Unit 6: Image Segmentation

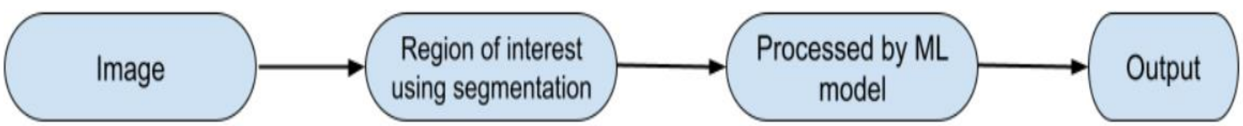
Introduction:

Image segmentation is a method in which a digital image is broken down into various subgroups called Image segments which helps in reducing the complexity of the image to make further processing or analysis of the image simpler.

Segmentation refers to the process of partitioning on image into multiple regions. It is typically used to locate objects and boundaries in images.

Segmentation in easy words is assigning labels to pixels. All picture elements or pixels belonging to the same category have a common label assigned to them.

For example: Let's take a problem where the picture has to be provided as input for object detection. Rather than processing the whole image, the detector can be inputted with a region selected by a segmentation algorithm. This will prevent the detector from processing the whole image thereby reducing inference time.



Similarity and discontinuity approach

Similarity approach: This approach is based on detecting similarity between image pixels to form a segment, based on a threshold. ML algorithms like clustering are based on this type of approach to segment an image.

Discontinuity approach: This approach relies on the discontinuity of pixel intensity values of the image. Line, Point, and Edge Detection techniques use this type of approach for obtaining intermediate segmentation results which can be later processed to obtain the final segmented image.

Discontinuity Based Technique:

In discontinuity-based approach, the partitions or sub-division of an image is based on some abrupt changes in the intensity level of images. Here, we mainly interest in identification of isolated points, lined and edges in an image. To identify these, we use 3X3 Mask operation.

The discontinuity-based segmentation can be classified into three approaches:

- Point detection
- Line detection
- Edge detection

Point Detection:

A point is the most basic type of discontinuity in a digital image. The most common approach to finding discontinuities is to run an (n X n) mask over each point in the image. The mask is as shown in figure bellow

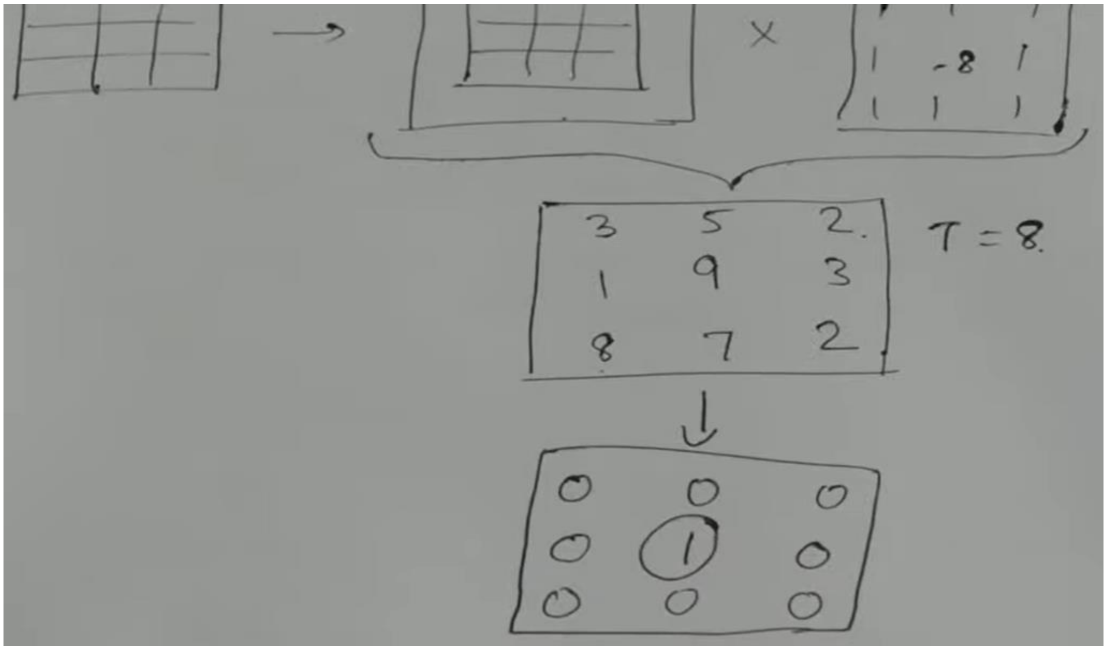
-1	-1	-1
-1	8	-1
-1	-1	-1

The point is detected at a location (x, y) in an image, where the convolution operation is done using this mask. If the absolute corresponding value of Z is greater than threshold value T, put label 1 for that point and put level 0 for others.

