UNIT 11: IMAGE SEGMENTATION

UNIT STRUCTURE

1.1 Learning Objectives

11.2 Introduction

11.3 Classification of Image Segmentation

11.4 Local and Global Segmentation

11.5 Image Segmentation Techniques

- 11.5.1 Edge-based Segmentation
- 11.5.2 Region Detection
- 11.5.3 Boundary Detection

11.6 Clustering Technique Image Segmentation Based on Thresholding Limitations

11.7 Edge Linking

11.8 Watershed Transformation

11.9 Let Us Sum Up

11.10 Further Reading

11.11 Answer to Check Your Progress

11.12 Model Questions

11.1 LEARNING OBJECTIVES

After going through this unit, you will be able to

- define image segmentation techniques
- apply edge-based segmentation, line-based segmentation
- apply region-based segmentation, and boundary detection

11.2 INTRODUCTION

Image segmentation involves dividing an image into meaningful structures. These structures depict the objects that constitute an image. Segmentation is the first step in image analysis and pattern recognition. Different features in images are identified and extracted by segmentation algorithms. Accuracy of extracted features decide the accuracy of automatic

recognition algorithms. Thus, a suitable and rough segmentation algorithm should be chosen very carefully. Selection of a suitable algorithm is highly application dependent. Image segmentation is one of the most difficult tasks in image processing. Generally, many image processing tasks are aimed at finding a group of pixels in an image that are similar or connected to each other in some way. We are showing the step image analysis, object representation, visualization, understanding, classification in Fig.11.1.

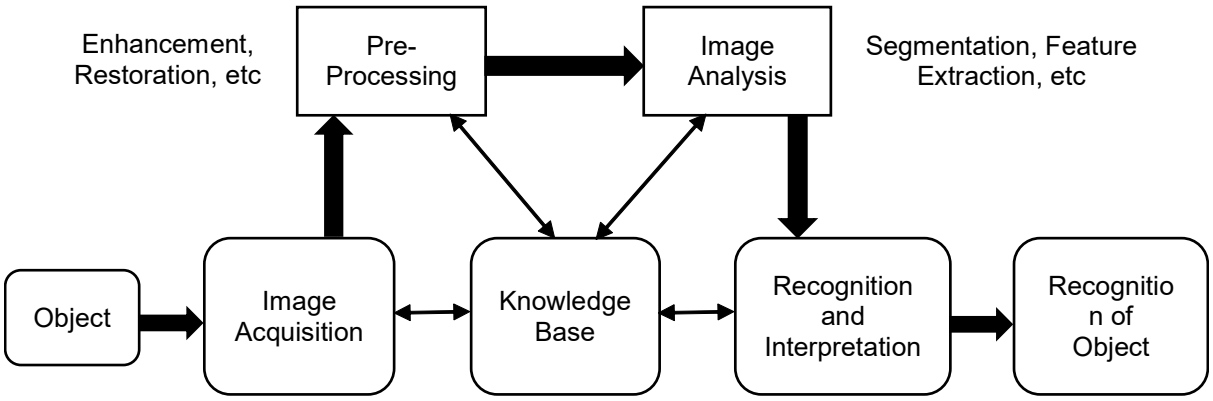


Fig.11.1: Image processing steps

An image scene is processed for autonomous machine perception and is mapped to knowledge. Here, the system tries to imitate human recognition and ability to make decision according to the information contained in the image, showing in Fig.11.2.

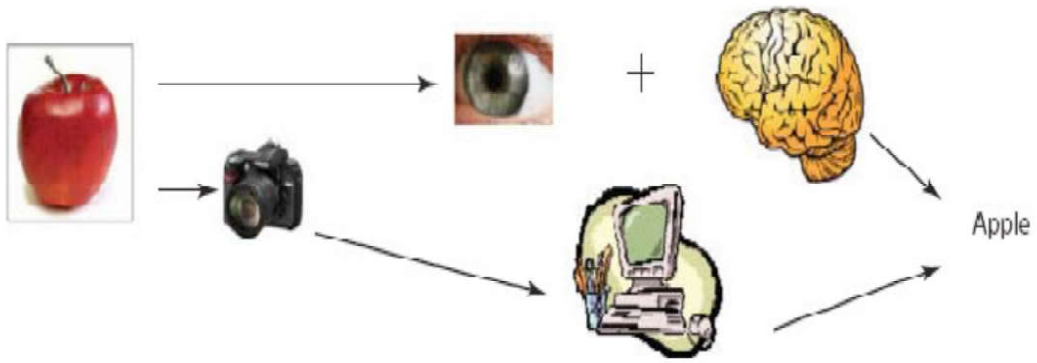


Fig.11.2: Image to knowledge mapping

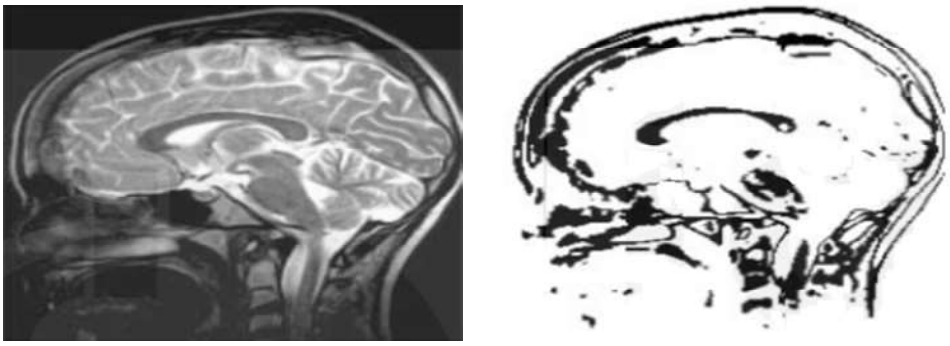
Image segmentation is the fundamental step in image analysis, understanding, interpretation and recognition tasks. It is the process of

decomposing a scene into different components. Segmentation partitions an image into multiple homogeneous regions with respect to some characteristics. In practice, it groups the pixels having similar attributes into one group. The result of image segmentation is a set of regions that collectively cover the entire image or a set of contours extracted from the image. Each pixel in a particular region is similar to the other pixel with respect to some characteristics such as colour, edge, texture, etc. Segmentation is an intermediate stage between low- level and high-level image processing tasks. Low-level tasks manipulate pixel values for irregularity correction or for image enhancement, whereas high-level tasks manipulate and analysis a group of pixels that convey some information.

Segmentation is the most important step in automated recognition system which has numerous applications and some of them are discussed below:

1. **Medical imaging**

   Segmentation is used to locate tumours, measure tissue volumes, computer aided surgery, diagnosis, treatment planning, study of anatomical structures, etc. Fig.11.3 shows the segmented portion of brain.



(a) Original image (b) Segmented image

Fig.11.3: Example of image segmentation in medical imaging

2. **Satellite imaging**

   Segmentation is used to locate objects like roads, forest, etc. in satellite images for monitoring of ecological resources like seasonal dynamics of vegetation, deforestation, mapping of underground

minerals etc. Segmentation methods based on texture, colour, wavelength, etc. are generally used here.

3. **Movement detection**

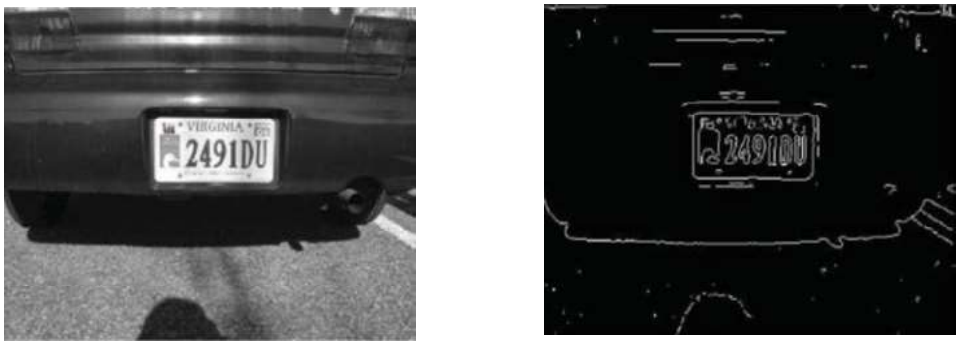
   Monitoring crowded urban environment is a goal of modern vision systems. Knowledge of the size of the crowd and tracking its motion can be used to monitor traffic intersection. Intelligent walk signal system can be designed based on the number of people waiting to cross the road. Knowledge of the size of the crowd is helpful in general safety, crowd control and planning urban environment.

4. **Security and surveillance**

   Security of the national assets such as bridges, dams, tunnels etc. is critical in today's world. Automated smart system to detect 'suspicious' movements or activities, to detect left baggage or vehicle is crucial for safety. Automated face detection systems try to match a criminal's face in a crowded place.

5. **License plate recognition (LPR)**

   Automated license plate reading is a very useful and practical approach as it helps in monitoring existing and illegally acquired license plates. LPR can be used in private parking management, traffic monitoring, automatic traffic ticket issuing, automatic toll payment, surveillance and security enforcement. Fig.11.4 shows the segmented license plate.



