Digital Image Processing

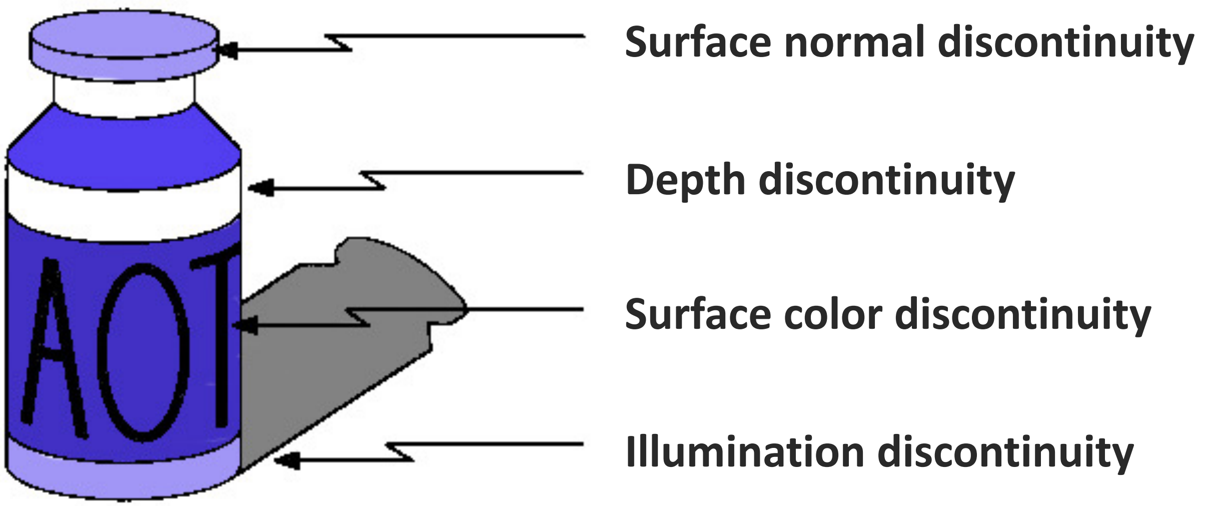
Edge Crispening

Ming-Sui (Amy) Lee

Lecture 03

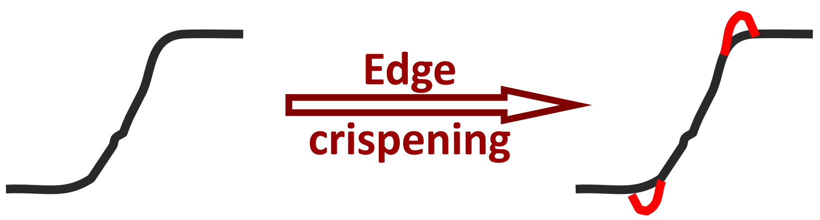
Edges

- Edges are caused by a variety of factors



Edge Crispening

- **Motivation**
  - A photograph with accentuated edges look more appealing



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

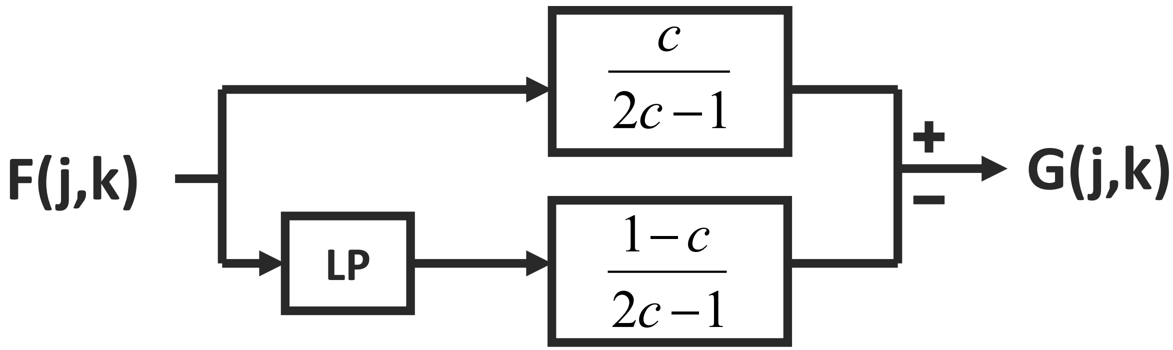
$$H = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

  - Edge → high frequency
  - High pass filtering

→ amplify the noise at the same time

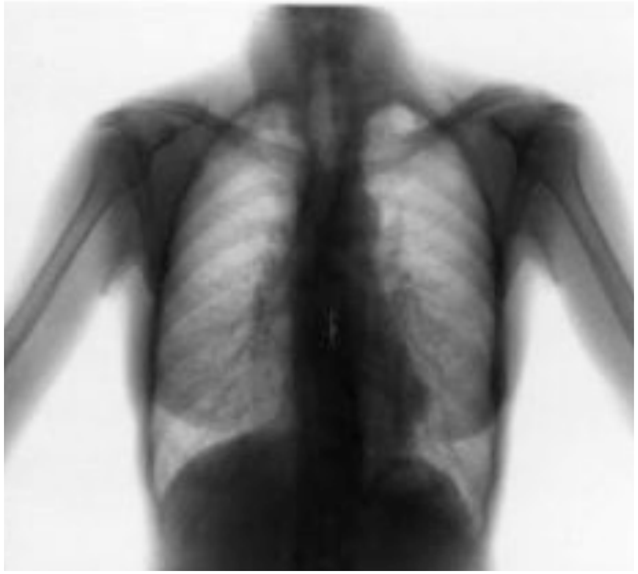
Edge Crispening

- Unsharp Masking
  - Appropriate combination of all-pass and low-pass filters

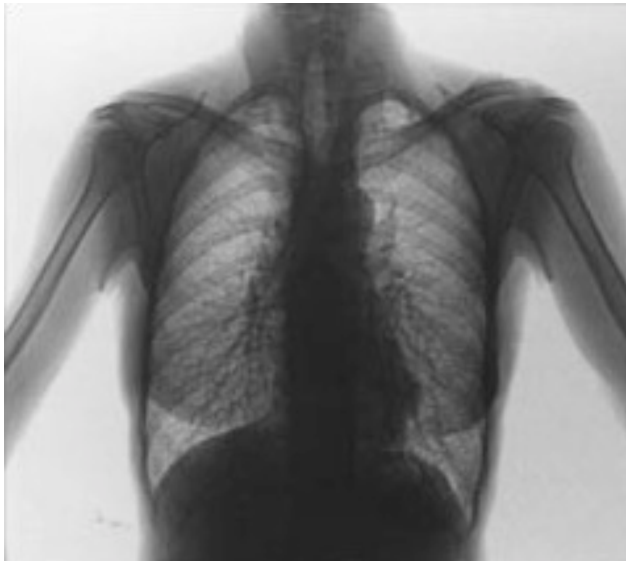


$$G(j,k)=\frac{c}{2c-1}F(j,k)-\frac{1-c}{2c-1}F_L(j,k), \quad where \quad \frac{3}{5}\le c\le\frac{5}{6}$$