$$g(x,y) = \begin{cases} 1 & \text{if } |Z(x,y)| > T \\ 0 & \text{otherwise} \end{cases}$$

Where Z is the response of the mask at any point in the image and T is non-negative threshold value.

For Example:



Here a point is detected in middle of the image.

Line Detection

Line detection is the next level of complexity in the direction of image discontinuity. For any point in the image, a response can be calculated that will show which direction the point of a line is most associated with. The mask for different direction is given bellow

horizontal
direction ma
badi discontinuity vnana khojeko ho
so tesko matra 2 garne aru -1

-1	-1	-1
2	2	2
-1	-1	-1

Horizontal direction

-1	2	-1
-1	2	-1
-1	2	-1

Vertical direction

-1	-1	2
-1	2	-1
2	-1	-1

45^0 direction

2	-1	-1
-1	2	-1
-1	-1	2

-45^0 direction

Perform convolution operation in given image using these masks.

Edge detection

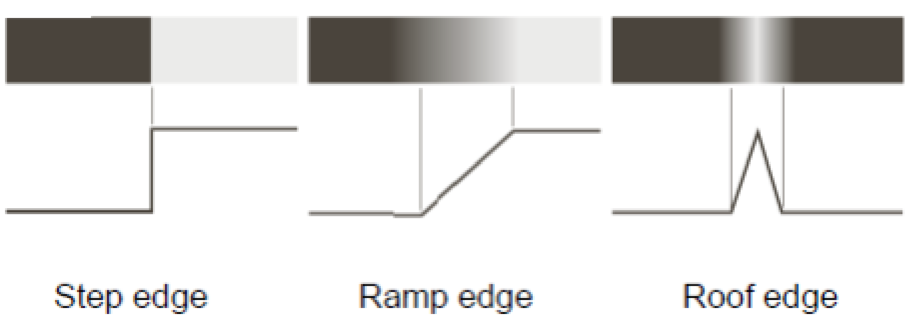
Since isolated points and lines of unitary pixel thickness are infrequent in most practical application, edge detection is the most common approach in gray level discontinuity segmentation. An edge is a boundary between two regions having distinct intensity level. It is very useful in detecting of discontinuity in an image, when the image changes from dark to white or vice-versa.

3 different edge types are observed:

Step edge: Transition of intensity level over 1 pixel only in ideal, or few pixels on a more practical use

Ramp edge: A slow and graduate transition

Roof edge: A transition to a different intensity and back



The first order derivative or gradient based filter such as Robert-cross, Prewitt, and Sobel operators are prefers for detecting thicker lines.

The second order derivative such as Laplacian is prefers for detecting thinner lines.

Robert (Cress Gradient) operator

This operator find the gradient difference in cross or diagonal pixel position.

The filter mask of Robert operator is:

g(x) =

-1	0
0	1

g (y) =

0	-1
1	0

By using one of this filter mask, we can perform convolution operation on input image to calculate g(x) and g(y).

Prewitt Operator

This method takes the central difference of the neighboring pixels.

The filter mask of Prewitt Operator is:

g(x) =

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

g(y) =

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

By using one of this filter mask, we can perform convolution operation on input image to calculate g(x) and g(y).

Sobel Operator

This method also takes the central difference of the neighboring pixels. It provides both a differentiating and a smoothing effect.

The filter mask of Sobel Operator is:

g(x) =

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

g(y) =

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

teivyerw euta 2 xa

By using one of this filter mask, we can perform convolution operation on input image to calculate g(x) and g(y).

Edge Linking and Boundary Detection

Edge Linking:

Ideally, edge detection should yield sets of pixels lying only on edges. In practice, these pixels rarely characterize edges completely because of non-uniform illumination, noise and breaks in the edges. Therefore, edge detection typically is followed by linking algorithms designed to assemble edge pixels into meaningful edges and/or region boundaries.