(a) Original license plate (b) Segmented license plate

Fig.11.4: Example of image segmentation in LPR

6. **Industrial inspection and automation**

   Here the objective is to find mission components (Integrated Circuits (IC)), in a Printed Circuit Board (PCB) or a missing pill in a packet of medicine or to check the quality of the baked biscuits etc. These automated systems help in reducing manpower in the plants and in reducing error committed by humans due to repetitive task.

7. **Robot navigation**

   Robots are used in a variety of industrial, medical and research applications. Pre-assigned tasks can be repeatedly and efficiently done by robots in hazardous environment. Image analysis and understanding help robots navigate and execute the given task.



CHECK YOUR PROGRESS A

1. What is image segmentation?

.....

.....

2. State two applications of image segmentation techniques.

.....

.....

11.3 CLASSIFICATION OF IMAGE SEGMENTATION

Image segmentation partitions an image into set of regions. In many applications, regions represent meaningful areas in an image. In other applications, regions might be set of border pixels grouped into structures such as line segments, edges, etc. The level of partitioning of an image depends on the problem to be solved. Segmentation should stop when the objects of interests have been isolated. Segmentation has two objectives:

- To decompose an image into regions for further analysis.
- To perform a change of representation of an image for faster analysis.

There are a number of segmentation techniques available. Segmentation is application dependent. A single segmentation technique may not be

suitable for different applications. Hence, these techniques have to be combined with the domain knowledge in order to effectively solve the problem. Generally, all segmentation algorithms are based on one of the two properties of gray level values of pixels.

a) Discontinuity

In this approach, images are portioned based on the difference or discontinuity of the gray level values. This means segmentation contour starts from the point where pixels appear to have very different gray values. Edge based segmentation methods fall in this category.

b) Similarity

Images are partitioned based on the similarity of the gray level values of the pixel according to a pre-defined criterion. This means as long as pixels have gray values close to each other, they are considered to be in the same segmentation block. Thresholding, region-based clustering, and matching-based segmentation techniques fall in this category.

We categorize segmentation techniques as follows:

1) Edge based segmentation

Edge based method is a commonly used technique to detect boundaries and discontinuities in an image. With this technique, detected edges are assumed to represent object boundaries and used to identify objects. The assumption here is that every part of the object is sufficiently uniform such that they can be separated on the basis of discontinuity alone.

2) Region based segmentation

Edge based techniques find the object boundaries and then locate the objects, whereas region-based techniques look for uniformity within a sub-region based on a suitable property like intensity, colour, texture etc. Region based segmentation starts in the middle of an object and then ‘grows’ outwards till it meets the object boundary.



CHECK YOUR PROGRESS B

1. Segmentation algorithms are generally based on which two properties of intensity?

.....

.....

11.4 LOCAL AND GLOBAL SEGMENTATION

In local segmentation, small windows are generated on a whole image and segmenting sub image. Global segmentation constitutes segmenting whole image. Global segmentation is used on large number of pixels. Local segmentation is used to deal with lesser number of pixel when compared to global segmentation. A global thresholding technique is one which makes use of a single threshold value for the whole image, whereas local thresholding technique makes use of unique threshold values for the partitioned sub images obtained from the whole image.

11.5 IMAGE SEGMENTATION TECHNIQUES

In this section, we shall discuss the various techniques of image segmentation.

11.5.1 Edge-based Segmentation

As object can be fully represented by its edges, segmentation of an image into separate objects can be achieved by location edges of these objects. Edge detection is a fundamental tool in image processing and computer vision. The aim is to identify points in an image at which the brightness changes sharply. Changes in image brightness can be due to discontinuities in depth, discontinuities in surface orientation, changes in material property or variation in scene illumination. Result of an edge detection algorithm is a set of connected curves that may indicate object boundaries. This significantly reduces the amount of data to be processed, making the overall object representation algorithm simpler and quicker.

If edge detection step is successful, subsequent task of interpreting the information is also successful. A typical approach to segmentation using edges is:

a) Compute an edge image, containing all edges of an original image.
b) Process the edge image so that only closed object boundaries remain.
c) Transform the result to an ordinary segmented image by filling in the object boundaries.

The first step of edge detection is discussed in the following sections. Difficulty lies in the second step which often requires removal of edges that are caused by noise or other artefacts, bridging the gaps at locations where no edge was detected and decision to connect the edges that make up a single object.

An edge may be closely defined as a line of pixels showing an 'observable' difference. For example, consider two sub-images shown in Fig.11.5. In the sub image Fig.11.5 (b), there is a clear difference between the gray levels in the second and third columns which can be easily picked by the human eye where as in the sub-image of Fig.11.5 (a) no such difference can be seen.

51	52	53	59
54	52	53	62
50	52	53	68
55	52	53	55

11.5 (a)

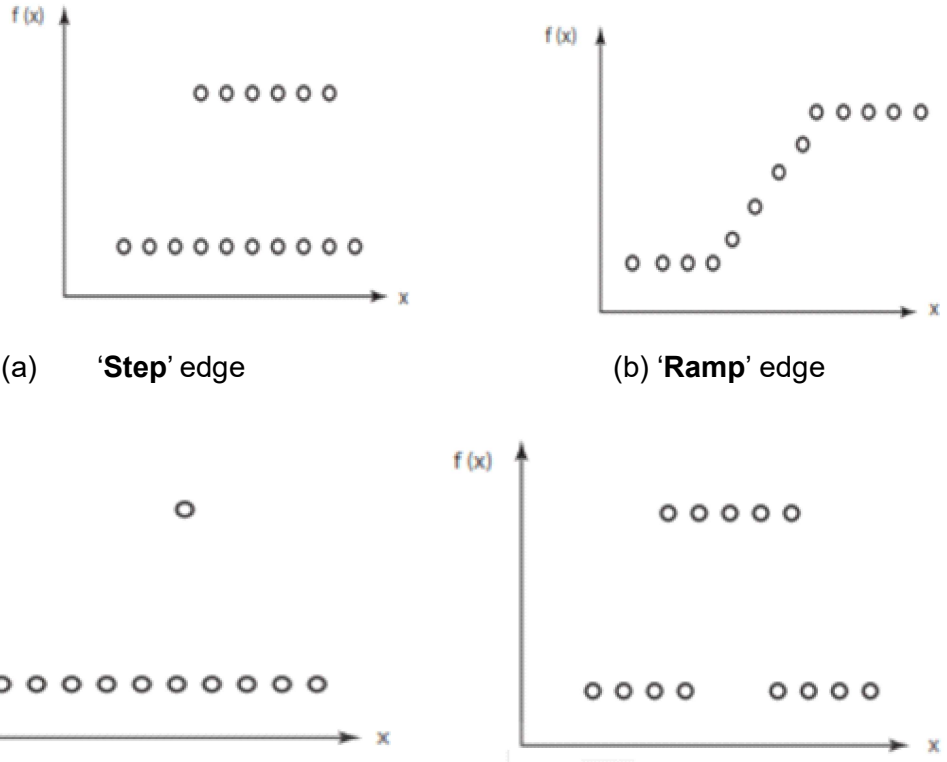
50	53	150	160
51	53	150	180
58	55	154	170
54	56	156	155

11.5 (b)

Fig.11.5: Two sub-images

Different edge models have been defined based on their intensity profiles. Fig.11.6 (a) shows a '**step**' edge which involves transition between two intensity values over a distance of one pixel. This is an ideal edge where no additional processing is needed for identification. A '**ramp**' edge is shown in Fig.11.6 (b), where the transition between two intensity levels takes place

over several pixels. In practice, all the edges get blurred and noisy because of focusing limitations and inherent noise present in electronic components. Fig.11.6 (c) shows a '**point**' which is defined as only one or two isolated pixels having different gray level values as compared to its neighbours. Fig.11.6 (d) shows a '**roof**' edge that is defined as multiple pixels having same or similar gray level values which are different from their neighbours.



(a) '**Step**' edge (b) '**Ramp**' edge

(c) '**IsolatedPoint**' edge (d) '**Roof**' edge

Fig.11.6: Gray Levels Across Edges

The first order derivative at a point x of a one-dimensional function $f(x)$ is given by

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

The second order derivative of is given by

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

Let the values of the ramp edge in Fig.11.6 (b) from left to right be

f(x)=	20	20	20	20	20	50	100	180	180	180	180	180
1st derivative f′(x)= [using Eq. (1)]	-	0	0	0	0	+30	+50	+80	0	0	0	-
2nd derivative f′′(x)= [using Eq. (2)]	-	0	0	0	0	30	20	-80	0	0	0	0

Generally, for the implementation of first or second derivative, masks are generated. These masks are convolved with the image to get the result.
For 3 x 3 mask shown in Fig.11.7, the output is calculated by,

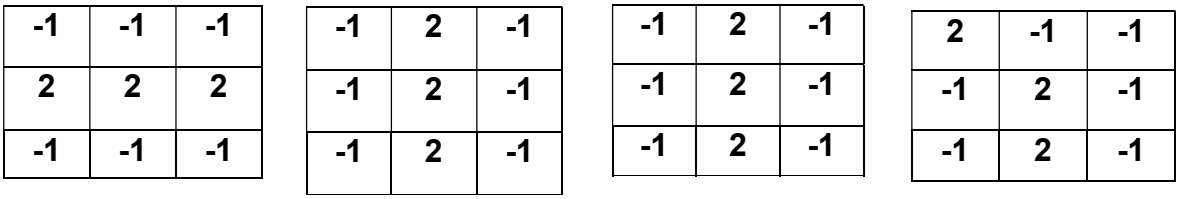
$$g(x,y) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} f(x+i, y+j)\, w(i,j) \qquad (3)$$

Where is the image with which mask w is being multiplied.

