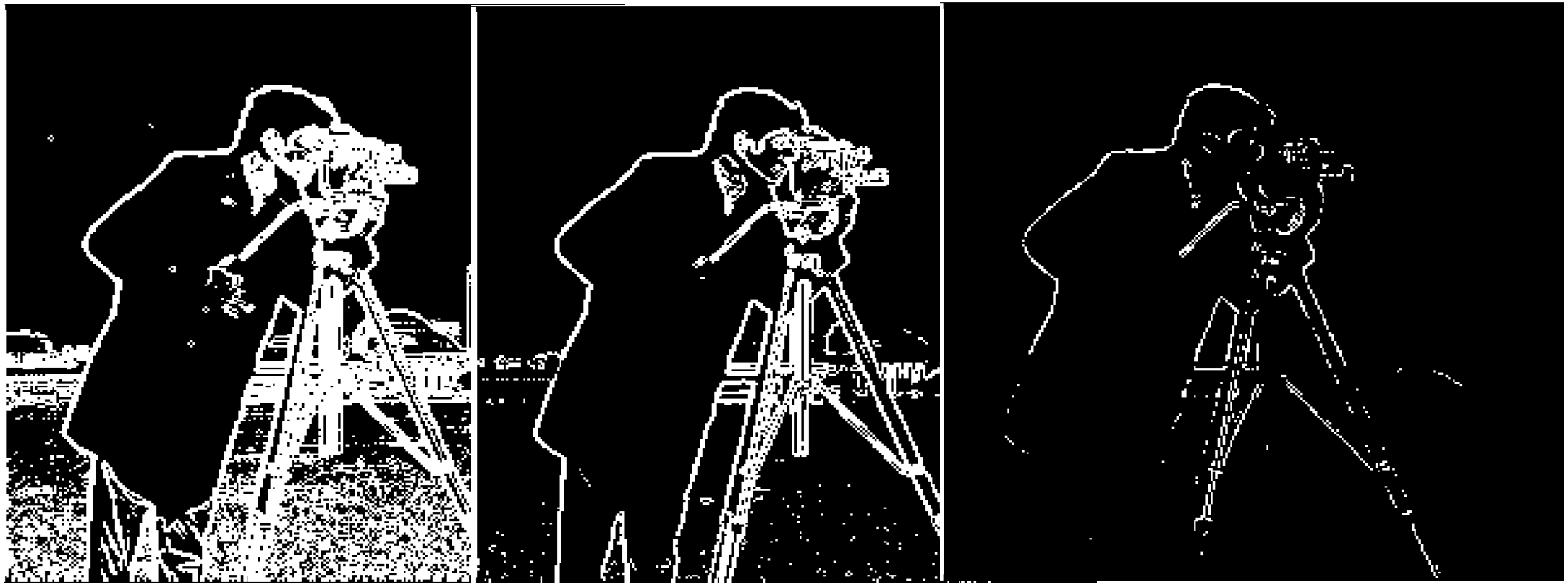
Binary Image



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Image Gradients to Edges

Problem 1: How to choose the threshold



>10 >30 >80

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Image Gradients to Edges

- We smooth to reduce the effect noise on image gradients
- Smoothed derivative removes noise but blurs edges

Image with Edge

Edge Location

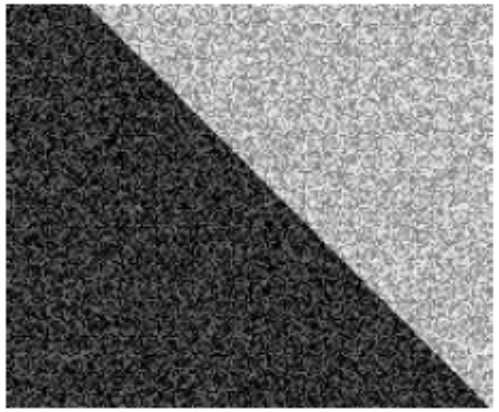
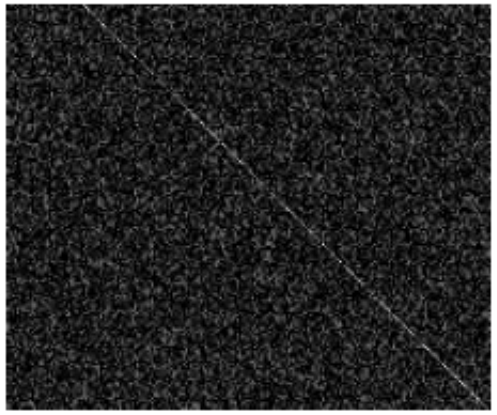


Image + Noise



Derivatives detect edge *and* noise

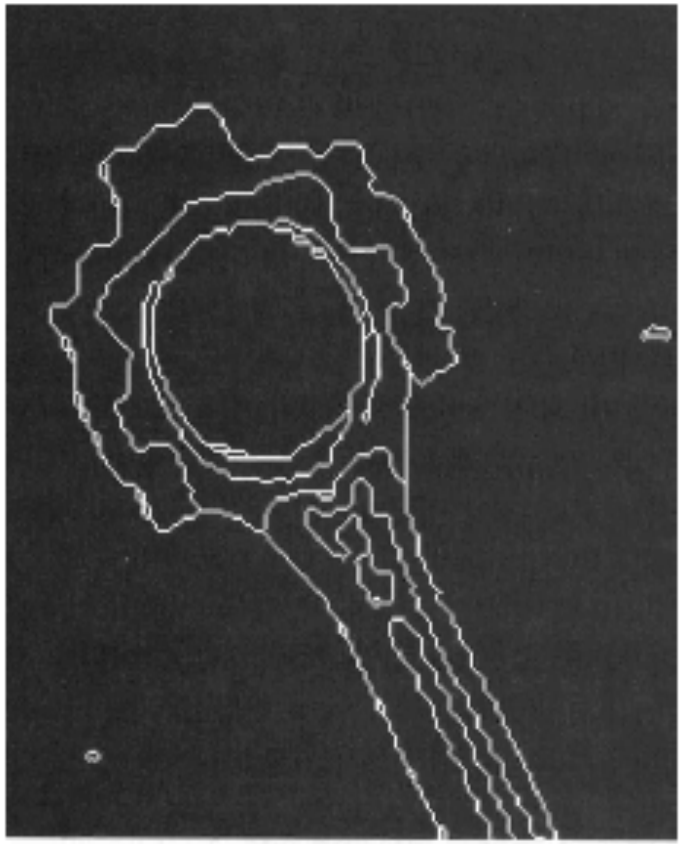
Smoothed derivative removes noise, but blurs edge

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Image Gradients to Edges

Problem 2: How to get thin localized edges

Smoothing+Thresholding gives a binary mask with “thick” edges



We want thin, one-pixel wide, connected contours

Marr Hildreth Edge Detector

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Second Derivative

$$\frac{\partial}{\partial x} f(i,j) \cong f(i,j) - f(i-1,j)$$

$$\frac{\partial}{\partial y} f(i,j) \cong f(i,j) - f(i,j-1)$$

$$[1 \quad -1]$$

$$\begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$$\frac{\partial^2}{\partial x^2} f(i,j) \cong \frac{\partial}{\partial x} f'(i+1,j) - \frac{\partial}{\partial x} f'(i,j)$$

$$= f(i-1,j) + f(i+1,j) - 2f(i,j)$$

$$[1 \quad -1] * [1 \quad -1]$$

$$= [1 \quad -2 \quad 1]$$

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Laplacian

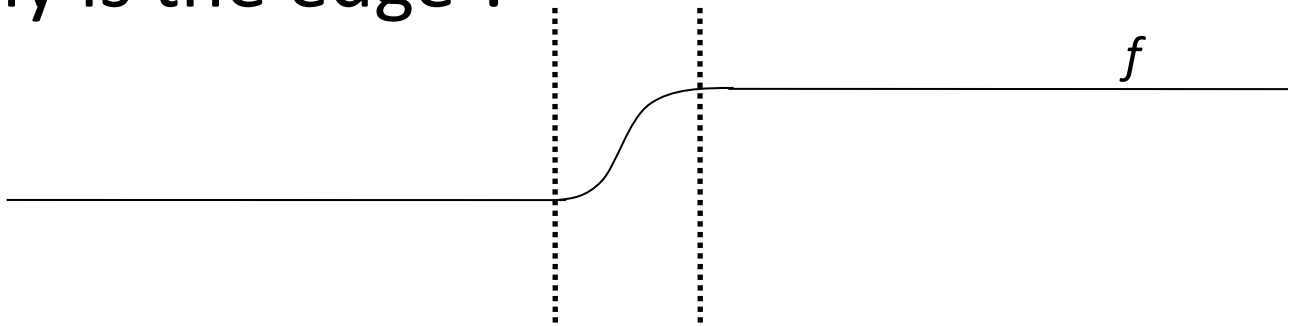
Equation: $$\nabla^2 f = \frac{\partial}{\partial x^2} f + \frac{\partial}{\partial y^2} f$$