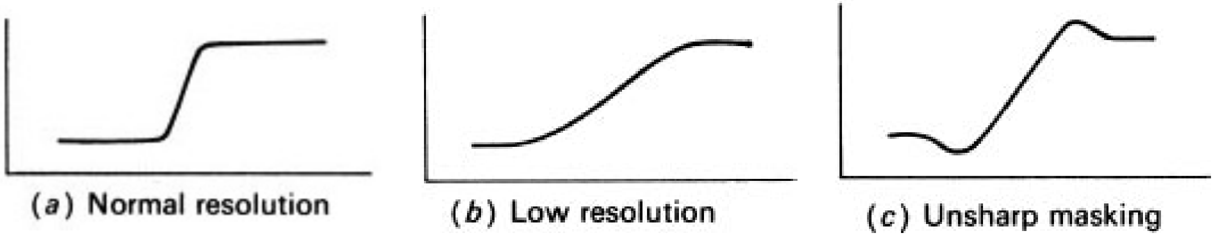
Edge Crispening



Original image



After sharpening
L=7, c=0.6



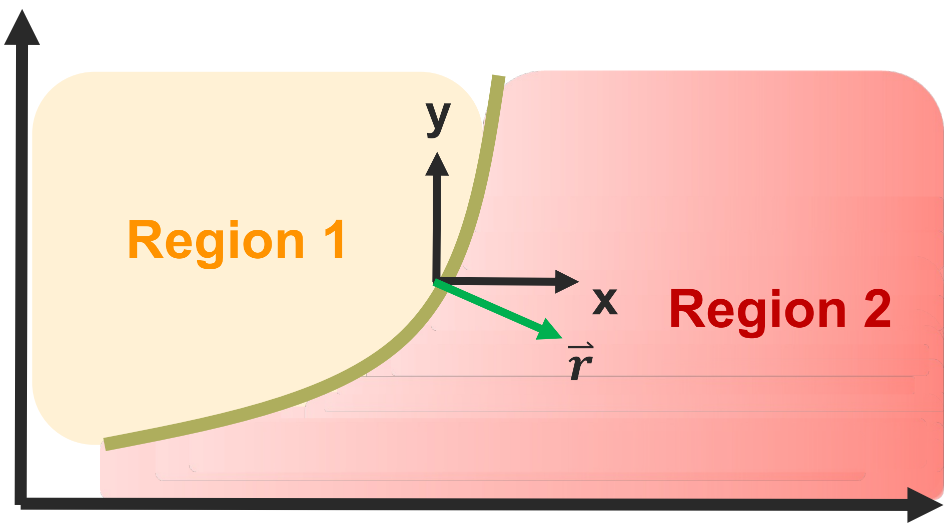
Edge Detection

Edge Detection

- **Motivation**
  - Human eyes are more sensitive to edges
  - Characterize object boundaries
  - Fundamental step in image analysis
    - Segmentation, registration, identification, etc.
- **Edge description**
  - Model-based methods
    - Rarely used
  - Non-parametric approaches
    - 1st and 2nd order derivatives

Edge Detection (1st order)

- **Orthogonal gradient generation**



$$\frac{dF}{dr} = \frac{\partial F}{\partial x}\frac{\partial x}{\partial r} + \frac{\partial F}{\partial y}\frac{\partial y}{\partial r} = \begin{pmatrix} \frac{\partial F}{\partial x} & \frac{\partial F}{\partial y} \end{pmatrix} \cdot \begin{pmatrix} \frac{\partial x}{\partial r} \\ \frac{\partial y}{\partial r} \end{pmatrix} = \begin{pmatrix} \frac{\partial F}{\partial x} & \frac{\partial F}{\partial y} \end{pmatrix} \cdot \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}$$

Edge Detection (1^{st} order)

- **Orthogonal gradient generation**
  - **When** $\begin{pmatrix} \dfrac{\partial F}{\partial x} \\ \dfrac{\partial F}{\partial y} \end{pmatrix}$ **(gradient direction) is parallel to** $\begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}$
    $\left\| \dfrac{dF}{dr} \right\|$ **has maximum value**

$$\Rightarrow \left\| \frac{dF}{dr} \right\| = \left\| \begin{pmatrix} \dfrac{\partial F}{\partial x} \\ \dfrac{\partial F}{\partial y} \end{pmatrix} \right\| = \sqrt{\left(\frac{\partial F}{\partial x} \right)^2 + \left(\frac{\partial F}{\partial y} \right)^2} \qquad \theta = \tan^{-1} \begin{pmatrix} \dfrac{\partial F}{\partial y} \\ \dfrac{\partial F}{\partial x} \end{pmatrix}$$

Magnitude **Orientation**

Edge Detection (1st order)

- Discrete case
  - Approximation I – 3 points
    - Row gradient

$$\frac{\partial F}{\partial x}(j,k) \cong F(j,k) - F(j,k-1) = G_R(j,k)$$

    - Column gradient

$$\frac{\partial F}{\partial y}(j,k) \cong F(j,k) - F(j+1,k) = G_C(j,k)$$

A_0	A_1	A_2
A_7	$F(j,k)$	A_3
A_6	A_5	A_4

$$\Rightarrow \quad G(j,k) = \sqrt{G_R^2(j,k) + G_C^2(j,k)} \qquad \theta(j,k) = \tan^{-1}\left(\frac{G_C(j,k)}{G_R(j,k)}\right)$$

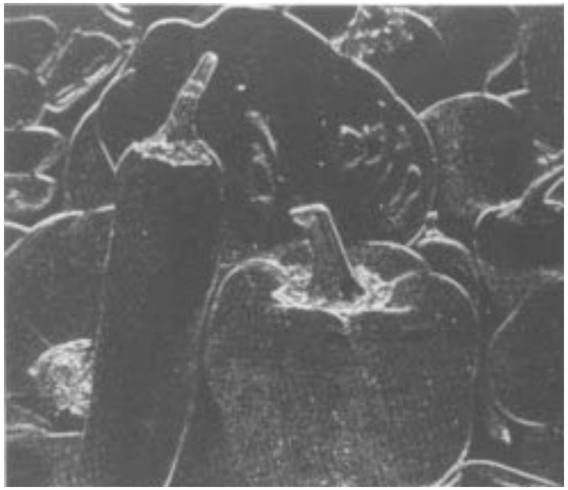
Edge Detection (1st order)

- **Example**

$$G_R(j,k) = F(j,k) - F(j,k-1)$$
$$G_C(j,k) = F(j,k) - F(j+1,k)$$

Original image

Horizontal magnitude



Vertical magnitude

Edge Detection (1st order)