W (-1,1)	W (0, -1)	W (1, -1)
W (-1,0)	W (0,0)	W (1,0)
W (-1,1)	W (0,1)	W (1,1)

Fig.11.7: 3 x 3 mask

Now, before we discuss the edge detection approaches, let us discuss line detection. A line can be a small number of pixels of a different colour or gray level on an otherwise unchanging background. For the sake of simplicity, it is assumed that the line is only single pixel thick. Fig. 11.8 shows line detection masks in various directions. The first mask in Fig.11.8 (a) responds strongly to lines (one pixel thick) oriented horizontally and the second mask in Fig.11.8 (b) responds to vertical lines. Third and fourth in Fig.11.8 (c) and Fig.11.8 (d) masks respond to lines at and respectively.



(a) Horizontal (w_1) **(b)** Vertical (w_2) **(c)** (w_3) **(d)** (w_4)

Fig.11.8: Line Detection Mask

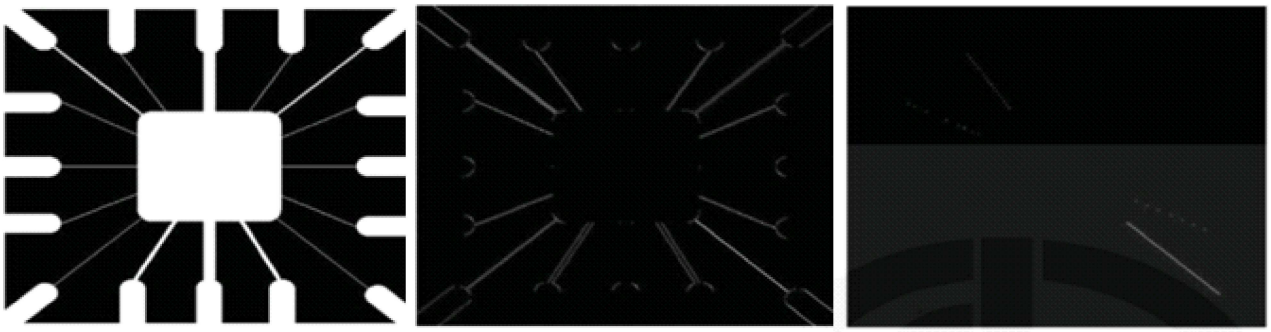
Let g_1, g_2, g_3, and g_4 be the response of each of these masks from left to right respectively. If then that point is more likely to be associated with horizontal line. Consider the electronic circuit shown in Fig.11.9 (a), the results of applying masks of Fig.11.8 (a) to Fig.11.8 (d) on this circuit are shown in Fig.11.9 (a) to Fig.11.9 (g).

Edge detection technique:

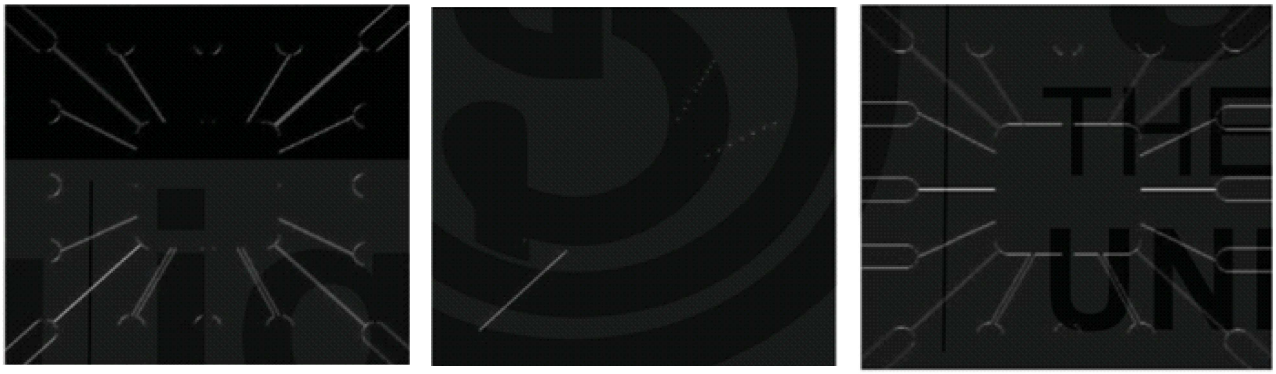
Edge detection is the most common approach used in segmentation. A typical edge may be the border between a block of red colour and a block of yellow colour. Edge can also be the boundary between two regions with relatively distinct gray level properties. Computation of a local derivative operator can enhance edges. Fig.11.10 shows the response of first and second derivative to light strip on dark background (a) and dark strip on light background (b). Edge is the smooth change in gray levels. **First derivative** is positive at the leading edge of transition and negative at the trailing edge of transition. It is zero in areas of constant gray level. The response of second derivative is different from first derivative. It is positive for dark side, negative for light side and zero in constant area. **'Magnitude' of first derivative** is used to detect the presence of an edge. **'Sign'** of second derivative is used to determine whether edge pixel lies on dark side or on light side of edge. **'Zero crossing'** is at the midpoint of a transition in gray level.

Edge detection is a non-trivial task. Edge-detection is not as simple as it looks. In practice, edges are corrupted by noise and blurring. This can be illustrated by the following examples of edge detection on a one-dimensional array. In the Fig.11.11 (a), we can intuitively say that there is an edge between 4^{th} and 5^{th} pixel. But, if the intensity difference between 4^{th} and 5^{th} pixel is

smaller because of noise and if intensity difference between the 5th and 6th pixels is higher as in Fig.11.11 (b), it would not be easy to identify the location of the edge precisely. As the edges are corrupted by noise, we observe multiple edges instead of a single edge.



(a) The original image (b) Result from -45 mask (c) -45 line



(d) Result of +45 mask (e) +45 line (f) Result of horizontal mask



(g) horizontal line

Fig.11.9: Circuit

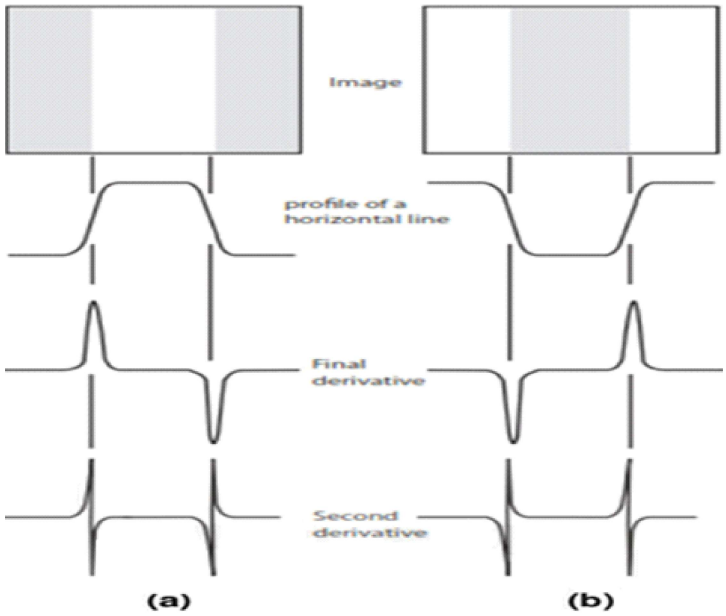


Fig.11.10: First and second derivative to (a) light strip on dark background (b) dark strip on light background

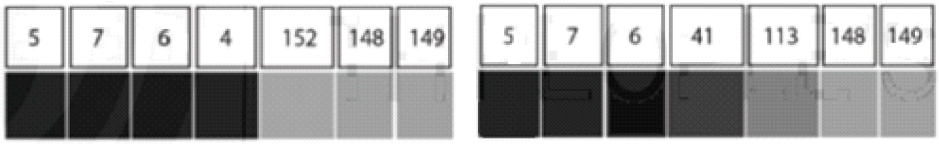


Fig.11.11 (a) **Fig.11.11 (b)**

Fig.11.11: Examples of edge detection on a one-dimensional array

11.5.1.1 Gradient operator

There are many edge detection methods. Broadly, we can classify them into two categories*:* **First order Gradient based search method**, and **Laplacian based zero crossing method**. The first order derivative-based search methods detect edges by computing the gradient magnitude and then searching for local directional maxima of the gradient magnitude. The zero-crossing based methods search for the zero crossing in a second order derivative computed from the image to find edges. Gradient is a first order derivative and is defined for an image $f(x, y)$ as

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \qquad (4)$$

It points in the direction of maximum rate of change of f at a point (x, y). Magnitude of the gradient is $mag\ (\nabla f) = \left[G_x^2 + G_y^2\right]^{1/2}$.