Convolution: $$[1 \quad -2 \quad 1] + \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

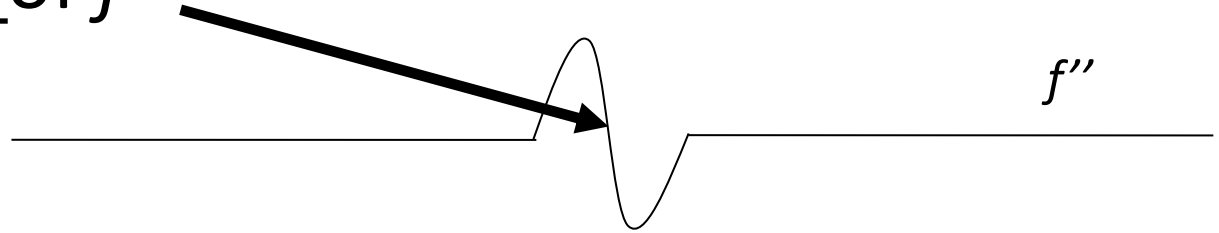
Edge Localization with Zero Crossing

Where exactly is the edge ?



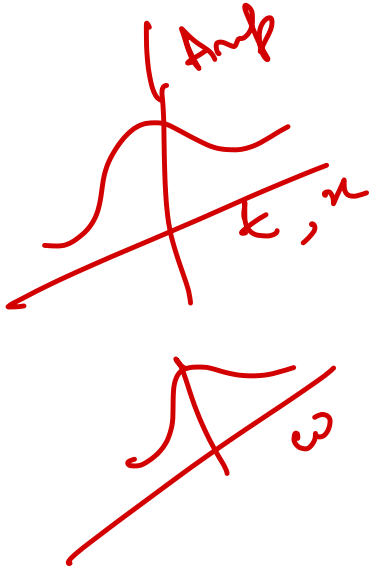
Maximum of *f'*

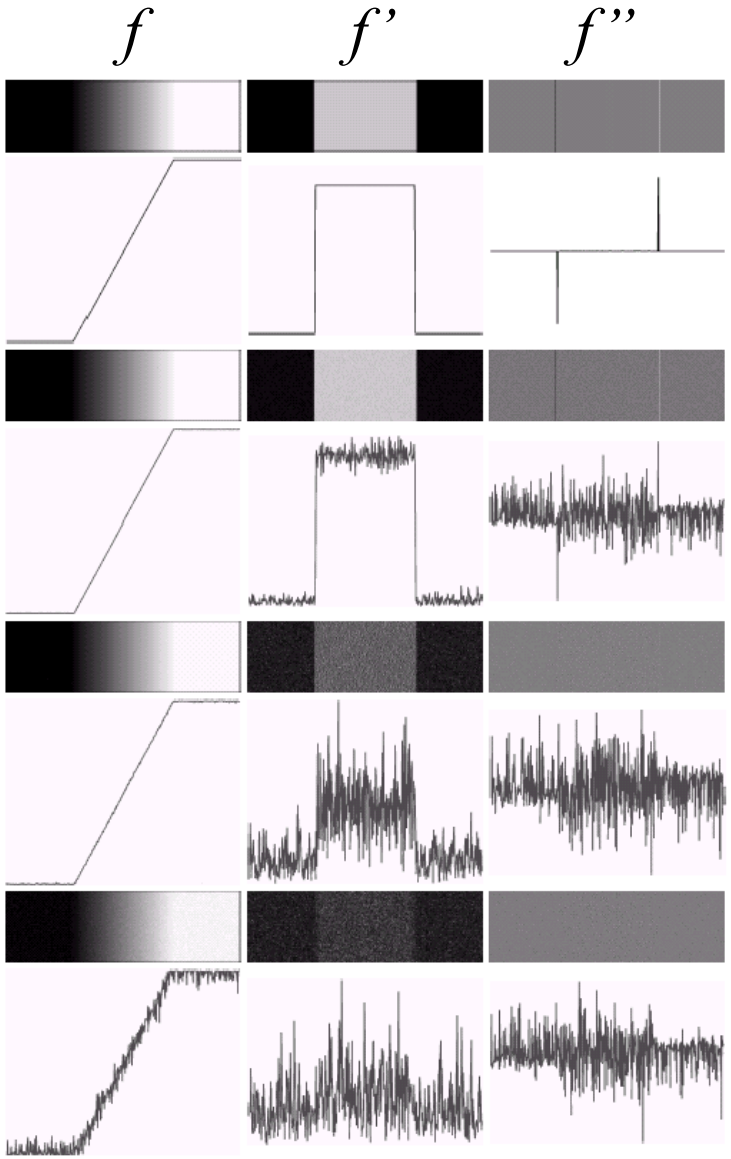
Zero crossing of *f''*



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

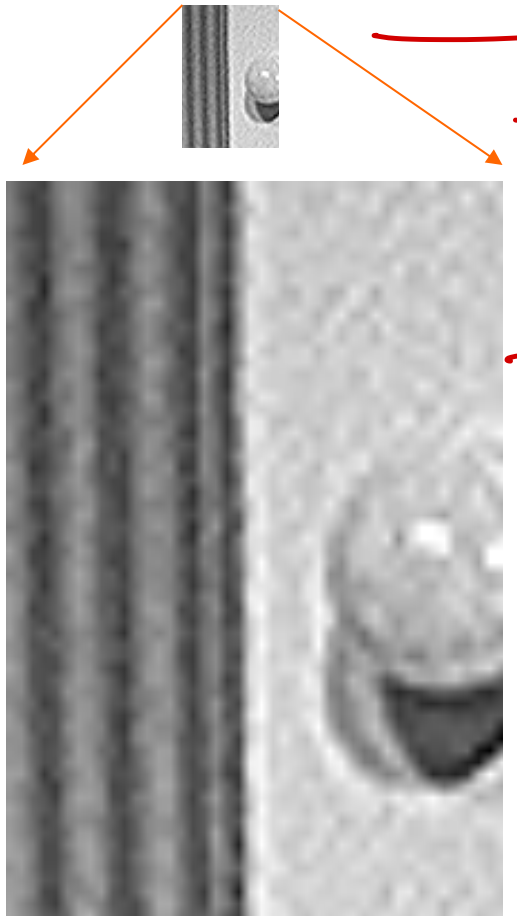
Effect of Noise on Derivatives





Gaussian filter
Low pass filter

Derivatives
High Pass filter



Noise
High frequency

$A \otimes B \equiv \left(A' B'\right)'$

A' = fourier transform

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Recall: Image Derivatives

- Good derivative filters are high-pass (edge) in one direction and low-pass (blur) in the orthogonal direction.