Edge linking may be:

Local: requiring knowledge of edge points in a small neighborhood.

Regional: requiring knowledge of edge points on the boundary of a region.

Global: the Hough transform, involving the entire edge image.

Edge Linking by Local Processing

All points that are similar according to predefined criteria are linked, forming an edge of pixels that share common properties.

Edge Linking by Regional Processing

Often, the location of regions of interest is known and pixel membership to regions is available. Approximation of the region boundary by fitting a polygon.

Polygons are attractive because:

- They capture the essential shape
- They keep the representation simple

Requirements

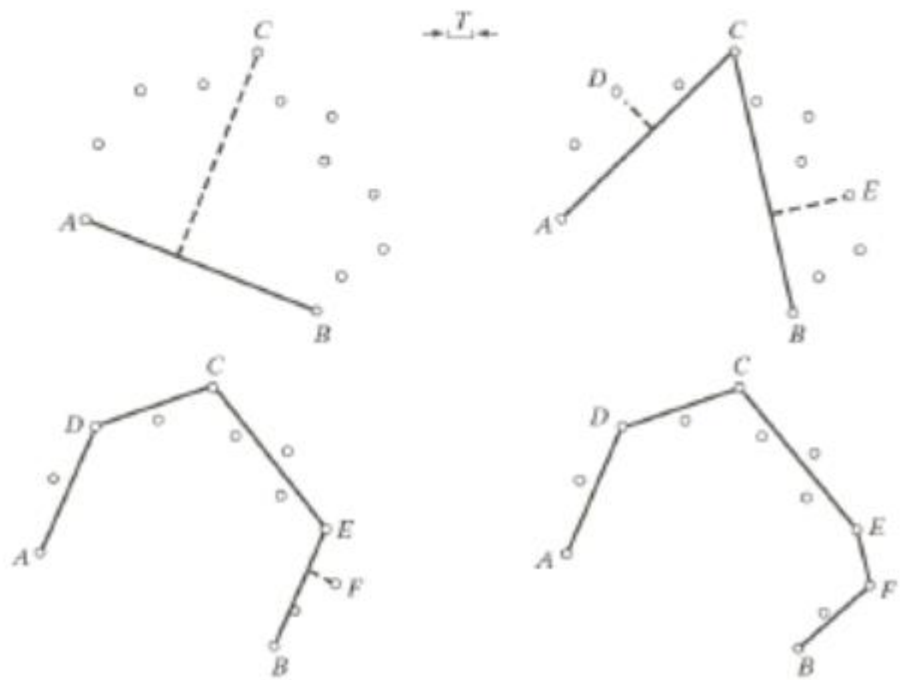
- Two starting points must be specified (e.g. rightmost and leftmost points).
- The points must be ordered (e.g. clockwise).
- Variations of the algorithm handle both open and closed curves.

If this is not provided, it may be determined by distance criteria:

- Uniform separation between points indicate a closed curve
- A relatively large distance between consecutive points with respect to the distances between other points indicate an open curve

We present here the basic mechanism for polygon fitting.

Given the end points A and B, compute the straight line AB. Compute the perpendicular distance from all other points to this line. If this distance exceeds a threshold, the corresponding point C having the maximum distance from AB is declared a vertex. Compute lines AC and CB and continue.



Edge Linking by Global Processing

Hough Transform

The Hough Transform is an algorithm patented by Paul V. C. Hough and was originally invented to recognize complex lines in photographs (Hough, 1962). Since its inception, the algorithm has been modified and enhanced to be able to recognize other shapes such as circles and quadrilaterals of specific types.

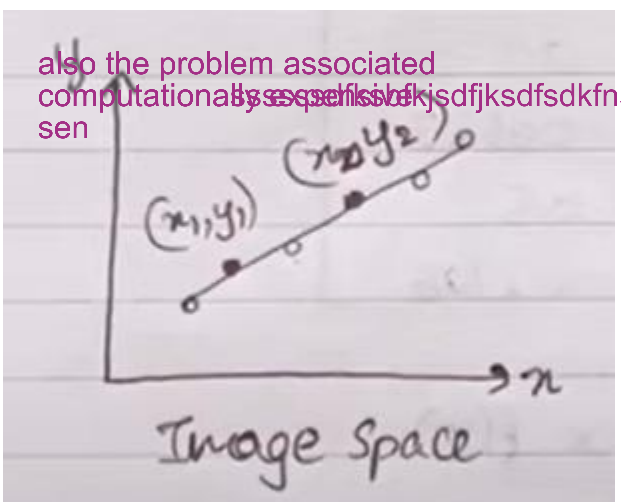
It is mainly used to connect disjoint edge points.

