Edge Detection

N. Ravitha Rajalakshmi

Edge Detection

- Find most relevant edges in the image.
- These edges should be connected into meaningful lines and boundaries resulting in segmented image containing two (or) more regions.
- The segmented results can be used for tasks such as object counting, feature extraction and classification.

Basic Concepts

- Edge represent boundary between two regions with different characteristics according to gray level, color (or) texture.
- Derivatives can be a better operator to detect the points corresponding to edge
- Given an one dimensional function f(x) , its first order and second order derivative can be expressed as the difference between two adjacent pixels.

$$\frac{\partial f}{\partial x} = f'(x) = f(x+1) - f(x)$$

$$\frac{\partial^2 f}{\partial x^2} = f''(x) = f(x+1) - 2f(x) + f(x-1)$$

Computing derivative in an image

- Since derivative is a linear operator , it can be computed using masks
- For example, the following mask can be used to determine the second order derivative

0	0	0
-1	2	-1
0	0	0

Detection of isolated point

- Second order derivative is very sensitive to sudden changes
- Hence, can be used to detect an isolated point
- Laplacian operator provides the second order derivative over a two dimensional function

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

$$\frac{\partial^2 f(x,y)}{\partial x^2} = f(x+1,y) - 2f(x,y) + f(x-1,y)$$

$$\frac{\partial^2 f(x,y)}{\partial y^2} = f(x,y+1) - 2f(x,y) + f(x,y-1)$$

$$\nabla^2 f(x,y) = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

Contd..

0	-1	0
-1	4	-1
0	-1	0

Decide a threshold T

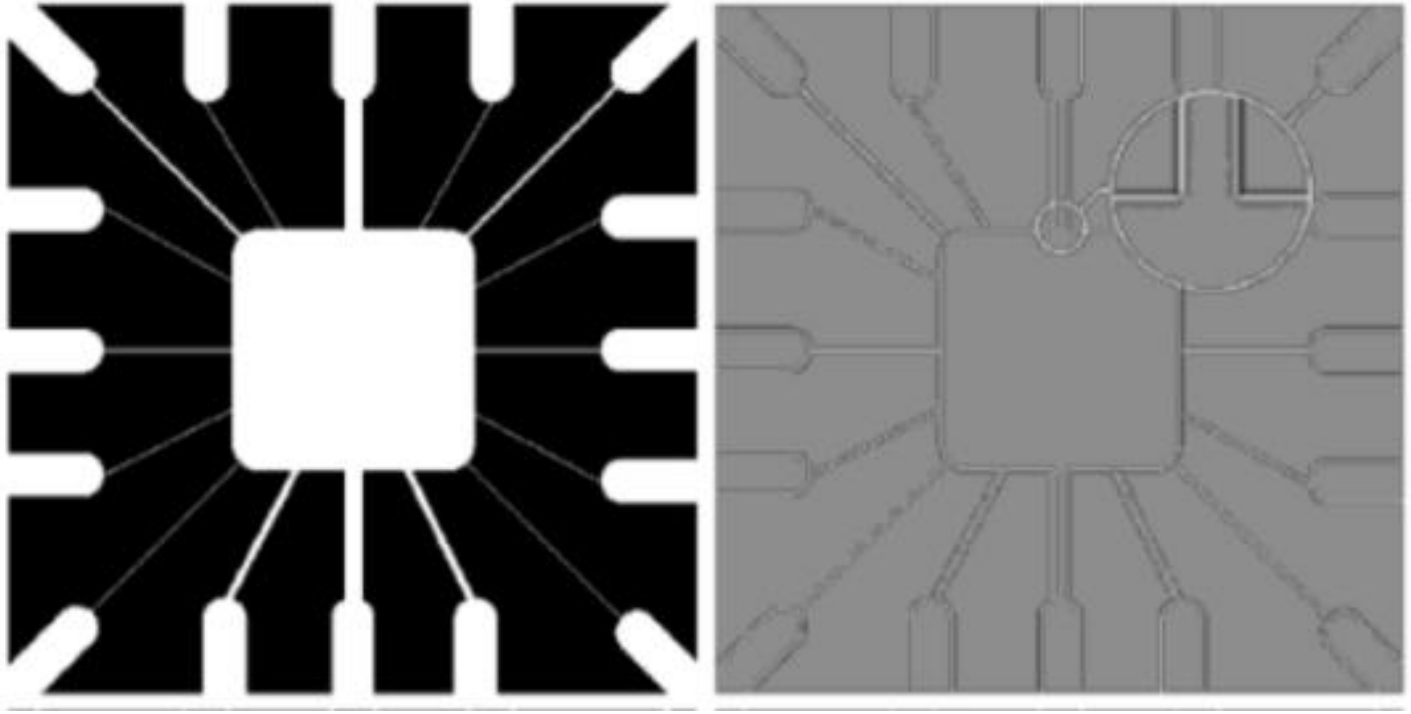
In the resulting filter response, those pixels whose value is larger than T can be considered as an isolated point.