Sobel (X)

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

Sobel (Y)

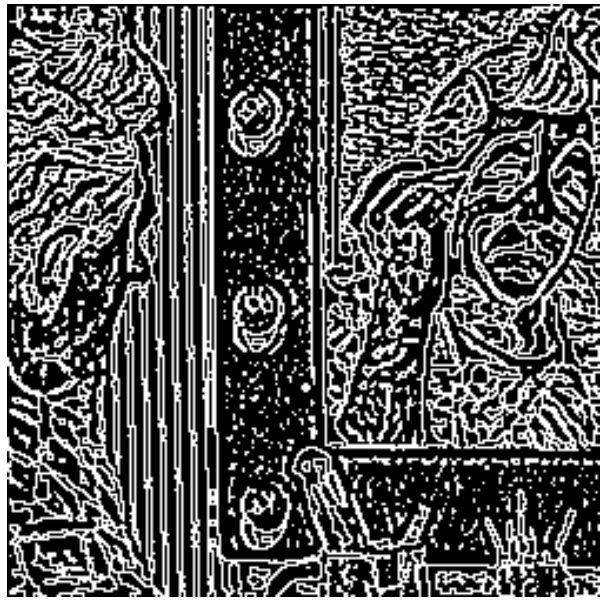
$$\frac{1}{8}\begin{bmatrix}1 & 2 & 1\\ 0 & 0 & 0\\ -1 & -2 & -1\end{bmatrix}$$

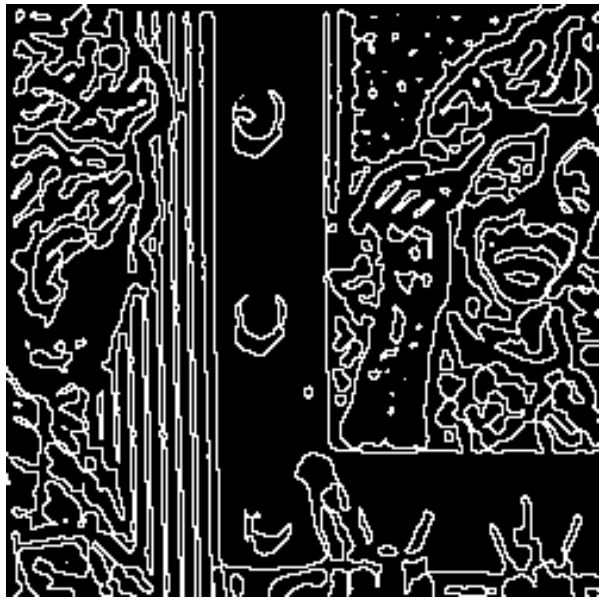
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

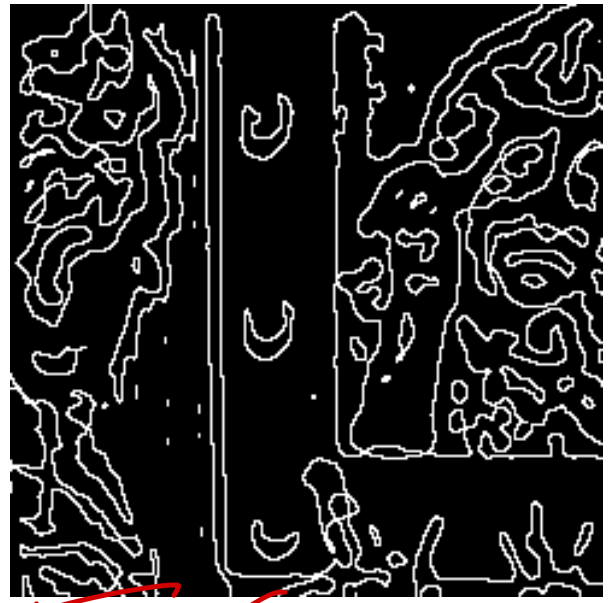
Zero Crossing After Smoothing

Input

Laplacian after increasingly strong smoothing







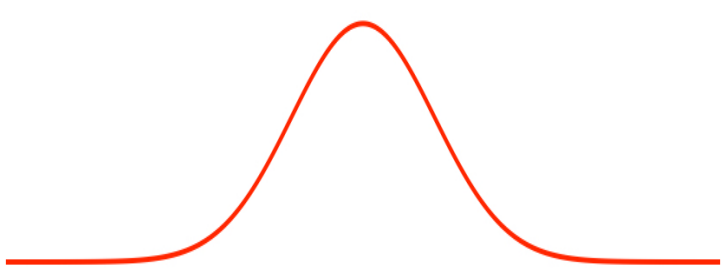
Canny Edge Detector

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

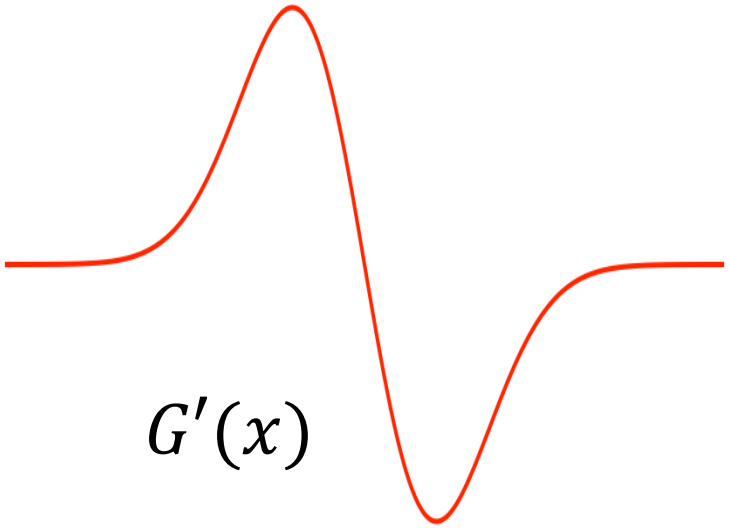
Canny Edge Detector

Step 1: Compute Edge Points

- Compute image derivatives G_x, G_y using derivative of Gaussian
  - Smooth with a Gaussian.
  - Use simple derivative kernels $[1 \quad -1]$.



$G(x)$



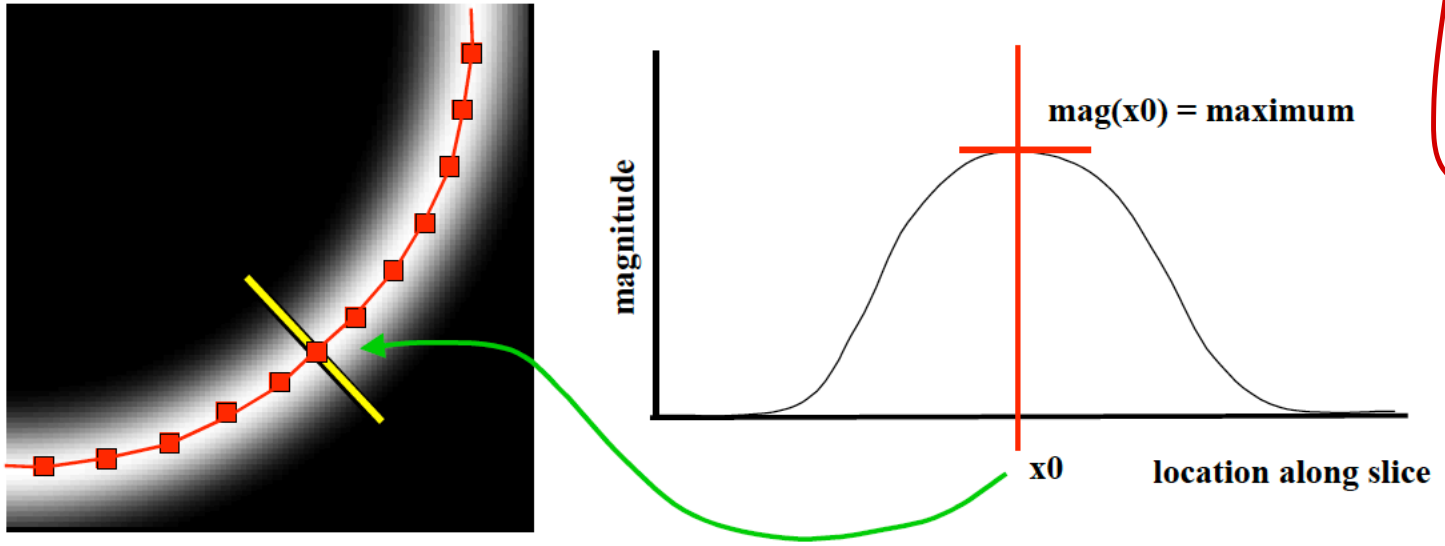
$G'(x)$

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Canny Edge Detector

Step 2: Thin Edges

- We want to mark points along curve where the magnitude is largest.
- We can do this by looking for a maximum along a 1D intensity slice normal to the curve (**non-maximum suppression**).
- These points should form a one-pixel wide curve.