Equation of line is

y = mx + c

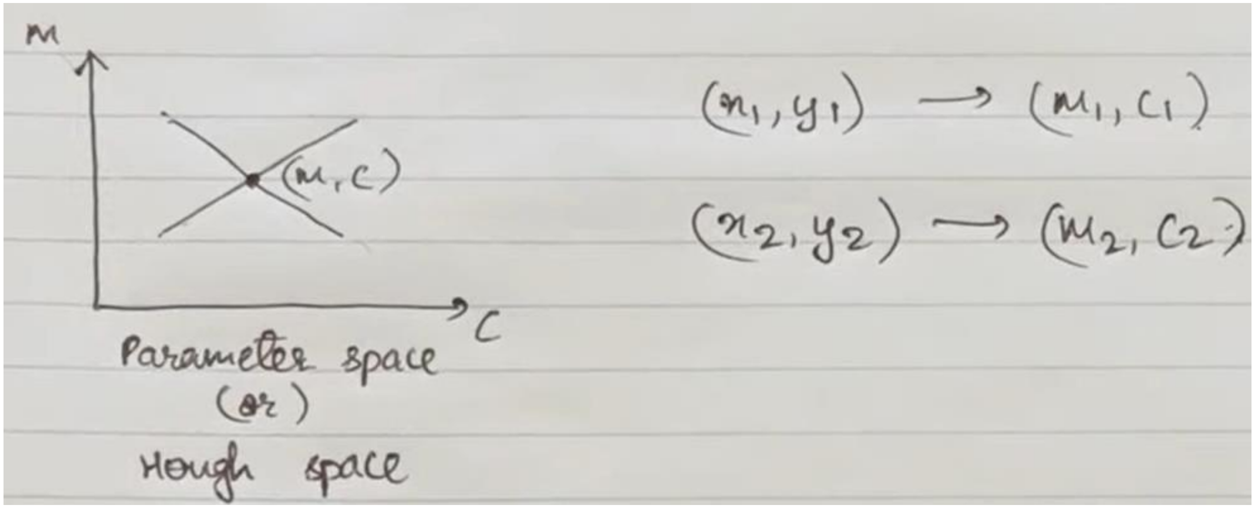
Where, m =slope and c = intercept of the line

also the problem associated
computationallsssedfksdfkjsdfjksdfsdkfnsdfnsdlkflly expensive
sen

