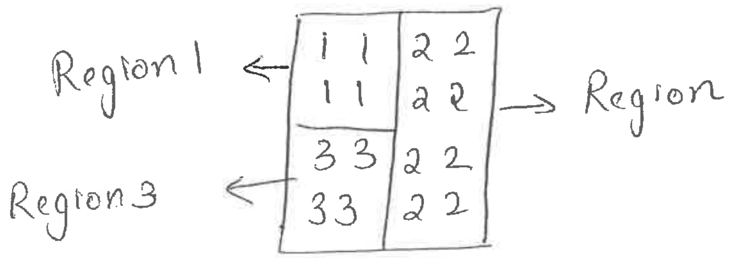
IMAGE SEGMENTATION

Segmentation:- helps to highlight particular region or part so that it is easy to analyze a particular region. the desired part can easily be analyzed.

Ex:- automated electronic assemblies, ~~medical diagnosis~~



Fundamentals:-

Consider an image & entire spatial region occupied by the image 'R'. By segmentation process, the entire image can be split into 'n' sub-regions of $R_1, R_2 - - - R_n$, then,

→ Entire spatial region $R = \bigcup_{i=1}^{n} R_i = R$

→ R_i is a connected set, $i = 1, 2 - - - - n$

→ $R_i \cap R_j = \phi$ (Null set) for all i & j & $i \neq j$.

→ $Q(R_i) =$ True for $i = 1, 2 - - n$

→ $Q(R_i \cup R_j) =$ False for any adjacent regions R_i & R_j

By considering all above conditions, an image is divided into regions

There are basically two types of segmentation:

Image Segmentation

- Discontinuity based Image Segmentation
  - Point Detection
  - Line Detection
  - Edge Detection
    - Basic edge detection
    - Advanced Edge Det
      - Marr-Hildreth Edge Detector
      - Canny edge detector
- Similarity based Image Segmentation
  - Thresholding
    - Basic global thresholding
    - Optimum thresholding
    - Adapti threshi
  - Region growing
  - Region splittin & Mergin

I DISCONTINUITY BASED IMAGE SEGMENTAT

(1) Point Detection :-

An isolated point can be identified as the change in intensity values compared to its surroundi. The isolated points can be detected by applying the second order derivative (Laplacian) mask.

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

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

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

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

Laplacian Mask =

0	1	0
1	-4	1
0	1	0

or

1	1	1
1	-8	1
1	1	1

(2) Line Detection :-

A line is nothing but continuous arrangement of points. For line detection, we use second order derivative so that it produces thinner lines than first derivatives.

Usage of second order derivative mask produces double line effect due to +ve & -ve values.

Negative values can be eliminated by taking absolute value of Laplacian, but this approach doubles the thickness of lines. Hence it is suitable to use only +ve values.

As second order derivative (Laplacian) is easily affected by noise, we use first order derivative. (Gradient)

-1	-1	-1
0	0	0
1	1	1

To find horizontal lines

-1	0	1
-1	0	1
-1	0	1

To find vertical lines

~~the~~ above two masks To detect lines at 0° & 90° angles.
At $+45^\circ$ & -45° the below masks are used.

$$\begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \quad \begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}$$