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Canny Edge Detector

Step 2: Thin Edges

- Compute edge direction: $\tan(\alpha) = {}^{G_y}/_{G_x}$.
- Edge point is the local maxima in edge direction. Find using non maximal suppression

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Canny Edge Detector

Step 3: Threshold and find connected edges

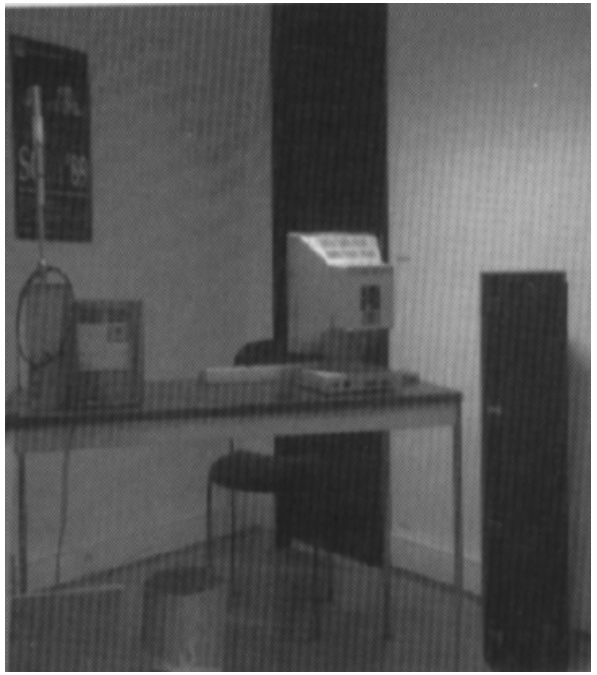
- How to pick threshold?
- How to connect edges?

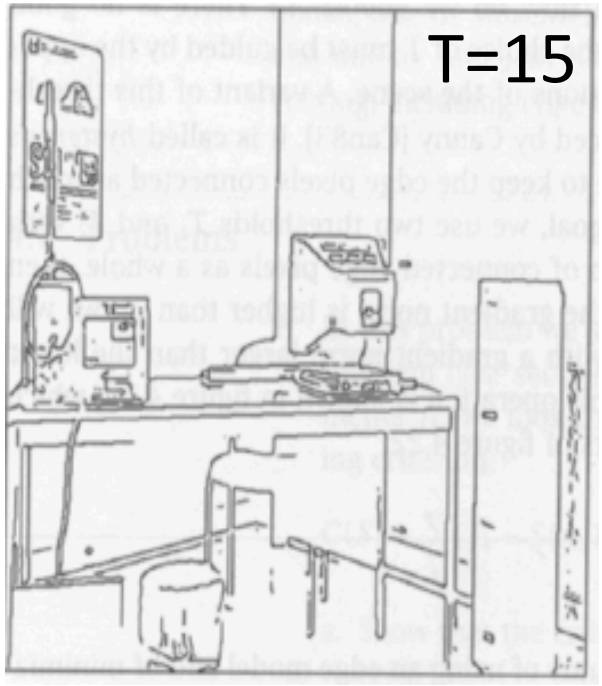
Hysteresis

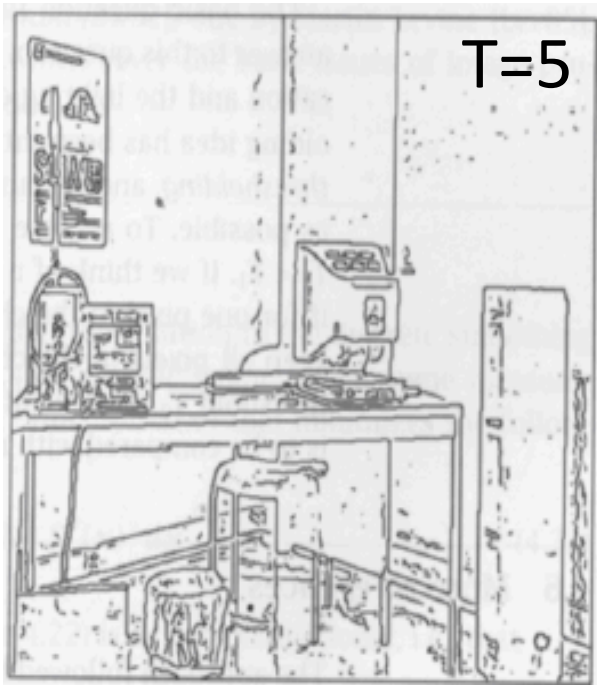
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

How to Choose the Gradient Threshold

- If the threshold is too high:
  - Very few (none) edges, High MISDETECTIONS, Many gaps
- If the threshold is too low:
  - Too many (all pixels) edges, High FALSE POSITIVES, many extra edges



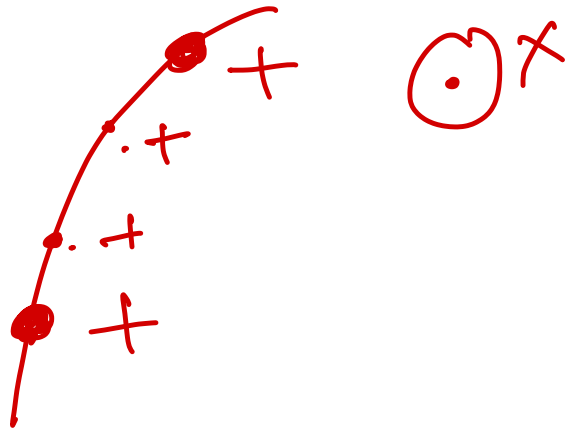




Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Hysteresis Thresholding and Edge Linking

- Keep both a high threshold H and a low threshold L.
  - Any edges with strength < L are discarded.
  - Any edge with strength > H are kept.



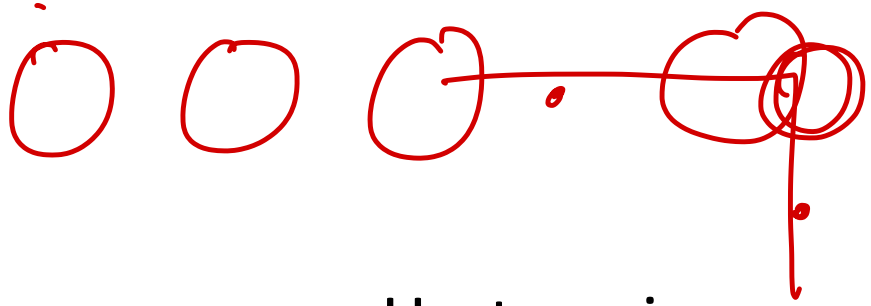
- An edge P with strength between L and H is kept only if there is a path of edges with strength > L connecting P to an edge of strength > H.

Edge Linking

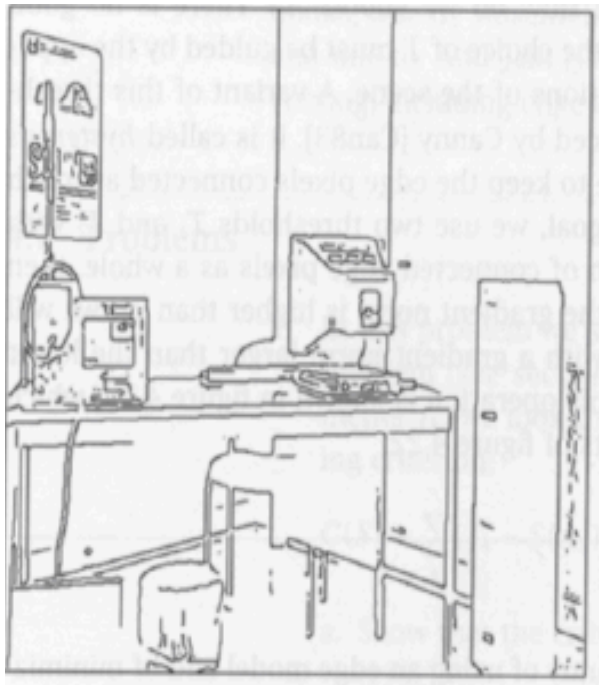
- In practice, hysteresis thresholding is combined with edge linking, by choosing the neighbors only along the edge direction. This links the edge pixels along the edge direction.

