Image Segmentation

In previous modules we tried to improve the image for better visualization

Now we will try to retrieve some information from image for high level analysis

Image segmentation

- Division of an image into regions or categories, which correspond to different objects or parts of objects

To represent meaningful areas

Other applications such as number Plate detection, satelite imaging etc.

Image segmentation

Discontinuity based

- Partition is carried out based on abrupt change in intensity values

Focus is on identifying

1. Points
2. Lines
3. Edges

Similarity based

- Group those pixels which are similar in some sense

Techniques used such as

1. Thresholding
2. Region growing
3. Region splitting and merging

Discontinuity based image segmentation

- Main focus is to **find** **isolated points**, **lines** and **edges** in the image
- This can be achieved by application of mask like we have discussed in spatial filtering

$$R = w_1 z_1 + w_2 z_2 + \ldots + w_9 z_9 = \sum_{i=1}^{9} w_i z_i$$

FIGURE 10.1 A general 3×3 mask.

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

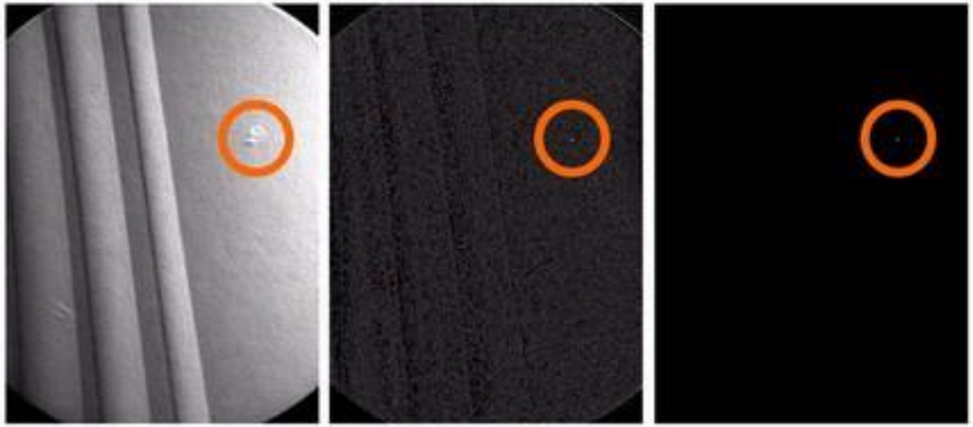
OR

$$R = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t)$$

Discontinuity based image segmentation

- **Isolated point detection**

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



X-ray image of a turbine blade | Result of point detection | Result of thresholding

$$|R| > T$$

T is threshold value (non negative)

Discontinuity based image segmentation

- **Line detection**

Moving first mask over entire image
Detects points lying on horizontal line
We can apply all masks over image and Find the R value

$$|R_i| > |R_j| \ \forall \ i \neq j$$

Then line is having angle more likely To the i^{th} mask

FIGURE 10.3 Line masks.

Horizontal

−1	−1	−1
2	2	2
−1	−1	−1

+45°

−1	−1	2
−1	2	−1
2	−1	−1

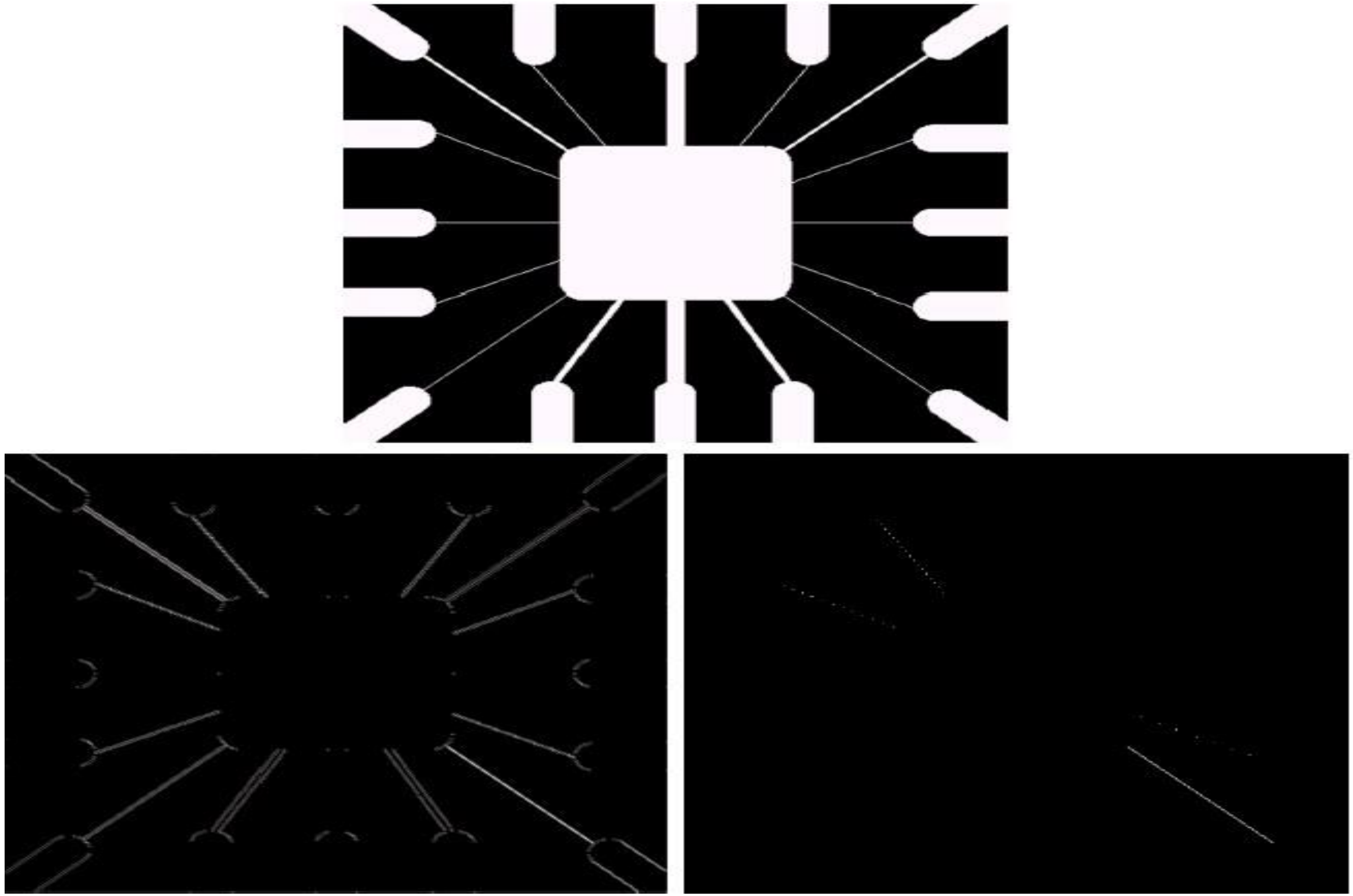
Vertical

−1	2	−1
−1	2	−1
−1	2	−1

−45°

2	−1	−1
−1	2	−1
−1	−1	2

Detection of Discontinuities Line Detection



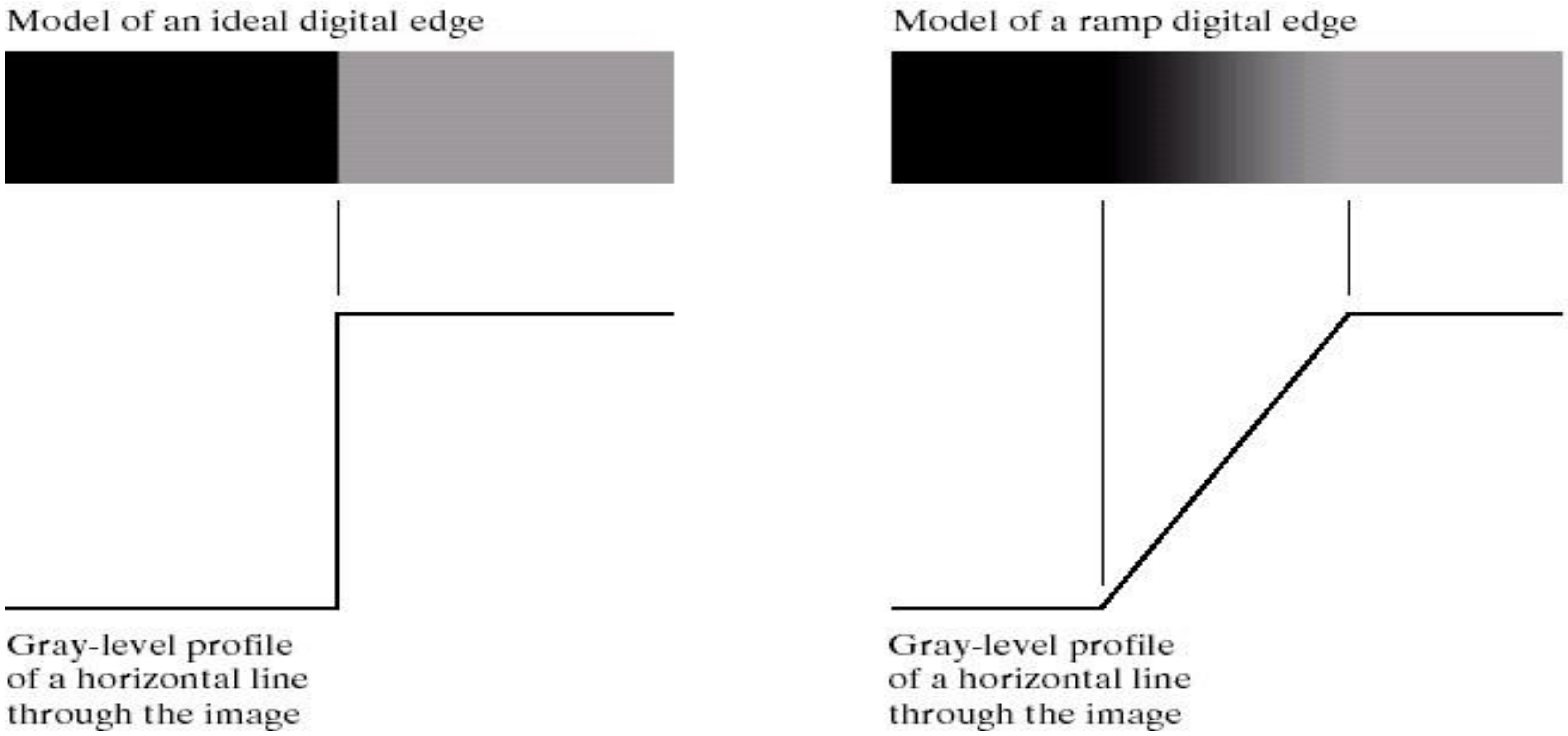
a
b c

FIGURE 10.4
Illustration of line detection.
(a) Binary wire-bond mask.
(b) Absolute value of result after processing with −45° line detector.
(c) Result of thresholding image (b).

Discontinuity based image segmentation

- **Edge detection:**
  Detecting the discontinuity in image
- Edge: Boundary between two regions having distinct intensity values

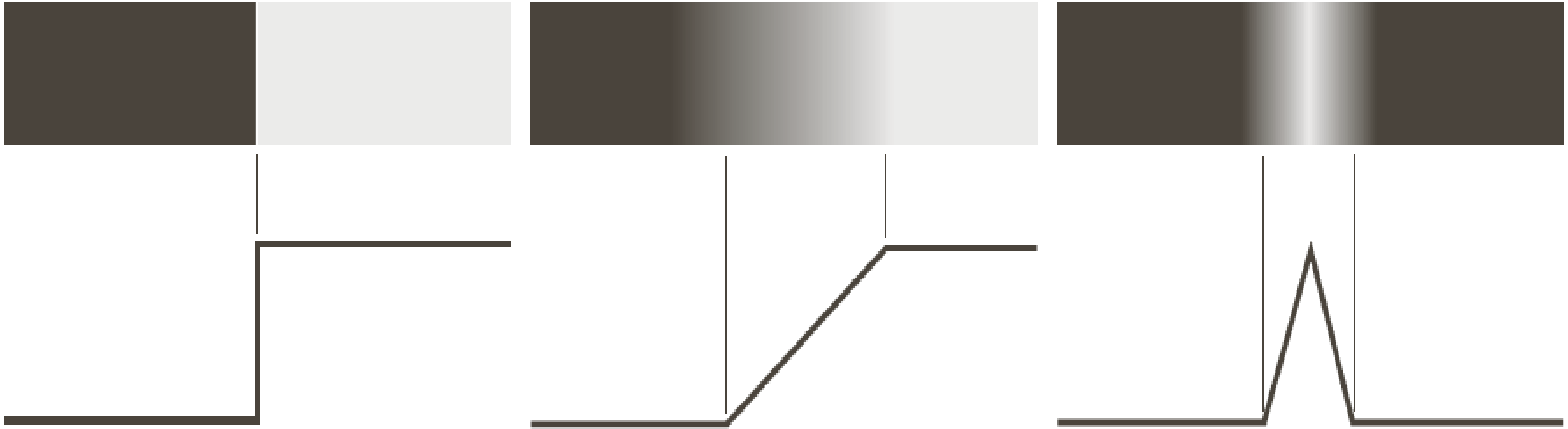
Detection of Discontinuities Edge Detection



a b

FIGURE 10.5 (a) Model of an ideal digital edge. (b) Model of a ramp edge. The slope of the ramp is proportional to the degree of blurring in the edge.

Detection of Discontinuities Edge Detection

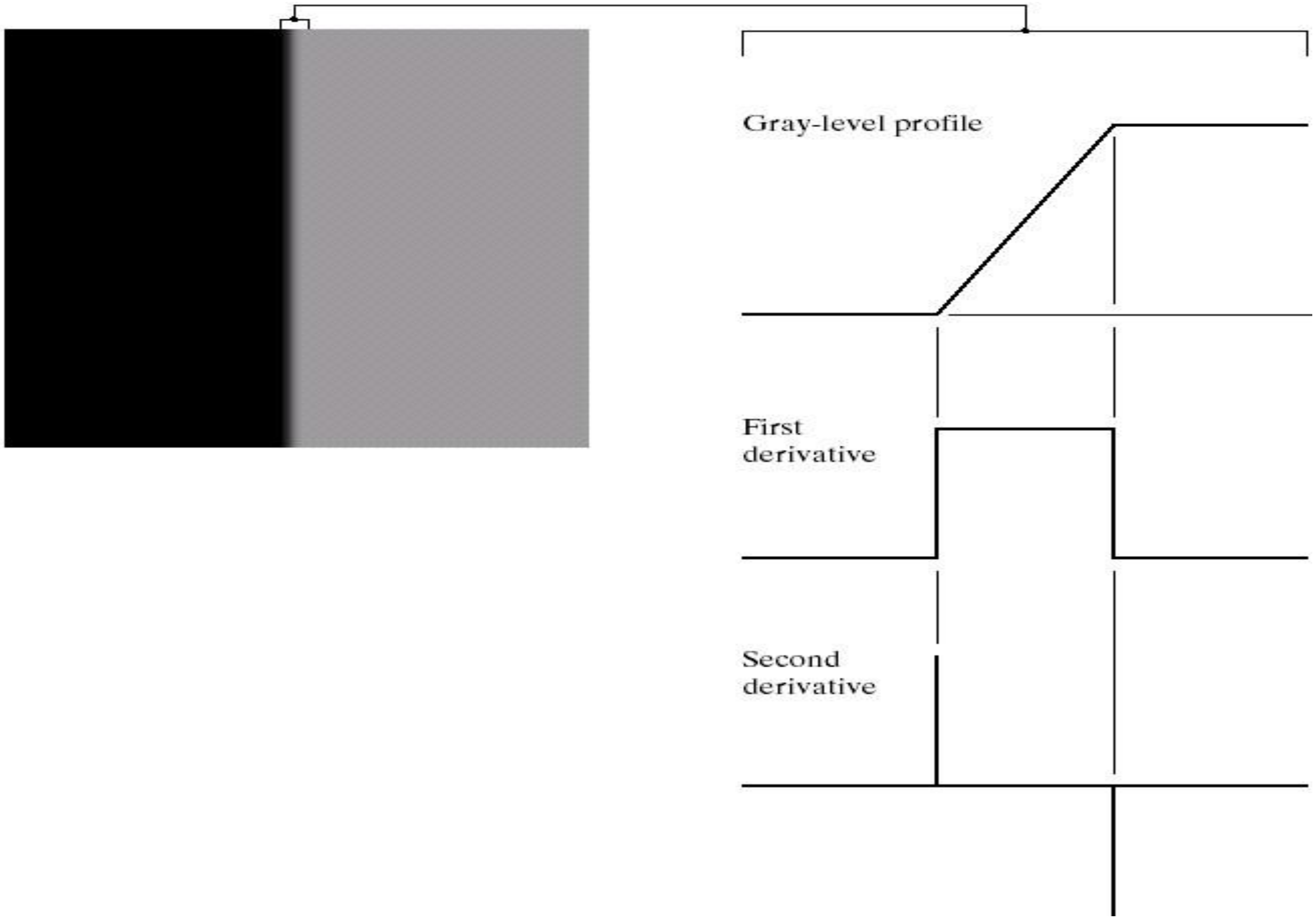


From left to right, models (ideal representations) of a step, a ramp, and a roof edge, and their corresponding intensity profiles.