This can be approximated as and direction of the gradient is given by

$(\nabla f) = |G_x| + |G_y|$, where is measured with respect to x-axis.

Now, we present again discussion on various gradient operators: **Prewitt operator** and **Sobel operator**.

1) Prewitt operator

It uses a 3 x 3 size mask that approximate first derivative. The x direction mask and y direction mask are shown in Fig.11.12. The approach used is

$G_x = (Z_7 + Z_8 + Z_9) - (Z_1 + Z_2 + Z_3)$

and $G_y = (Z_7 + Z_8 + Z_9) - (Z_1 + Z_2 + Z_3)$

The convolution results in greatest magnitude indicating gradient direction. These are also called 'Compass operators' because of its ability to determine gradient direction.

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

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

Fig.11.12: Prewitt operator

2) Sobel operator

Sobel operator is similar to Prewitt's, but it uses a weight of 2 in the centre coefficient to give it little prominence. Soble x and y direction mask is given in Fig.11.13.

$G_x = (Z_7 + 2Z_8 + Z_9) - (Z_1 + 2Z_2 + Z_3)$

and $G_y = (Z_3 + 2Z_6 + Z_0) - (Z_1 + 2Z_4 + Z_7)$

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

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

Fig.11.13: Sobel operator

Sobel operator has the advantage of providing both a derivative and a smoothing effect. This smoothing effect has noise suppression characteristics. Diagonal edges can be detected by Prewitt and Sobel masks by rotating the earlier masks by counter clockwise. Fig.11.14 shows the two masks.

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

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

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

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

(a) Prewitt (**b)** Sobel

Fig.11.14: Prewitt and Sobel Masks for Diagonal Edge Detection

11.5.1.2 Laplacian operator

Laplacian, a second order derivative is defined for a 2D function $f(x, y)$, as

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

Two Laplacian masks are shown in Fig. 11.15. The Laplacian equation for Fig.11.15 (a) is

$$\nabla^2 f = 4Z_5 - (Z_2 + Z_4 + Z_6 + 8_8)$$

Similarly, the Laplacian equation for Fig.11.15 (b) is

$$\nabla^2 f = 8Z_5 - (Z_1 + Z_2 + Z_3 + Z_4 + Z_6 + Z_7 + Z_8)$$

The centre coefficient of these operators are positive and other coefficients are negative. Sum of all the coefficients is zero.

0	1	0
1	-4	1
0	1	0

(a)

1	1	1
1	-8	1
1	1	1

(b)

Fig.11.15: 3 x 3 Laplacian

Laplacian is hardly used in practice for edge detection because it is very sensitive to noise, and produces double edges. Edge direction is not detected by Laplacian. It is used to find the location of edge using zero crossing detection. When the image contains noise, the best approach is to use Laplacian of 2D Gaussian function for edge detection application. 2D Gaussian filter is given by

$$H(x,v) = e^{-1x^2+y^2/2\pi^2} \text{ or } H(r) = e^{-r^2/2\sigma^2},$$

Where, $r^2 = x^2 + y^2$ and σ is the standard deviation.

First derivation of Gaussian filter is $H'(r) = \frac{1}{\sigma^2} r e^{-r^2/2\sigma^2}$.

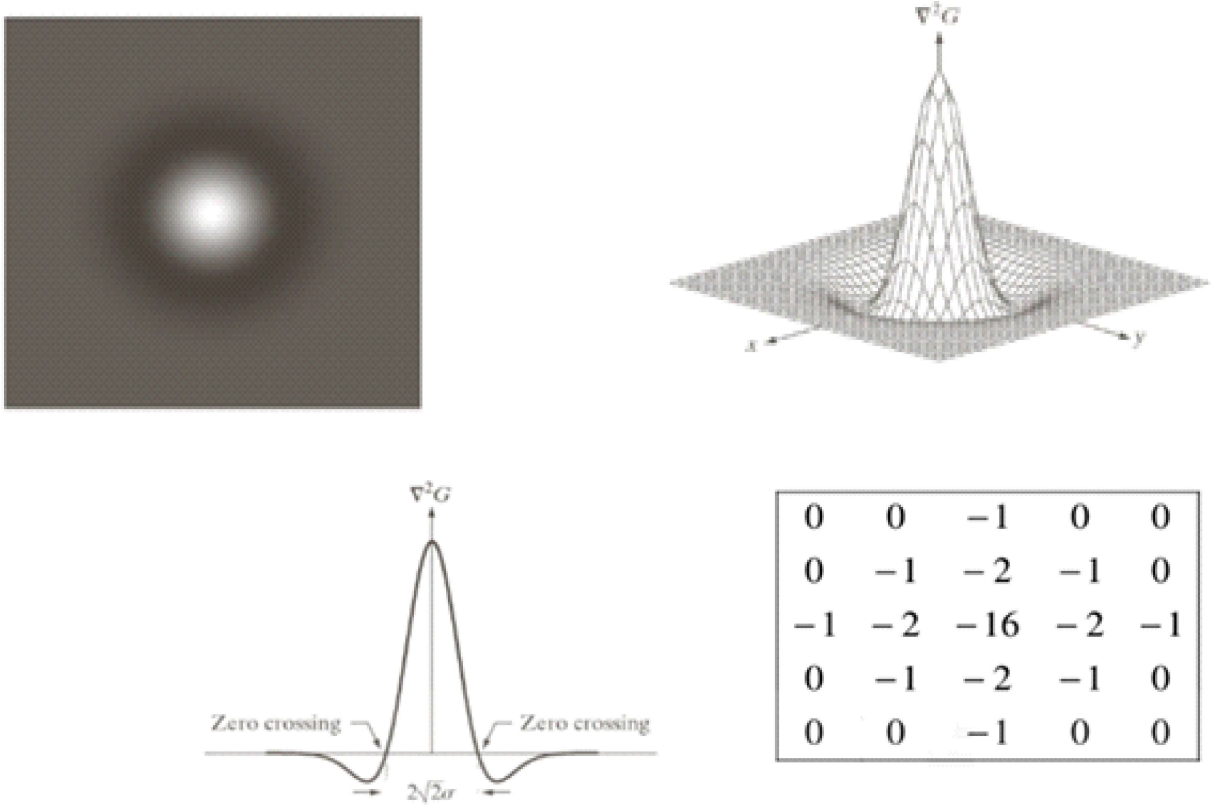
Second derivative of Gaussian filter is

$$H'(r) = \frac{1}{\sigma^2}\left(\frac{r^2}{\sigma^2} - 1\right)e^{-r^2/2\sigma^2} .$$

After returning to original coordinates x, y and introducing a normalizing coefficient C, a convolution mask of LOG (Laplacian of Gaussian) operator is given by

$$H''(x,y) = C\left(\frac{x^2 + y^2 - \sigma^2}{\sigma^4}\right)e^{-(x^2+y^2)/2\sigma^2}$$

A 5×5 LOG mask is given in Fig.11.16. Due to its shape, LOG is also known as 'Mexicanhat'. Computing second derivative in this way is robust and efficient. Zero crossings are obtained at $r = \pm\sigma$.



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

Fig.11.16: LOG as an image, LOG 3D plot, LOG

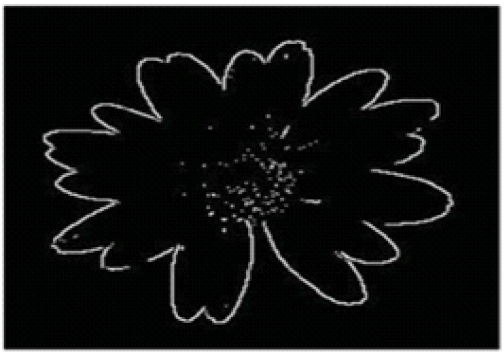
Fig.11.17 (a) is the input image, Fig. 11.17 (b) is the output of Prewitt filter, Fig.11.17 (c) is the output of Robert filter, Fig.11.17 (d), Fig.11.17 (e), and Fig.11.17 (f) are outputs of Laplacian, Canny and Sobel filters respectively. As it is clear from the figures, each filter extracts different edges in the image. Laplacian and Canny filters extract lot if inner details while Sobel and Robert filters extract only the boundary. Prewitt filter extracts the entire boundary of the flower without any gaps in the boundary.



(a) Original image



(b) Output of Prewitt filter



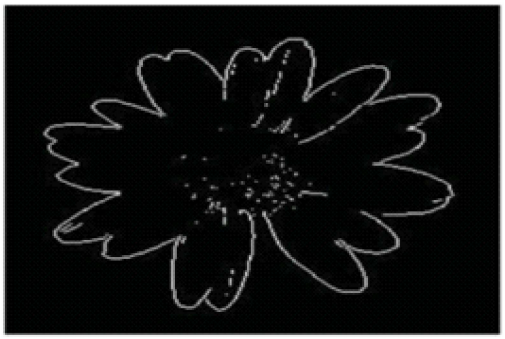
(c) Output of Robert filter



(d) Output of Laplacian filter



(e) Output of Canny filter



(f) Output of Sobel filter

Fig.11.17: (a) Original image (b) Output of Prewitt filter (c) Output of Robert filter (d) Output of Laplacian filter (e) Output of Canny filter (f) Output of Sobel filter



CHECK YOUR PROGRESS C

1. What is an edge?
2. List the properties of the second derivative around an image?

3. Define Gradient operator?
4. Why is Laplacian generally not used in its original form for edge detection?
5. Give a 5 x 5 LOG mask.