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

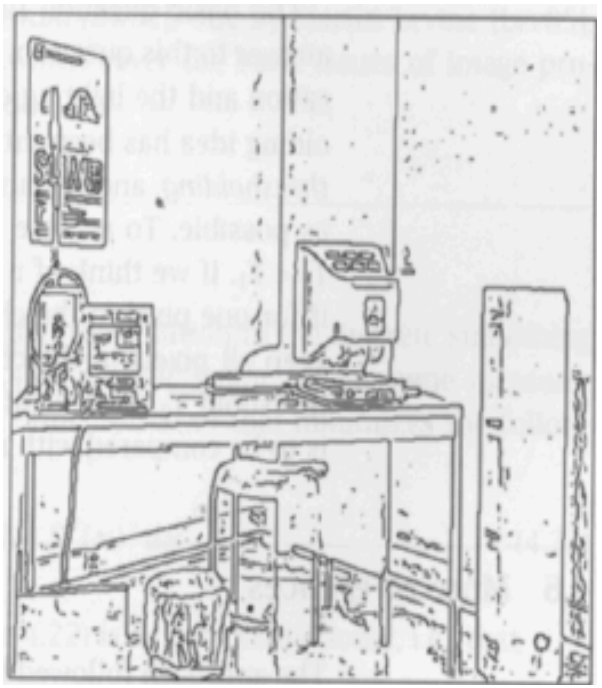
Hysteresis



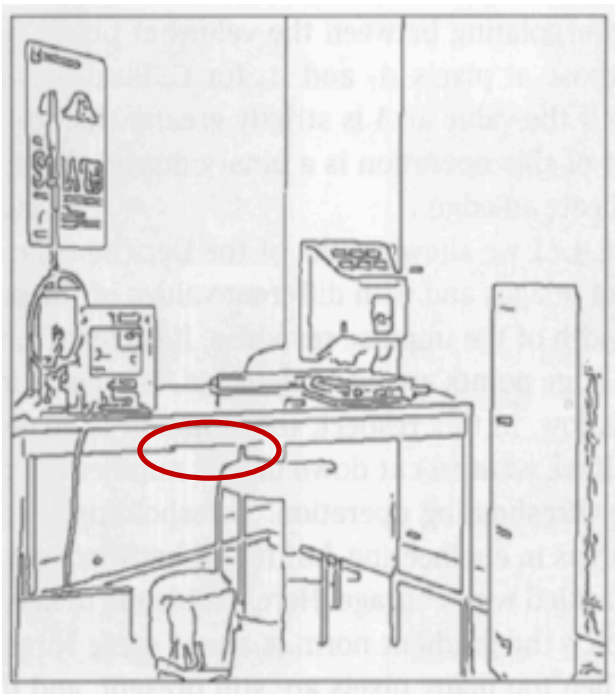
T=15



T=5



Hysteresis
H=15, L=5




Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Canny Edge Detector

- Compute image derivatives G_x, G_y using derivative of Gaussian
  - Smooth with a Gaussian.
  - Use simple derivative kernels $[1 \quad -1]$.

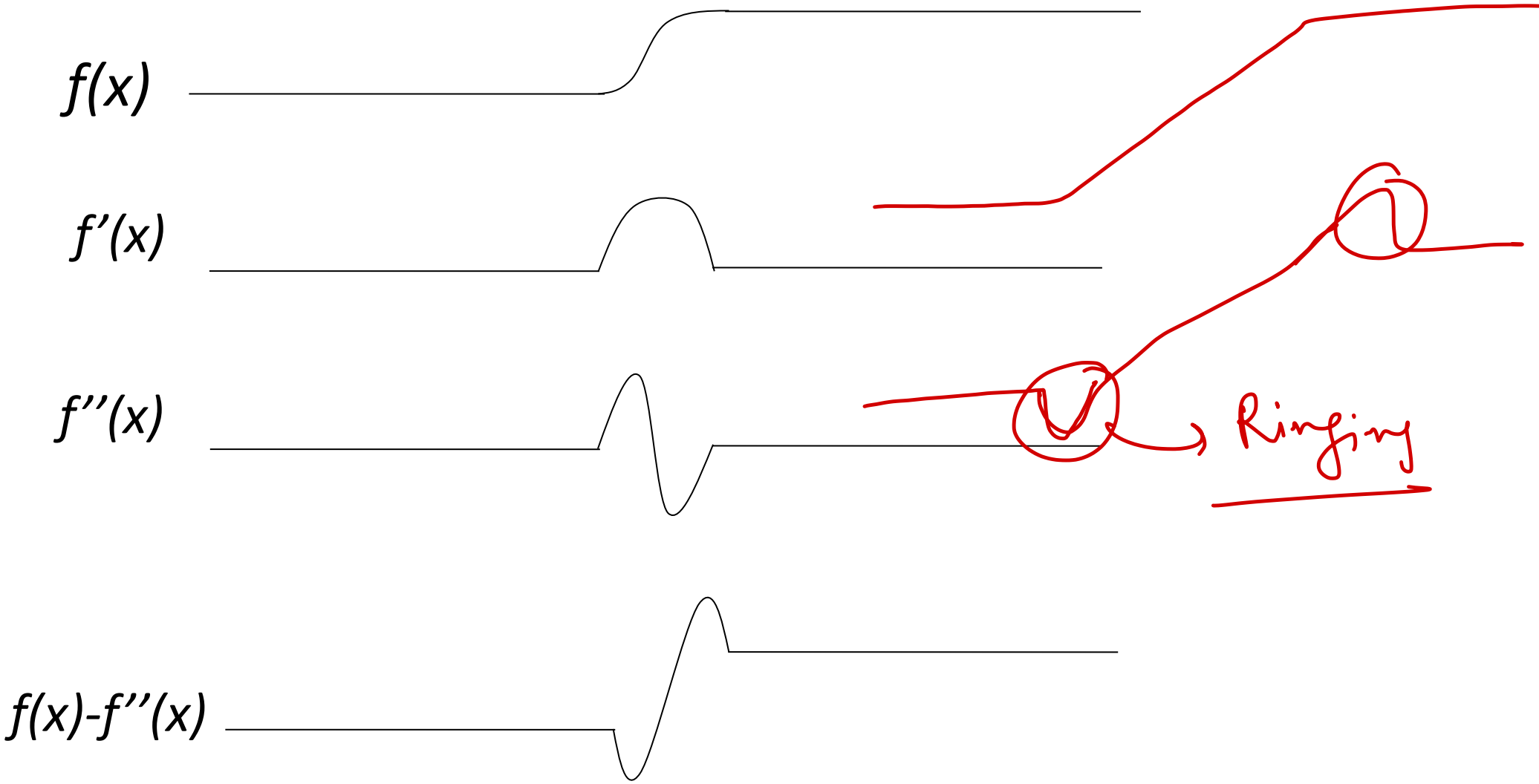
- Compute edge direction: $\tan(\alpha) = {}^{G_y}/_{G_x}$. Edge point is the local maxima in edge direction.

- Use only edge points with gradient above threshold. Use hysteresis based thresholding and edge linking with two thresholds. Low threshold accepted only if neighbor to high threshold

Image Sharpening

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Sharpening by Subtracting the Laplacian



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Sharpening by Subtracting the Laplacian

Equation: $$\nabla^2 f = \frac{\partial}{\partial x^2} f + \frac{\partial}{\partial y^2} f$$

Convolution: $$[1 \quad -2 \quad 1] + \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Subtracting the Laplacian from the image:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

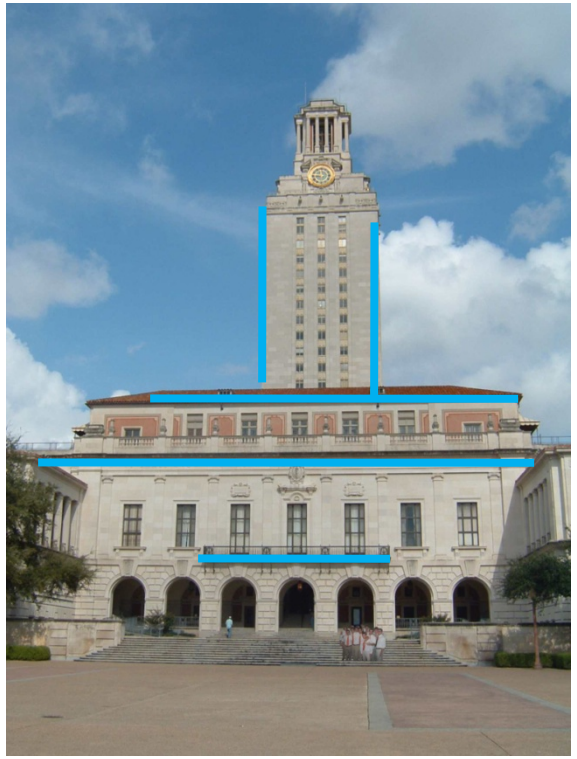
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

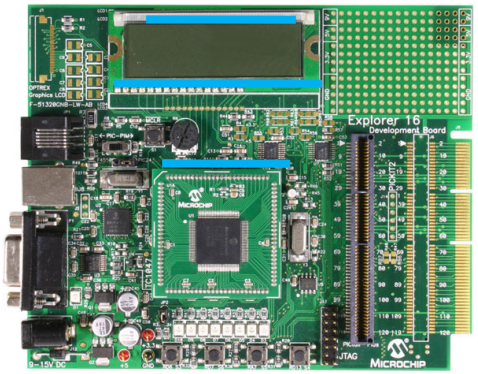
Sharpening Example

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Line Detection

- Why detect lines? Many objects characterized by presence of straight lines

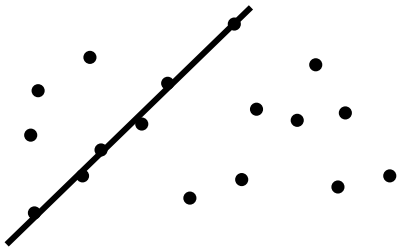




- Why aren’t we done just by running edge detection?