A_0	A_1	A_2
A_7	$F(j,k)$	A_3
A_6	A_5	A_4

- Discrete case
  - Approximation II – 4 points
  - Roberts cross differentiation (0~90→45~135)
    - Row gradient

$$G_1(j,k) = F(j,k) - F(j+1,k+1)$$

    - Column gradient

$$G_2(j,k) = F(j,k+1) - F(j+1,k)$$

$$\Rightarrow G(j,k) = \sqrt{G_1^2(j,k) + G_2^2(j,k)} \qquad \theta(j,k) = \tan^{-1}\left(\frac{G_2(j,k)}{G_1(j,k)}\right) + \frac{\pi}{4}$$

Edge Detection (1st order)

- Discrete case
  - Approximation III – 9points

A_0	A_1	A_2
A_7	$F(j,k)$	A_3
A_6	A_5	A_4

    - Row gradient

$$G_R(j,k) = \frac{1}{K+2}\left[(A_2 + KA_3 + A_4) - (A_0 + KA_7 + A_6)\right]$$

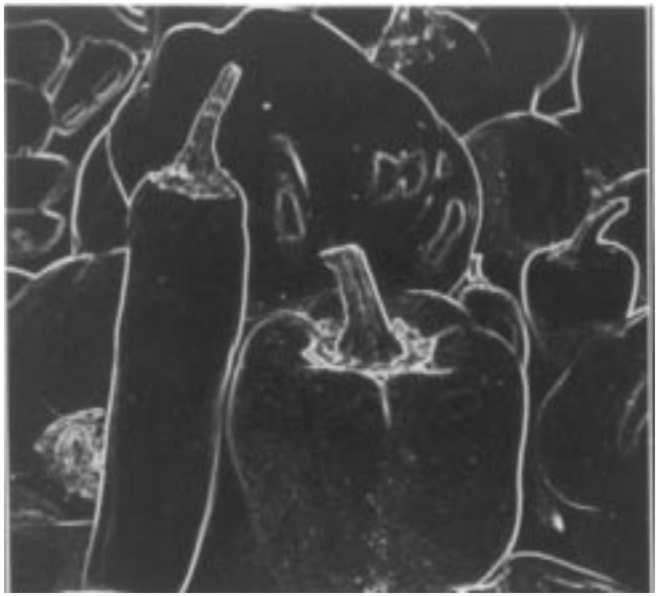
    - Column gradient

$$G_C(j,k) = \frac{1}{K+2}\left[(A_0 + KA_1 + A_2) - (A_6 + KA_5 + A_4)\right]$$

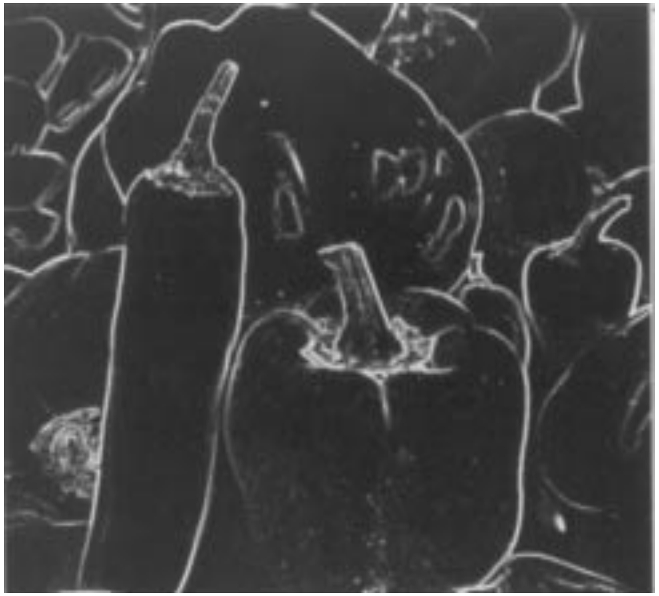
$$\Rightarrow \quad G(j,k) = \sqrt{G_R^2(j,k) + G_C^2(j,k)} \qquad \theta(j,k) = \tan^{-1}\left(\frac{G_C(j,k)}{G_R(j,k)}\right)$$

- K=1: Prewitt Mask ; K=2: Sobel Mask

Edge Detection (1st order)



Roberts magnitude



Roberts square root

Edge Detection (1st order)

- Compute row and column gradients

- Analyze the statistics of the magnitude (histogram)

- Pick a threshold T
- If G(j,k)>=T
  → set it as an edge point
  otherwise (If G(j,k)<T)
  → non-edge point
- Q: How to select T?

- Examine the cumulative distribution function

Canny Edge Detector

- **Why Canny?**
  - **Good Detection**
    - The optimal detector must minimize the probability of false positives as well as false negatives
  - **Good Localization**
    - The edges detected must be as close as possible to the true edges
  - **Single Response Constraint**
    - The detector must return one point only for each edge point

Canny Edge Detector

- Five Steps:
  - Noise reduction
  - Compute gradient magnitude and orientation
  - Non-maximal suppression
  - Hysteretic thresholding
  - Connected component labeling method

Smooth the image with a Gaussian filter

Example:
5x5 Gaussian filter with σ = 1.4

$$F_{NR} = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} * F$$

Canny Edge Detector

- **Five Steps:**
  - Noise reduction
  - Compute gradient magnitude and orientation
  - Non-maximal suppression
  - Hysteretic thresholding
  - Connected component labeling method

$$G(j,k) = \sqrt{G_R^2(j,k) + G_C^2(j,k)}$$

$$\theta(j,k) = \tan^{-1}\left(\frac{G_C(j,k)}{G_R(j,k)}\right)$$

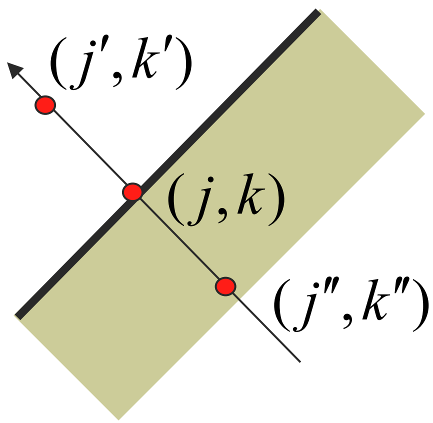
Magnitude

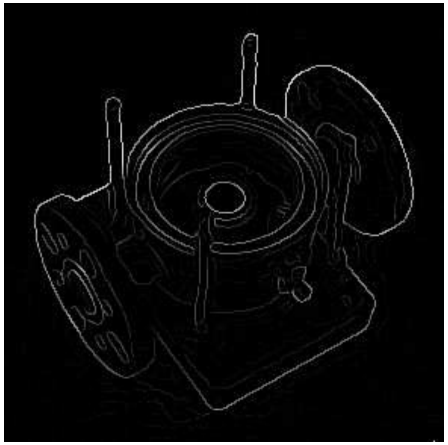
Canny Edge Detector

- **Five Steps:**
  - Noise reduction
  - Compute gradient magnitude and orientation
  - Non-maximal suppression
  - Hysteretic thresholding
  - Connected component labeling method