Line detection

- Second order derivative can be used to detect lines
- Another variant of Laplacian operator is shown below. It is isotropic, uniform in all directions.
- Using Laplacian creates double edges

-1	-1	-1
-1	8	-1
-1	-1	-1

Double edge effect

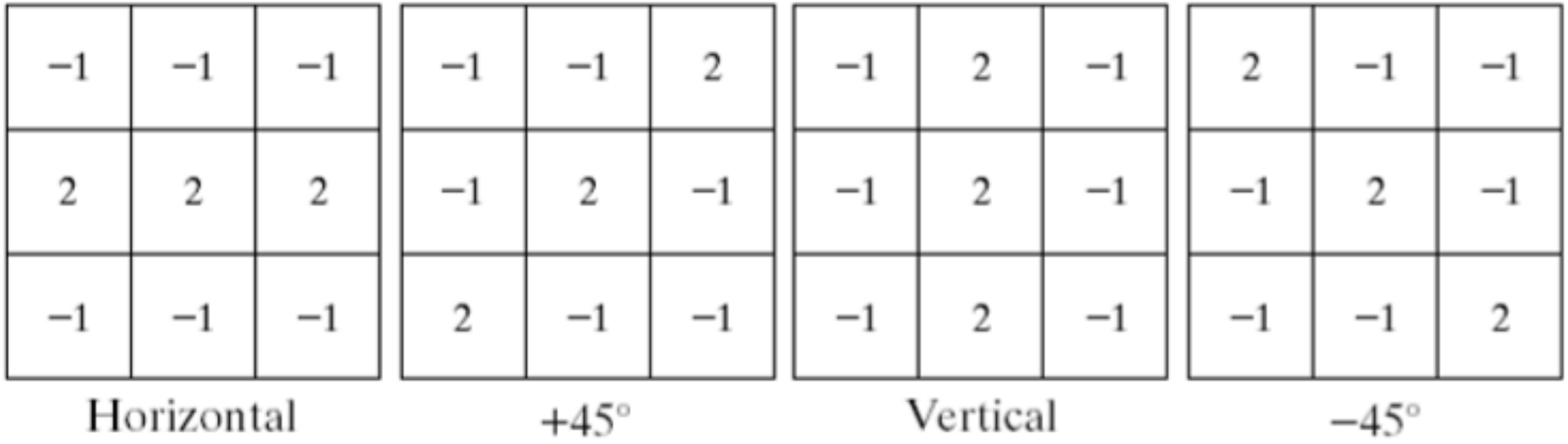


Original Image and Its Laplacian Response

Double edge effect is attributed to the fact that second order derivates creates two responses for every edge indicating the Pixels that are occurring on the either side of the edge pixel.

Line Masks

To detect lines in specific directions, use masks that would emphasize a certain direction and be less sensitive to other directions. Popular line masks are shown below