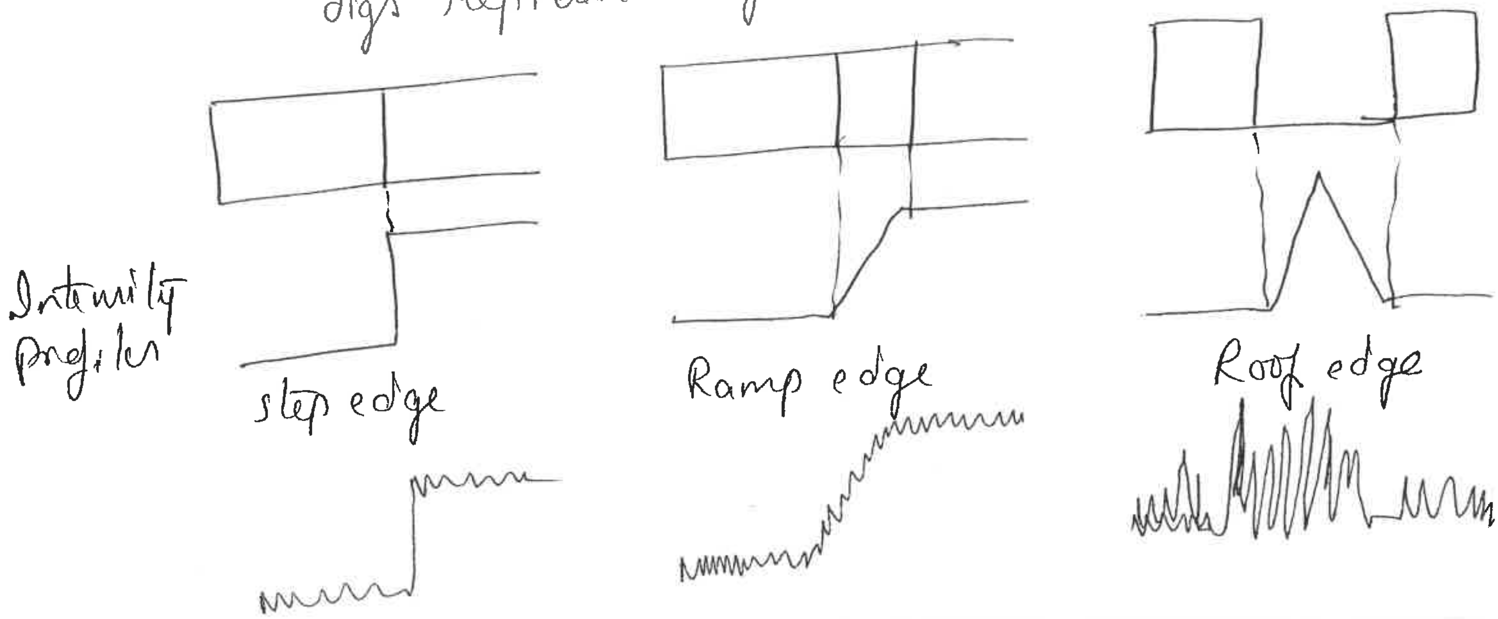
0° (horizontal) $\quad +45^\circ \quad -45^\circ \quad$ vertical

Usually, the first order (Gradient) mask for 'Edge detection' is preferred compared to second order (Laplacian) because second order derivative produces double edge response

(3) <u>Edge Detection</u>:-

Edge is nothing but boundary b/w two regions.
It is the most frequently used method for segmentation based on abrupt changes in the intensity.

figs represent edge models



To detect edges properly we have three fundamental steps

> Image smoothing for noise reduction
> Detection of edge points
> Edge localisation

(a) <u>Basic Edge detection</u> :-

With the help of gradient (first order mask) we will find edges.

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

The magnitude of the vector ∇f is denoted as

$$M(x,y) = \text{mag}(\nabla f) = \sqrt{g_x^2 + g_y^2}$$

The direction of gradient vector

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

To obtain gradient components g_x & g_y, we use two masks namely Sobeal & Prewitt mask

0	1	2
-1	0	1
-2	-1	0

-2	-1	0
-1	0	+1
0	1	+2

Sobeal Mask

0	1	1
-1	0	1
-1	-1	0

-1	-1	0
-1	0	1
0	1	1

Prewitt Mask.

(b) Advanced Techniques for edge detection:-

(i) Marr-Hildreth or Laplacian on Gaussian (LOG) Edge detection:-

Usually Gaussian mask blurs the image to reduce noise both in spatial & freq domains. This mask is used to detect abrupt changes in intensity but they are directional operators.

The Laplacian has advantage over Gaussian because it responds equally to changes in intensity in any mask direction.

Hence $g(x,y) = \nabla^2[G(x,y)] * f(x,y)$ is LOG filter used to find zero crossings of $g(x,y)$ & determine the location of edges in $f(x,y)$

As it is a linear process.

$$g(x,y) = \nabla^2 [G(x,y) * f(x,y)]$$

First smooth the i/p image with gaussian filter.

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

where σ is the standard deviation.

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

you can either derive or write final sol.

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

$$= \frac{\partial}{\partial x}\left[\frac{-\cancel{2}x}{\cancel{2}\sigma^2} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}\right] + \frac{\partial}{\partial y}\left[\frac{-\cancel{2}y}{\cancel{2}\sigma^2} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}\right]$$

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

Apply uv rule

$$= \frac{-x}{\sigma^2} \cdot \frac{(-2x)}{2\sigma^2} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)} + e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)} \cdot \left(\frac{-1}{\sigma^2}\right)$$

$$- \frac{y}{\sigma^2}\left(\frac{-2y}{2\sigma^2}\right) e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)} + e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)} \left(\frac{-1}{\sigma^2}\right)$$

$$\nabla^2 G(x,y) = \left[\frac{\cancel{2}x^2}{\cancel{2}\sigma^4} - \frac{1}{\sigma^2} + \frac{\cancel{2}y^2}{\cancel{2}\sigma^4} - \frac{1}{\sigma^2}\right] e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}$$