Search the nearest neighbors (j',k') and (j'',k'') along the edge normal

$$G_N(j,k)=\begin{cases} G(j,k) & if\ G(j,k)>G(j',k') \\ & and\ G(j,k)>G(j'',k'') \\ 0 & \end{cases}$$



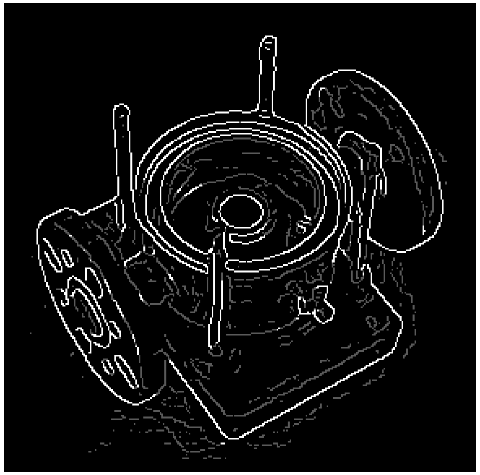


Canny Edge Detector

- **Five Steps:**
  - Noise reduction
  - Compute gradient magnitude and orientation
  - Non-maximal suppression
  - Hysteretic thresholding
  - Connected component labeling method

Label each pixels according to two threshold: T_H, T_L

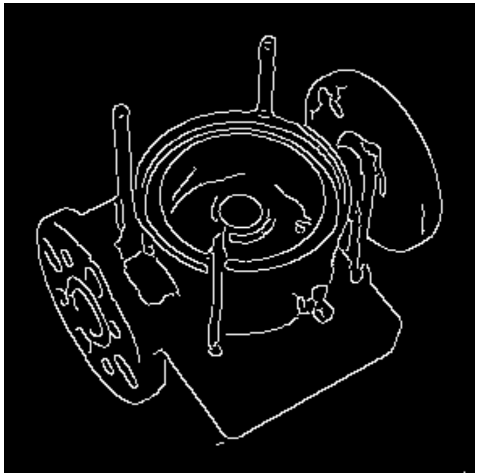
$G_N(x,y) \geq T_H$	Edge Pixel
$T_H > G_N(x,y) \geq T_L$	Candidate Pixel
$G_N(x,y) < T_L$	Non-edge Pixel



Canny Edge Detector

- Five Steps:
  - Noise reduction
  - Compute gradient magnitude and orientation
  - Non-maximal suppression
  - Hysteretic thresholding
  - Connected component labeling method

If a candidate pixel is connected to an edge pixel directly or via another candidate pixel then it is declared as an edge pixel