Detection of Discontinuities Edge Detection

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.



Detection of Discontinuities Edge Detection

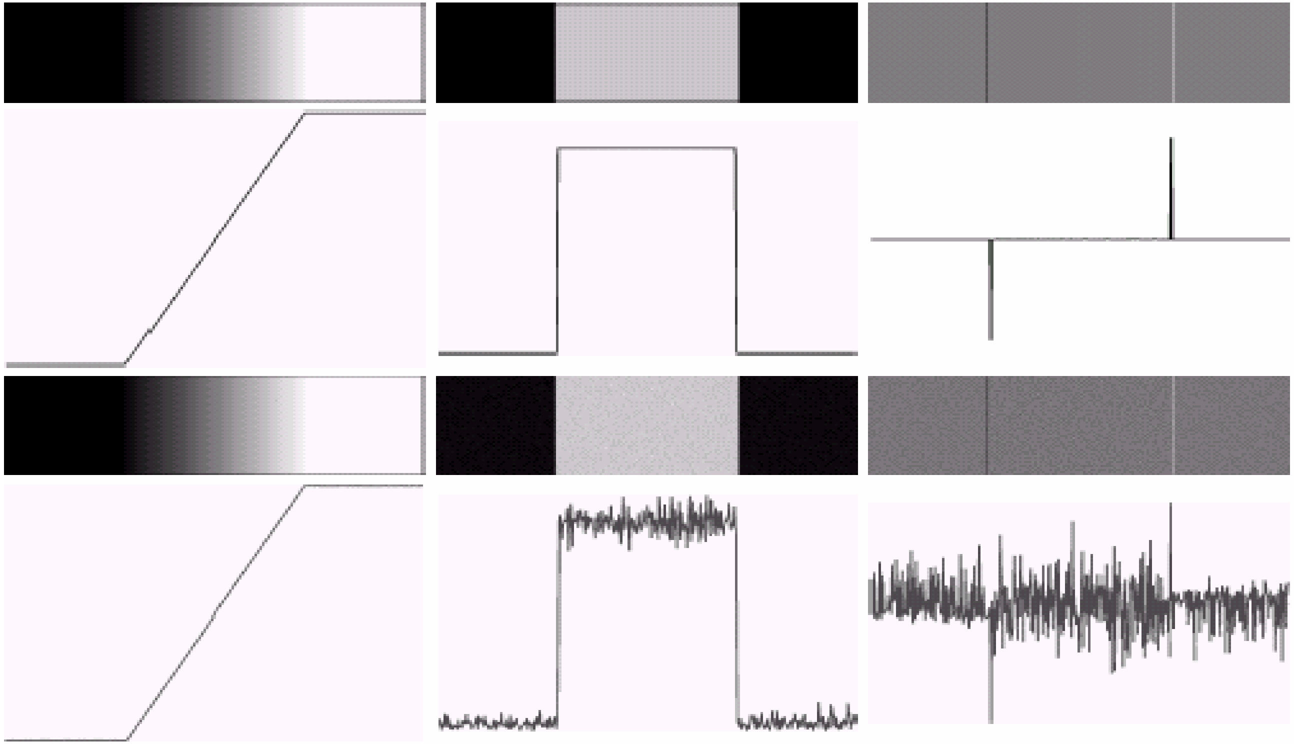


FIGURE 10.7 First column: images and gray-level profiles of a ramp edge corrupted by random Gaussian noise of mean 0 and $\sigma = 0.0, 0.1, 1.0,$ and 10.0, respectively. Second column: first-derivative images and gray-level profiles. Third column: second-derivative images and gray-level profiles.

a
b
c
d

Detection of Discontinuities Edge Detection

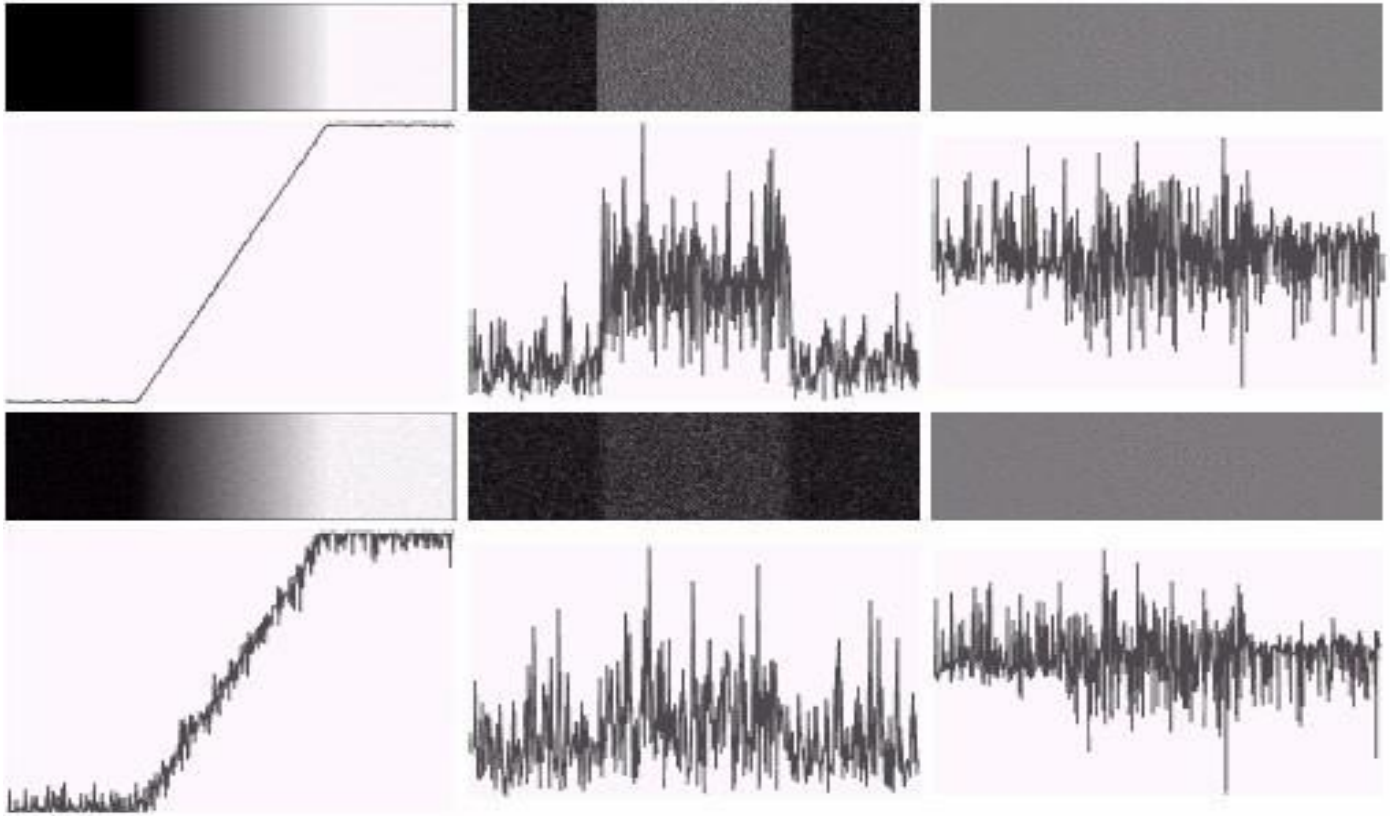


FIGURE 10.7 First column: images and gray-level profiles of a ramp edge corrupted by random Gaussian noise of mean 0 and $\sigma = 0.0, 0.1, 1.0,$ and 10.0, respectively. Second column: first-derivative images and gray-level profiles. Third column: second-derivative images and gray-level profiles.

a
b
c
d

Detection of Discontinuities Gradient Operators

- First-order derivatives:
  - The gradient of an image $f(x,y)$ at location (x,y) is defined as the vector:

$$\nabla \mathbf{f} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

  - The magnitude of this vector: $\nabla f = \text{mag}(\nabla \mathbf{f}) = \left[G_x^2 + G_y^2\right]^{\frac{1}{2}}$
  - The direction of this vector: $\alpha(x, y) = \tan^{-1}\left(\frac{G_x}{G_y}\right)$
  - It points in the direction of the greatest rate of change of f at location (x,y)

Detection of Discontinuities Gradient Operators

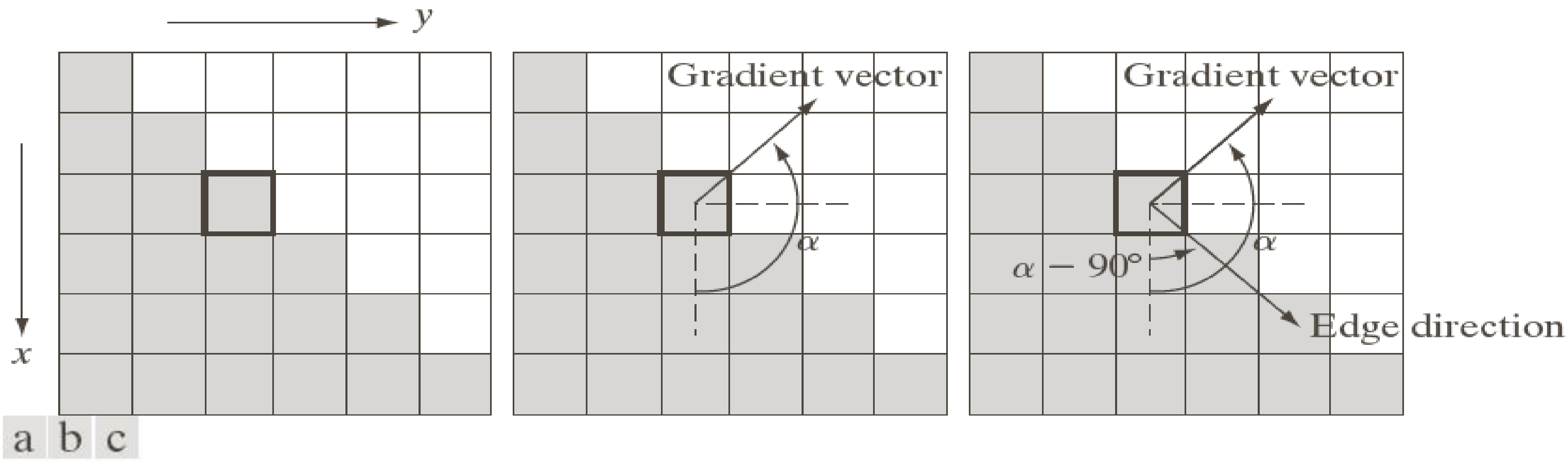


FIGURE 10.12 Using the gradient to determine edge strength and direction at a point. Note that the edge is perpendicular to the direction of the gradient vector at the point where the gradient is computed. Each square in the figure represents one pixel.

Detection of Discontinuities Gradient Operators

Roberts cross-gradient operators

−1	0		0	−1
0	1		1	0

Roberts

Prewitt operators

−1	−1	−1		−1	0	1
0	0	0		−1	0	1
1	1	1		−1	0	1

Prewitt

Sobel operators

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

Sobel

Detection of Discontinuities Gradient Operators

Prewitt masks for detecting diagonal edges

0	1	1	−1	−1	0
−1	0	1	−1	0	1
−1	−1	0	0	1	1

Prewitt

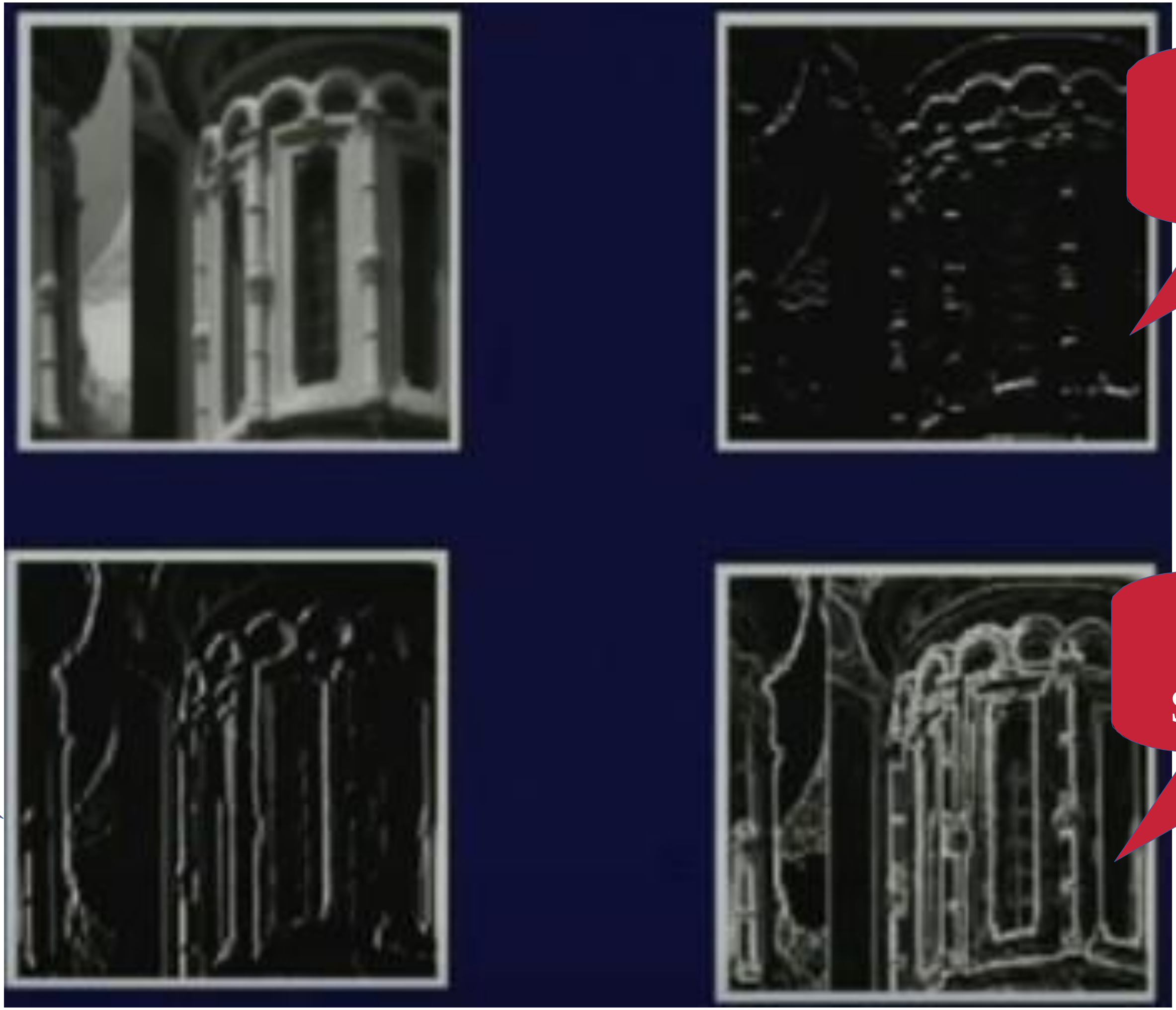
Sobel masks for detecting diagonal edges

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

Sobel

a	b
c	d

FIGURE 10.9 Prewitt and Sobel masks for detecting diagonal edges.