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

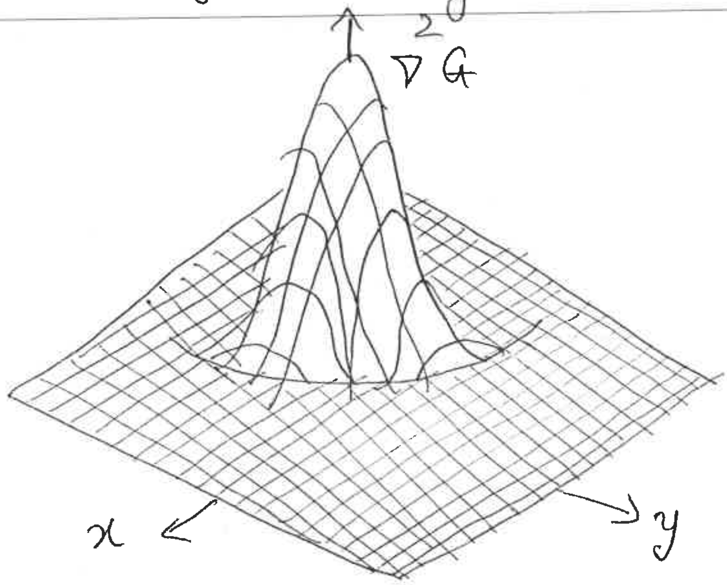
This expression is called Laplacian of Gaussian (LOG)

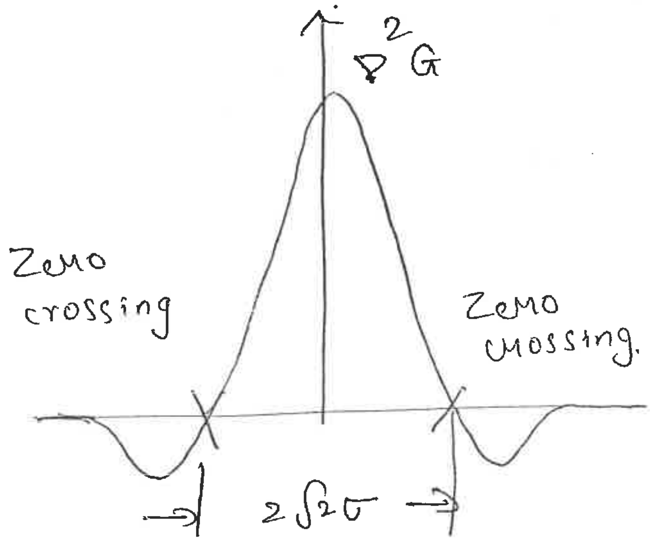
The Marr - Hildreth edge - detection algorithm :-

→ Apply gaussian low pass filter on an i/p image.

→ Compute the Laplacian of the image

→ Find the zero crossing of the image.





Drawback :-

Here we get non-maxima information also.

(Along with actual edge, sub edges are obtained)

Mask for LOG function :-

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

Canny Edge Detector:-

→ This is used to suppress non-maxima information. The objectives of edge detection are

~~→ low error rate~~

→ Edge points should be well localized.

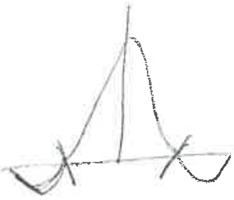
→ Single edge point response.

Step 1:- Smooth i/p image with a Gaussian filter

→ Let $f(x,y)$ be i/p image. ~~Now apply the laplacian of~~ ~~i.e. Gaussian function~~

$G(x,y)$ – Gaussian fn. to the image.