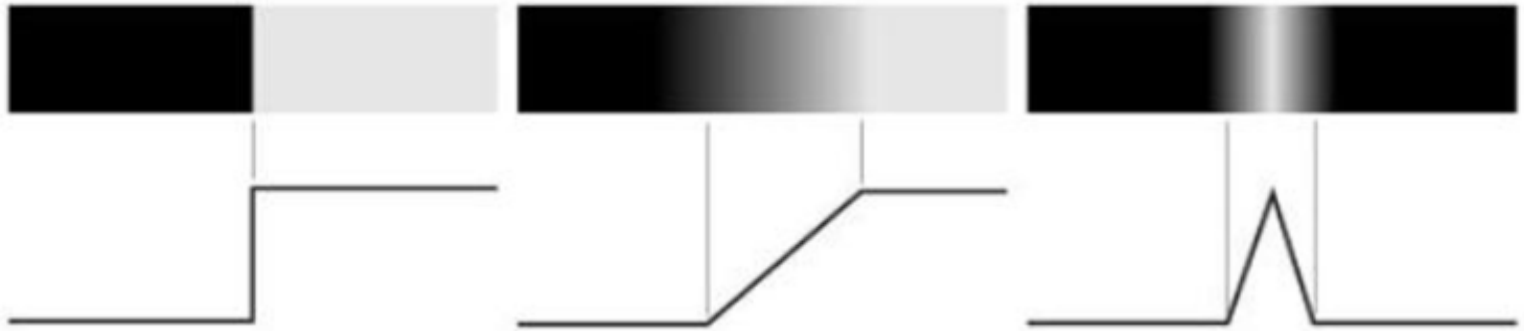


Edge Types

- Three types of edge types which occurs in an image include
  - Step edge – Transition of intensity level is over 1 pixel only in ideal, or few pixels on a more practical use
  - Ramp edge – A slow and graduate transition
  - Roof edge – Transition is back and forth into varying intensity levels

a b c

FIGURE 10.8
From left to right, models (ideal representations) of a step, a ramp, and a roof edge, and their corresponding intensity profiles.

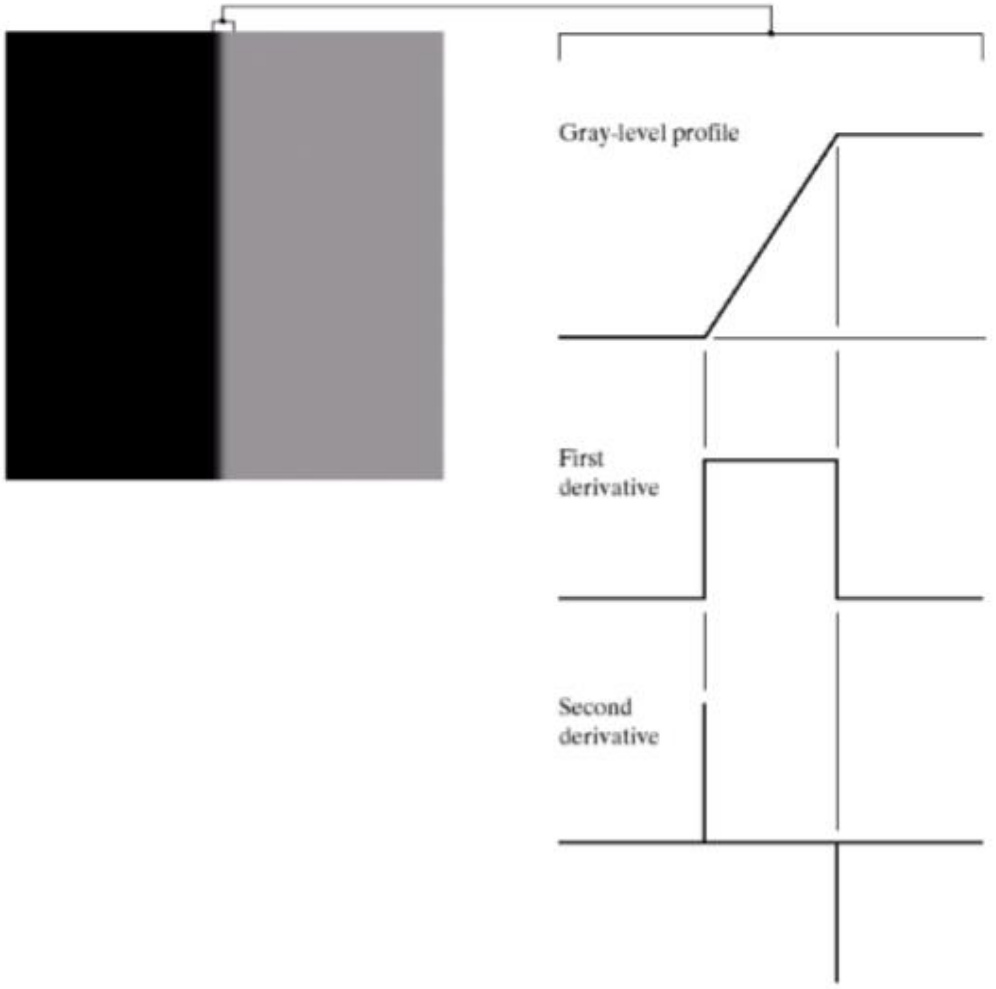


Edge Profile and its derivatives of ramp edge

- First derivative
  - Magnitude can be used to detect the presence of edge at a certain point
  - Produces thick edges
- Second derivative
  - Provide very strong response to fine details and noise
  - Sign determine whether edge pixel lie on the dark side (or) bright side.
  - Zero crossing between the negative and positive peaks of second order derivative can be used to detect the centre of thick edges