Input Image
Output Image By horizontal Sobel operator
Output Image By vertical Sobel operator
Output Image By combined Sobel operator

Detection of Discontinuities Gradient Operators: Example

a b
c d

FIGURE 10.10 (a) Original image. (b) $|G_x|$, component of the gradient in the x-direction. (c) $|G_y|$, component in the y-direction. (d) Gradient image, $|G_x| + |G_y|$.

$$\nabla f \approx |G_x| + |G_y|$$

Detection of Discontinuities Gradient Operators: Example

FIGURE 10.11 Same sequence as in Fig. 10.10, but with the original image smoothed with a 5×5 averaging filter.

Detection of Discontinuities Gradient Operators: Example

a b

FIGURE 10.12 Diagonal edge detection. (a) Result of using the mask in Fig. 10.9(c). (b) Result of using the mask in Fig. 10.9(d). The input in both cases was Fig. 10.11(a).

0	1	2
−1	0	1
−2	−1	0

−2	−1	0
−1	0	1
0	1	2

Detection of Discontinuities Gradient Operators: Example

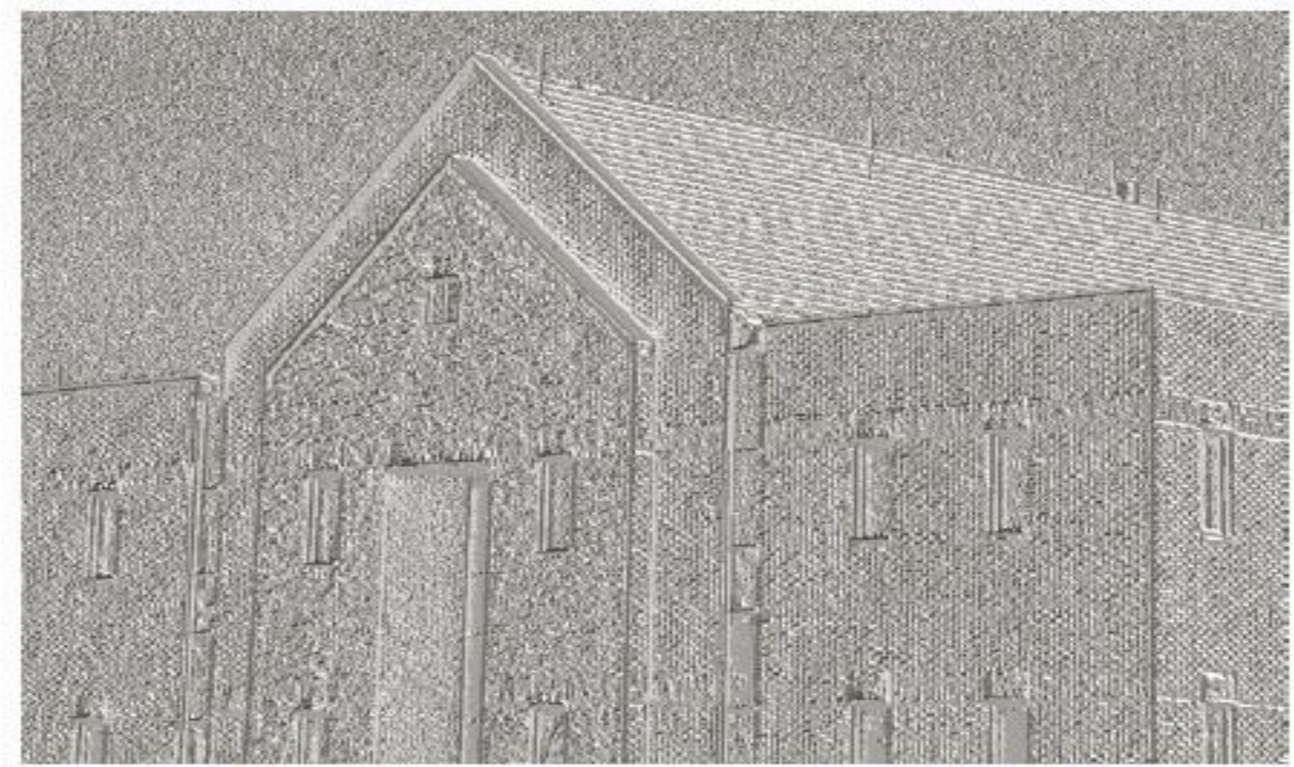


FIGURE 10.17 Gradient angle image computed using Eq. (10.2-11). Areas of constant intensity in this image indicate that the direction of the gradient vector is the same at all the pixel locations in those regions.

Detection of Discontinuities Gradient Operators

- Second-order derivatives: (The Laplacian)
  - The Laplacian of an 2D function *f*(*x*,*y*) is defined as

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

  - Two forms in practice:

FIGURE 10.13 Laplacian masks used to implement Eqs. (10.1-14) and (10.1-15), respectively.

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

Edge detection

- Second derivative operator -Laplacian

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

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

- Generally not used, because it is very **sensetive to noise**
- To **reduce the effect of noise** first image is **smoothened**

Edge detection

- For smoothening purpose **Gaussian** operator is used
- After smoothening Laplacian operator is applied
- This is called as **Laplacian of Gaussian (LoG)**
- This will reduces the effect of noise

Edge detection

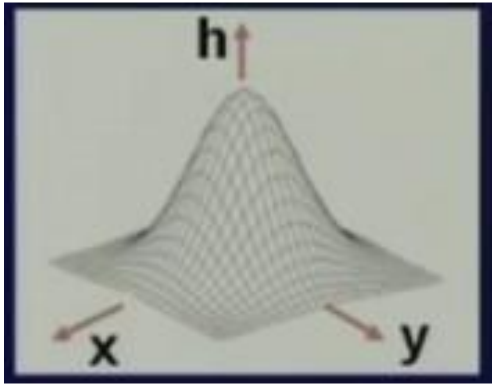
- Gaussian mask:

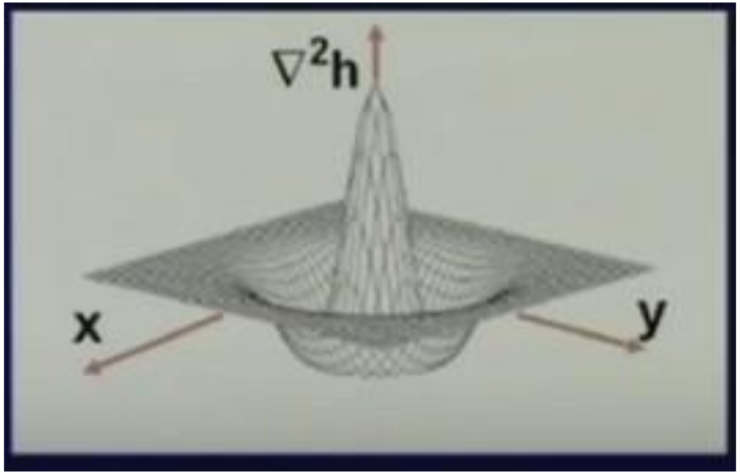
$$h(x,y) = e^{\frac{-x^2 + y^2}{2\sigma^2}}$$

- Laplacian of Gaussian

$$\nabla^2 h(x,y) = \frac{(r^2 - \sigma^2)}{\sigma^4} \cdot e^{\frac{-x^2 + y^2}{2\sigma^2}}$$

Ref: https: //en.wikipedia.org/wiki/Laplace%27s_equation





Edge detection

LoG identifies location
Of edge in image

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

LoG mask

Input Image

Sobel Output Image

LoG Output Image

Local Processing Approach

- Apply edge operator on image to get edge detected image.
- Analyse each pixel in small neighborhood of (x,y)

1. All points similar in nature are linked
2. This forms a boundary of pixels that are similar in nature

- Similarity measures

3. Strength of the response of gradient
4. Direction of gradient

Local Processing Approach

$$(x', y') \in N_{(x,y)}$$

(x', y') and (x, y) are similar if

$$|\nabla f(x,y) - \nabla f(x', y')| < T$$ Strength of gradient at location (x,y) and at location (x',y') must be close

$$|\alpha(x,y) - \alpha(x', y')| < A$$ Angle of gradient at location (x,y) and at location (x',y') must be close

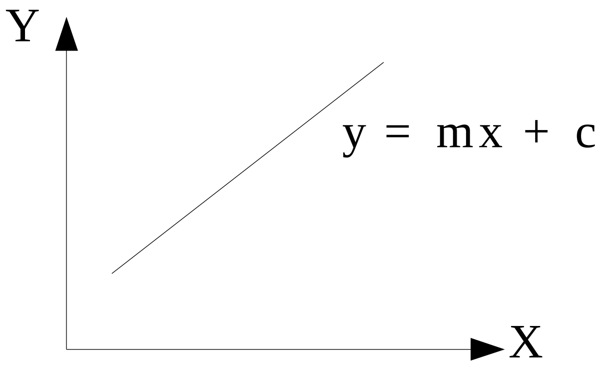
T is gradient strength threshold and Alpha is gradient angle threshold

Global Processing Approach

- Ideally discontinuity detection techniques should idetify pixels lying boundary
- Boundary is a region where transition from low intensity value to high intensity value
- But in practice often these boundary points are not connected due to poor illumination or noise
- Local processing is based on neighborhoods, but it is a very small region
- This may not link the pixels which are far away than neighborhood