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



→ Smoothed image is obtained by convolution

$$f_s(x,y) = G(x,y) * f(x,y)$$

Step 2 → Compute gradient magnitude & direction (angle)

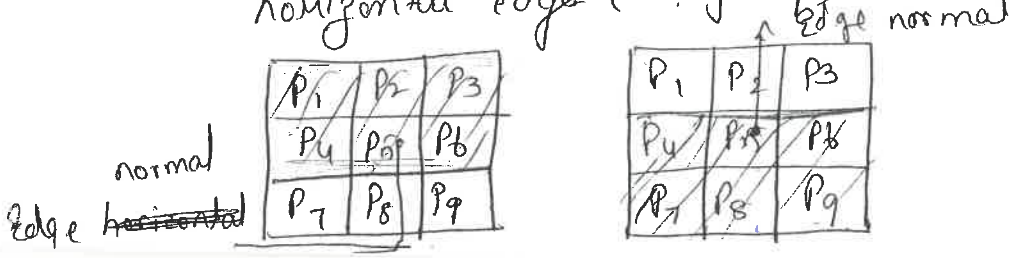
$$M(x,y) = \sqrt{g_x^2 + g_y^2} \quad ; \quad \alpha(x,y) = \tan^{-1}\left(\frac{g_x}{g_y}\right)$$

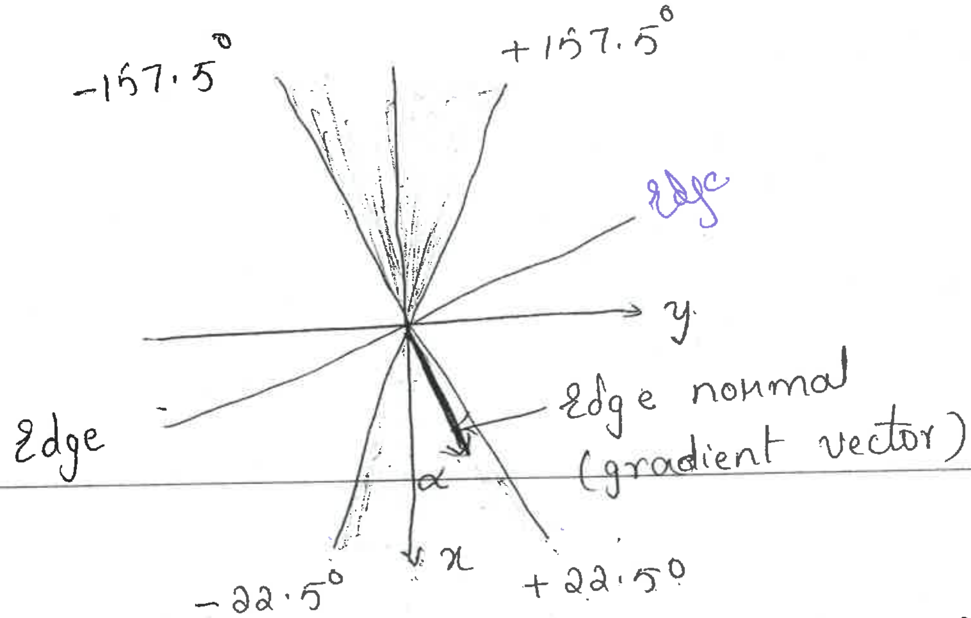
where $g_x = \frac{\partial f_s}{\partial x}$; $g_y = \frac{\partial f_s}{\partial y}$

$M(x,y)$ & $\alpha(x,y)$ are arrays of same size as the image.

Step 3 → To avoid ridges (non-maxima information), use non-maxima suppression.
(note: $M(x,y)$ contains wide ridges around local maxima)

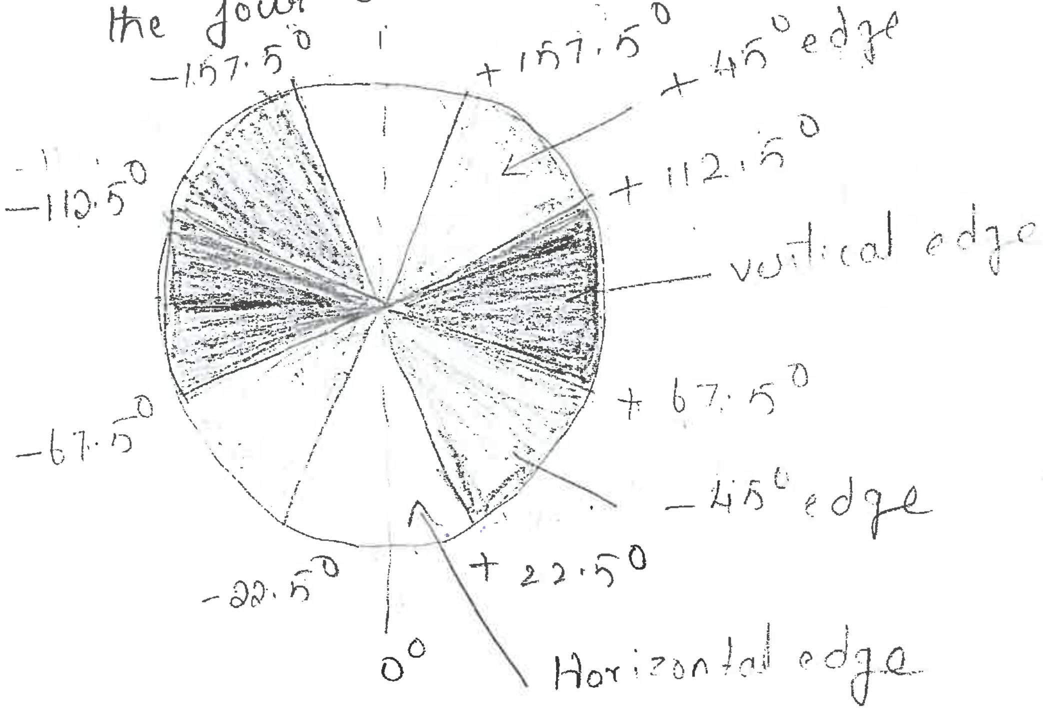
→ Figs show two possible orientations of a horizontal edge (in gray) in a 3×3 neighbourhood