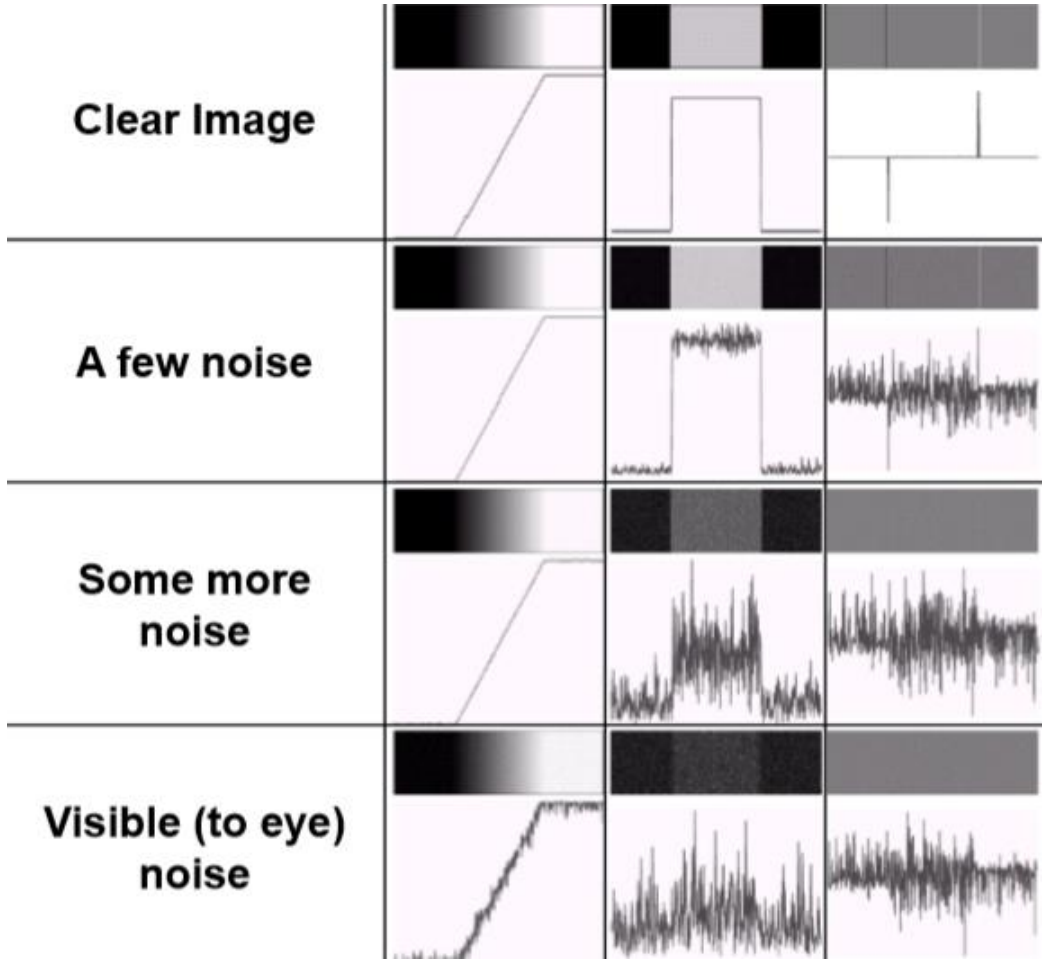
a b

FIGURE 10.6
(a) Two regions separated by a vertical edge. (b) Detail near the edge, showing a gray-level profile, and the first and second derivatives of the profile.



Edge Model Sensitivity

Edges are highly sensitive to noise. Hence, it is a common practice to apply noise reduction techniques before edge detection.