Edge linking

- Hough Transform: Mapping of a spatial domain into parameter domain



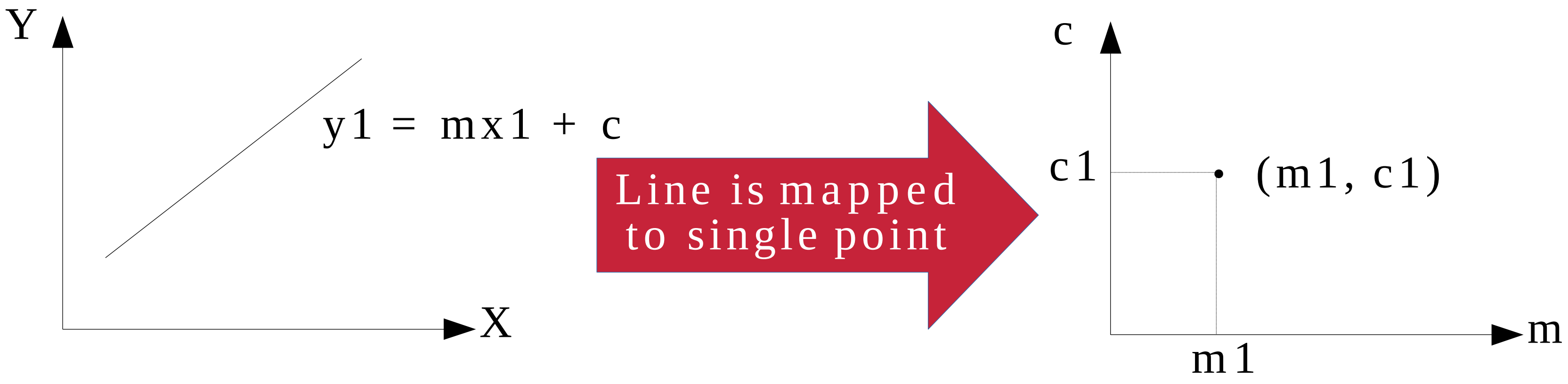
Slope intercept form

For a particular straight line slope and Intercept is constant $y = m1x + c1$

m1 is slope of the line and
c1 is the intercept of the line

Edge linking

- Hough Transform: Mapping of a spatial domain into parameter domain

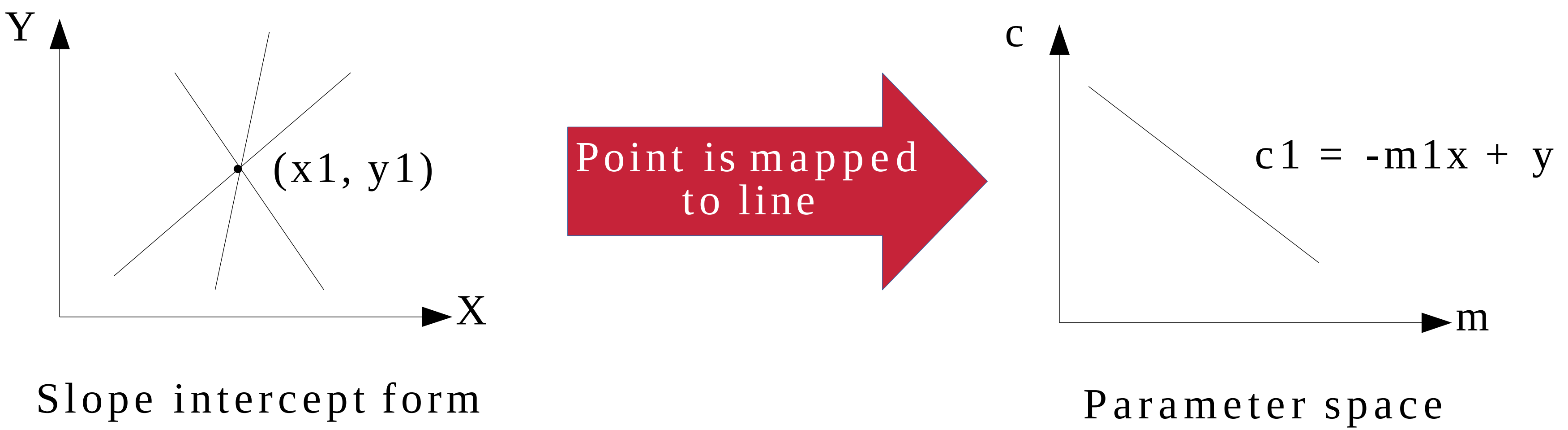


Slope intercept form

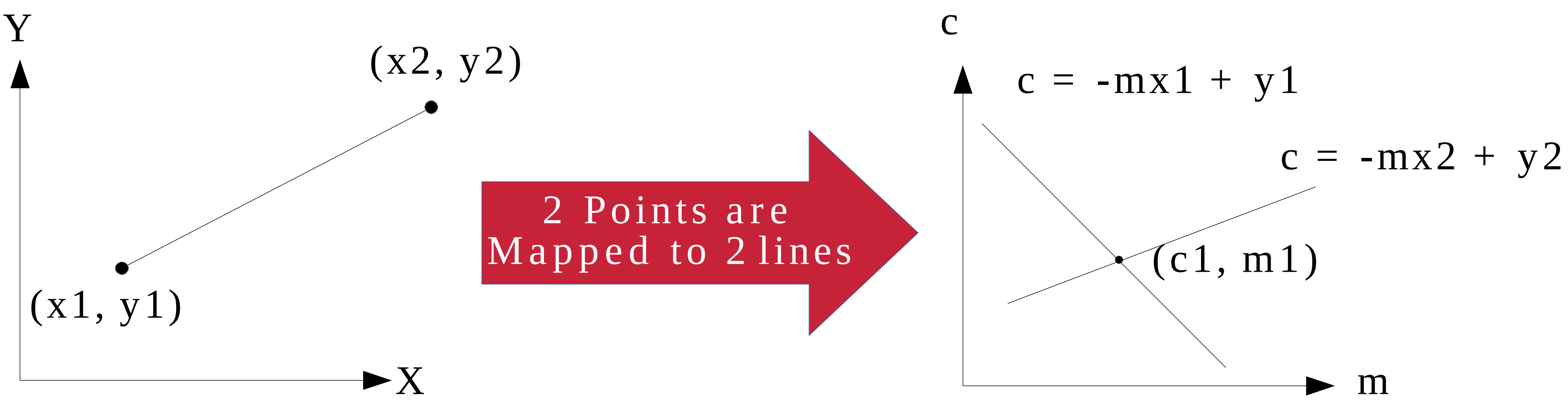
Parameter space

Edge linking

- **Hough Transform:** Mapping of a spatial domain into parameter domain



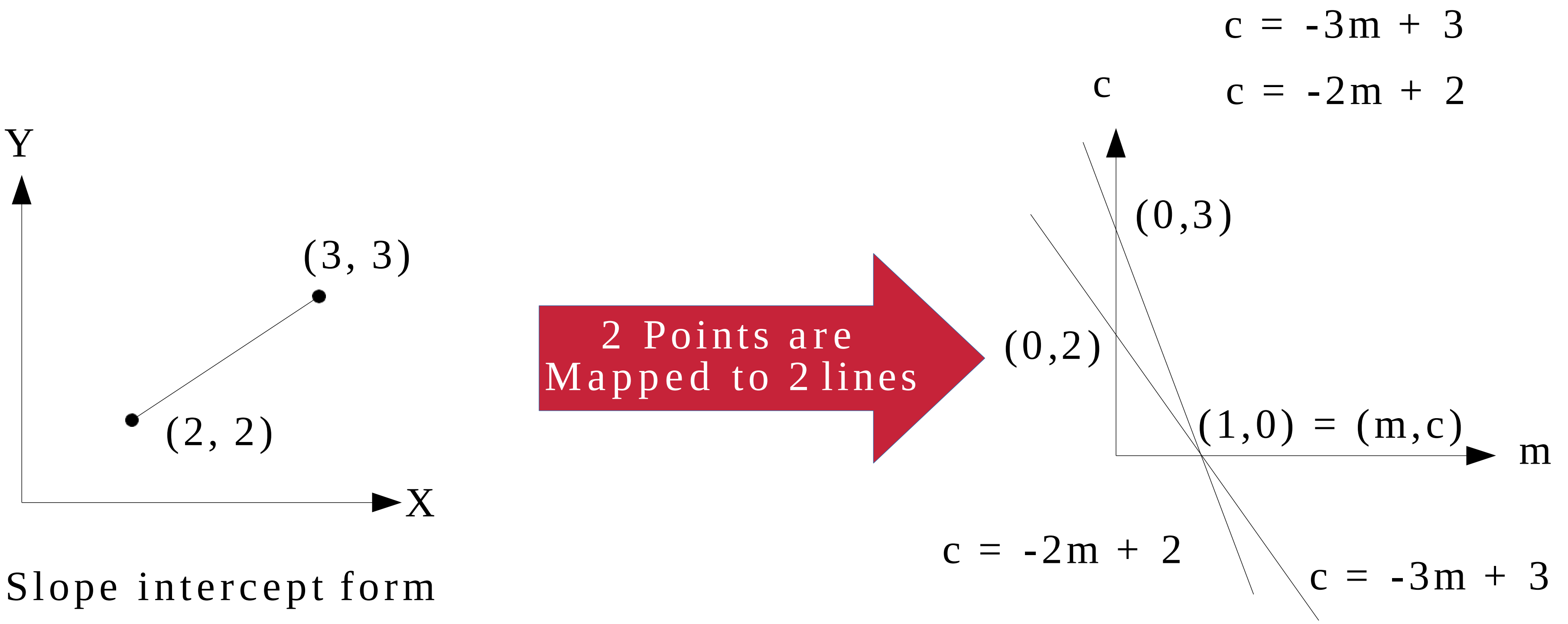
Edge linking



Slope intercept form

Parameter space

Edge linking



Slope intercept form

Edge linking

- Using Hough transform we find whether points are colinear or not.
- Problem:

  Check whether the points (1,1), (2,2), (3,3) are colinear or not

Edge linking Hough Transform Example

- Step 1 Convert points from (x,y) plane to (m,c) plane

  Equation of line is $y = mx + c$

  given points (1,1), (2,2), (3,3)

  1. $(x,y) = (1,1) \Rightarrow 1 = 1m + c \Rightarrow c = -m + 1$

  if $m = 0$ then $c = 1$

  if $c = 0$ then $m = 1$ therefore $(m,c) = (1,1)$

Edge linking Hough Transform Example

- Step 1 Convert points from (x,y) plane to (m,c) plane

  Equation of line is $y = mx + c$

  given points (1,1), (2,2), (3,3)

  2. $(x,y) = (2,2) \Rightarrow 2 = 2m + c \Rightarrow c = -2m + 2$

  if $m = 0$ then $c = 2$

  if $c = 0$ then $m = 1$ therefore $(m,c) = (1,2)$

Edge linking Hough Transform Example

- Step 1 Convert points from (x,y) plane to (m,c) plane

  Equation of line is $y = mx + c$

  given points (1,1), (2,2), (3,3)

  3. $(x,y) = (3,3) \Rightarrow 3 = 3m + c \Rightarrow c = -3m + 3$

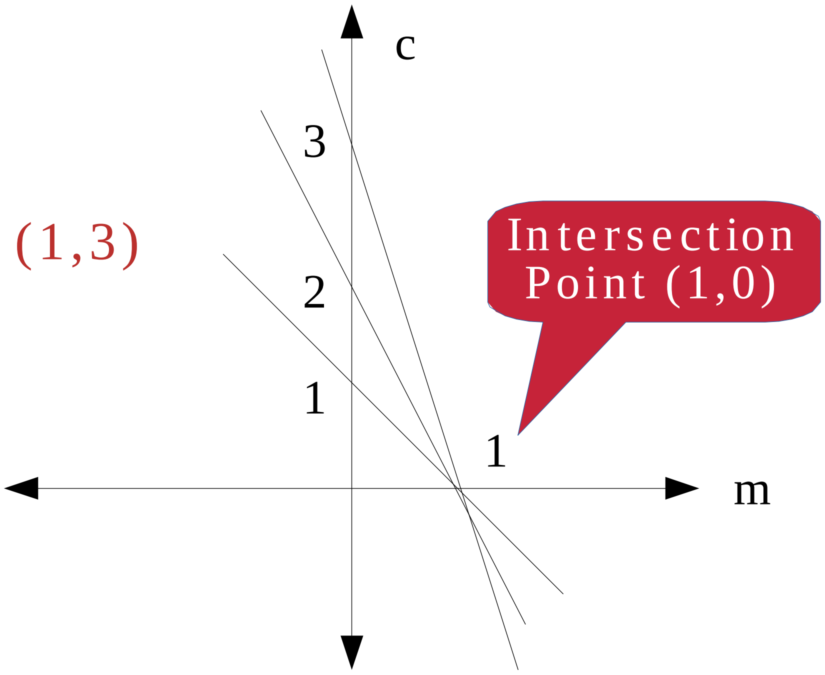
  if $m = 0$ then $c = 3$

  if $c = 0$ then $m = 1$ therefore $(m,c) = (1,3)$

Edge linking Hough Transform Example

- Step 2: Using (m,c) points draw lines in m-c plane

Points are
(1,1), (1,2), (1,3)



All lines intersect at
$(m,c) = (1,0)$

Edge linking Hough Transform Example

- Step 3: Put the value of $(m,c) = (1,0)$ in the original equation of line

  $y = mx + c \Rightarrow y = x$

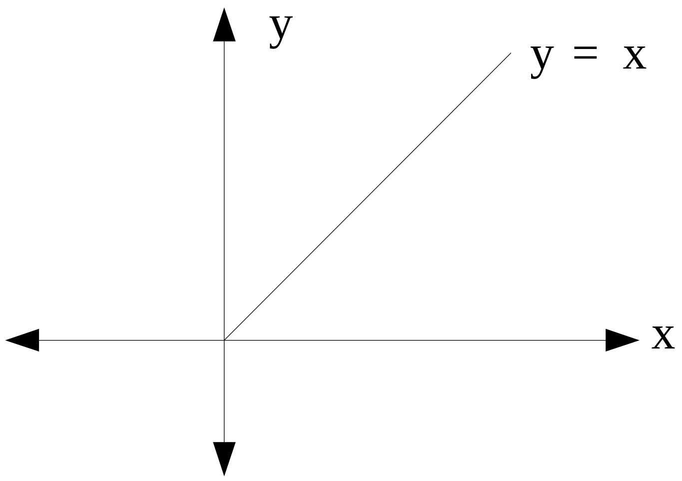
- Step 4: Check the original points (from x-y plane) satisfy the equation $y = x$

  (1,1), (2,2), (3,3) points satisfy the equation hence these points are colinear

Edge linking Hough Transform Example

Step 4: Check the original points (from x-y plane) satisfy the equation $y = x$

(1,1), (2,2), (3,3) points satisfy the equation hence these points are colinear



Hough Transform

- Advantages:

1)Conceptually simple technique.
2)Handles missing occluded data gracefully.
3) Can be adapted for many other forms.

- Disadvantages:

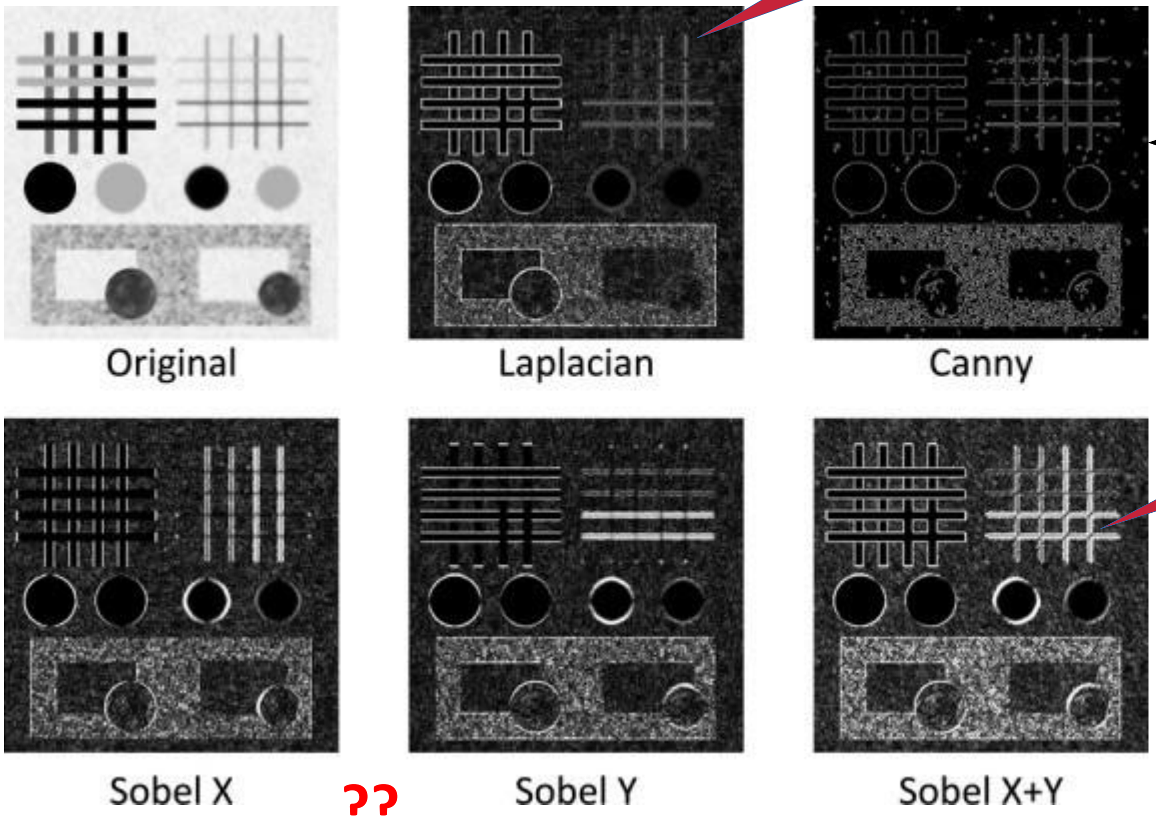
1)Large storage space required.
2)Checks for only one type of object.

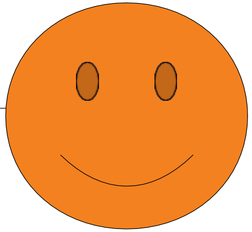
Although it is the commonly preferred method for lines & circle detection, the HT in general has several limitations making it challenging to detect anything other than lines and circles. This is especially the case when more parameters are needed to describe shapes, this add more complexity.

Canny Edge detection

Thick edges and noise

Optimal





Thick edges

??

Canny Edge Detection Algorithm

- Smoothing: Blurring of the image to remove noise.
- Finding gradients: The edges should be marked where the gradients of the image has large magnitudes.
- Non-maximum suppression: Only local maxima should be marked as edges.
- Double thresholding: Potential edges are determined by thresholding.
- Edge tracking by hysteresis: Final edges are determined by suppressing all edges that are not connected to a very certain (strong) edge.

Canny Edge Detection Algorithm

1. Smoothing: Blurring of the image to remove noise.

- Eg. Apply Gaussian filter to smoothen the image

$$B = \frac{1}{159} \cdot \begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$


(a) Original


(b) Smoothed

Canny Edge Detection Algorithm

2. Finding gradients: To find the edges use sobel operators in X and Y direction G_x and G_y

$$K_{GX} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$K_{GY} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Calculate strength of the gradient G

$$|G| = \left[G_x^2 + G_y^2\right]^{\frac{1}{2}}$$

Canny Edge Detection Algorithm

2. Finding gradients: Output showing Gradient magnitudes in image i.e edges inimage

(a) Smoothed (b) Gradient magnitudes

Thick edges

Edges are very thick and broad
Hene exact location isunknown!!