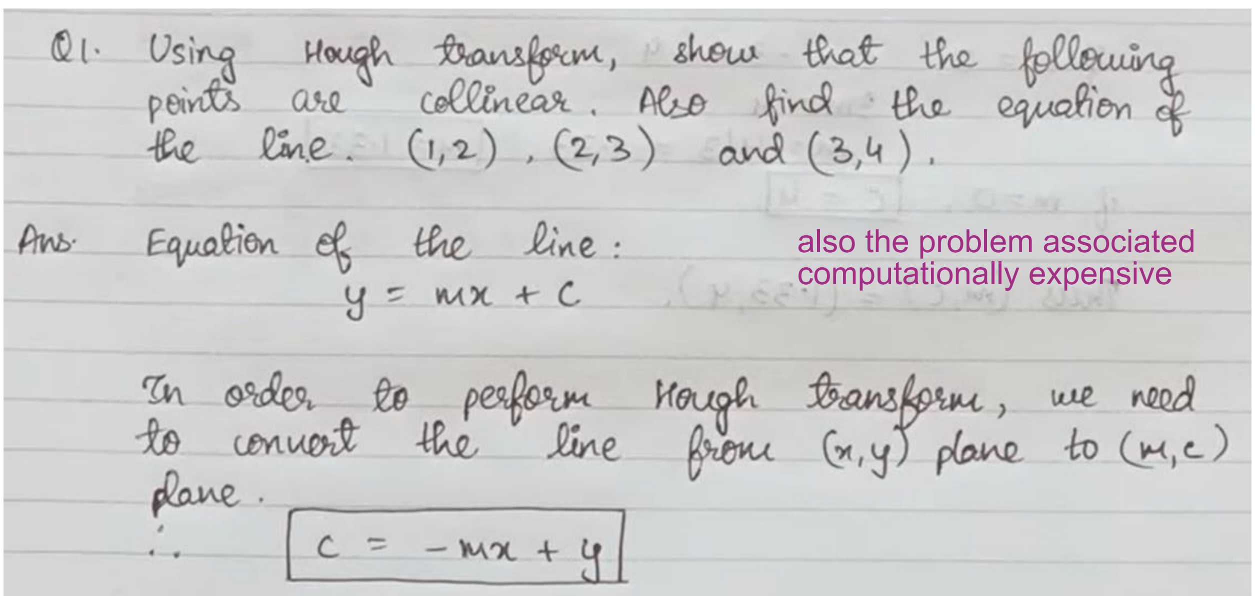


A single point can be a part of infinite line. Therefore we transform that point in the x-y plane, into a line in the m-c plane.

x1, y1 rw x1 , y2 points of same line thyoni tw image space ma so tyo hough space ma chai intersepting line in hough space



If A and B are two points connected by a line in the special domain, they will be intercepting line in the Hough space.



Q1. Using Hough transform, show that the following points are collinear. Also find the equation of the line. (1,2), (2,3) and (3,4).

Ans. Equation of the line:

$$y = mx + c$$

In order to perform Hough transform, we need to convert the line from (x,y) plane to (m,c) plane.

$$\therefore \boxed{c = -mx + y}$$

also the problem associated computationally expensive

(i) For $(x, y) = (1, 2)$, $c = -m + 2$

if $c = 0$, $0 = -m + 2$

$\boxed{m = 2}$

if $m = 0$, $\boxed{c = 2}$

Thus, $(m, c) = (2, 2)$.

(ii) For $(x, y) = (2, 3)$, $c = -2m + 3$.

if $c = 0$, $0 = -2m + 3$

$2m = 3$

$m = 3/2 = 1.5$. $\boxed{m = 1.5}$

if $m = 0$, $\boxed{c = 3.}$

Thus, $(m, c) = (1.5, 3)$.

(iii) For $(x, y) = (3,4)$, $c = -3m + 4$.

if $c = 0$, $0 = -3m + 4$

$3m = 4$

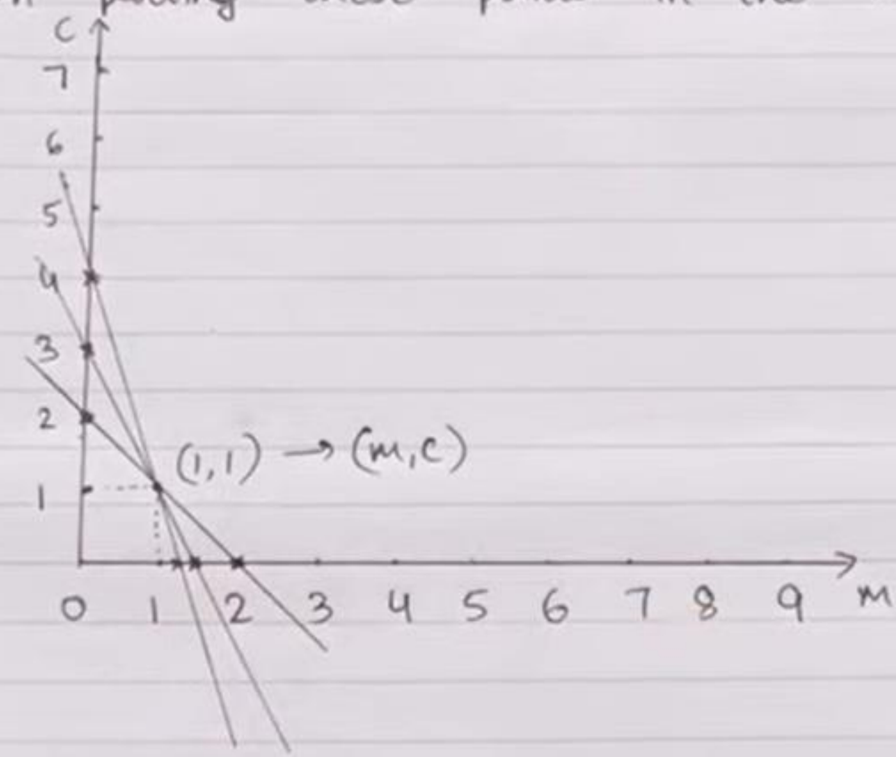
$m = 4/3 = 1.33$. $\boxed{m = 1.33}$

if $m = 0$, $\boxed{c = 4}$.