11.5.2 Region detection

Object boundary was detected to locate an object. In this section, object region will be used to locate the object. In theory, locating an object by locating either its boundary or region should result in the same object as boundary and region are just different representations of the same object. But in practice, edge-based segmentation approach may give totally different result than a region-based approach. Imperfect images, imperfect methods can be the reason which causes that the result of object detection using boundary-based approach to be different from region-based approach. Region based methods rely on the assumption that neighbouring pixels within one region have similar values. The common approach is to compare a pixel with its neighbours. A pixel belongs to the same cluster if the pixel and its neighbours satisfy the same similarity criterion. Region based segmentation methods employ two basic steps:

a) Merging and
b) Splitting.

Image segmentation using **merging** has the following steps:

Step 1: Obtain the initial segmentation of the image.

Step 2: Merge two adjacent segments to form a single segment if they are similar in same way.

Step 3: Repeat step 2 until no segment to be merged remains.

The initial segmentation can be all individual pixels. The basic idea is to combine two pixels (regions) if they are similar. The similarity criteria can be based on gray level similarity, texture of the segment, etc.

Image segmentation using **Splitting** has the following steps:

Step 1: Obtain an initial segmentation of an image.

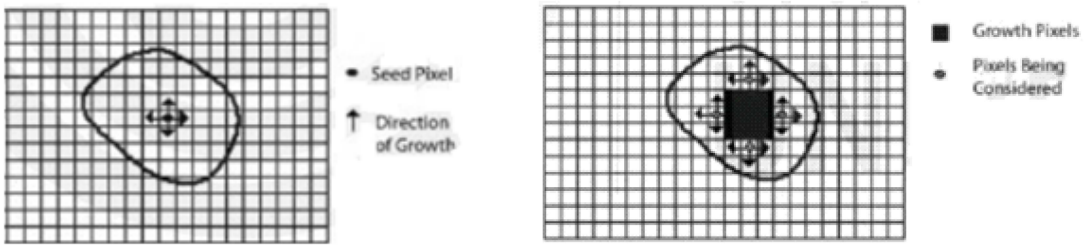
Step 2: Split each segment that is in homogeneous in some way.

Step 3: Repeat step 2 until all segments are homogeneous.

Here, initial segmentation may be the entire image (no segmentation). The criterion for inhomogeneity of a segment may be the variance of gray levels or the difference in its textures etc. Both splitting and merging methods seem to be top-down and bottom-up approach of the same method. But there is a basic difference. Merging two segments is straight forward, but in splitting, we need to know the subsegment boundary.

Region growing:

Region growing is a process of merging adjacent pixel segments into one segment. It is one of the simplest and very popular method of segmentation which is used in many applications. It needs a set of starting pixels called '**seed**' points. The process consists of picking a seed from the set and examining all 4 or 8 connected neighbours of this seed and merging similar neighbours to the seed as shown in Fig.11.18 (a). The seed point is modified based on all merged neighbours in Fig.11.18 (b). The algorithm continuous until the seed set is empty.



(a) Start of growing region (b) Growing process after a few iterations

Fig.11.18: Region

Region growing algorithm for a single seed point and same gray level value as similarity measure to merge the pixels is as follows:

The algorithm uses a data structure entity stack to keep track of seed points. Two operations push and pop are used here.

Push: puts the pixel coordinates on the top of the stack.

Pop: takes a pixel from the top of the stack.

Input to the algorithm is an image 'f'. Initial seed has coordinates of (x, y) and gray level value g at = (x, y) $f(x, y) = g$. The goal is to grow a region with all pixels having gray level value of g and assigning them gray level $k = 1, k \neq 1, k \neq g$. Let (x, y) be the coordinates of initial seed, and let be the coordinates of pixel under investigation.

The algorithm

```
Push (x, y)
        Do till stack is not empty
Pop (a, b)                      / (take input from top of the stack)
        If f (a, b) = g         / if input = desired value of g
                Set f (a, b) = 1        / segmented pixel is assigned
                                          as 1
                Push (a, b + 1          / Test all four neighbours of
                                          (a, b)
                Push (a, b – 1)         / Test all four neighbours of
                                          (a, b) by pushing it on the top
                                          of the stack
                Push (a +1, b)
                Push (a – 1, b)
        End
```

This is a recursive algorithm. The final region is extracted by selecting all pixels having gray level value as 1(k). The algorithm can be modified by changing the similarity measure to incorporate a range of values for merging. The statement if $f(a, b) = g$ can be changed to $g_1 < f(a, b) < g_2$.

Thus, if the gray level value of pixel (a, b) is between g_1 and g_2 then, it is segmented. The algorithm can be further modified to incorporate multiple seed points. In the above algorithm, only four neighbours are considered. It can be modified for eight neighbourhood. Instead of using four push operation, eight push instruction can be used with coordinates of all eight neighbours.

There are several issues about region growing which are listed below:

1) Suitable selection of seed points is important. Seed point selection is user dependent. Histogram and image properties can be taken into account by user to select the seed point.
2) Selection criterion for similarity measure is important. It generally depends on the original image and object to be segmented. Band of intensity values, colour, texture, shape are some of the similarity measures generally used.
3) Formulation of a stopping rule is also very important in region growing. Growing of a region should stop when no more pixel satisfies the similarity criteria. Algorithm can be made more powerful, if size and shape of the grown region are also considered.

Advantage: The main advantage of region growing algorithm is that it correctly separates the regions based on the property defined by the user. The algorithm is very simple to implement. Only input needed are the seed points and selection criterion. Multiple criteria can also be applied. The algorithm works well in noisy environment also.

Disadvantage: The major disadvantage of region growing is that the seed points are user dependent. Selection of wrong seed points can lead to wrong segmentation results. The algorithm is highly iterative and requires high computational time and power.

Example 1:

In the image segment given in Fig.11.19 (a) seed points are given at (3, 2) and (3, 4). Similarity criterion is gray level difference. Find segmented image, if a) T=3 and b) T=8.

	1	2	3	4	5
1	0	0	5	6	7
2	1	[1]	5	8	7
3	0	1	6	[7]	7
4	2	0	7	6	6
5	0	1	5	6	5

Fig.11.19 (a): Input image segment for example

Solution:

For $T = 3$, region growing starts with pixel $(3, 2)$. All the pixels having gray level difference <3 are assigned as a and denoted as region R_1. Another region growing starts at $(3, 4)$. All pixels with gray level value <3 are assigned as b and denoted as region R_2. The output is shown in Fig.11.19 (b).

For $T = 8$, all pixels have gray level difference less than $3 \Rightarrow$ only one region is formed, with all pixels being assigned as 'a'. The output is shown in Fig.11.19 (c).

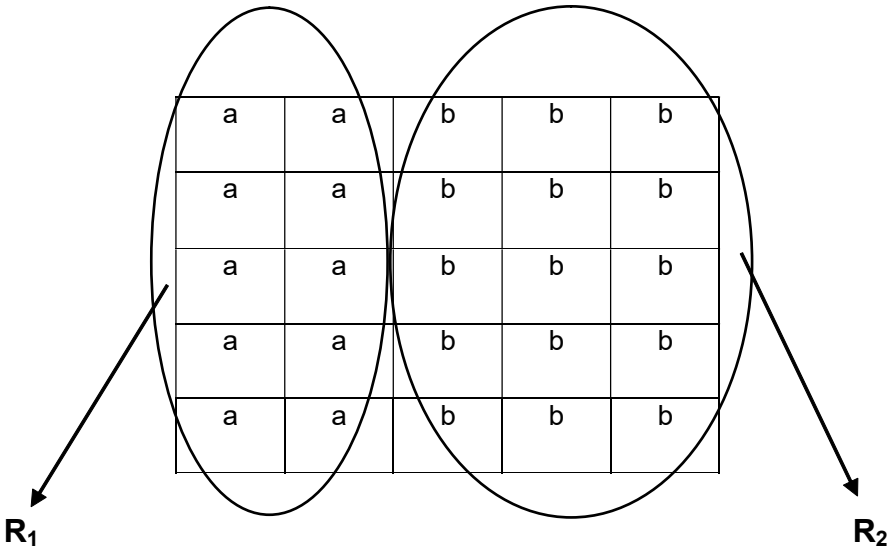


Fig.11.19 (b): Output for $T = 3$

Output for $T = 3$

a	a	a	a	a
a	a	a	a	a
a	a	a	a	a
a	a	a	a	a
a	a	a	a	a

Fig.11.19 (c): Output for $T = 8$

Example 2: Use region growing to segment the object in the image given in Fig.11.20. The seed is the centre pixel of the image. Region is grown in the following directions.

a) In horizontal and vertical directions only (4 neighbourhood)

b) In horizontal and vertical and diagonal directions (8 neighbourhood).