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

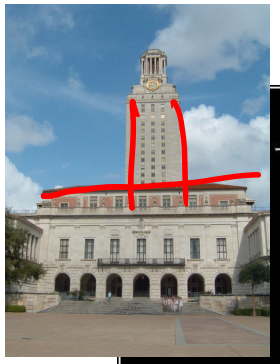
Difficulty of Line Detection

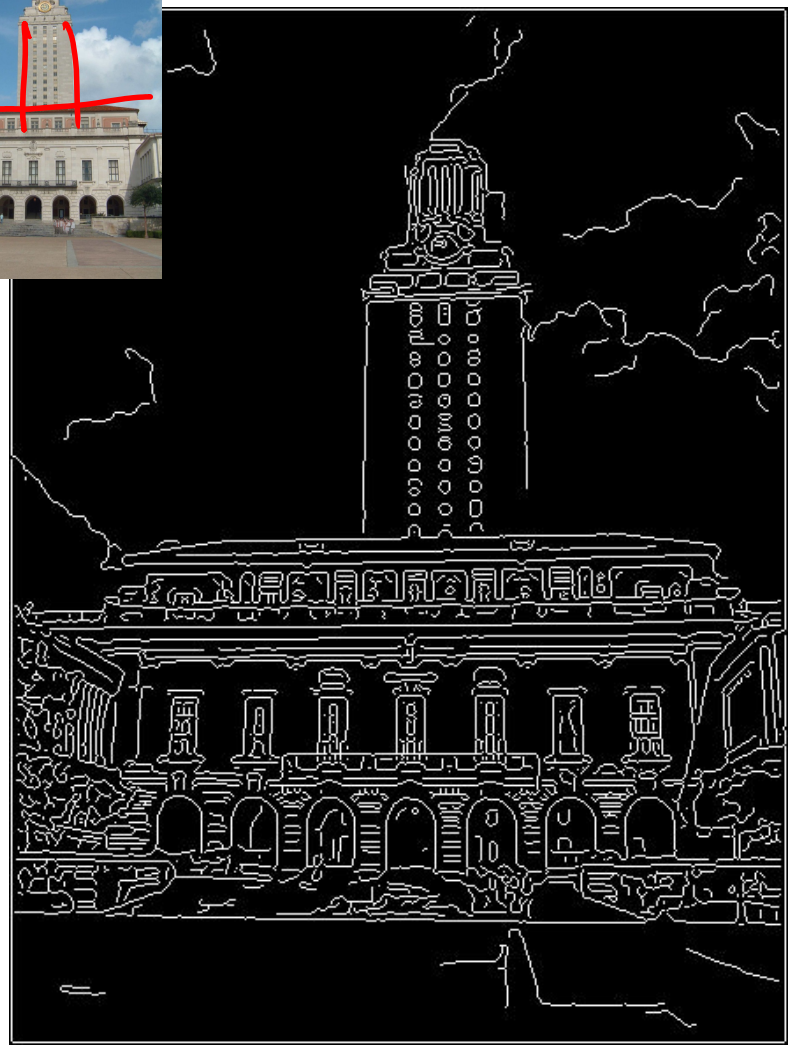


- Extra edge points (clutter), multiple models:
  - Which points go with which line, if any?

- Only some parts of each line detected, and some parts are missing:
  - How to find a line that bridges missing evidence?

- Noise in measured edge points, orientations:
  - How to detect true underlying parameters?

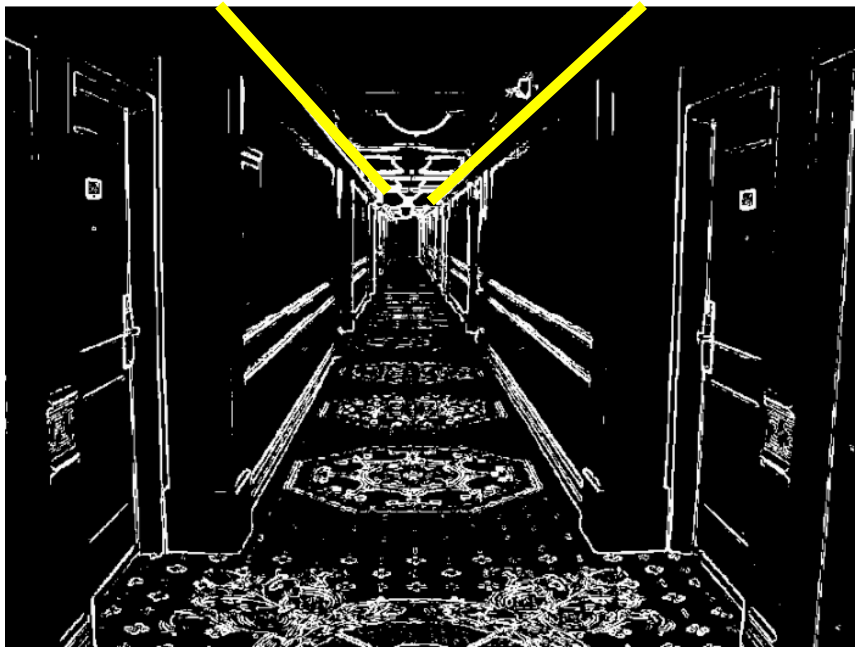




Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Edges to Lines: Problem Statement

- Assume that we have performed some edge detection, and a thresholding of the edge magnitude image.
- Thus, we have some pixels that may partially describe the boundary of some objects.
- We wish to find sets of pixels that make up straight lines.




Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Hough Transform

- Can be used to detect lines, circles etc.
- Introduced in 1962 (Hough 1962) and first used to find lines in images a decade later (Duda 1972).
- The goal is to find the location of lines in images. It can give robust detection under noise and partial occlusion
- **Caveat**: Hough transform can detect structures **ONLY** if their parametric equation is known.


Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Detecting Lines using Hough Transform

- Consider a point of known coordinates (x_i, y_i)
  - There are many lines passing through the point (x_i, y_i)

- Straight lines that pass that point have the form $y_i = \boldsymbol{a}x_i + \boldsymbol{b}$
  - Common to them is that they satisfy the above equation for some set of parameters $(\boldsymbol{a}, \boldsymbol{b})$