Basic Edge Detection Algorithm

It is a three stage process which involves

- Smoothen the image for removing noise
- Detect edge pixels
- Combine them to form meaningful edge boundaries.

Edge Detection - Basics

- Gradient is the tool used to find edge strength and direction at any given location (x,y) of an image.
- Magnitude of gradient

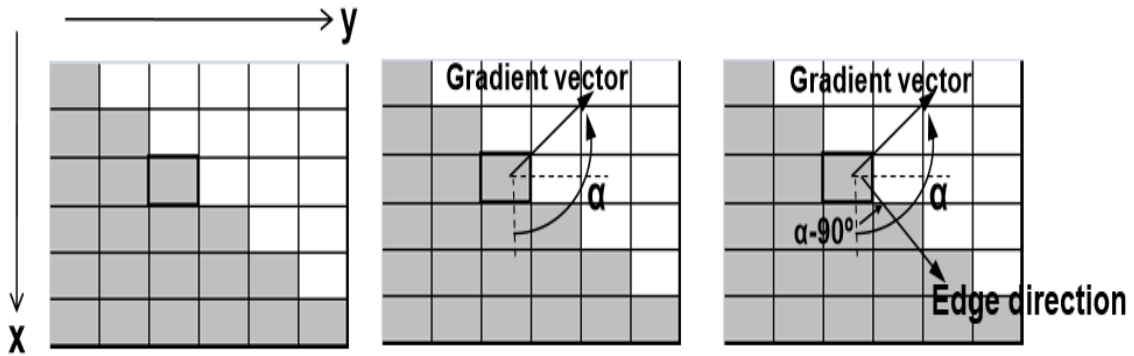
$$\nabla f \equiv grad(f) \equiv \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

Value of rate of change in the intensity $M(x, y) = \sqrt{g_x^2 + g_y^2}$

- Direction of gradient

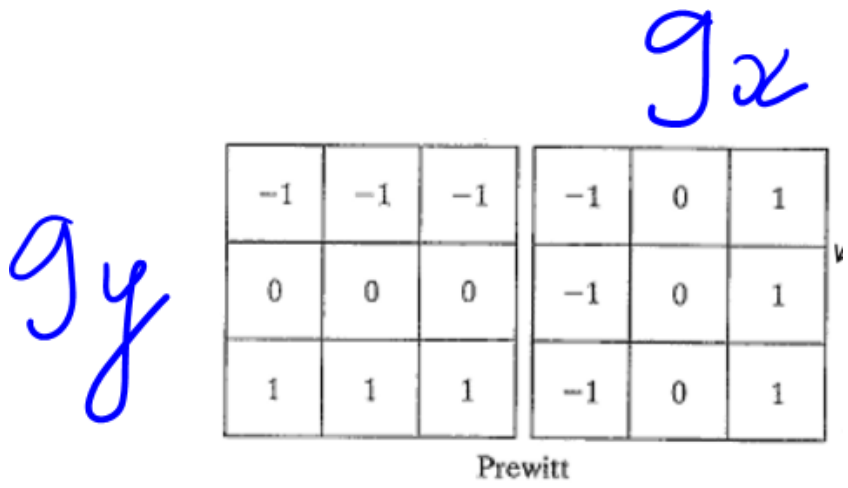
$$\alpha(x, y) = \tan^{-1}\left(\frac{g_y}{g_x}\right)$$

Gradient is always perpendicular to edge direction.

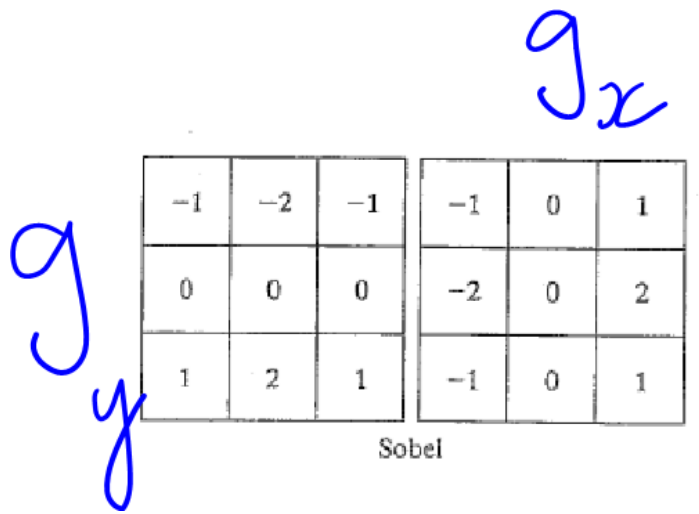


Prewitt and Sobel Operator

- First order derivative operator centered around a pixel
- It prevents the computation of the gradient at every pixel position via approximation over computation