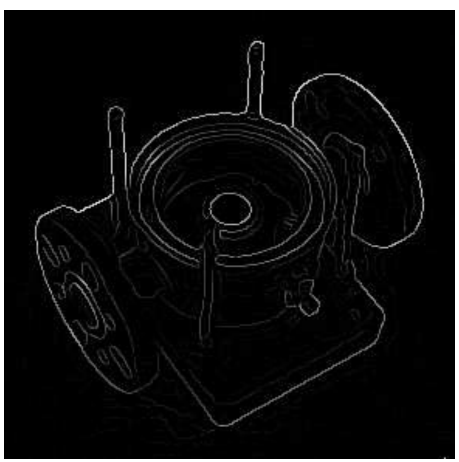
Edge Map

Canny Edge Detector

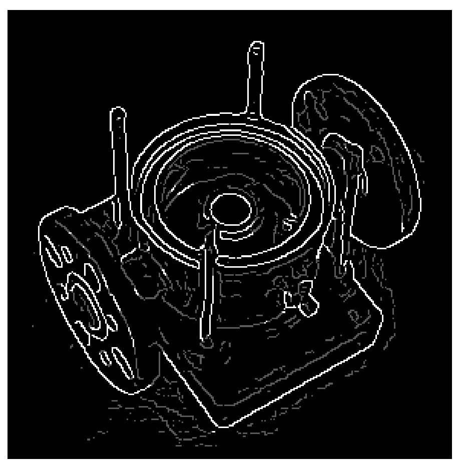
Input

Step 1

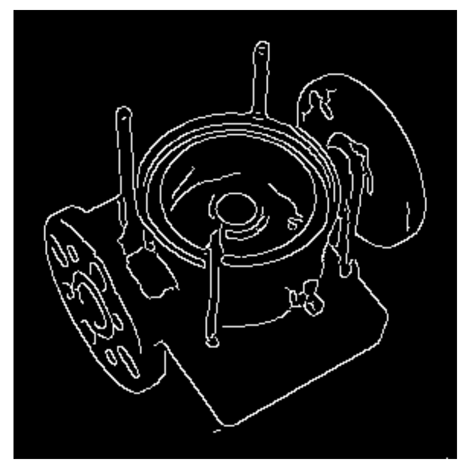
Step 2



Step 3



Step 4

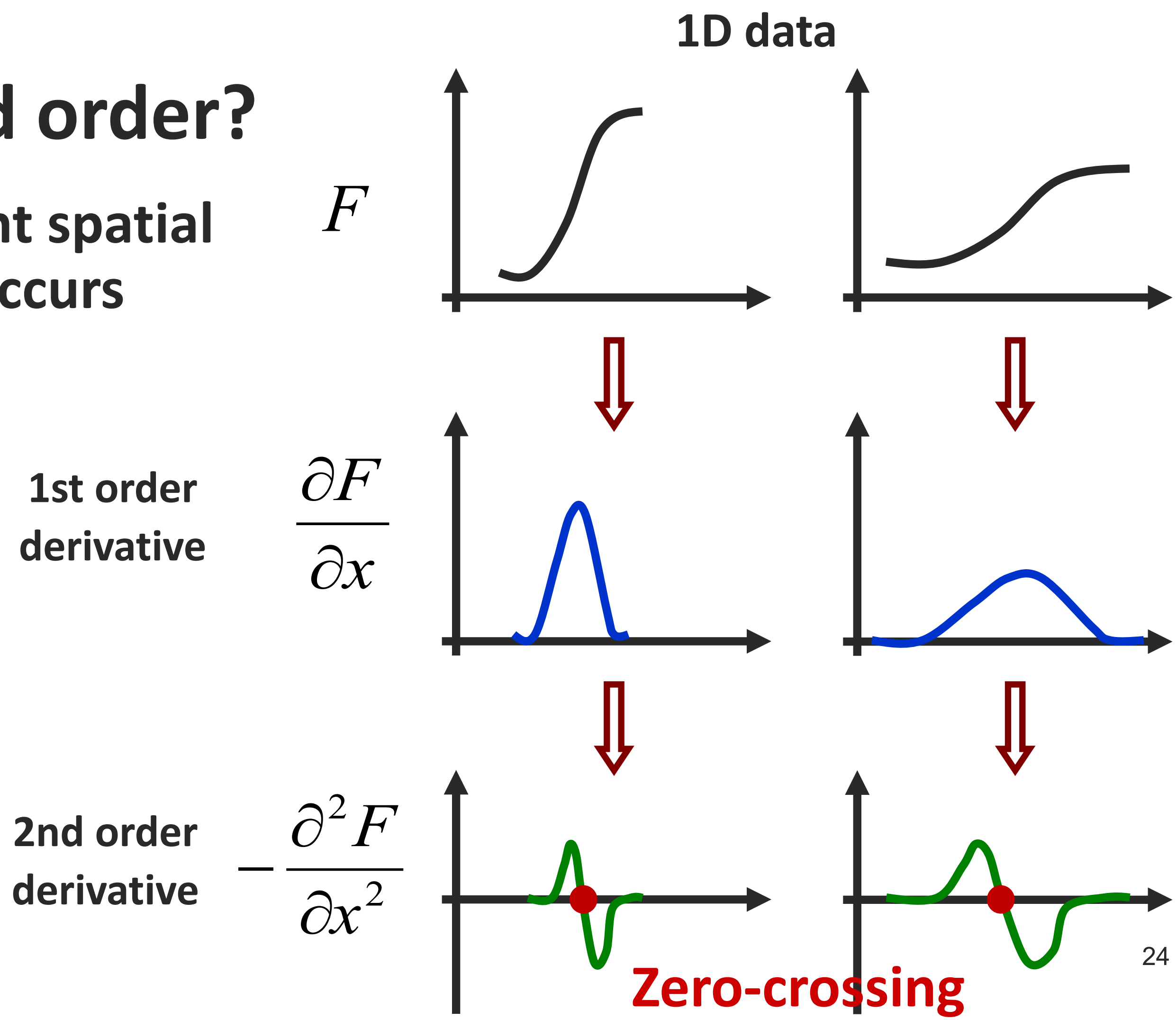


Step 5

Edge Detection – Part II

Edge Detection (2^{nd} order)

- **Why 2nd order?**
  - **Significant spatial change occurs**



Edge Detection (2nd order)

- **Laplacian Generation**

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \Longrightarrow \nabla^2 F(x,y) = \frac{\partial^2 F(x,y)}{\partial x^2} + \frac{\partial^2 F(x,y)}{\partial y^2}$$

- **Discrete Approximation**

$$-\frac{\partial^2}{\partial x^2} \cong \begin{bmatrix} -1 & 2 & -1 \end{bmatrix} \Longrightarrow 2f(x) - (f(x+h) + f(x-h))$$

  - **By Taylor series expansion**

$$\begin{aligned} & 2f(x) - (f(x+h) + f(x-h)) \\ = \quad & 2f(x) - \left[f(x) + hf'(x) + \frac{h^2}{2} f''(x) + f(x) + (-h) f'(x) + \frac{h^2}{2} f''(x) + \cdots \right] \\ \cong \quad & -h^2 f''(x) \end{aligned}$$

Edge Detection (2^{nd} order)

- **Discrete Approximation**

$$-\frac{\partial^2}{\partial x^2} \cong \begin{bmatrix} -1 & 2 & -1 \end{bmatrix} \qquad -\frac{\partial^2}{\partial y^2} \cong \begin{bmatrix} -1 & 2 & -1 \end{bmatrix}^T$$

  - **combine together**

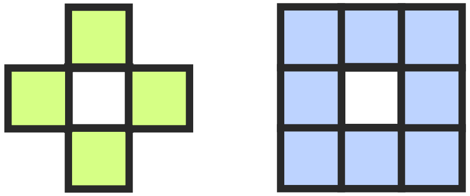
$$-\nabla^2 \cong \frac{1}{4}\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}; \qquad \nabla^2 \cong \frac{1}{4}\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

gain-normalized

Edge Detection (2nd order)

- Laplacian impulse response
  - four-neighbor

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



4-neighbor 8-neighbor

  - eight-neighbor

$$H_1 = \frac{1}{8}\begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}$$

$$H = \frac{1}{8}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

non-separable

$$H = \frac{1}{8}\begin{bmatrix} -2 & 1 & -2 \\ 1 & 4 & 1 \\ -2 & 1 & -2 \end{bmatrix}$$

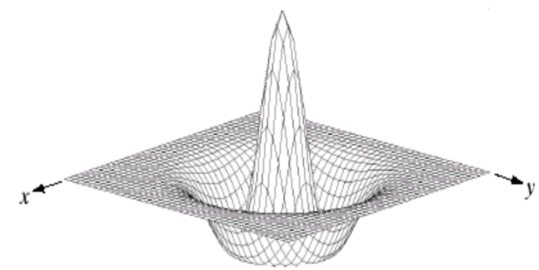
separable

$$H_2 = \frac{1}{8}\begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}$$

Edge Detection (2^{nd} order)

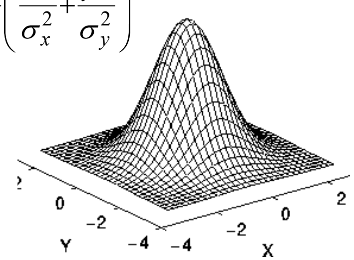
- **Laplacian of Gaussian (LOG)**

→ **Gaussian Low-pass Filter** → **2nd-order Derivative (Laplacian)** →



Mexican Hat Low-pass Filter

$$g(x,y) = Ae^{-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2}\right)}$$



$\frac{1}{273}$

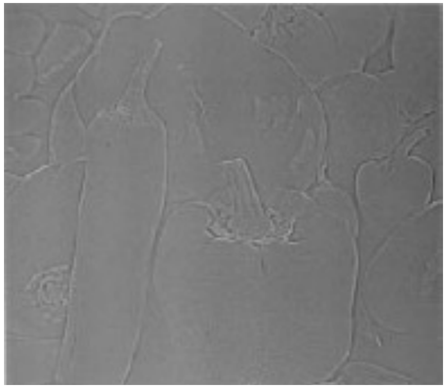
1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

- **Difference of Gaussians (DOG)**

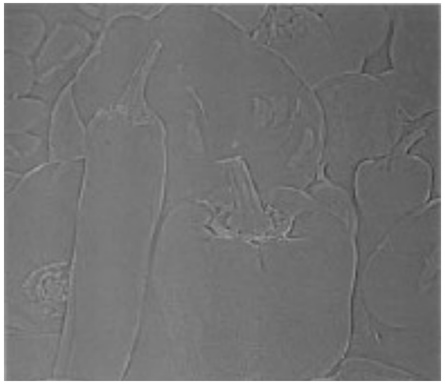
→ **Gaussian Low-pass Filter** → **1st-order Derivative** →

Edge Detection (2nd order)

- Examples

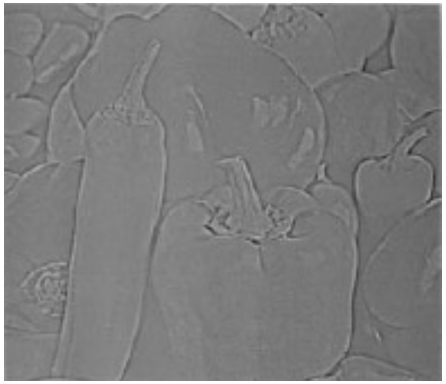


Four-neighbor



Eight-neighbor

Separable eight-neighbor



Laplacian of Gaussian (LOG)

Are we done yet?