Canny Edge Detection Algorithm

2. Finding gradients angle: To find the exact location of the edge angle of gradient is important

Calculate angle of the gradient G using

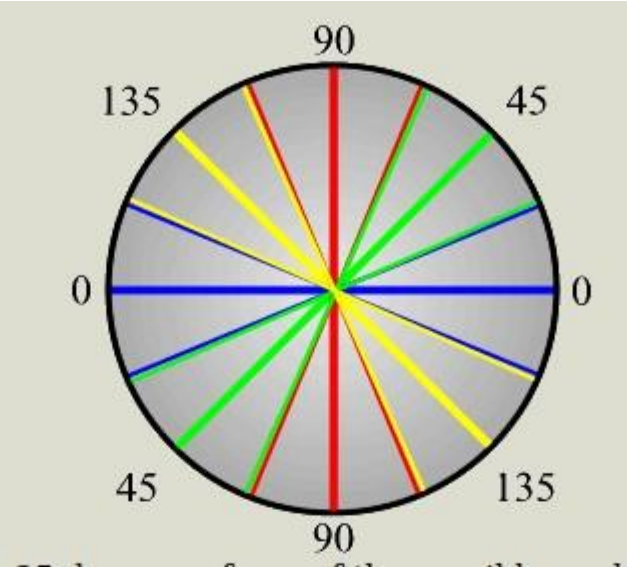
$$\theta = \tan^{-1}\frac{|G_y|}{|G_x|}$$

This angle information is used in non maximal suppression

Canny Edge Detection Algorithm

3. Non-maximum suppression: The purpose of this step is to convert the “blurred” edges in the image of the gradient magnitudes to “sharp” edges

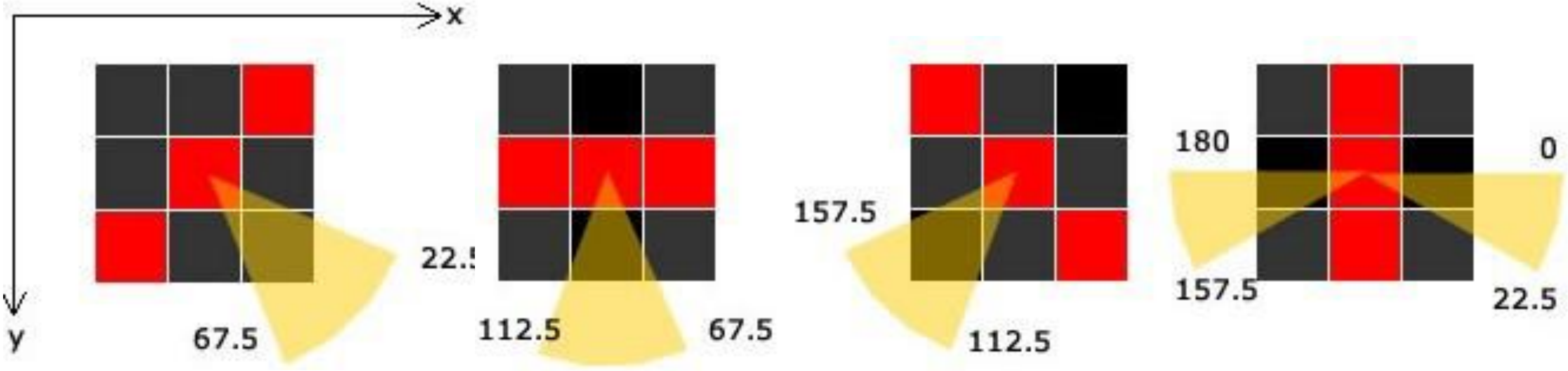
- Preserve local maxima in gradient image and remove everything else.
- Approximate angle of gradient at every pixel by its nearest 45 degree multiple



Canny Edge Detection Algorithm

3. Non-maximum suppression: Compare the pixels in +ve and -ve direction of gradient

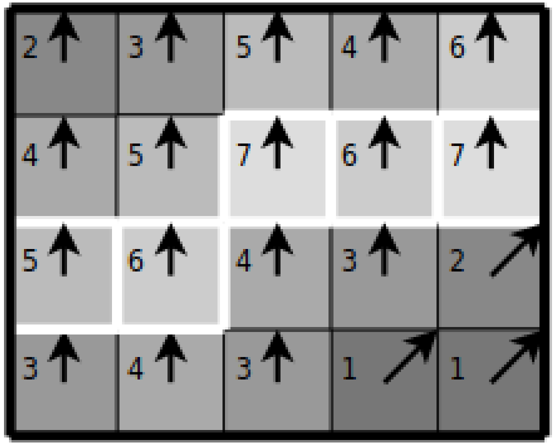
- If maximum and its magnitude is greater than the upper threshold then mark it as a edge pixel

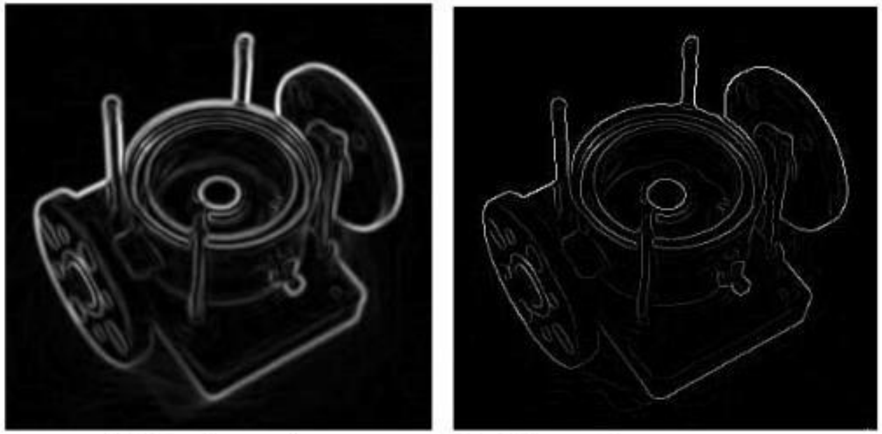


Canny Edge Detection Algorithm

3. Non-maximum suppression: Compare the pixels in +ve and -ve direction of gradient

- If maximum and its magnitude is greater than the upper threshold then mark it as a edge pixel





(a) Gradient values (b) Edges after non-maximum suppression

Canny Edge Detection Algorithm

4. Double Thresholding: Many of the edges in the image after non maximal suppression are true but some are false due to noise inimage

- Edge pixels stronger than the high threshold are marked as strong edge pixels; weaker than the low threshold are suppressed and edge pixels between the two thresholds are marked as weak.

Canny Edge Detection Algorithm

5. Edge tracking by hysteresis: Weak edges are kept only if they are connected to strong edges

(a) Double thresholding (b) Edge tracking by hysteresis (c) Final output

Segmentation using Similarity Based Approach

- Similarity based segmentation
- Thresholding

  1. Global thresholding
  2. Dynamic or adaptive threshold
  3. Optimal threshold
  4. Local thresholding

- Region growing technique
- Region splitting and merging technique

Only local and global are to be reviewed

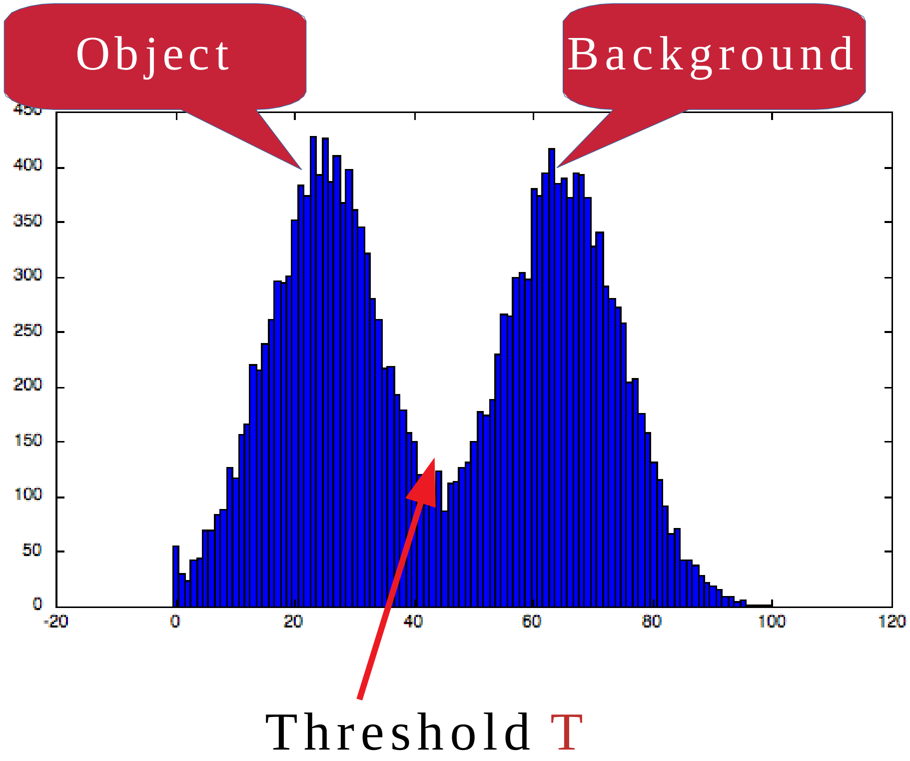
Similarity based Segmentation

Thresholding

- Suppose an image f(x,y) is having a dark object against bright background
- Such image generate bimodal histogram

$$f(x,y) < T \quad implies \quad object$$

$$f(x,y) \geq T \quad implies \quad background$$

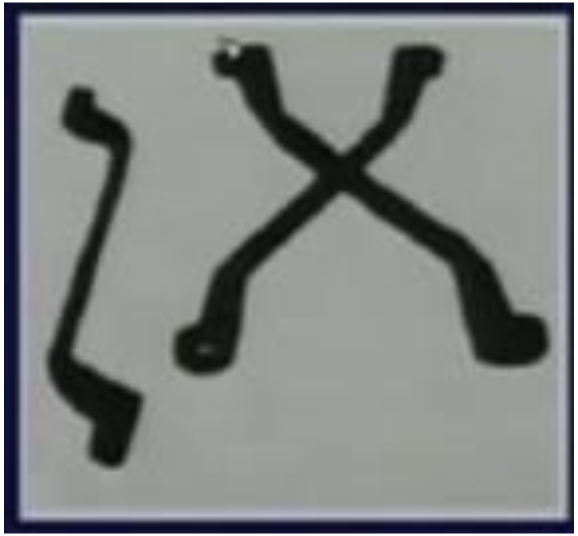


Similarity based Segmentation

- Thresholding example

Input image

Thresholding



Segmented output image

Similarity based Segmentation

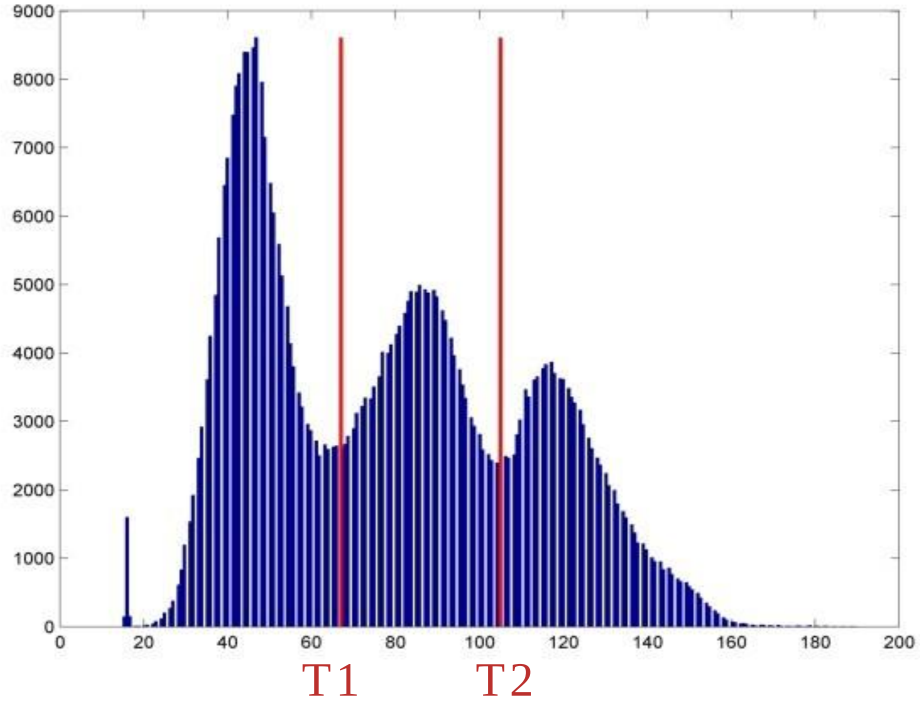
Thresholding

Trimodal histogram

$$f(x,y) > T_2 \;\; implies \;\; object\,2$$

$$T_1 < f(x,y) \le T_2 \;\; implies \;\; object\,1$$

$$f(x,y) < T_1 \;\; implies \;\; background$$



Similarity based Segmentation

- Selection of threshold value
- Thresholding function which test the image against threshold value

$$T = T[x, y, p(x, y), f(x, y)]$$

Depending on combination
Of these three values
Thresholding can be
1. Local
2. Global
3. Adaptive
4. Optimal

(x,y) = coordinates of the pixels

f(x,y) = intensity value of the pixels

p(x,y) = local property in neighborhood
Centered at (x,y)

Similarity based Segmentation

- If T[f(x,y)] then it is global thresholding
- If T[f(x,y), p(x,y)] then it is local thresholding
- If T[(x,y), f(x,y), p(x,y)] then it is Adaptive thresholding

$$g(x,y) = 0 \;\; if \;\; f(x,y) > T$$ Object pixel

$$g(x,y) = 1 \;\; if \;\; f(x,y) \leq T$$ Background pixel

Similarity based Segmentation

- How to find threshold value?
- Every time looking at the histogram and deciding is not feasible
- Need some automated process for threshold selection

Similarity based Segmentation

- Automatic global thresholding:

1. Initialize value of threshold T
2. Perform segmentation using threshold value T to get two regions G1 and G2

Pixel intensity values from group G1 and G2 are similar but two groups different

Similarity based Segmentation

- Automatic global thresholding:

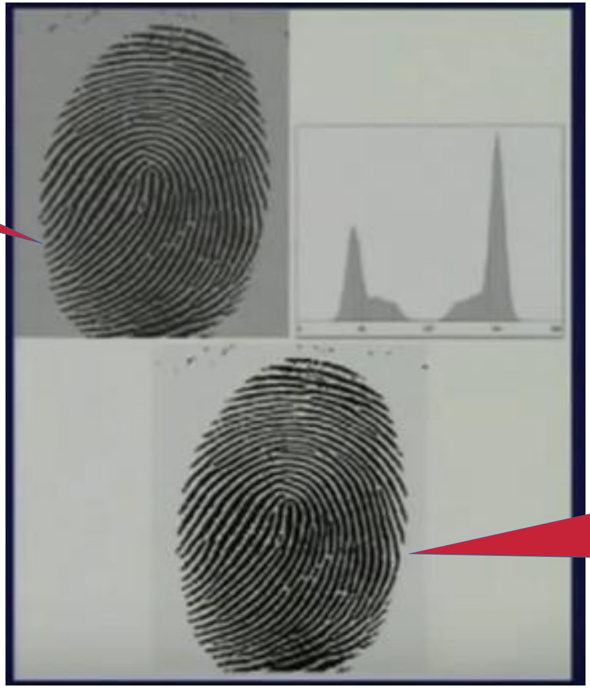
1. Initialize value of threshold T
2. Perform segmentation using threshold value T to get two regions G1 and G2
3. Compute mean M1 and M2 using pixel intensity values of G1 and G2
4. New threshold value T = (M1 + M2)/2
5. If (|Ti – T(i+1)| < = T’) then STOP
6. else goto step 2

T’ is some tolarance value

Similarity based Segmentation

- Automatic global thresholding example:

Input Image



Output Image after application of Automatic global thresholding

Similarity based Segmentation

- Automatic global thresholding:

  Gobal thresholding gives very good results if intensity of illumination is uniform