The edge direction is determined from the direction of normal.
In fig. edge normal is in the range of directions from −22.5° to +22.5° or from −157° to +15[7.5°] horizontal edge.
This edge is called horizontal edge.

→ fig below shows the angle ranges corresponding to the four directions (orientations) under consideration for edge passing through center point of the region. (horizontal, vertical, +45° & −45°)



→ If normal is in the range of directions from −22.5° to +22.5° or from −157.5° to +157.5°, we ca[ll] that edge as horizontal edge.

→ Let d_1, d_2, d_3, d_4 be four basic edge directions for a 3×3 region: horizontal $-45°$, vertical $+45°$

→ The non-maxima suppression scheme for a 3×3 ~~scheme~~ region centered at every point (x, y) in $\alpha(x,y)$ can be formulated as follow

1) Find the direction d_K closest to $\alpha(x,y)$

2) If the gradient value of $M(x,y)$ is less than atleast one of its two neighbours along d_K

let $g_N(x,y) = 0$ (suppression)

otherwise $g_N(x,y) = M(x,y)$

where $g_N(x,y)$ - non-maxima suppressed image.

Step 4:- Use Double Threshold on $g_N(x,y)$ to reduce false edge points

> If threshold is too low, there will still be false edges (false positives)

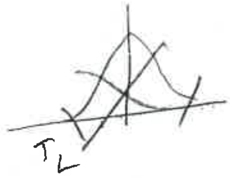
> If threshold is too high, the actual valid edge points will be eliminated. (false negatives)

> Canny's algorithm improves the situation by using hysterisis thresholding i.e two thresholds are used, a low threshold T_L & a high threshold T_H.

> The ratio of high to low threshold must be 2:1 or 3:1

we can visualize the thresholding operation as creating two additional images

$$g_{NH}(x,y) = g_N(x,y) \geq T_H$$



$$\& \quad g_{NL}(x,y) = g_N(x,y) \geq T_L$$

Initially both $g_{NH}(x,y)$ & $g_{NL}(x,y)$ are set to 0.

$g_{NH}(x,y)$ will have fewer nonzero pixels than $g_{NL}(x,y)$ but all non zero pixels in $g_{NH}(x,y)$ will be in $g_{NL}(x,y)$ because the latter image is formed with a lower threshold. Hence eliminate from $g_{NL}(x,y)$ all non zero pixels from $g_{NH}(x,y)$

i.e $g_{NL}(x,y) = g_{NL}(x,y) - g_{NH}(x,y)$

Hence after thresholding operations, all strong pixels in $g_{NH}(x,y)$ are assumed to be valid pixels.

Longer edges are formed using the following procedure:

(a) Locate the next unvisited edge pixel p, in $g_{NH}(x)$

(b) Mark as valid edge pixels all weak pixels in $g_{NL}(x,y)$ that are connected to p using 8-connectivity

(c) If all nonzero pixels in $g_{NH}(x,y)$ have been visited go to step (d) else return to step (a)

(d) Set to zero all pixels in $g_{NL}(x,y)$ that were not marked valid edge pixels.

EDGE LINKING & BOUNDARY DETECTION :-

The need of edge linking is due to the following two reasons:

→ Small pieces of edges may be missing.

→ small edge segments may appear to be present due to noise where there is no real edge.