Similarly, criterion is the difference between two-pixel values is less than or equal to 5.

10	10	10	10	10	10	10
10	10	10	69	70	10	10
59	10	60	64	59	66	60
10	59	10	60	70	63	62
10	60	59	65	67	10	65
10	10	10	10	10	10	10
10	10	10	10	10	10	10

Fig.11.20: Input image

Solutions:

1. Region growing starts with seed point pixel with gray value 60 in the centre. It moves horizontally up and down, vertically up and down to check how much given pixel value differs from 60.

If the difference is less than equal to 5, then it is assigned as 'a' and merges with the region, else it is assigned as 'b'. Fig.11.21 (a) shows the output.

2. If diagonal elements are also included then the region grows more as shown in Fig.11.21 (b).

b	b	b	b	b	b	b
b	b	b	b	b	b	b
b	b	a	a	a	b	b
b	a	b	a	b	b	b
b	b	a	a	b	b	b
b	b	b	b	b	b	b
b	b	b	b	b	b	b

Fig.11.21 (a):
Output for 4-neighbourhood

b	b	b	b	b	b	b
b	b	b	b	b	b	b
a	b	a	a	a	b	a
b	a	b	a	b	a	a
b	a	a	a	b	b	a
b	b	b	b	b	b	b
b	b	b	b	b	b	b

Fig.11.21 (b):
Output for 8-neighbourhood

Now, let us discuss Split and Merge method. As mentioned before, there is a fundamental problem in splitting procedure. It needs suitable sub segments to be established before performing splitting. The problem of how to split has to be resolved. We can decide whether a particular segment needs to be split or not by checking if

- Gray level difference (variance) exceeds a threshold or
- The variance of texture measure exceeds a threshold or
- Histogram entropy or any other histogram measure exceeds threshold, or
- Edge pixels exist in the segment.

We start by splitting the entire image into four quadrants. The object can be in any or some or of all the four quadrants, as the object can be anywhere in the image. Then, further subdivision of the quadrants is done. Merging operation is added to the segmentation process, recursive splitting and merging of image segments is done as shown in Fig.11.22. Original image is shown in Fig.11.22 (a) and Fig.11.222 (b) shows, entire image split into four segments. Fig.11.22 (c) shows

a further split of four segments. In Fig.11.22 (d), further segmentation is done if gray level variance in the sub block is non-zero. Merging of the sub blocks having same gray levels is shown in Fig.11.22 (e). Continuing this process, we end up in two segments one being the object and other being the background.

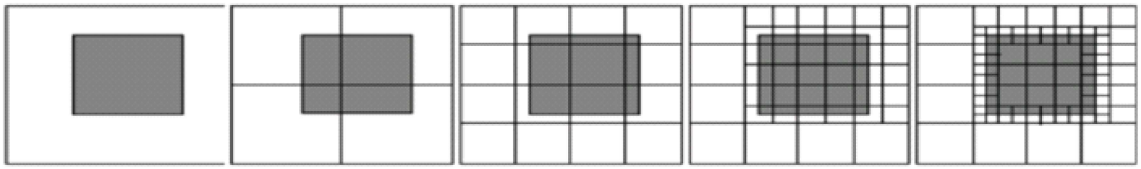


Fig.11.22 (a) **Fig.11.22 (b)** **Fig.11.22 (c)** **Fig.11.22 (d)** **Fig.11.22 (e)**

Fig.11.22: Example of split and merge segmentation

Split and merge algorithm uses '**Quad tree**' for representing segments. Fig.11.23 shows that there is a one-to-one relation between splitting an image recursively into quadrant sand the corresponding quad tree representation. However, there is a limitation in this representation, as we cannot model merging of two segments at different level of the pyramid. Let R represent the entire image. A homogeneity criterion it is selected. If the region R is not homogeneous $(H(R) = false)$, then split R into four quadrants R_1, R_2, R_3, and R_4. Any four regions with the same parent can be merged into a single homogeneous region if they are homogeneous. The steps of split and merge algorithm are as follows:

Step 1: Define an initial segmentation into regions, a homogeneity criterion and a pyramid data structure.

Step 2: If any region R in the pyramid data structure is not homogeneous (H(R) = False), split it into four child regions.

Step 3: When no further splitting is possible, merge two adjacent regions Ri which are homogeneous $(H(R_i \cup R_j) = True)$.

Step 4: Stop when no further merging is possible.

<table>
<tr><td rowspan="2">R_1</td><td colspan="2">R_{21}</td><td>R_{22}</td></tr>
<tr><td colspan="2">R_{23}</td><td>R_{24}</td></tr>
<tr><td rowspan="3">R_2</td><td>R_{411}</td><td>R_{412}</td><td rowspan="2">R_{42}</td></tr>
<tr><td>R_{413}</td><td>R_{414}</td></tr>
<tr><td colspan="2">R_{44}</td><td>R_{43}</td></tr>
</table>

Fig.11.23 (a):
Naming convention of quadrants

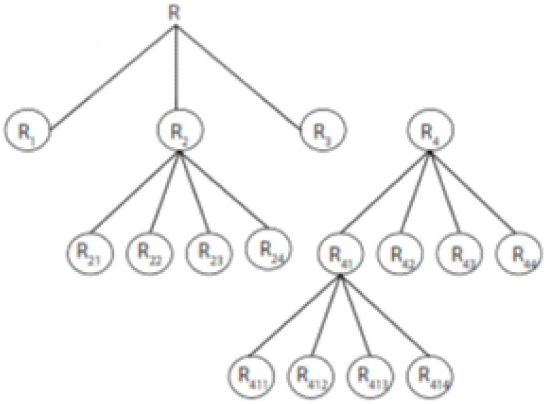


Fig.11.23 (b):
Quadtree representation

Fig.11.23: Example of recursive splitting of an image by a quad

Several modifications to this basic algorithm are possible. For example, in Step 2, merging of homogeneous regions is allowed which results in a simpler and faster algorithm.

Advantage of this algorithm: The image could be split progressively according to our required resolution because the number of splitting levels is decided by the user.

Disadvantage of this algorithm: It may produce blocky segments as splitting is done in rectangular quadrants. This problem can be solved by splitting at higher level but this will increase the computational time.

Example 3: Segment the image in the Fig.11.24 (a) by split and merge algorithm. Homogeneity criteria is the gray levels of the pixels.

Solution: The image is divided into four quadrants. Fig.11.24 (b) shows the image and its quad tree. Quadrants 2 and 3 are homogeneous. Thus, no further splitting is done. Quadrant 1 and 4 are non-homogeneous, hence they are divided further into 4 quadrants. Fig.11.24 (c) shows the splitting and corresponding quad tree. Now only one quadrant, 12 is still non-homogeneous. Hence, it is further subdivided. Fig.11.24 (d) shows the segmented image and its final quad tree structure. Now all regions are homogeneous and hence no further splitting is possible.

Now merging operation takes place between adjacent quadrant. Quadrants 43 and 44 are merged into a single region. Quadrants 123 and 124 are merged, quadrant 11 and 14 are merged. Finally merge all quadrants which are homogeneous. Fig.11.24 (e) is the final segmented image after merging is complete.