- In such case getting a global threshold value is difficult

Similarity based Segmentation

Automatic global thresholding for poor illuminated image

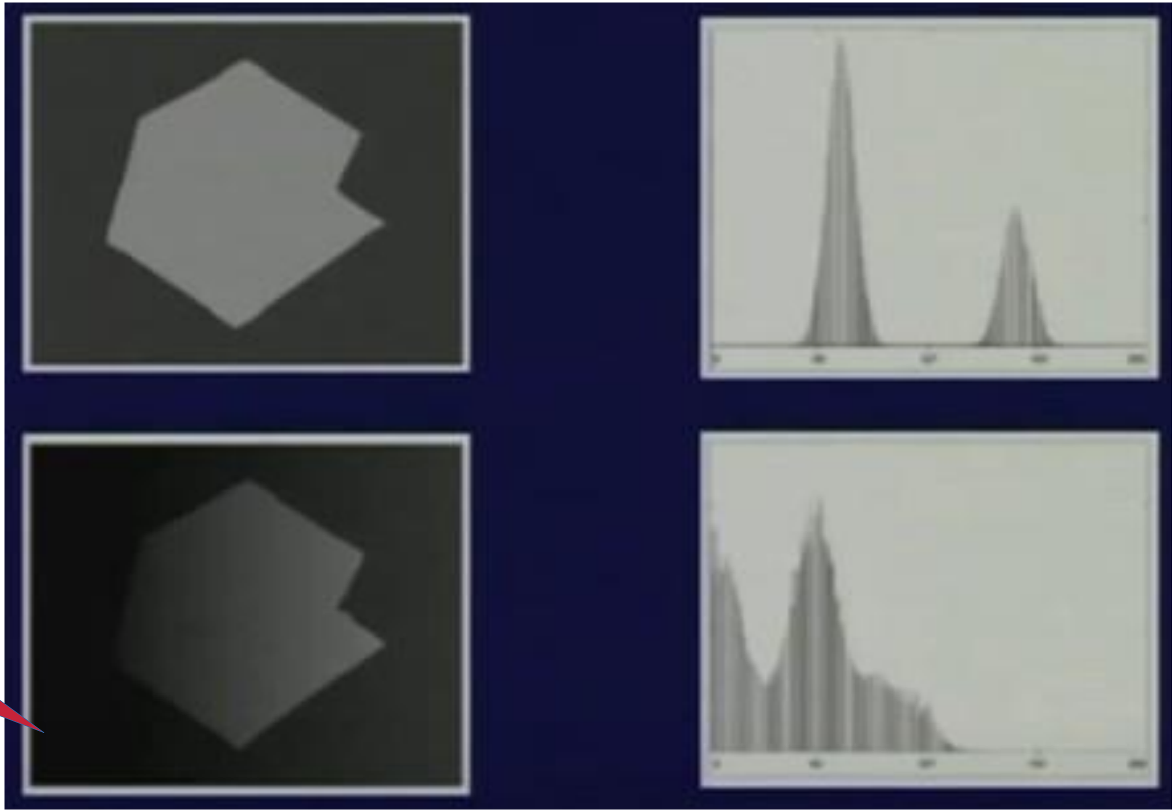


Image in non uniform illumination source

Similarity based Segmentation

Automatic global thresholding for poor illuminated image

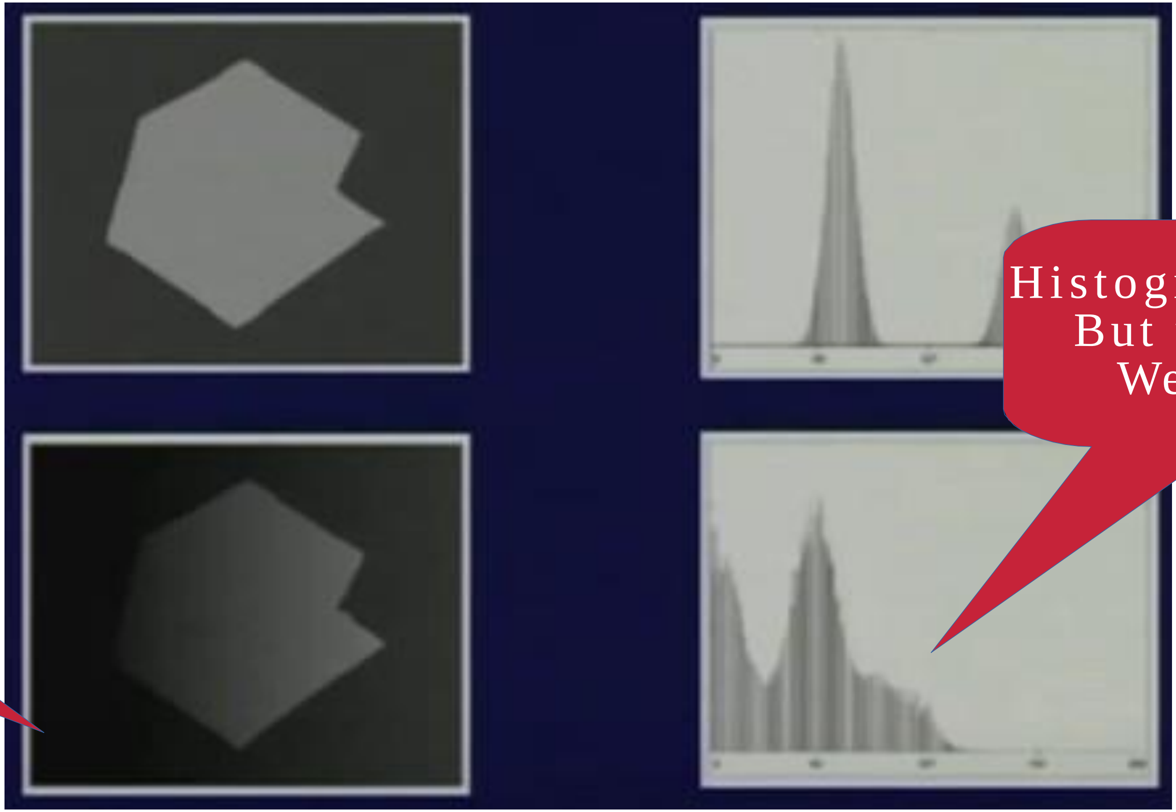
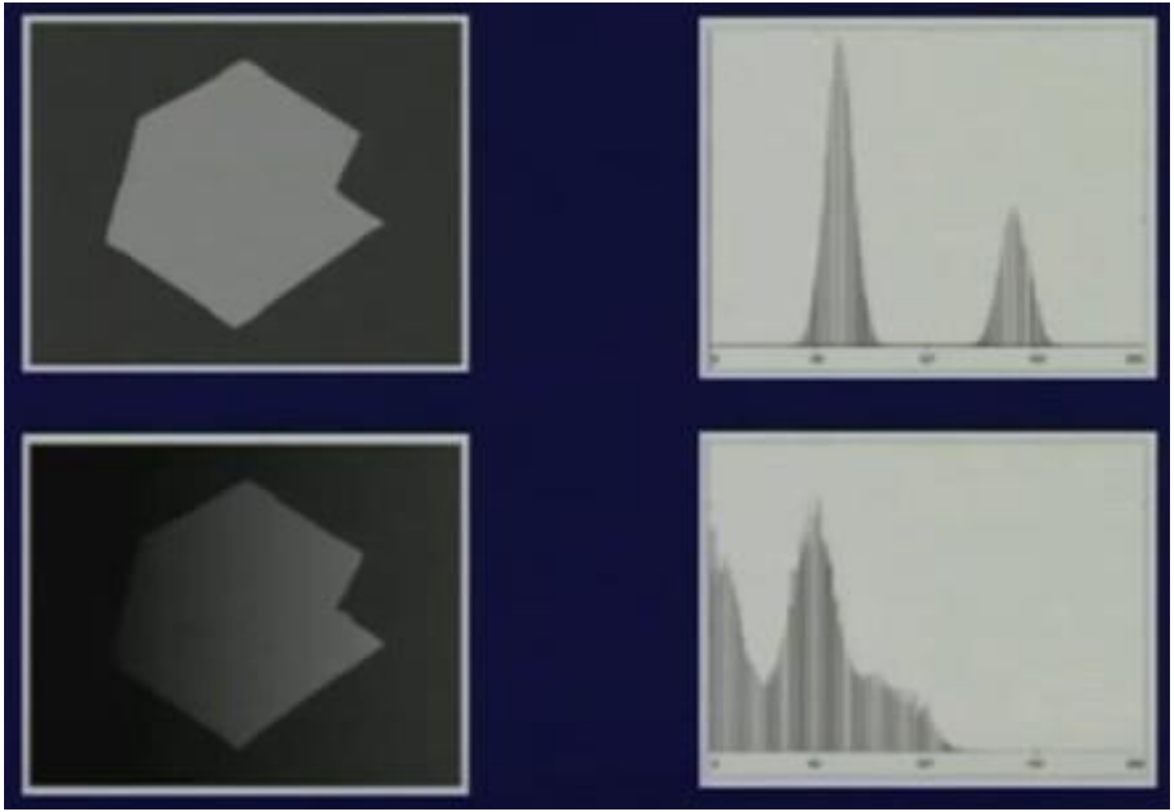


Image in non uniform illumination source

Histogram is bimodal But valley is not Well defined

Similarity based Segmentation

Automatic global thresholding for poor illuminated image



Automatic global thresholding
is likely to fail!!!

Similarity based Segmentation

- Solution for poor illumination problem of global thresholding

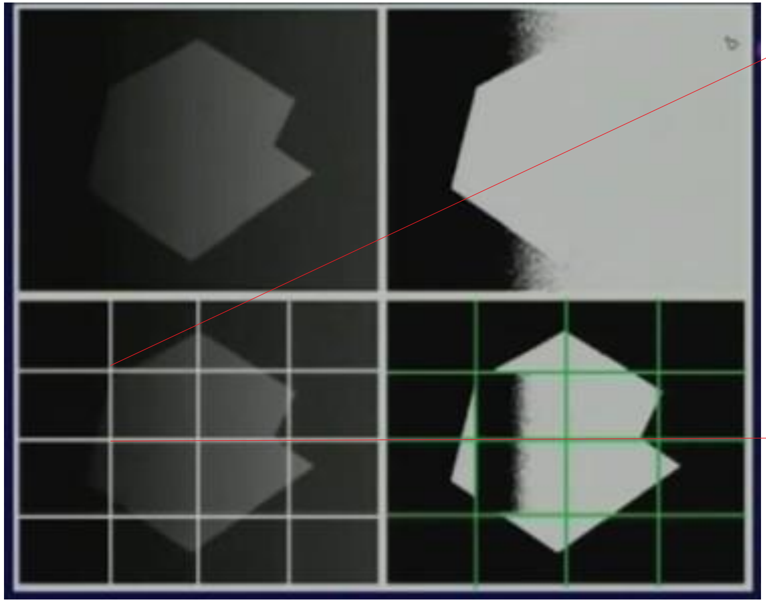
  Divide the image into number of sub-images so that illumination can be uniform
- Find the global threshold value for sub-images
- Union of all threshold values gives the final threshold
- This is called as an adaptive thresholding as the threshold value is depend on the location in image

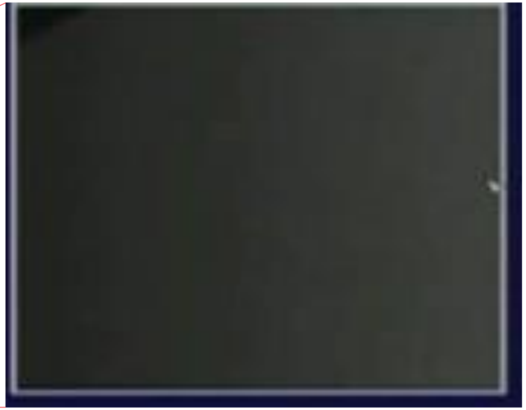
Similarity based Segmentation

- Solution for poor illumination problem of global thresholding

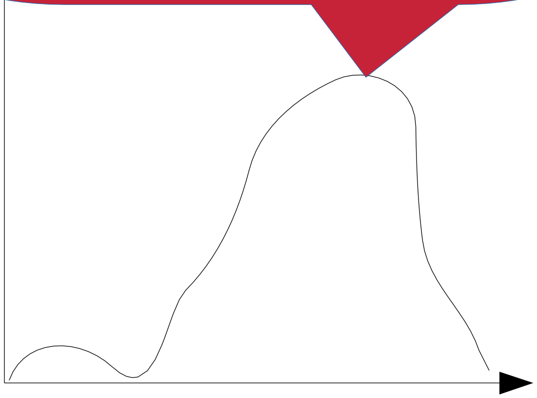
Similarity based Segmentation

- Solution for poor illumination problem of global thresholding





Threshold is dominated By this region



Similarity based Segmentation

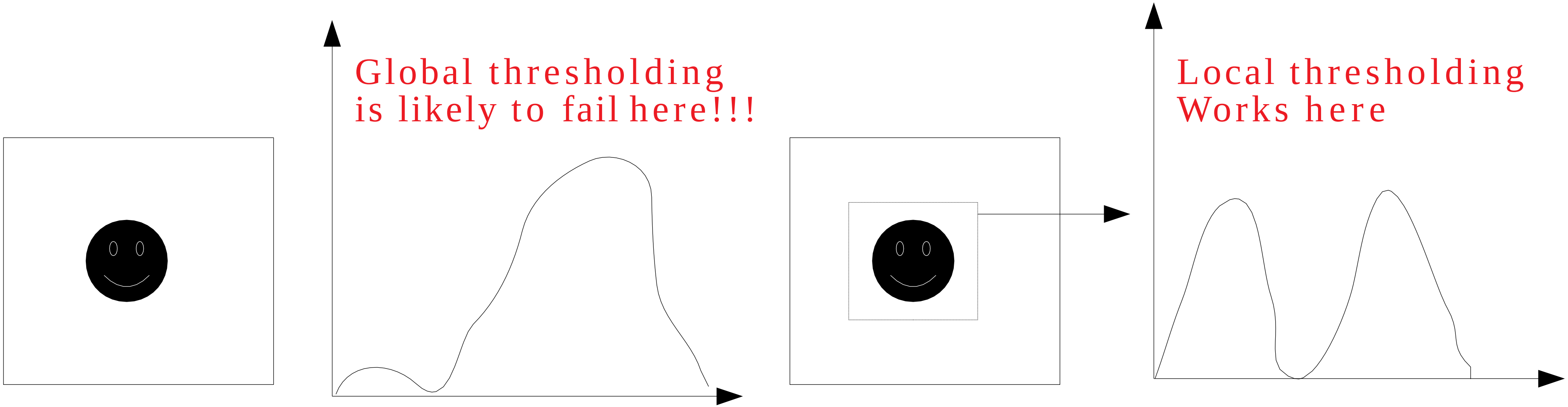
- Local Thresholding

Solution:
Consider local image and apply the thresholding

This converts the histogram into bimodal
and equally distributed in object and background
region

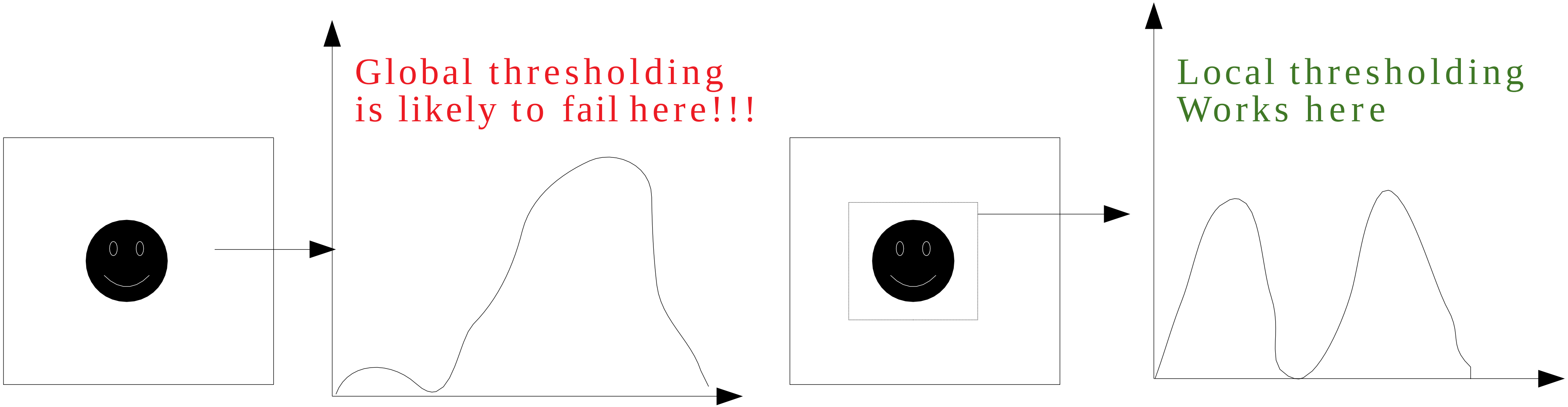
Similarity based Segmentation

- Local Thresholding



Similarity based Segmentation

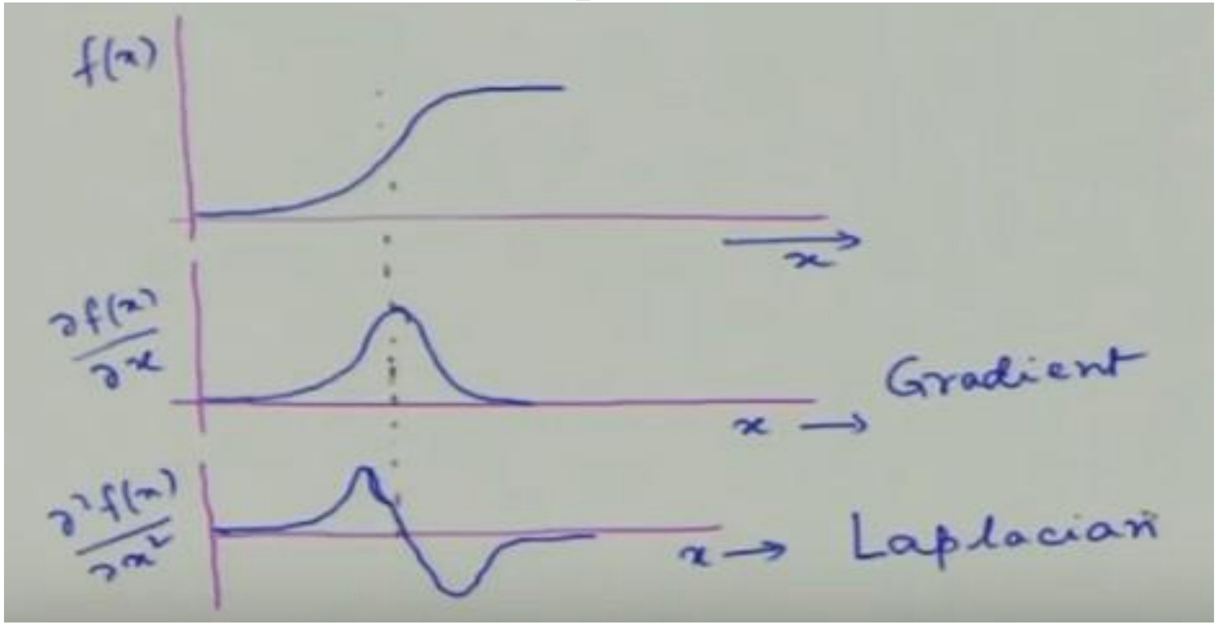
- Local Thresholding



BUT HOW TO DETECT LOCAL BOUNDARY BETWEEN OBJECT AND BACKGROUND?