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

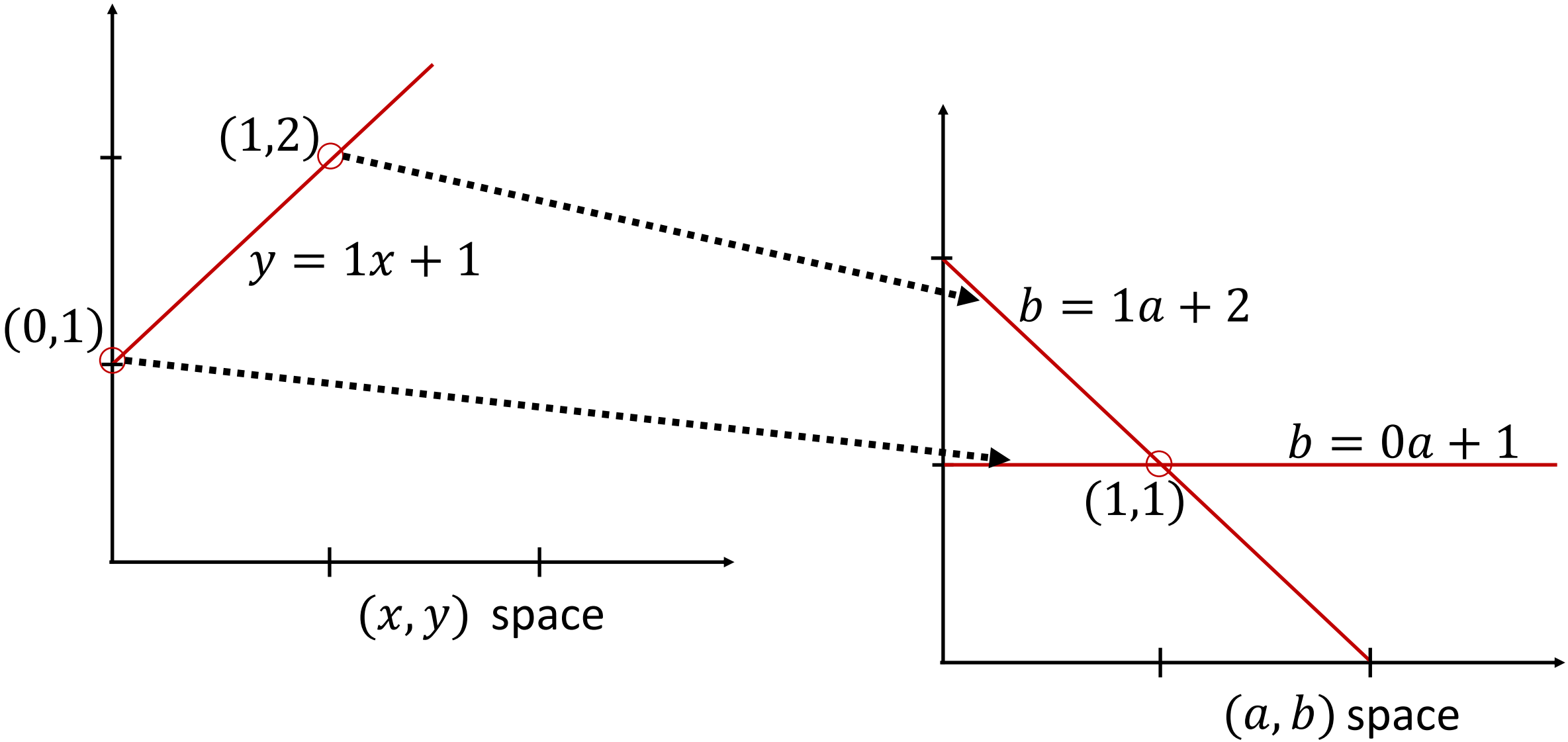
Detecting Lines using Hough Transform

- This equation can obviously be rewritten as follows:
  - $\boldsymbol{b} = -\boldsymbol{a}x_i + y_i$
  - We can now consider x and y as parameters
  - $\boldsymbol{a}$ and $\boldsymbol{b}$ as variables.

- This is a line in $(\boldsymbol{a}, \boldsymbol{b})$ space parameterized by x and y.
  - A single point (x_i, y_i) gives a line in (a, b) space.
  - Another point (x_j, y_j) will give rise to another line in (a, b) space.

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Detecting Lines using Hough Transform




Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Detecting Lines using Hough Transform

- Two points (x_i, y_i) and (x_j, y_j) define a line in the (x, y) plane with parameters (a_k, b_k).
- These two points give rise to two different lines in (a, b) space.
- In (a, b) space these lines will intersect at a point (a_k, b_k)
- All points on the line (a_k, b_k) in (x, y) space will parameterize lines that intersect at point (a_k, b_k) in (a, b) space.

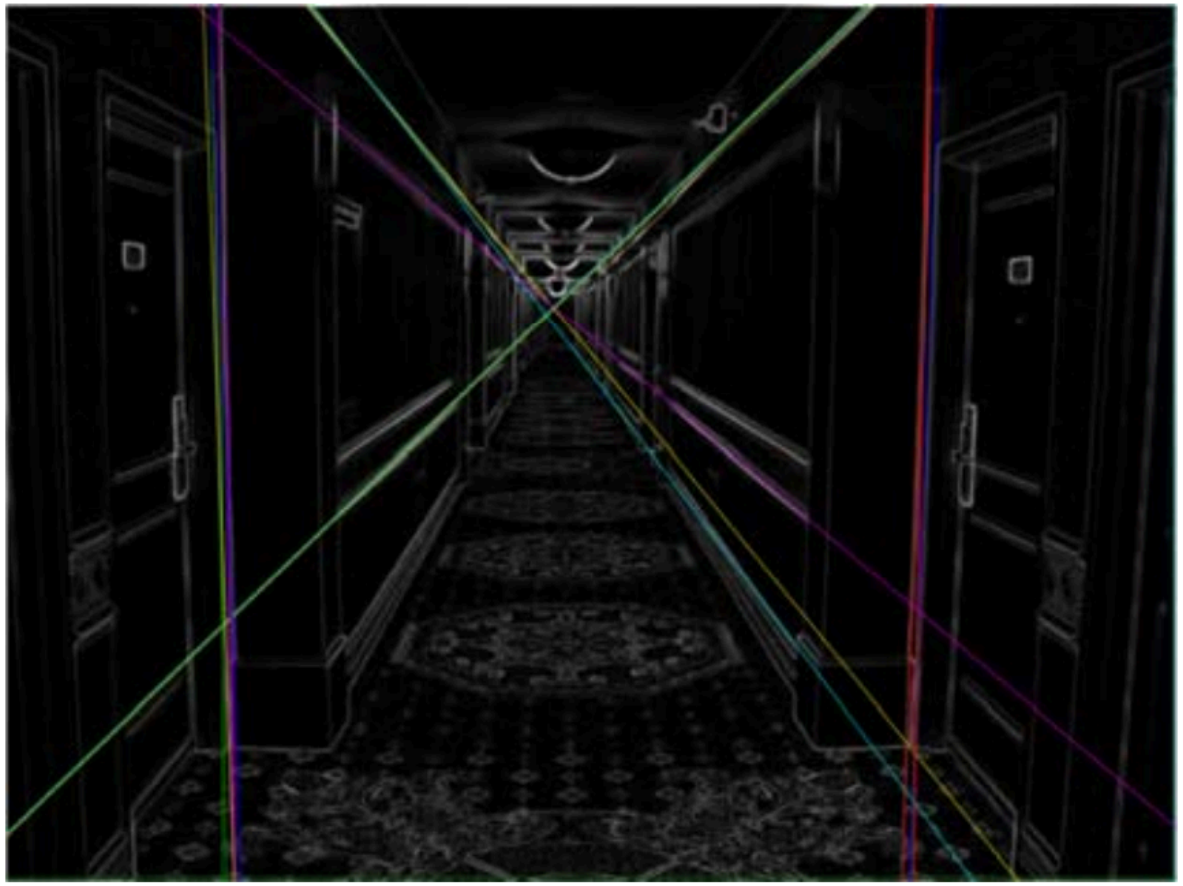

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Detecting Lines using Hough Transform

- Quantize the parameter space (a, b) by dividing it into **accumulator** cells
- For each pair of points (x_i, y_i) and (x_j, y_j) detected as an edge:
  - Find the intersection (a_k, b_k) in (a, b) space.
  - Increase the value of cell $[[a_{\min}, a_{\max}], [b_{\min}, b_{\max}]]$ containing (a_k, b_k)
- Cells receiving more than a certain number of counts (also called **votes**) are assumed to correspond to lines in (x, y) space.