Prewitt



Sobel

Prewitt and Sobel Operator Characteristic

- Both the operator mask coefficients sum to zero and also provide a response of zero in areas of constant intensity.
- Sobel Provides noise suppression as it amplifies the center pixel values
- Both filters are isotropic (provides uniform result across horizontal and vertical edges.)

Advanced Edge Detectors

- Marr-Hildreth
  - Intensity changes are not independent of image scale
  - Hence the detector must be tuned to any scale
  - Used an operator Laplacian of Gaussian

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Gaussian Operator

$$\nabla^2 G(x,y) = \left[\frac{x^2+y^2-2\sigma^2}{\sigma^4}\right] e^{-\frac{x^2+y^2}{2\sigma^2}},$$

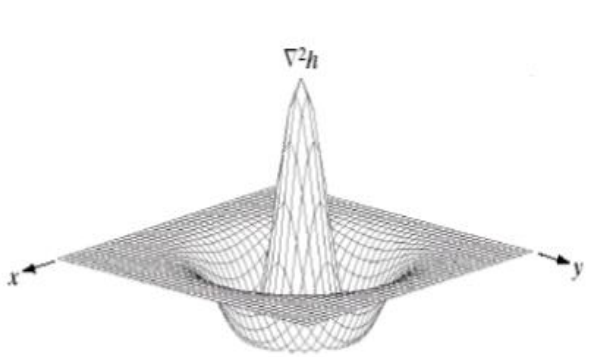
Laplacian of Gaussian Operator

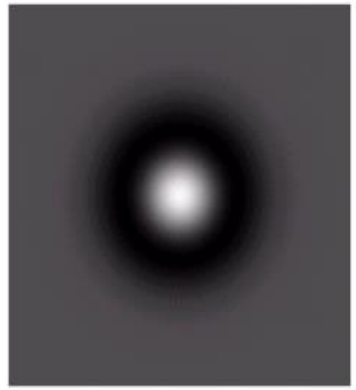
Contd..

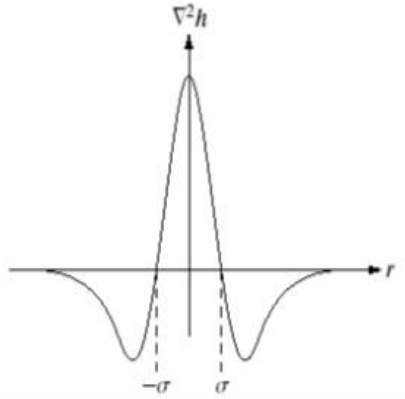
- Gaussian and Laplacian operator can be applied separately
- Gaussian reduces noise (intensity structures less than $\propto$)
- Laplacian detect the edges

Generic Algorithm:

1. Filter image using Gaussian
2. Detect edges using Laplacian
3. Find zero crossings to detect the center of the edge







0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

a b
c d

FIGURE 10.14 Laplacian of a Gaussian (LoG). (a) 3-D plot. (b) Image (black is negative, gray is the zero plane, and white is positive). (c) Cross section showing zero crossings. (d) 5 × 5 mask approximation to the shape of (a).

Advanced Edge Detectors

- Canny Edge Detector
  - Smooth the image using Gaussian Filter
  - Detect the edges using gradient
  - For each detected edge (Let M(x,y) indicate the magnitude $\alpha(x, y)$ indicate the direction of detected edge pixels), perform non maximal suppression. Here, k denote the neighborhood.

1. Find the direction d_k that is closest to $\propto (x, y)$
2. If the value of $M(x, y)$ is less than at least one of two neighbors along d_k, $g_N(x, y) = 0$ otherwise $g_N(x, y) = M(x, y)$.

Contd..