Similarity based Segmentation

- Local Thresholding – Boundary detection using Gradient and Laplacian

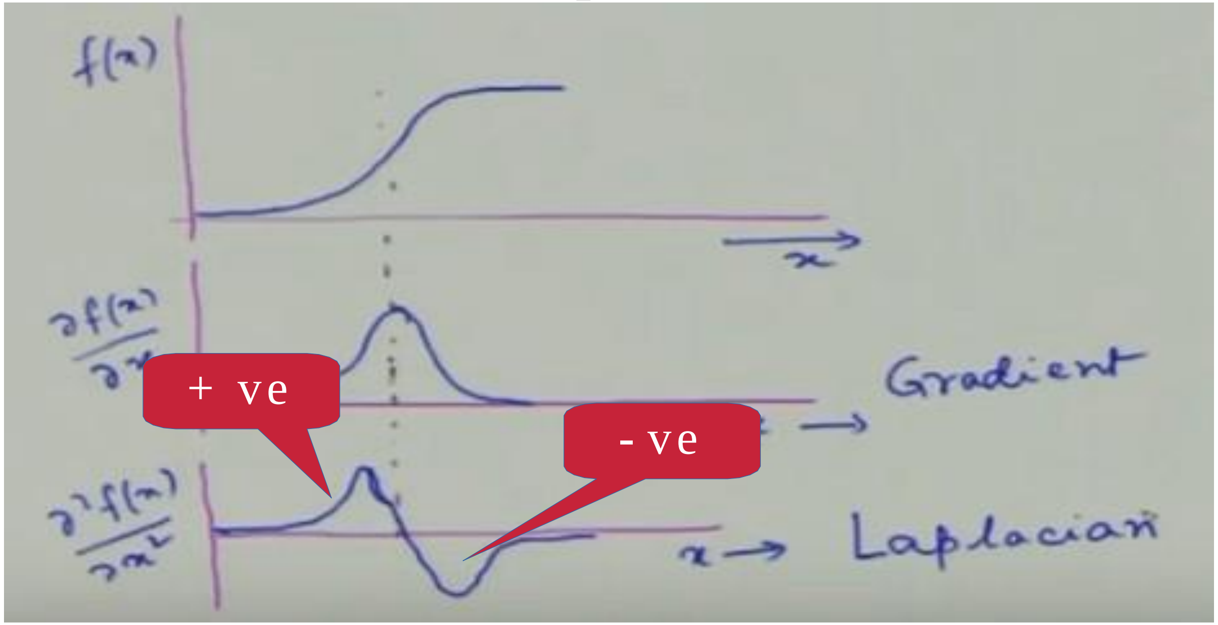


Gradient gives position of The edge whereas

Laplacian Determines whether point Lies on darker side or brighter Side of the edge

Similarity based Segmentation

- Local Thresholding – Boundary detection using Gradient and Laplacian



Gradient gives position of The edge whereas

Laplacian Determines whether point Lies on darker side or brighter Side of the edge

Similarity based Segmentation

- Local Thresholding – Boundary detection using Gradient and Laplacian

$$using\ \ three\ \ properties\ \ f(x,y)\ \ \nabla f(x)\ \ \nabla^2 f(x,y)$$

$$s(x,y) = 0 \ \ if \ \ \nabla f(x,y) < T$$ Not belong to boundary

$$= +ve \ \ if \ \ \nabla f(x,y) \geq T \quad \nabla^2 f(x,y) \geq 0$$ Belongs to Object

$$= -ve \ \ if \ \ \nabla f(x,y) \geq T \quad \nabla^2 f(x,y) < 0$$ Belongs to Background

Similarity based Segmentation

Local Thresholding –
Boundary detection using Gradient and Laplacian

This image is used to
Findout object region
And background image

Similarity based Segmentation

Local Thresholding –
Boundary detection using Gradient and Laplacian

Scan each row of the image from left to right

- (- , +) indicates transition from background to object
- (+, -) indicates transition from object to background

- (......***)(-, +)(0 or +)(+, -)(***.....)

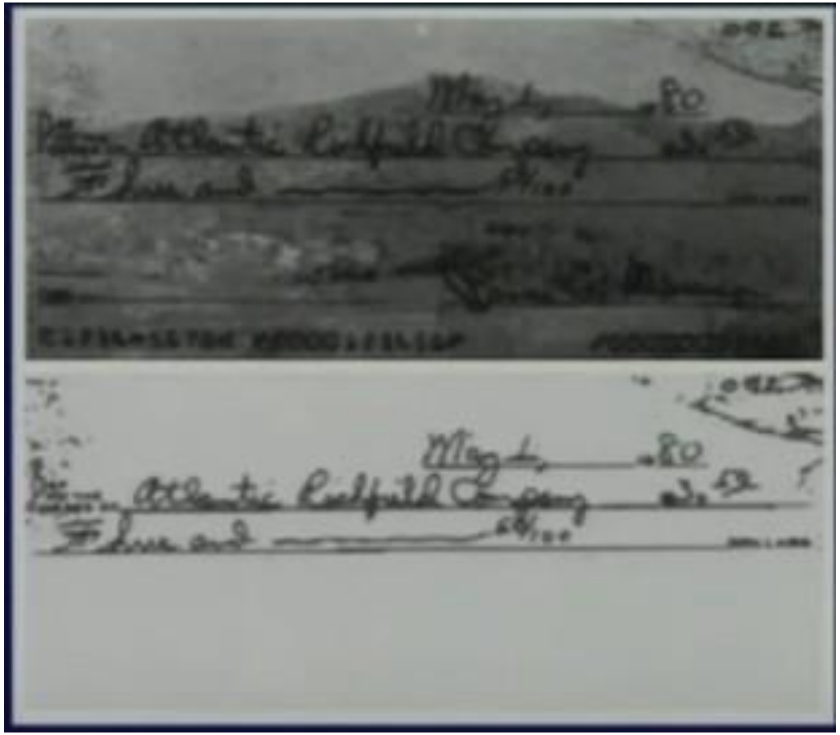
Similarity based Segmentation

Local Thresholding –
Boundary detection using Gradient and Laplacian

(……***)(-, +)(0 or +)(+, -)(***…..)

Object

Any combination of
-, + or 0



Region-Based Segmentation

- Edges and thresholds sometimes do not give good results for segmentation.
- Region-based segmentation is based on the connectivity of similar pixels in a region.
  - Each region must be uniform.
  - Connectivity of the pixels within the region is very important.
- There are two main approaches to region-based segmentation: region growing and region splitting.

Region-Based Segmentation
Basic Formulation

Let R represent the entire image region.

- Segmentation is a process that partitions R into subregions,
- $R_1, R_2, \ldots, R_n$, such that

$$\text{(a)} \quad \bigcup_{i=1}^{n} R_i = R$$

$$\text{(b)} \quad R_i \text{ is a connected region, } i = 1,2,\ldots,n$$

$$\text{(c)} \quad R_i \cap R_j = \phi \quad \text{for all } i \text{ and } j, i \neq j$$

$$\text{(d)} \quad P(R_i) = \text{TRUE for } i = 1,2,\ldots,n$$

$$\text{(e)} \quad P(R_i \cup R_j) = \text{FALSE} \quad \text{for any adjacent regions } R_i \text{ and } R_j$$

where $P(R_k)$: a logical predicate defined over the points in set R_k

For example: $P(R_k)$=TRUE if all pixels in R_k have the same gray level.

Region Growing

- Thresholding still produces isolated image
- Region growing algorithms works on **principle of similarity**

•**It states that a region is coherent if all the pixels of that region are homogeneous with respect to some characteristics such as colour, intensity, texture, or other statistical properties**

•Thus idea is to pick a pixel inside a region of interest as a starting point (also known as a **seed point**) and allowing it to grow

•**Seed point is compared with its neighbours, and if the properties match , they are merged together**

•This process is repeated till the regions converge to an extent that no further merging is possible

Region Growing Algorithm

- It is a process of grouping the pixels or subregions to get a bigger region present in an image

- **Selection of the initial seed:** Initial seed that represent the ROI should be given typically by the user. Can be chosen automatically. The seeds can be either single or multiple

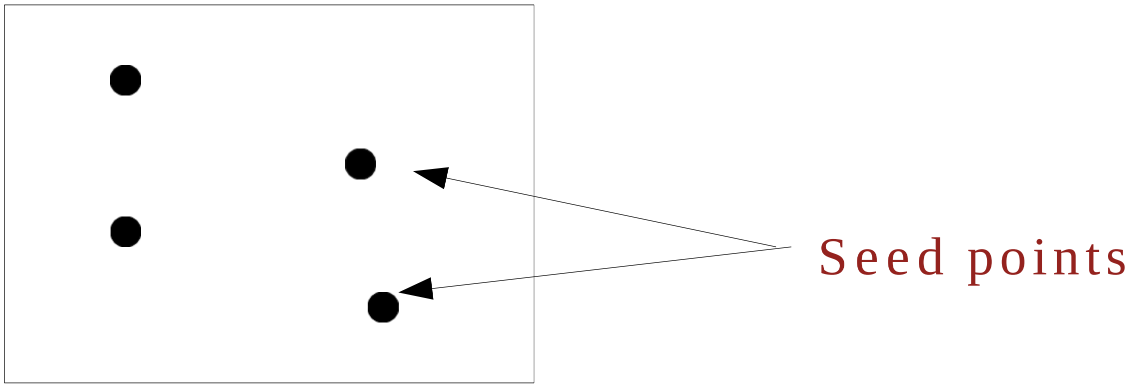
- **Seed growing criteria:** Similarity criterion denotes the minimum difference in the grey levels or the average of the set of pixels. Thus, the initial seed 'grows' by adding the neighbours if they share the same properties as the initial seed

- **Terminate process:** If further growing is not possible then terminate region growing process

Similarity based Segmentation

- Region growing segmentation

  Grouping the pixels to make larger subregions such that properties of newly added pixels must be same

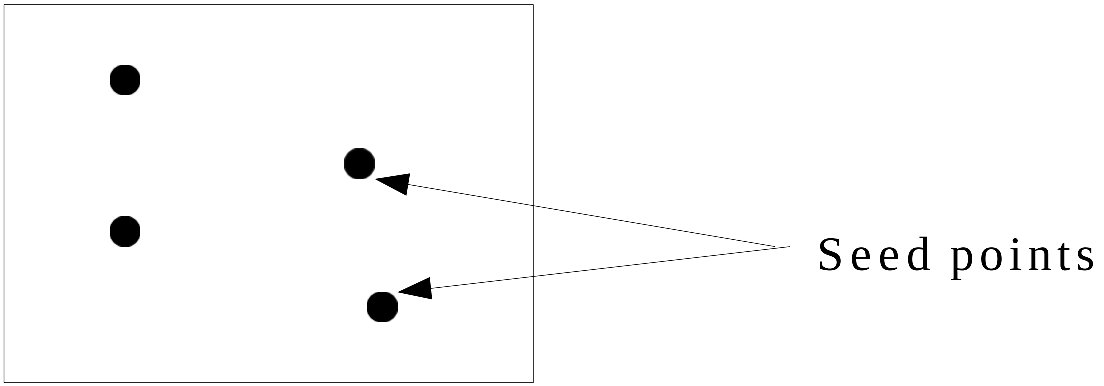


Similarity based Segmentation

- Region growing segmentation

  Make 3X3 neighborhood (4, 8 or m-connected) around seed point and check for the intensity values

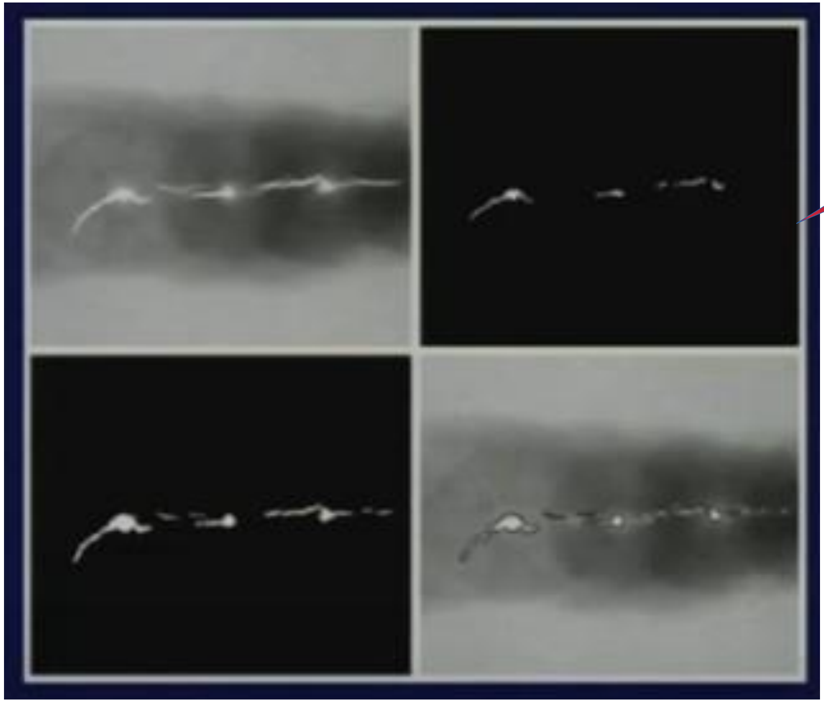
If difference in intensity values is very large then Don’t include in the set



Similarity based Segmentation

- Region growing segmentation

X-Ray image of
Welded part



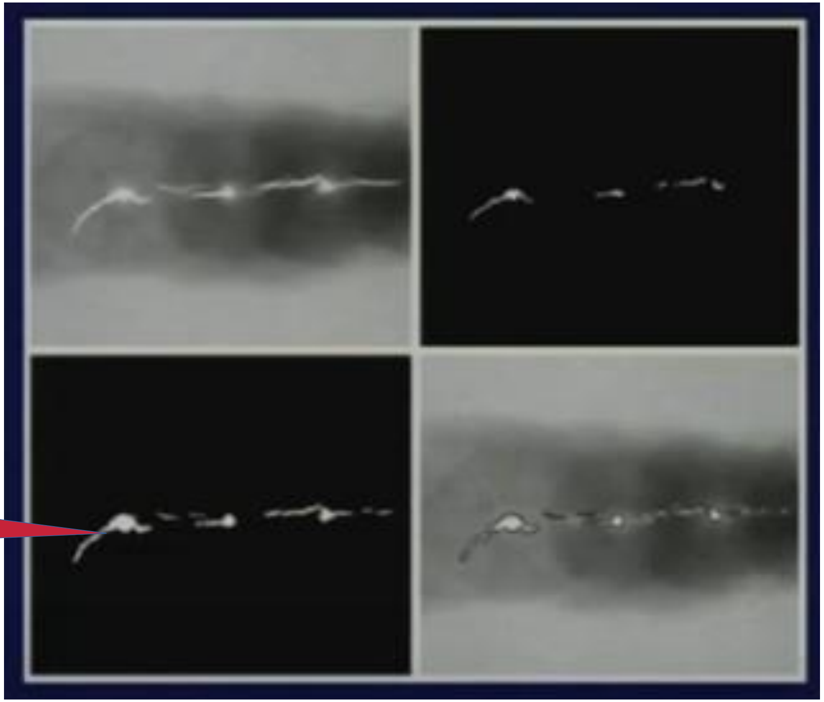
Thresholding
Output

Cracks are indicated
Near 255 value

Similarity based Segmentation

- Region growing segmentation

X-Ray image of
Welded part



Select seed points
With value 255

Perform region
Growing operations
On each seed point
In Original Image

Region Growing Algorithm

- Consider image shown in figure:

1	0	7	8	7
0	1	8	$\underline{9}$	8
0	0	7	9	8
0	$\underline{1}$	8	8	9
1	2	8	8	9

•Assume seed point indicated by underlines. Let the seed pixels 1 and 9 represent the regions C and D, respectively

•Subtract pixel from seed value

•If the difference is less than or equal to 4 (i.e. T=4), merge the pixel with that region. Otherwise, merge the pixel with the other region.