Edge linking methods can be classified into two categories:

Local edge linking

Global Edge linking

(1) Local Edge linking :-

- Detect the edge points
- Analyze characteristics of pixels in a small neighbourhood (say 3×3 or 5×5) about every point (x, y) in image
- The basic criteria is to determine ⓐ the strength of the response of the gradient operator used to produce the edge pixel.

ⓑ The direction of gradient vector

- For a point (x_0, y_0)

strength of gradient = $M(x_0, y_0)$

Direction of gradient $= \alpha(x_0, y_0)$

for a point (x, y) at the edge
strength of the gradient $= M(x, y)$
Direction of gradient $= \alpha(x, y)$
Point (x_0, y_0) is similar to (x, y) if

$$|M(x, y) - M(x_0, y_0)| \leq E$$

whe E = +ve threshold.

$$|\alpha(x, y) - \alpha(x_0, y_0)| \leq E$$

Thus (x_0, y_0) & (x, y) are linked together.

(2) Global Edge linking by using Hough transform

Hough transform is used to connect disjoint edge points. It is used to fit the points as plane curves. The plane curves are lines, circles & parabolas.

The line eq of a line is given as

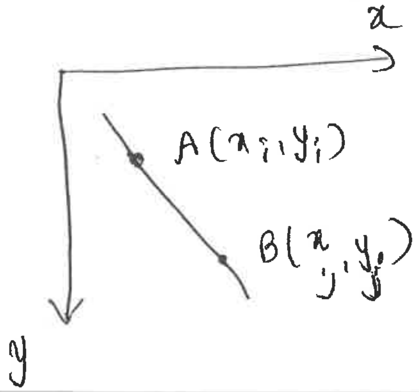
$$y = mx + c$$

where m is slope & c is the y-intercept.

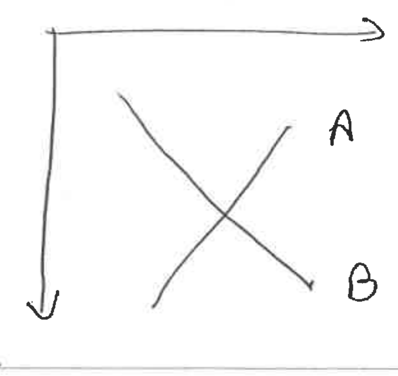
However, there are infinite lines that can be drawn connecting these points. Therefore an edge point in an x-y plane is transformed into a c-m plane.

The line eq, is $c = -mx + y$.

xy plane



cm plane, Hough space



All points in xy plane are represented as lines in c-m plane. The objective is to find intersection point, to indicate that edge points are part of the same line. If A & B are points connected by a line in the spatial domain, then they will be intersecting lines in the Hough space.

To check whether lines are intersecting or not, the c-m plane is partitioned as an accumulator array. For every edge point (x, y), the corresponding accumulator element is incremented in the accumulator array. At the end of this process, the accumulator values are checked. The largest value of the accumulator is called (m', c'). This point gives us the line equation of the (x, y) space.

$$y = m'x + c'$$

The Hough Transform can be stated as follows:

1) Load the image
2) Find the edges of the image using any edge detector
3) Quantize the parameter space P.
4) Repeat the following for all the pixels of the image:
   If the pixel is an edge pixel, then
   (a) $c = (-x)m + y$
   (b) $P(c.m) = P(c.m) + 1$
5) Show Hough space
6) Find local maxima in parameter space.
7) Draw the line using local maxima.

→ The Hough transform for a polar co-ordinate line is as follows.

Except steps 4 all remaining steps are same.

Step 4:- If the pixel is an edge pixel, then for all (x, y) & θ

(a) Calculate ρ for the pixel
(b) Increment the position (ρ, θ) in the accumulator array P.

→ For circle detection

Step 4:- If pixel is an edge pixel, then for all values r, calculate

(a) $x_0 = x - r\cos\theta$
(b) $y_0 = y - r\sin\theta$
(c) $P(x_0, y_0, r) = P(x_0, y_0, r) + 1$

THRESHOLDING :-

Thresholding plays a major role in segmentation. There are two types of thresholding
(1) Single level thresholding
(2) Multi-level thresholding.