- Hysteresis thresholding is performed to reduce the false edges
  - Two thresholds T_L and T_H are used.
  - Create two thresholded images $g_{NL}(H)$ using T_L (weak pixels) and $g_{NH}(H)$ using T_H (strong pixels)

1. Scan the image $g_{NH}(H)$ and mark all pixels as unvisited
2. For each edge pixel x in $g_{NH}(H)$,

   Scan the image $g_{NL}(H)$ and mark the pixels that are connected to x as valid
3. Set all unmarked pixels in $g_{NL}(H)$ as 0
4. Return $g_{NH}(H)$ + $g_{NL}(H)$

Those weak pixels that are connected to strong pixels are considered as edge pixels

Edge Linking

- These edge pixels from previous module result in fragmented edges.
- So the method is usually followed by linking algorithm designed to assemble edge pixels into meaningful edges.
- Two methods include
  - Local Processing
    - Pixels which are similar according to predefined criteria are linked.
  - Global Processing (Hough Transform)
    - Detect all pixels that lie on particular shape

Local Processing – Edge Linking

1. Compute the gradient magnitude and direction of the input image
2. Construct a binary image

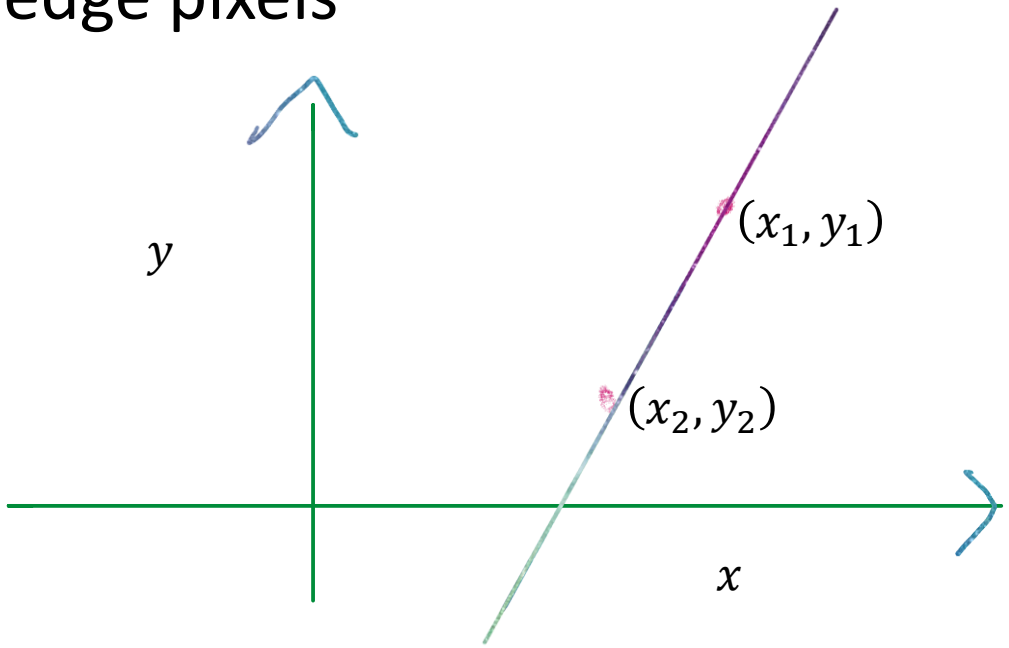
$$g(x,y) = \begin{cases} 1 & M(x,y) > T_M \; and \; \alpha(x,y) = A \pm T_A \\ 0 & otherwise \end{cases}$$