Split and Merge Algorithm

- Region growing algorithm is slow
- So seed point can be extended to a seed region
- Instead of a single pixel, a node of a Regional adjacency graph (RAG) a region itself is now considered as a starting point.

- The split process can be stated as follows:

1)Segment the image into regions R1, R2,….Rn using a set of thresholds
2)Create RAG. Use a similarity measure and formulate a homogeneity test
3)The homogeneity test is designed based on the similarity criteria such as intensity or any image statistics
4)Repeat step 3 until no further region exits that requires merging

Split and Merge Algorithm

-

2	2	2	2	2
2	8	8	8	1
2	8	8	8	1
2	1	1	1	1
2	1	1	1	1

2	2	2	2	2
2	8	8	8	1
2	8	8	8	1
2	1	1	1	1
2	1	1	1	1

8	8	8
8	8	8

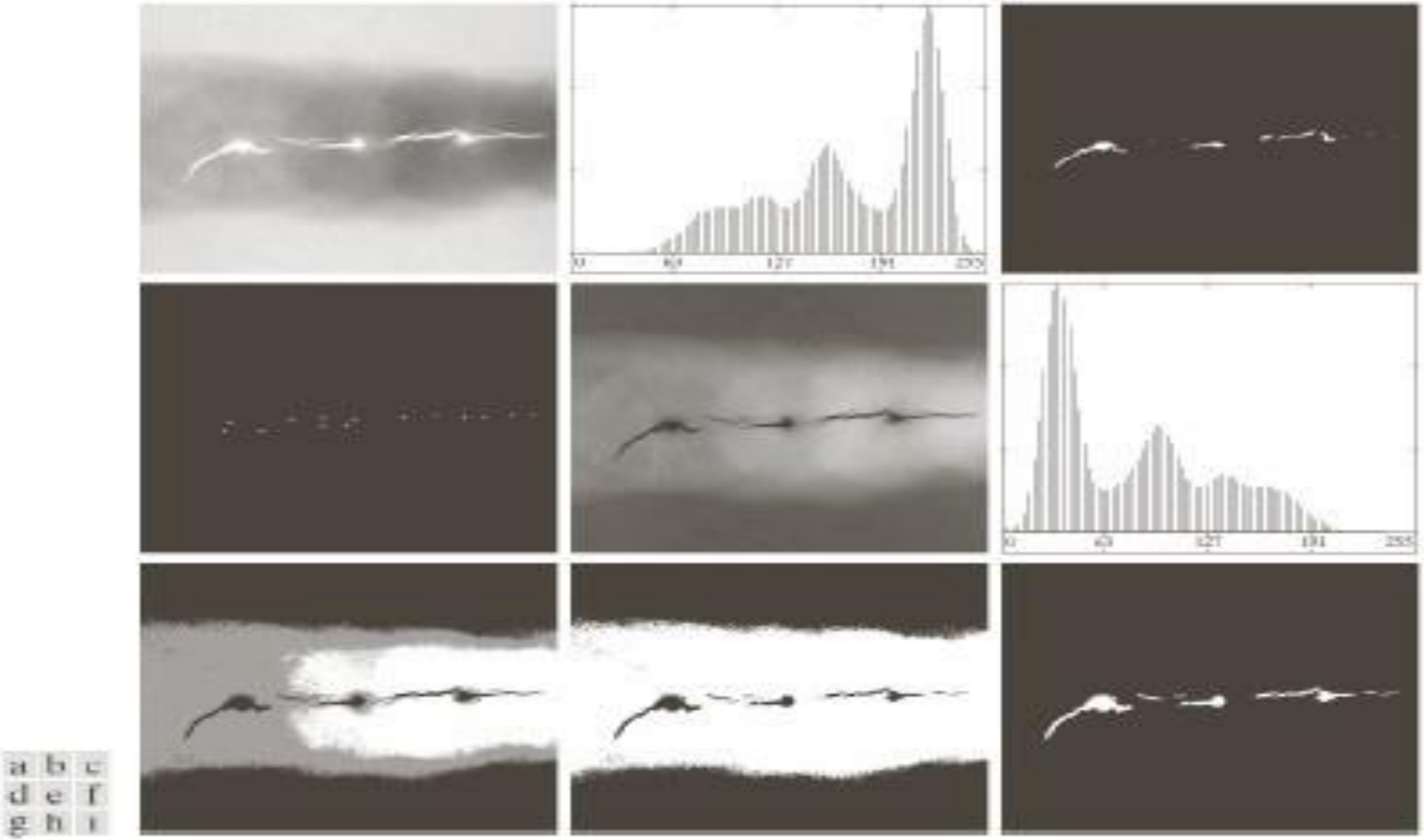
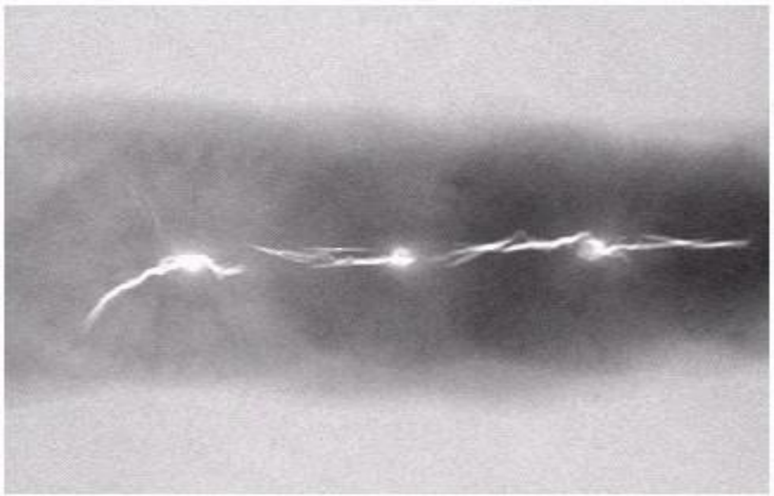
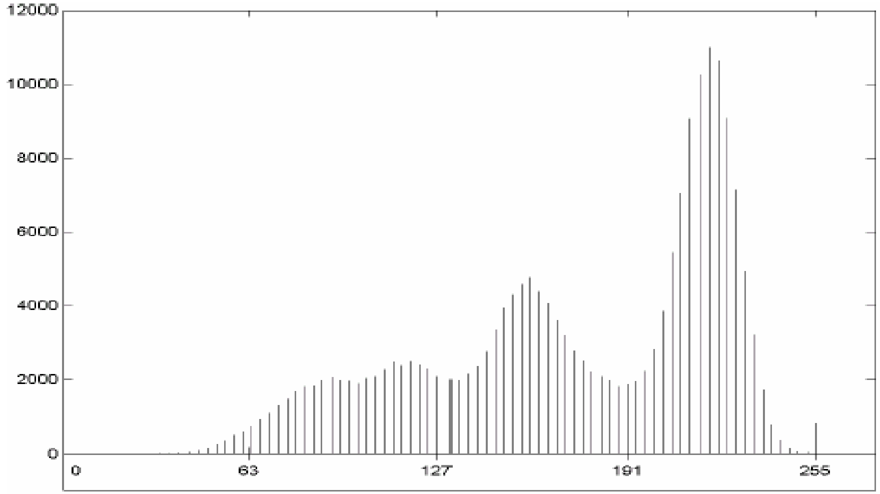


FIGURE 10.51 (a) X-ray image of a defective weld. (b) Histogram. (c) Initial seed image. (d) Final seed image (the points were enlarged for clarity). (e) Absolute value of the difference between (a) and (c). (f) Histogram of (e). (g) Difference image thresholded using dual thresholds. (h) Difference image thresholded with the smallest of the dual thresholds. (i) Segmentation result obtained by region growing. (Original image courtesy of X-TEK Systems, Ltd.)

Region-Based Segmentation
Region Growing

- (a). It is difficult to segment the defects by thresholding methods. (Applying region growing methods are better in this case.)





Split and Merge using Quadtree

- Entire image is assumed as a single region. Then the homogeneity test is applied. If the conditions are not met, then the regions are split into four quadrants.

- This process is repeated for each quadrant until all the regions meet the required homogeneity criteria. If the regions are too small, then the division process is stopped.

- 1) Split and continue the subdivision process until some stopping criteria is fulfilled. The stopping criteria often occur at a stage where no further splitting is possible.
- 2) Merge adjacent regions if the regions share any common criteria. Stop the process when no further merging is possible

Region-Based Segmentation
Region Splitting and Merging

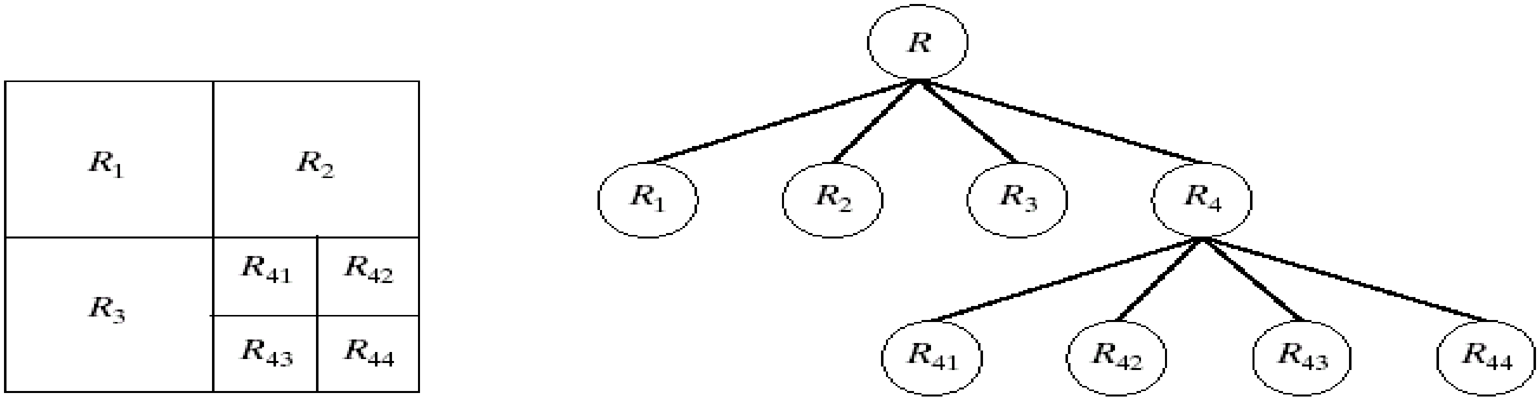
- Region splitting is the opposite of region growing.
  - First there is a large region (possible the entire image).
  - Then a predicate (measurement) is used to determine if the region is uniform.
  - If not, then the method requires that the region be split into two regions.
  - Then each of these two regions is independently tested by the predicate (measurement).

Region-Based Segmentation
Region Splitting

- The main problem with region splitting is determining where to split a region.
- One method to divide a region is to use a quadtree structure.
- Quadtree: a tree in which nodes have exactly four descendants.

a b

FIGURE 10.42
(a) Partitioned image.
(b) Corresponding quadtree.



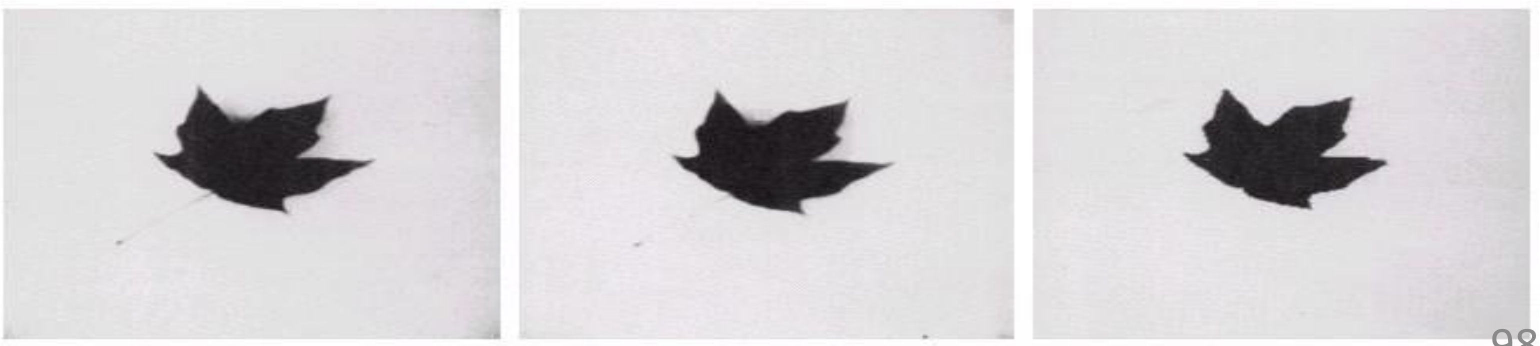
Region-Based Segmentation
Region Splitting and Merging

The split and merge procedure:

– Split into four disjoint quadrants any region *Ri* which $P(R_i) = \text{FALSE}$.

– Merge any adjacent regions *Rj* and *Rk* for which $P(R_j \cup R_k) = \text{TRUE}$. (the quadtree structure may not be preserved)

– Stop when no further merging or splitting is possible.



a b c

FIGURE 10.43 (a) Original image. (b) Result of split and merge procedure. (c) Result of thresholding (a).

Expected Questions..

- Short Notes on Image Segmentation Types- Discontinuity based and similarity based
- Discontinuity based operators- Robert, Sobel, Prewitt
- Discontinuity based Canny Edge Detection Algorithm

- Short note on similarity based segmentation and Thresholding
- Short note on similarity based segmentation and Region Growing Technique
- Short note on similarity based segmentation and Splitting-Merging Technique