Thus $(m, c) = (1.33, 4)$.

$(m, c) = (2,2), (1.5,3), (1.33,4)$.

On plotting these points in the m-c plane:



All three lines are intercept in single point (1,1) that means all points (1,2) (2,3) (3,4) are collinear and they are part of the same line.

Original equation :

$$y = mx + c$$

Substituting (1,1) :

$\boxed{y = x + 1}$ → Final equation

Thresholding

The simplest method for segmentation in image processing is the threshold method. It divides the pixels in an image by comparing the pixel's intensity with a specified value (threshold). It is useful when the required object has a higher intensity than the background (unnecessary parts).

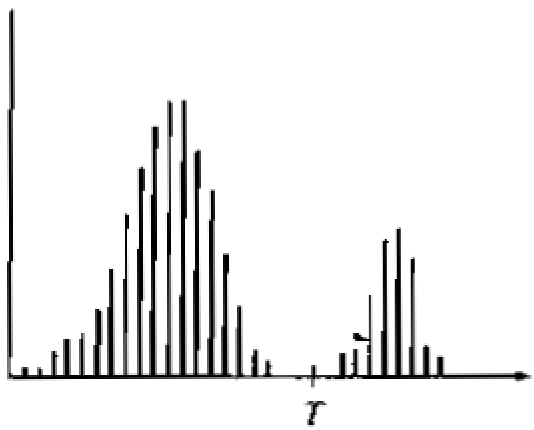
You can consider the threshold value (T) to be a constant but it would only work if the image has very little noise (unnecessary information and data). You can keep the threshold value constant or dynamic according to your requirements.

The thresholding method converts a grey-scale image into a binary image by dividing it into two segments (required and not required sections).

Steps to apply threshold

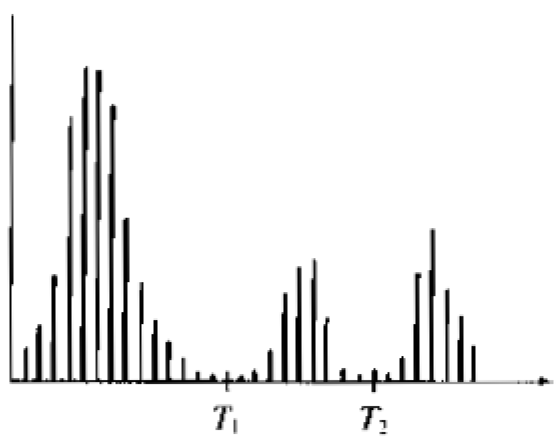
1. Select a threshold value ***T***
2. Any point ***(x,y)*** in the image at which ***f(x,y)* > *T***, this point is called an object point and ***f(x,y)* <= *T*** is called background point.
3. The segmented image ***g(x,y)*** is denoted by

$$g(x, y) = \begin{cases} 1 & \text{if } f(x, y) > T \\ 0 & \text{if } f(x, y) \leq T \end{cases}$$



Here in histogram, the region greater than T is object and less than T is background.

The histogram bellow shows the thresholding problem involving three dominant modes. For example two objects on a dark background.



Here in histogram multiple thresholding classifies a point ***(x,y)*** as belonging to the background if $f(x,y) <= T_1$, one object classes if $T_1 < f(x,y) <= T_2$, and to the other object classes $f(x,y) > T_2$

$$g(x,y) = \begin{cases} a & \text{if } f(x,y) > T_2 \\ b & \text{if } T_1 < f(x,y) \le T_2 \\ c & \text{if } f(x,y) \le T_1 \end{cases}$$

highest limit xa t2 tyo vnda ni thulo vye tw best a payo

According to the different threshold values, we can classify thresholding segmentation in the following categories:

Global Thresholding:

A constant threshold value is apply for both object and background is called global thresholding. In this method, you replace the image's pixels with either white or black.

If the intensity of a pixel at a particular position is less than the threshold value, you'd replace it with black. On the other hand, if it's higher than the threshold, you'd replace it with white.