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Hough Transform: Example



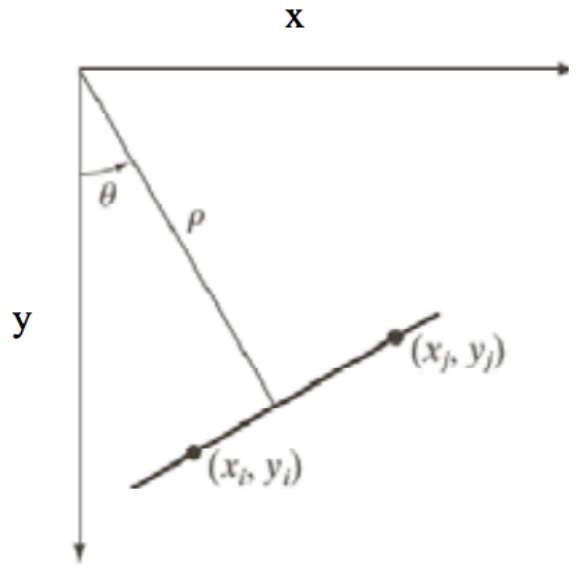
Top 20 most voted lines

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Other Hough Transformations

- We can represent lines as polar coordinates instead of $y = ax + b$

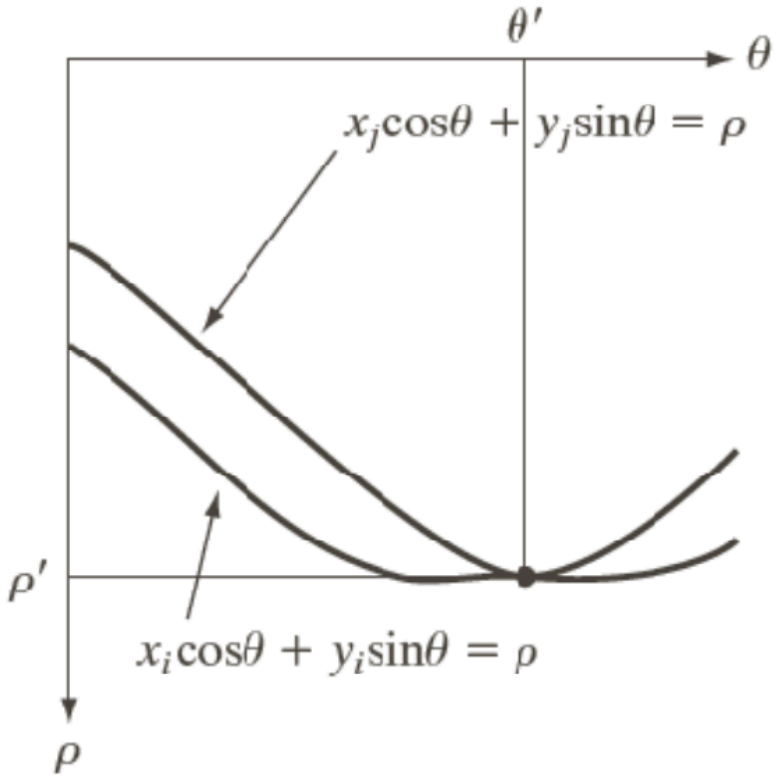
$$x * \cos(\theta) + y * \sin(\theta) = \rho$$



- Lines in (x, y) space are not lines in (ρ, θ) space

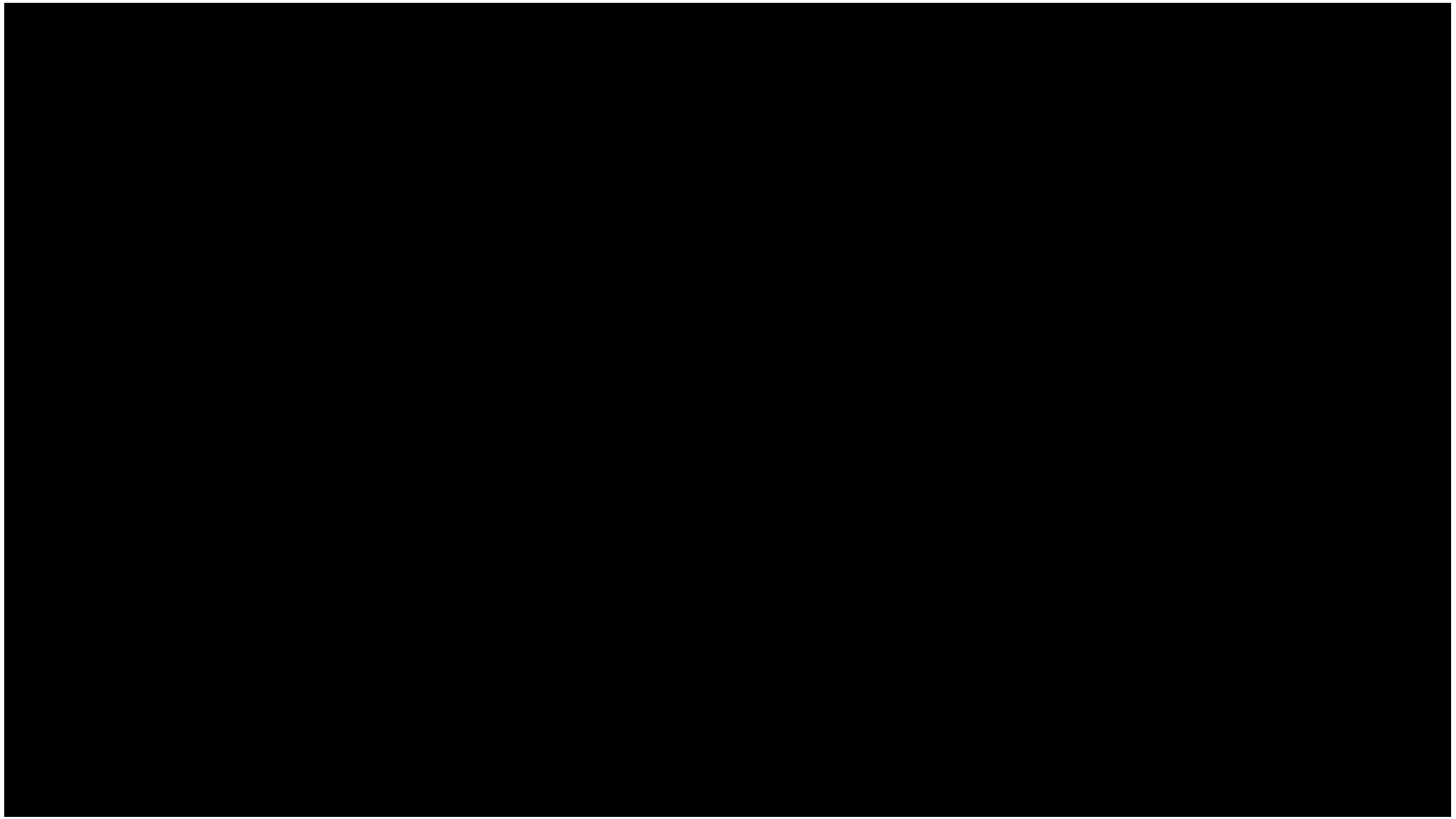
- Horizontal line: $\theta = 0$, $\rho =$ intercept with the y-axis.

- Vertical line: $\theta = 90$, $\rho =$ intercept with the x-axis.



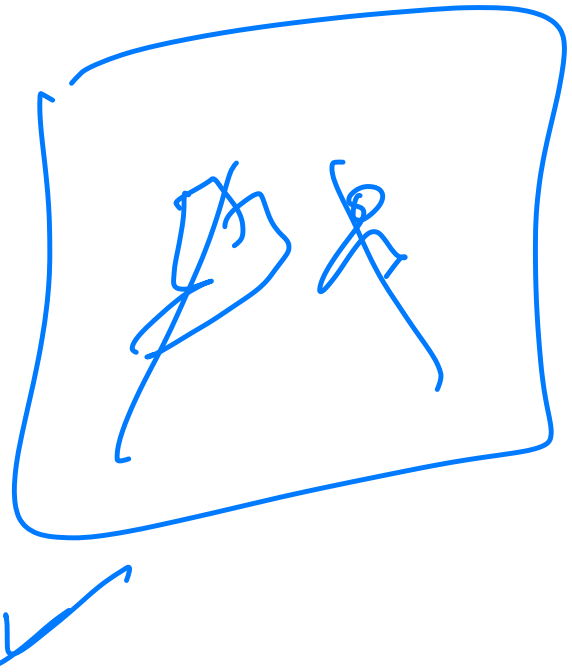

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Other Hough Transformations



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Hough Transform: Advantages



- Conceptually simple.
- Easy implementation
- Handles missing and occluded data very gracefully.
- Can be adapted to many types of forms, not just lines

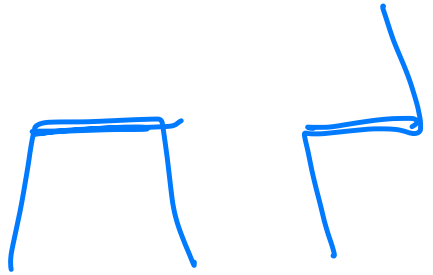

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Hough Transform: Disadvantages

- Computationally complex for objects with many parameters.
- Accuracy of estimation (accumulator cell size) needs to be traded off with complexity of estimation.
- Looks for only one single type of object

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Hough Transform: Disadvantages



- Can be “fooled” by “apparent lines”.

- The length and the position of a line segment cannot be determined.

- Co-linear line segments cannot be separated.