A denotes specified direction T_A denotes band of acceptable directions about A

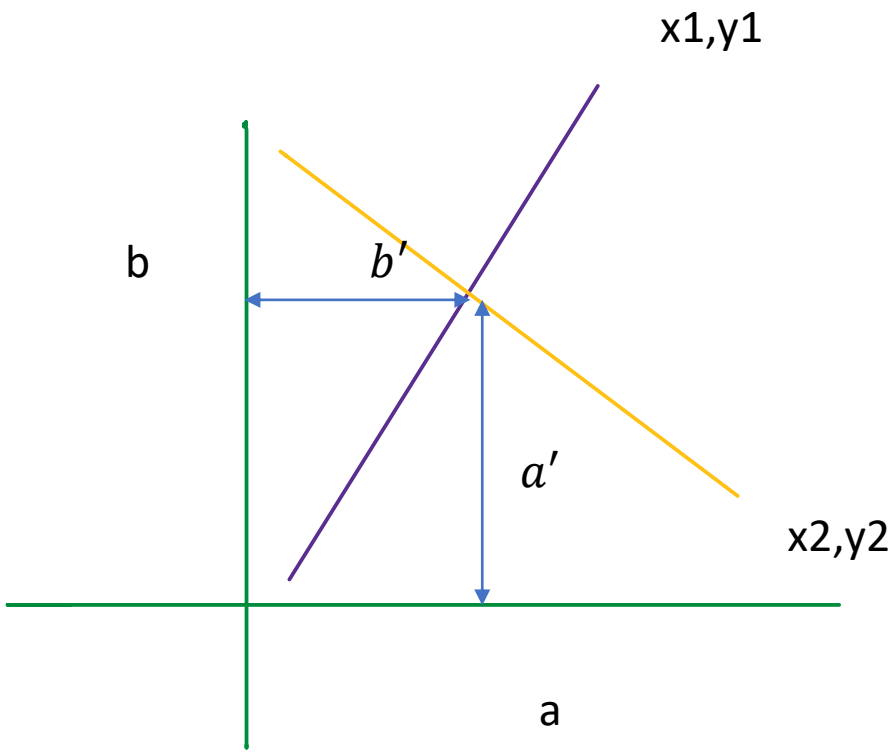
3. Scan rows in g(x,y) and fill in the gaps that do not exceed specified length K
4. To detect the gaps along other directions, rotate the image by angle θ and repeat step 3

Global Processing – Edge Linking

- Given that pixels lie on a straight line, it can be detected using Hough Transform
- It utilizes parameter space for detection of line from the available edge pixels



Provided that (x1,y1) and (x2,y2) are edge pixels, there is inherently a line connecting them which needs to be detected by Hough transform



a' and b' indicate the slope and intercept of the line on which the point (x1,y1) and (x2,y2) passes through in the cartesian plane

Parameter space

Line equation can be expressed as

Y = a x + b

Here a and b corresponds to parameter space, they determine the slope and intercept of a line in the cartesian plane

In a deterministic parameter space, check whether the (x,y) passes through the line specified by a and b. There are multiple values of (a, b) denoting set of lines in which (x,y) passes through

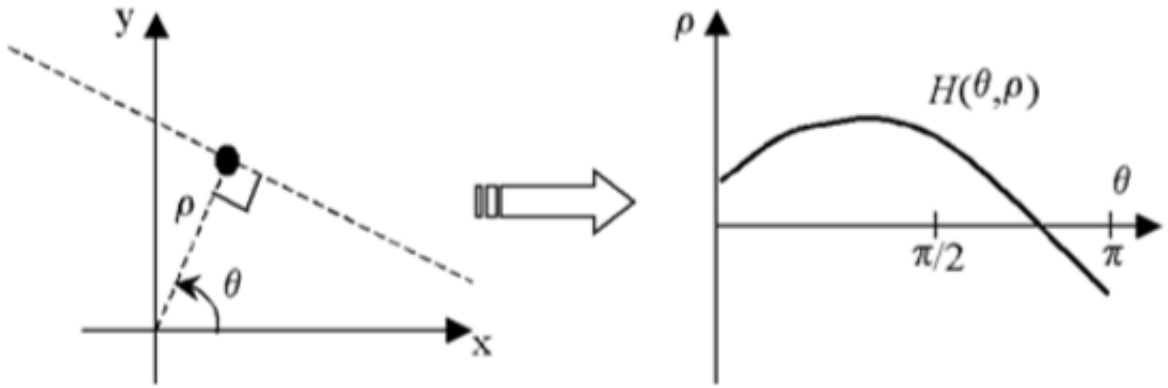
If there are two edge pixels say (x1,y1) and (x2,y2) lie on the same line in cartesian plane, then they intersect in parameter space at some point

- Since (x_i, y_i) approaches infinity in the vertical direction, normal representation of the line is used.

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

Here, ρ denotes the distance between line and origin and θ denote the angle between the distance vector and positive x axis

- Each line in xy plane is transformed into sinusoidal in ρ θ plane as shown below



Hough Transform Implementation

- Divide the ρ θ into discrete sections (They are known as accumulator cells) and initialize its corresponding value to zero.
- For an edge pixel (xi,yi) , compute ρi for every possible θ
- Increase the value in the accumulator **(ρi θ)**

Inspect the accumulator cells with high values to identify the ρ' θ'