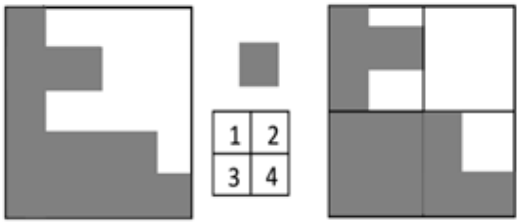


Fig.11.24 (a): Split and merge algorithm

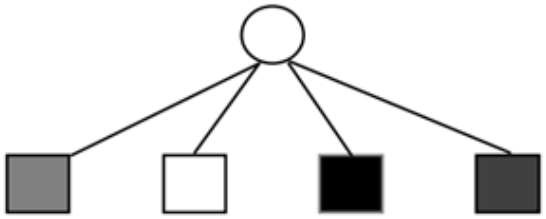


Fig.11.24 (b): Image and its quad tree

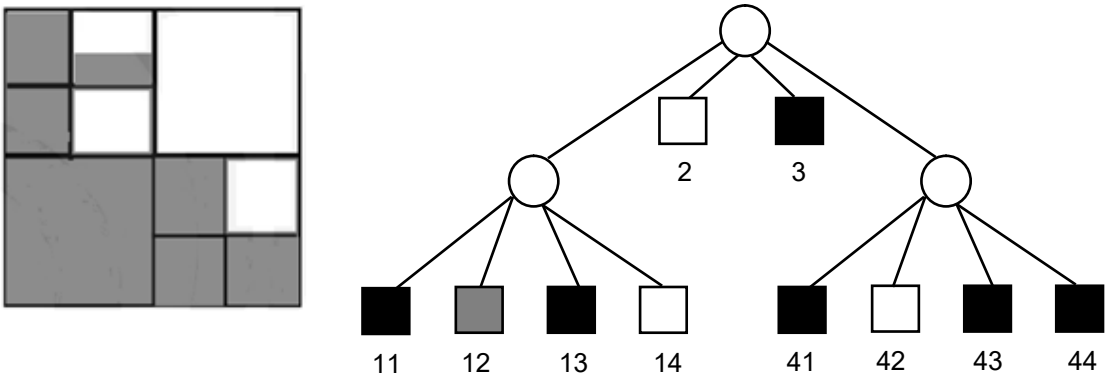


Fig.11.24 (c): Splitting and corresponding quad

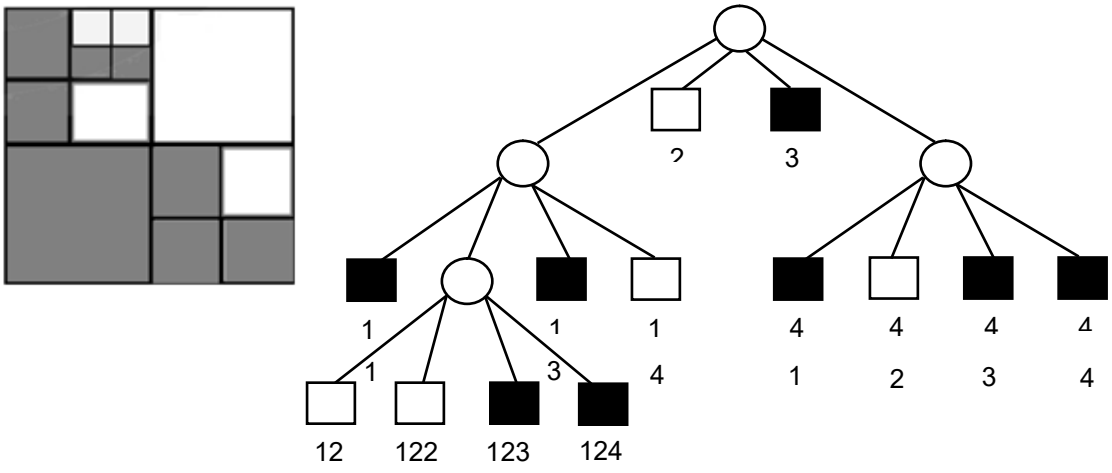


Fig.11.24 (d): Segmented image and its final quad tree

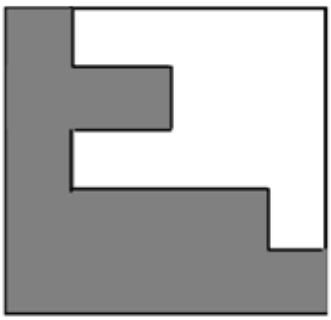


Fig.11.24 (e): Split and merge algorithm



CHECK YOUR PROGRESS D

1. Distinguish between image segmentation based on thresholding with image segmentation based on region-growing techniques.
2. Consider the image segment
3. What are the advantages or disadvantages if we use more than on seed in a region-growing technique?

11.5.3 Boundary detection

After performing image segmentation, a region may be represented in terms of:

- External characteristics (boundaries).
- Internal characteristics (texture).

A region may be described by its boundary in terms of features such as its length, the orientation of the straight line joining its extreme points and the number of concavities in the boundary. Many algorithms require the points in the boundary of a region in an ordered clockwise (or anticlockwise) direction. For boundary detection or following or tracking, the following assumptions are made:

1) The image is binary where 1=foreground and 0=background
2) The image is padded with a border of 0's so an object cannot merge with the border.
3) W limit the discussion to single regions. The extension is straight forward.

Moore Boundary Tracking Algorithm:

Given a binary region R or its boundary.

Step 1: Let the starting point b_0, be the uppermost, leftmost point in the image labelled 1.

Step 2: Denote by c_0 the west neighbour of b_0. C_0 is always a background point.

Step 3: Examine the 8-neighbours of b_0, starting at c_0 and proceeding in a clockwise direction.

Step 4: Let b_1 denote the first neighbour encountered whose value is 1.

Step 5: Let c_1 denote the background point immediately preceding b_1 in the sequence.

Step 6: Store the locations of b_0 and b_1 for use in Step 10.

Step 7: Let $b = b_1$ and $c = c_1$.

Step 8: Let the 8-neighbours of b starting at c and proceeding clockwise be denoted as n_1, n_2, …, n_8. Find the first n_k which is foreground (i.e., a "1").

Step 9: Let $b = n_k$ and $c = n_{k-1}$.

Step 10: Repeat steps 8 and 9 until $b = b_0$, that is, we have reached the first point and the next boundary point found is b_1.

The sequence of b points found when the algorithm stops is the set of ordered boundary points.

		1	1	1	1	
	1			1		
		1		1		
	1			1		
	1	1	1	1		

Fig.11.25 (a): Image

	c_0	b_0	1	1	1	
	1			1		
		1		1		
	1			1		
	1	1	1	1		

Fig.11.25 (b): Step 1 and Step 2

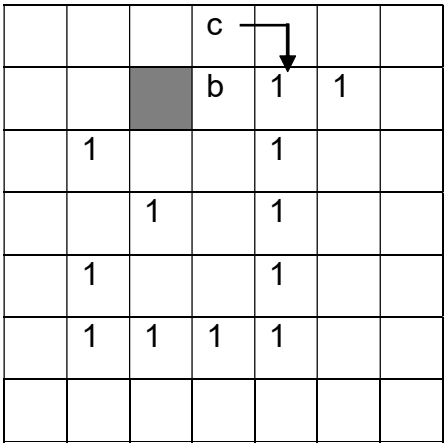


Fig.11.25 (c): Step and Step 9

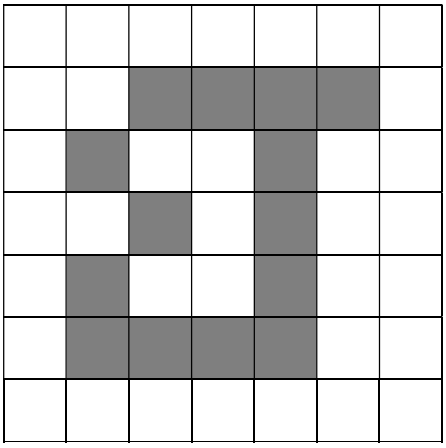


Fig.11.25 (d): Step 10

Fig.11.25: Illustration in Boundary following algorithm.

There is a need for stopping rule as stated in Step 10. We would only include the spur at the right if we stop when we reach the initial point without checking the next point. Starting from topmost leftmost point in Fig.11.26 (a) results in the Fig.11.26 (b). In the Fig.11.26 (c), the algorithm has returned to the starting point again. Rest of the boundary could not be traced.

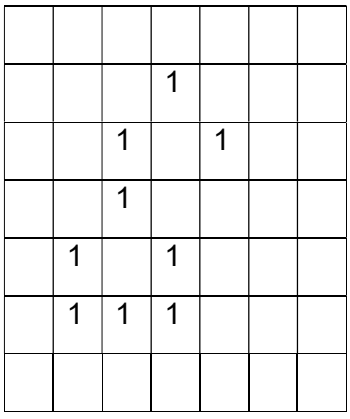


Fig.11.26 (a)

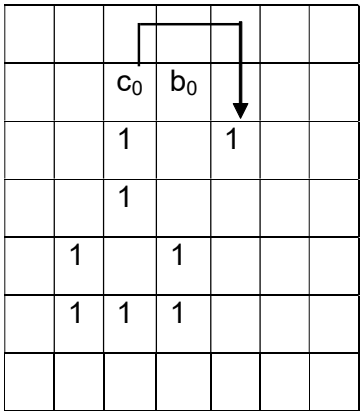


Fig.11.26 (b)

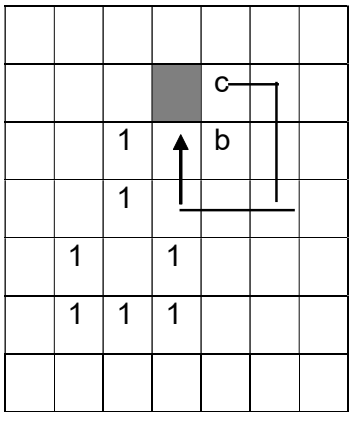


Fig.11.26 (c)

Fig.11.26: Example of erroneous result of Boundary Detection Algorithm

Chain Codes:

Chain codes are used to represent a boundary by a connected sequence of straight-line segments of specified length and direction. Freeman code [1961] represent a boundary by the sequence of straight-line segments of specified length and direction. The direction

is coded by a numbering scheme (4 or 8 connectivity) as shown in Fig.11.27.