Edge Detection (2nd order)

- **2nd Order Edge Detection**

  - **How to detect zero-crossing?**
    - many ways

Edge Detection (2^{nd} order)

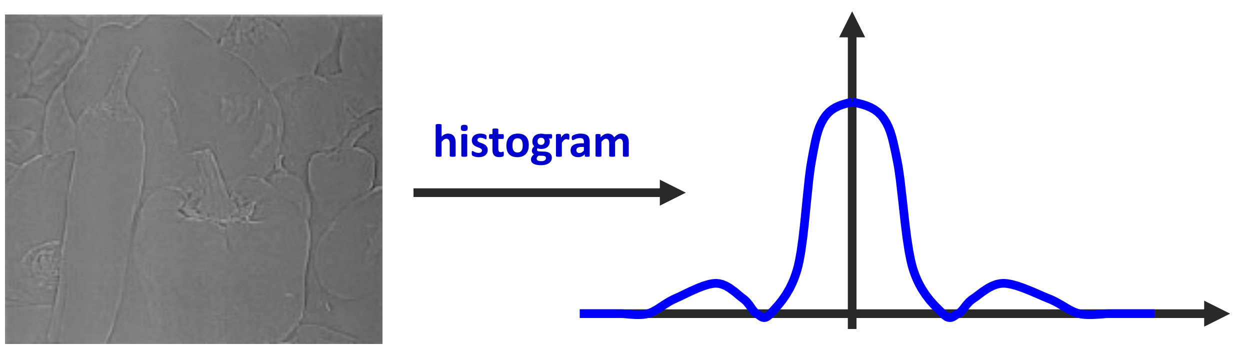
- **Zero-crossing**

3 steps:

- Generate the histogram of G
- Set up a threshold to separate zero and non-zero, output as G'
- For G'(j,k)=0, decide whether (j,k) is a zero-crossing point

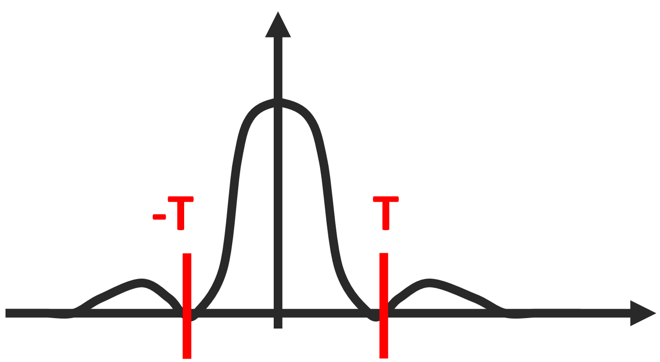
Edge Detection (2nd order)

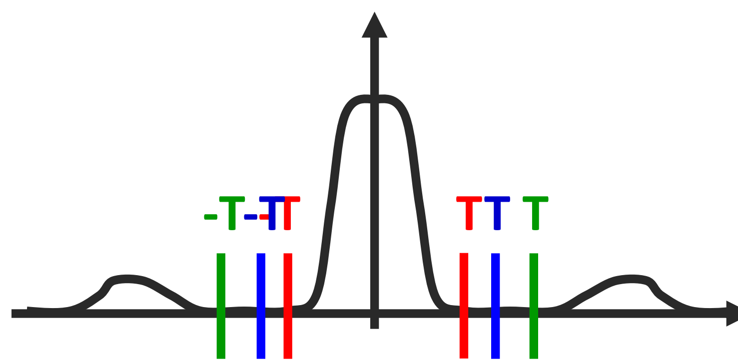
- Zero-crossing
  - 3 steps:
    - **Generate the histogram of G**
    - Set up a threshold to separate zero and non-zero to get G’
    - For G’(j,k)=0, decide whether (j,k) is a zero-crossing point



Edge Detection (2nd order)

- **Zero-crossing**
  - **3 steps:**
    - Generate the histogram of G
    - **Set up a threshold to separate zero and non-zero to get G'** $|G(j,k)| \leq T \Rightarrow G'(j,k) = 0$
    - For G'(j,k)=0, decide whether (j,k) is a zero-crossing point

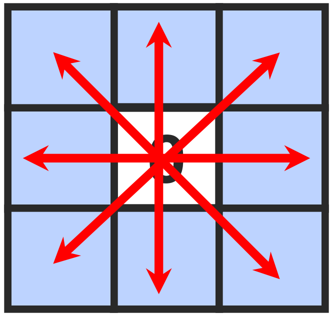




Edge Detection (2^{nd} order)

- **Zero-crossing**
  - **3 steps:**
    - Generate the histogram of G
    - Set up a threshold to separate zero and non-zero to get G’
    - **For G’(j,k)=0, decide whether (j,k) is a zero-crossing point → edge map**

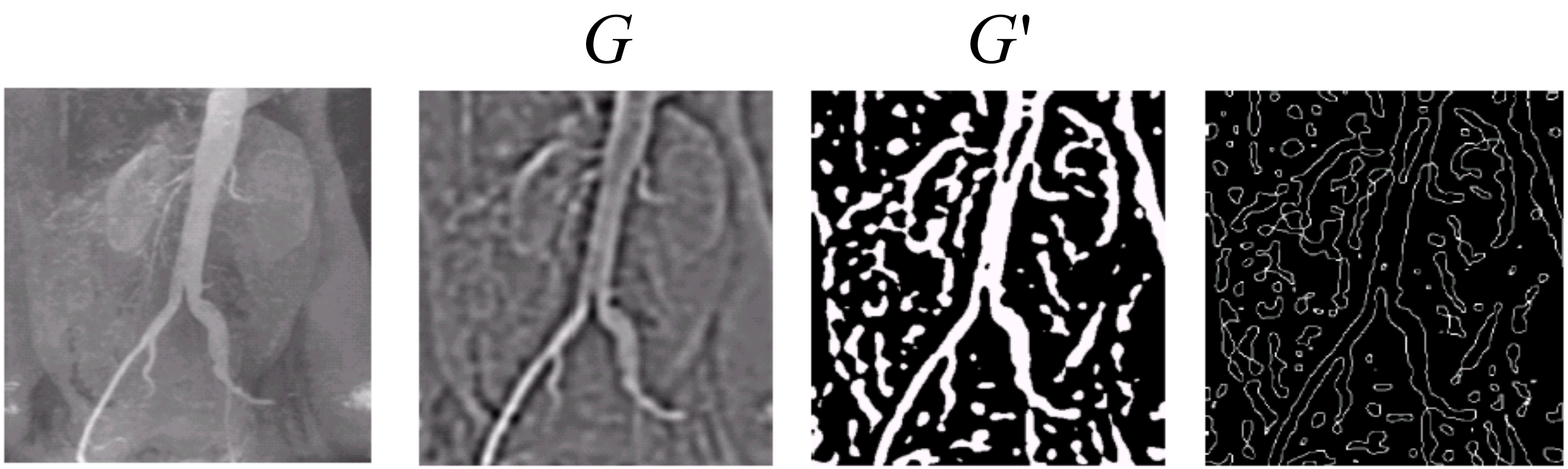
$$G'(j,k) = 0$$



e.g. $\{-1,0,1\}$

Edge Detection (2nd order)