Procedure for Global thresholding thresholding le sadhae suruma greater wala lekhxa

1. Select initial threshold value T (*choose average value of intensity*)
2. Segment the image using T. this will produce two groups G1 and G2 which are G1 contains values > T and G2 contains values <= T
3. Compute the average gray level values $\mu 1$ and $\mu 2$ for the pixel in region G1and G2
4. Compute the new threshold value
   $T= ½(\mu1+ \mu2)$
5. Repeat step 2 to step 4 until the T in successive iterations is same.

Example: Find the Global Threshold value of given image

5	3	9
2	1	7
8	4	2

Solution:

Calculate the threshold value T_0 by taking average of all the pixel value

$T_0 = (5+3+9+2+1+7+8+4+2)/9$

$T_0 = 4.55 = 5$

Segment the image using T = 5, we get

G1 = {9,7,8}

Calculate the mean value of G1

$\mu1 = (9+7+8)/3$

$\mu1 = 8$

G2 = {5,3,2,1,4,2}

Calculate the mean value of G2

$\mu2 = (5+3+2+1+4+2)/6$

$= 2.83$

$\mu2 = 3$

Now, the new value of T say T_1

$T_1= ½(\mu1+ \mu2)$

$T_1= ½ (8+3)$

$T_1= 5.5 = 6$

Here the threshold value for successive iteration is different,

So, again segment the image using new threshild value of T_1 i.e. 6,

G1 = {9,7,8}

$\mu1 = 8$

G2 = {5,3,2,1,4,2}

$\mu2= 3$

Now, the new value of T say T_2

$T_2 = ½ (8+3)$

$T_2 = 5.5 = 6$

Here the threshold value for successive iteration is same, so final value of T is 6, which is the global threshold value.

Local or Regional Thrisholding

When the value of T changes over an image is called Variable thresholding. The term local or Regional thresholding is used sometimes to denote variable thresholding in which the value of T at any point (x,y) in an image depends on properties of a neighborhood of (x,y).

For example: The average intensity of the pixels in the neighborhood.

Adaptive Thresholding

If T depends on special coordinates (x,y) themselves, then variable thresholding is called dynamic or adaptive thresholding.

Having one constant threshold value might not be a suitable approach to take with every image. Different images have different backgrounds and conditions which affect their properties.

Thus, instead of using one constant threshold value for performing segmentation on the entire image, you can keep the threshold value variable. In this technique, you'll keep different threshold values for different sections of an image.

This method works well with images that have varying lighting conditions. You'll need to use an algorithm that segments the image into smaller sections and calculates the threshold value for each of them.

Procedure for Adaptive thresholding

1. Divide original image in to different regions
2. Apply global thrsholding method in each region separately
3. Merge the resulted region based image

Region-Based Segmentation

Region-based segmentation algorithms divide the image into sections with similar features. These regions are only a group of pixels and the algorithm find these groups by first locating a seed point which could be a small section or a large portion of the input image.

After finding the seed points, a region-based segmentation algorithm would either add more pixels to them or shrink them, so it can merge them with other seed points.

Based on these two methods, we can classify region-based segmentation into the following categories:

Region Growing

In this method, you start with a small set of pixels and then start iteratively merging more pixels according to particular similarity conditions. A region growing algorithm would pick an arbitrary seed pixel in the image, compare it with the neighbor pixels and start increasing the region by finding matches to the seed point.

When a particular region can't grow further, the algorithm will pick another seed pixel which might not belong to any existing region. One region can have too many attributes causing it to take over most of the image. To avoid such an error, region growing algorithms grow multiple regions at the same time.

You should use region growing algorithms for images that have a lot of noise as the noise would make it difficult to find edges or use thresholding algorithms.

Algorithm for Region Growing

1. Choose a seed point
2. Check the condition

   If |seed point – pixel value| <=T

      Add pixel to seed point region

Else

Leave as it is

3. Repeat step 2 for all pixel

Example: Apply region growing on following image with seed point at (2, 2) and threshold value as 2.