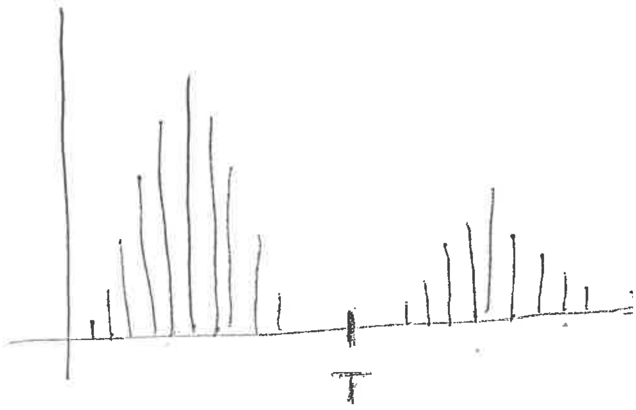
(1) Single level Thresholding :-

Suppose that an image $f(x,y)$ has light objects on a dark background. If we draw a histogram of this type of image, it will have two dominant mode of gray levels. as shown in fig.

So at any point (x,y),

if $f(x,y) > T$; object is light

if $f(x,y) < T$; object is dark

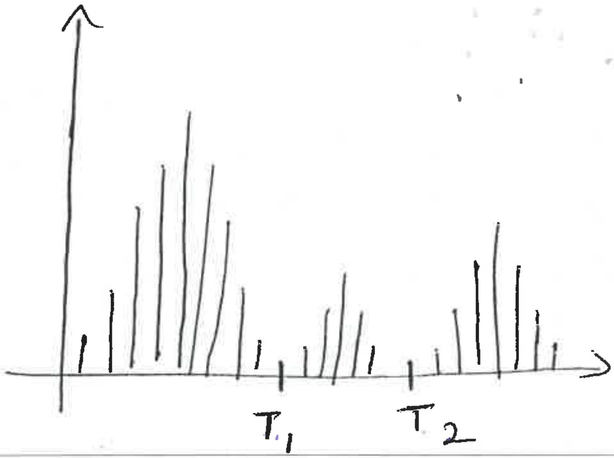


This is called single-level thresholding because we have separated information of an image by single threshold.

(2) Multilevel thresholding :- Suppose we have three dominant modes that characterize the image histogram This is called multilevel thresholding because we have used more than one threshold for separation of image formation

Multilevel thresholding classifies a point (x, y) as belonging to one object class if

$T_1 < (x, y) < T_2$



→ If $f(x,y) \leq T_1$, the object has dark background

→ If $f(x,y) \geq T_2$, the object has white

Hence thresholding is a test function of

(1) spatial co-ordinates (x,y)

(2) $f(x,y)$ - gray level point (x,y)

(3) $P(x,y)$ - some local property at point (x,y)

Hence $Th = T[x, y, f(x,y), P(x,y)]$

There are three types of thresholding functions:

(1) Global Thresholding (2) Local Thresholding

(3) Adaptive thresholding

Global Thresholding :-

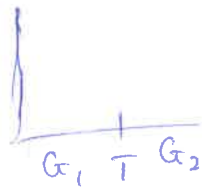
(1) Select an initial estimate of threshold T.

(2) Segment the image using T. This will produce ~~ta~~ two groups of pixel.

G_1 with pixel values $> T$

G_2 with pixel values $\leq T$

3) Compute the average gray level values u_1 & u_2 for the pixels in regions G_1 & G_2.



4) Compute a new threshold value

$$T = \frac{1}{2}(u_1 + u_2)$$

5) Repeat steps 2 to 4 until difference in T in successive iterations is smaller than a predefined parameter T_0.

(2) Local Thresholding :-

A good threshold may be selected based on whether the histogram peaks are tall, narrow, symmetric & separated by deep valleys. Hence this depends on local property of images (edges) to select a threshold called local thresholding.

(3) Basic Adaptive Thresholding :-

Due to poor illumination, it is practically seen that sometimes, we get histogram that cannot be partitioned by single Global thresholding.

To handle such situation, image is divided into sub images & different thresholds are used for different subimages. The threshold depends upon location of pixels & hence called adaptive thresholding.

REGION BASED SEGMENTATION :-

There are two main approaches to region based segmentation.

(1) Region splitting & merging

(2) Region growing.

(1) Region Splitting & Merging :-

Region splitting is to break the image into a set of disjoint regions which are coherent within themselves.

The process is defined as follows:

1) Initially take the image as a whole to be the area of interest.

2) Look at the area of interest & decide if all pixels contained in the region satisfy some similarity constraint.

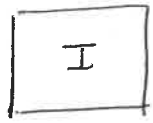
3) If 'TRUE' then area of interest is splitted into subareas & now consider each of subareas as area of interest.

4) This process is performed continuosly until no further splitting is possible.