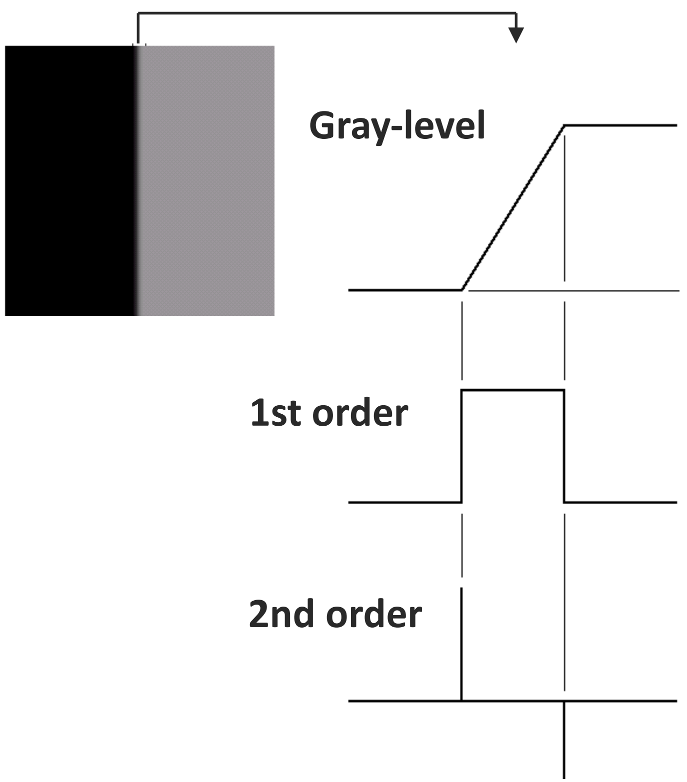
- **Example**



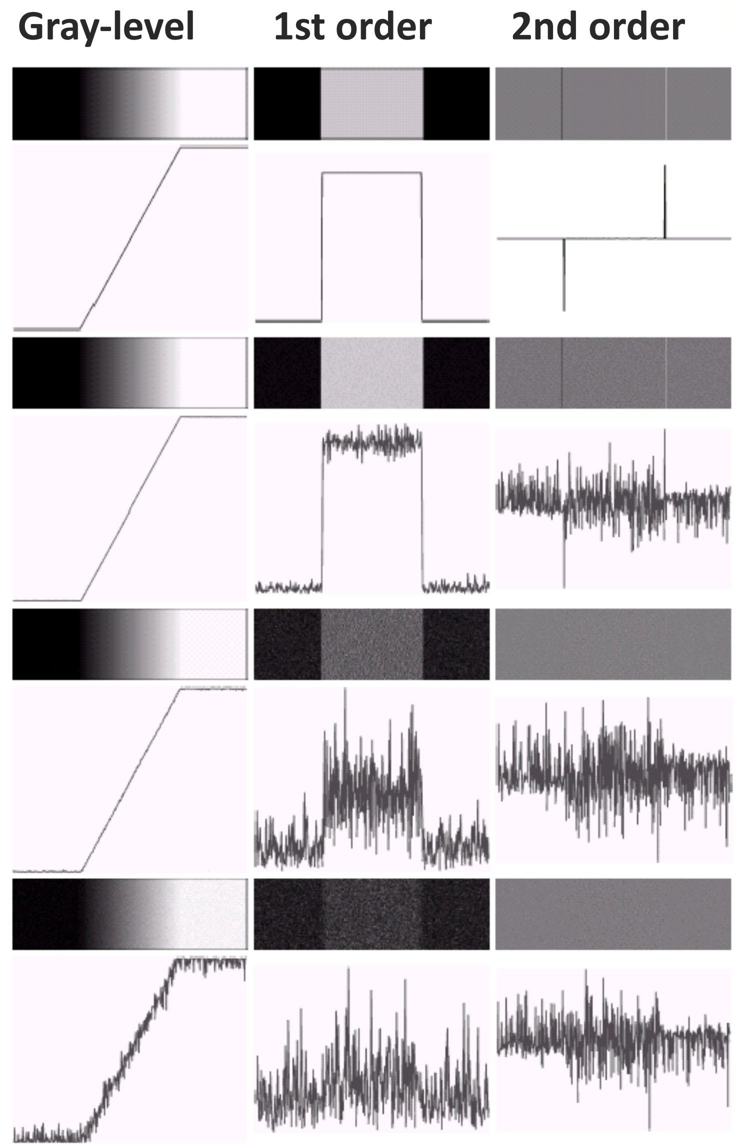
Original | Laplacian | Thresholded | Edge map