0	1	2	0
2	5	6	1
1	4	7	8
0	9	5	1

Solution:

Here the seed point is 7 which is at location (2,2)

0	1	2	0
2	5	6	1
1	4	7	8
0	9	5	1

Threshold value (T) =2

Therefore the condition is

|seed point – pixel value| <=2

The possible pixel values which satisfy the condition are {5,6,7,8,9}

Now, let's say region A which is denoted by 1 for condition satisfied values and region B which is denoted by 0 for other values

0	0	0	0
0	1	1	0
0	0	1	1
0	1	1	0

Region Splitting and Merging

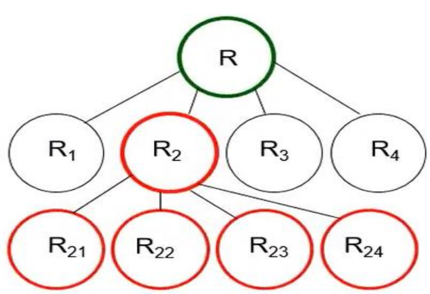
As the name suggests, a region splitting and merging focused method would perform two actions together – splitting and merging portions of the image.

This method is also called divide and conquer.

It would first split the image into regions that have similar attributes and merge the adjacent portions which are similar to one another. In region splitting, the algorithm considers the entire image, while in region growth, the algorithm would focus on a particular point. region growth ma seed point ma matra focus garerw garthyo

The region splitting and merging method follows a divide and conquer methodology. It divides the image into different portions and then matches them according to its predetermined conditions. Another name for the algorithms that perform this task is split-merge algorithms.

Quad tree of splitting



Algorithm for Region Splitting

1. Select max and min intensity value from given image
2. Check the condition
   if ((max – min) > T)
            split the image in to equal 4 regions
   else
            leave as it is
3. Apply step 2 in all regions

Algorithm for Region Merging

1. Select max and min intensity value from neighbor regions
2. Check the condition
   if ((max_1 – min_2) <= T && (max_2 – min_1) <= T)
            merge those regions
   else
            leave as it is
3. Apply step 2 in all neighbor regions

Example: Apply split and merge in given image. The threshold value is 3.

6	5	6	6	7	7	6	6
6	7	6	7	5	5	4	7
6	6	4	4	3	2	5	6
5	4	5	4	2	3	4	6
0	3	2	3	3	2	4	7
0	0	0	0	2	2	5	6
1	1	0	1	0	3	4	4
1	0	1	0	2	3	5	4

Solution:

Here, given T = 3

Max pixel value = 7

Min pixel value = 0

Now, check the condition (max – min) > T

(7 – 0) > 3

Then split the image in to 4 regions

A

B

6	5	6	6	7	7	6	6
6	7	6	7	5	5	4	7
6	6	4	4	3	2	5	6
5	4	5	4	2	3	4	6
0	3	2	3	3	2	4	7
0	0	0	0	2	2	5	6
1	1	0	1	0	3	4	4
1	0	1	0	2	3	5	4

C

D

Now, apply rule for all the regions A, B, C and D

7
4

A

7
2] 5 ≯ 3

B

6	5	6	6	7	7	6	6
6	7	6	7	5	5	4	7
6	6	4	4	3	2	5	6
5	4	5	4	2	3	4	6
0	3	2	3	3	2	4	7
0	0	0	0	2	2	5	6
1	1	0	1	0	3	4	4
1	0	1	0	2	3	5	4

C

3
0

D

7
0] 7

3

Now, merging

Check the condition $((max_1 – min_2) <= T \ \&\& \ (max_2 – min_1) <= T)$

Here, for region A and B1

$((7 – 5) <= 3 \ \&\& \ (7 – 4) <= 3)$

Both the condition is satisfied so merge these two regions

Apply same rule for all the regions

6	5	6	6	7	7	6	6
6	7	6	7	5	5	4	7
6	6	4	4	3	2	5	6
5	4	5	4	2	3	4	6
0	3	2	3	3	2	4	7
0	0	0	0	2	2	5	6
1	1	0	1	0	3	4	4
1	0	1	0	2	3	5	4

In the resulted image we can see two regions one is shaded region and another is un-shaded region.

also the problem associated

-computationally expensive

-sensitive to noise

-splitting is quite simple but merging is very complex

End of Unit-6

-``Meaningful'' regions may not be uniform: surface properties of a solid body will vary in brightness or colour dependent on the existence of slowly varying gradients due to lighting conditions.

-Lighting effects or curvature affect the appearance, e.g. a sphere illuminated by a point light source may have intensities varying from pure white to black, yet is a single surface.

-It is very unusual in practice for an image to be composed of uniform regions of similar intensity, or colour, or texture etc.

-Regional segmentation works best with binary data as the limited range of values lead to more uniform regions.