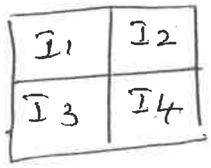
In the worst case, the areas may be of just one pixel. This is also called divide conquer 'or' top down method.

If only splitting is used, final segmentation would probably contain many neighbouring regions that have identical or similar properties. Hence merging is used after each splitting process.

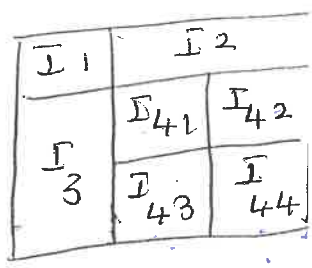
1) Consider a whole image shown in fig



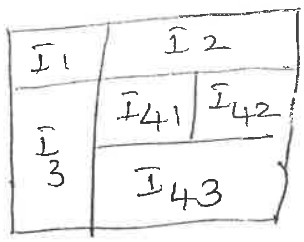
2) Suppose that all pixels in I are not similar, so we have to split the region I in four sub regions.



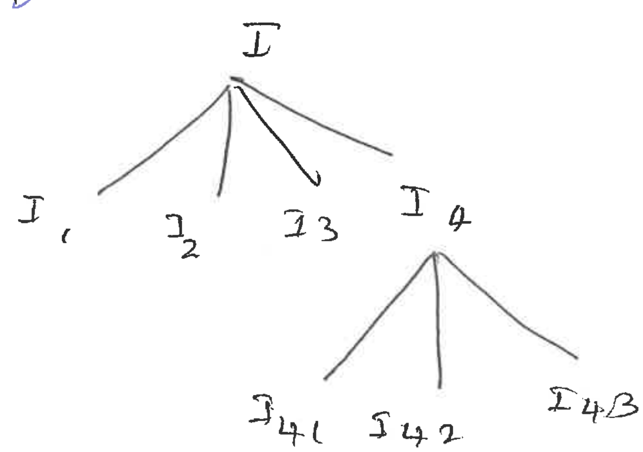
3) Assume that all pixels in region I_1, I_2 & I_3 are similar but those in I_4 are not. So I_4 is splitted again in four subregions



4) Suppose after splitting I_{43} & I_{44} are found to be identical, we have to merge them again



5) All splitting process can also be represented by a quad tree



Ex:-

R2, R1, R3

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

To merge set a threshold '4'

$8 - 2 > 4$

$8 - 1 < 4$

$2 - 1 < 4$

6

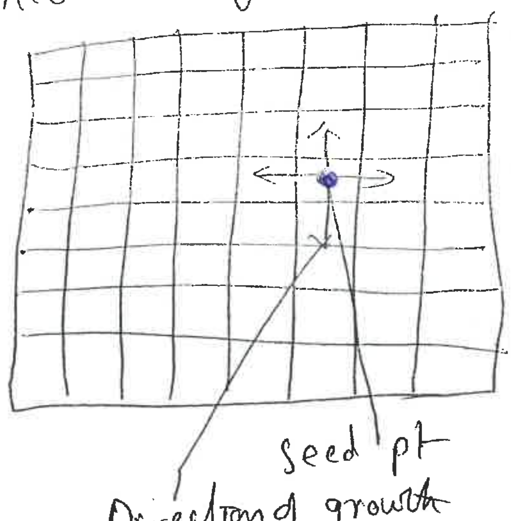
~~Hence me~~

Hence after merging

	8	8	8	
	8	8	8	

(2) Region Growing:-

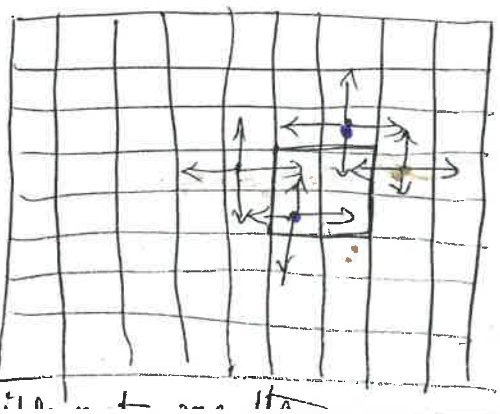
It is the opposite of region splitting & merging. The process is defined as follows:



(1) Start by choosing an arbitrary seed pixel & compare it with neighbouring pixels.

(2) Region is grown from seed pixel by adding in neighbouring pixels that are similar, increasing the size of the region

(3) When the growth of one region stops we simply choose another seed pixel which does not yet belong to any region & start again



Disadvantage (i) Current region dominates growth process ...