Edge Detection

- Noisy image

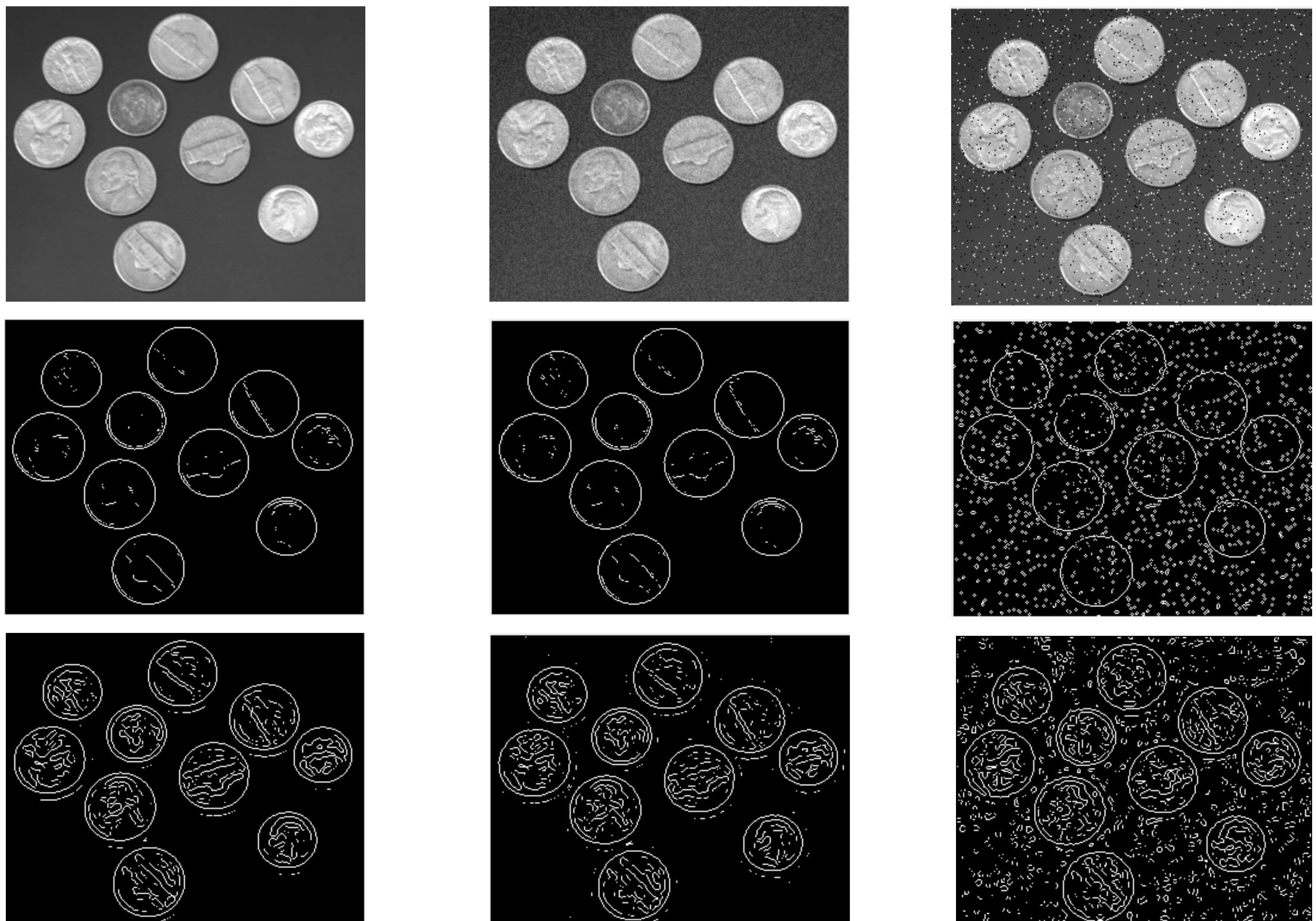


"Pre-processing" ⇐



Edge Detection

I | I + Gaussian noise | I + impulse noise



1^{st} order

2^{nd} order

Edge Detection

- **Post-Processing**

Edge Map	Binary image Edge points → 1; non-edge points → 0
↓	
Edge Linking	Local processing: Based on edge orientation Global processing: Hough transform
↓	
Boundary Curves	A complete edge contour

How can we detect lines?

- **Option 1:**
  - **Search for the line at every possible position/orientation**

- **Option 2:**
  - **Hough transform**
    - **performed after edge detection**
  - **Advantages**
    - **tolerant of gaps in the edges**
    - **relatively unaffected by noise**
    - **unaffected by occlusion in the image**

Hough Transform

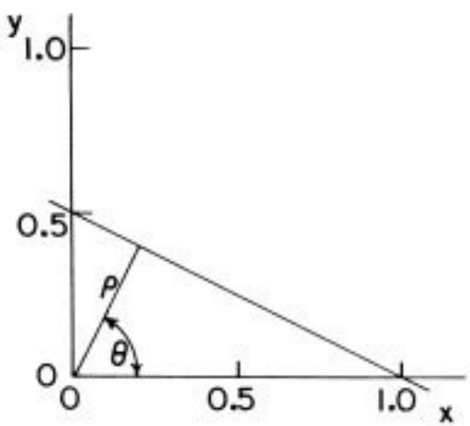
- A straight line can be represented as

$$y = mx + b$$

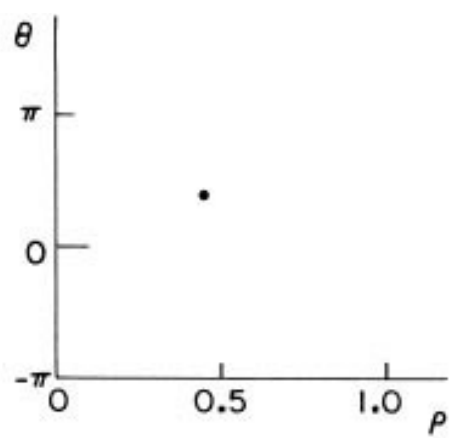
  - fails in case of vertical lines

- A more useful representation is

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



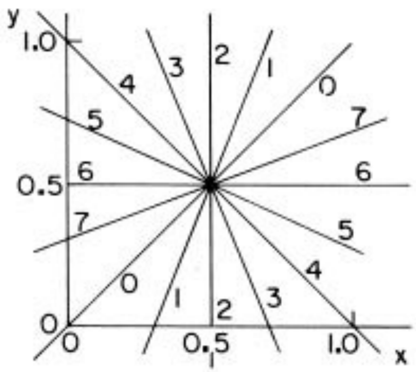
(a) Parametric line



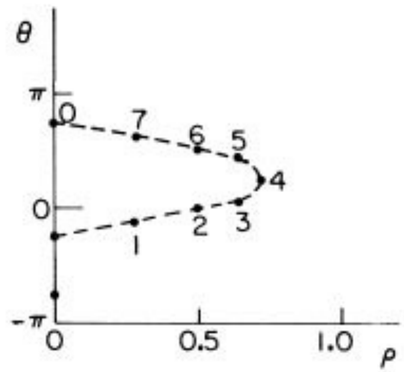
(b) Hough transform of (a)

Hough Transform

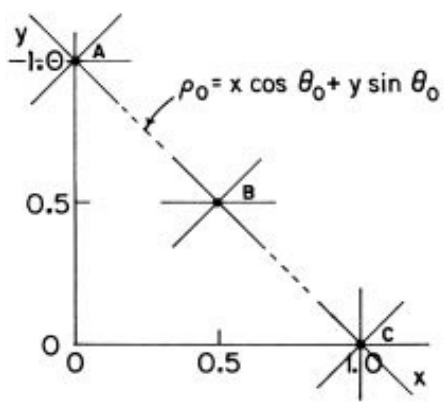
$$\rho = x\cos\theta + y\sin\theta$$



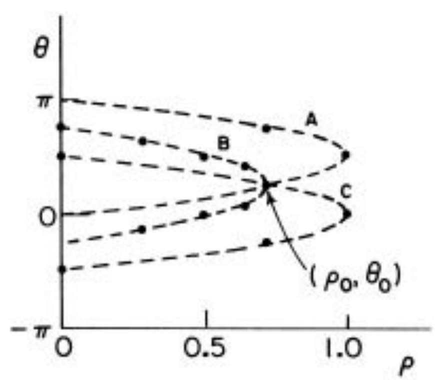
(c) Family of lines, common point



(d) Hough transform of (c)



(e) Colinear points



(f) Hough transform of (e)

Review

Review

- Noise Cleaning
  - Uniform noise → low-pass filtering
  - Impulse noise → non-linear filtering
  - Mixed noise → ?
- Edge Crispening
  - Unsharp masking
- Edge Detection
  - 1^{st}-order edge detection -- threshold
  - 2^{nd}-order edge detection -- zero-crossing