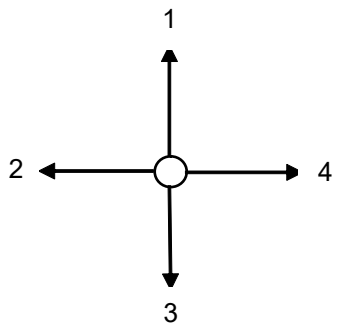


Fig.11.27 (a): 4-direction chain code

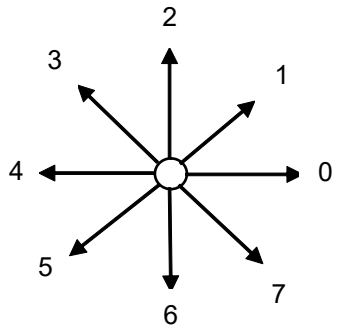


Fig.11.27 (b): 8-direction chain code

Fig.11.27: Direction numbers

Images are acquired and processed in a grid format with equal spacing in the x direction and y direction. Directions, so a chain code can be generated by the following a boundary in clockwise direction and assigning a direction to the segment connecting every pair of pixels. To avoid noise degradation and long chains a resampling of the image grid is commonly used to describe the boundary at a coarser level as shown in Fig.11.28 (a), Fig.11.28 (b) shows resampled points and Fig.11.28 (c) shows 8-direction chain code.

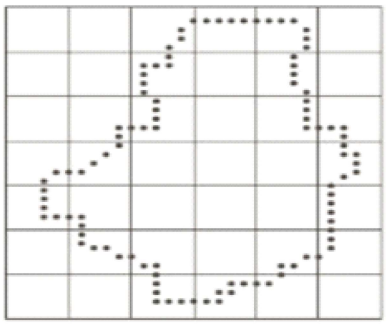


Fig.11.28 (a): Digital boundary

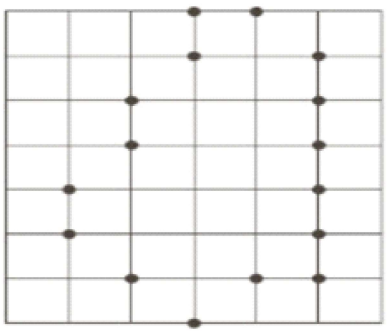


Fig.11.28 (b): Result of with resampling grid resampling

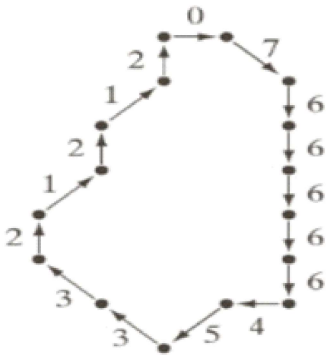


Fig.11.28 (c): 8-direction chain coded

Fig.11.28

If we start from topmost leftmost corner, chain code for Fig.11.28 is

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

The chain code depends on the starting point. To normalize it, we treat the code as a circular sequence of direction number sand redefine the starting point so that the resulting sequence forms an integer of minimum magnitude. To account for rotation, we use the first differences of the chain code instead of the code itself.

The first difference is obtained by counting the number of direction changes in counter clockwise direction that separate two adjacent elements of the code.

First difference is **6 7 7 0 0 0 0 6 1 1 0 7 7 1 7 1**

First value is calculated by considering the code as a circular sequence of integers, hence coding the sign changes from 2 to 0 counter clockwise and so on.

Example 4: Find chain code and first difference of the following boundary shapes.

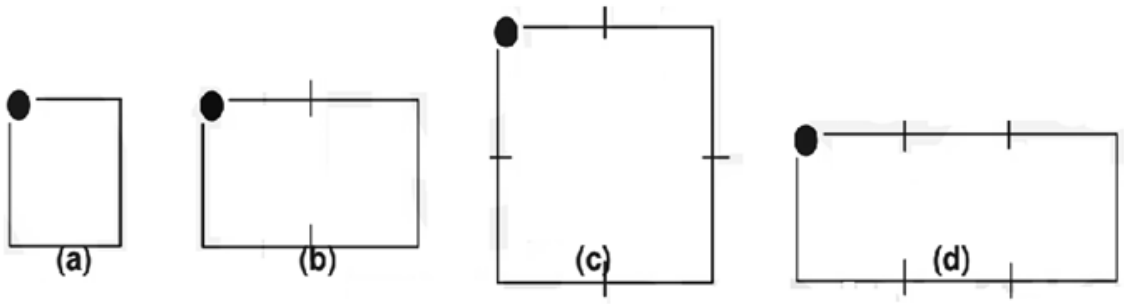
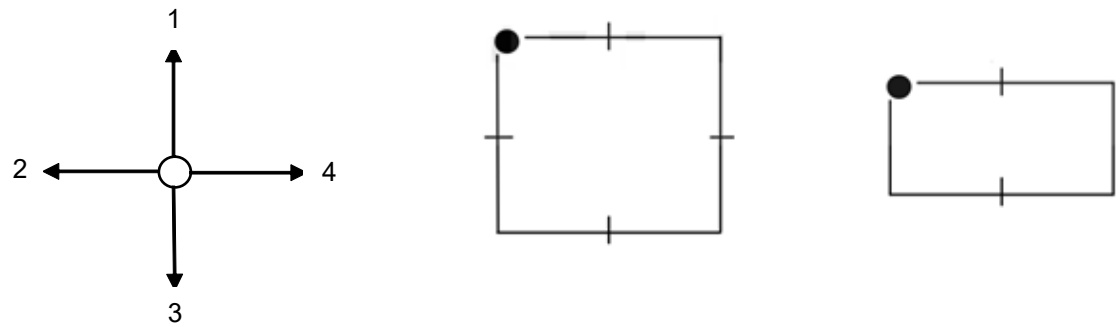
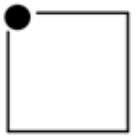


Fig.11.29

Solution:



Chain code:	0 0 3 3 2 2 1 1	0 0 3 2 2 1
Difference:	3 0 3 0 3 0 3 0	3 0 3 3 0 3





Chain code: 0 0 3 2 2 1 0 0 0 3 2 2 2 1

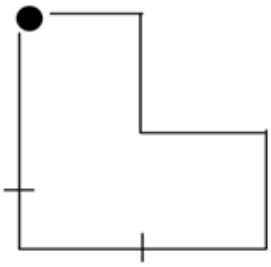
Difference: 3 0 3 3 0 3 3 0 0 3 3 0 0 3

4-connected codes are used in this example. First differences are computed by treating the chain as a circular sequence.



CHECK YOUR PROGRESS:

1. Find chain code and first difference of the following boundary shape.



11.6 CLUSTERING TECHNIQUE ON IMAGE SEGMENTATION

Clustering is a method to perform image segmentation of pixel-wise segmentation. Clustering-based segmentation groups pixels into clusters based on their similarity to the feature. It is used in image compression and content-based image retrieval. It is also used when objects in an image have distinct features that feature vectors can capture. In this type of segmentation, we try to cluster the pixels that are together. There are two approaches for performing the segmentation by hierarchical clustering.

1) Clustering by Merging
2) Clustering by Divisive

1) Clustering by Merging or Agglomerative Clustering:

This method uses a bottom-up approach, which involves assigning the pixel nearest to the cluster. The algorithm for performing the agglomerative clustering as follows:

- Take each point as a separate cluster.
- For a given number of epochs or until clustering is satisfactory.
- Merge two clusters with the smallest inter-cluster distance.
- Repeat the above step.

2) Clustering by Division or Divisive Clustering:

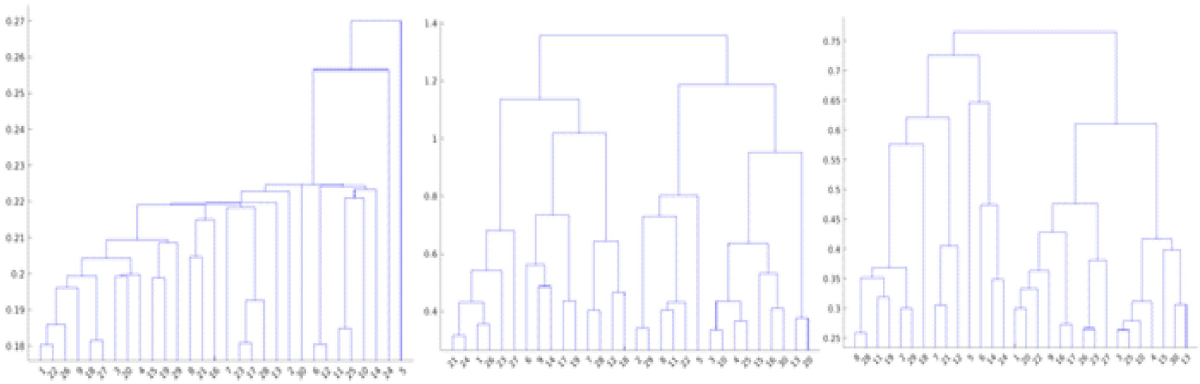
This method uses a top-down approach, which involves assigning the pixel nearest to the cluster. The algorithm for performing the agglomerative clustering as follows:

- Construct a single cluster containing all points.
- For a given number of epochs or until clustering is satisfactory.
- Split the cluster into two clusters with the largest inter-cluster distance.
- Repeat the above steps.

The agglomerative clustering is represented by Dendrogram. It can be performed in three methods:

- By selecting the closest pair for merging.
- By selecting the farthest pair for merging.
- By selecting the pair which is at an average distance (neither closest nor farthest).

The following dendrogram represents various types of clustering:



(a) Nearest clustering (b) Average clustering (c) Farthest clustering

Fig.11.30: Different types of agglomerative clustering

Here, we will be discussing how to perform the K-Means Clustering.

K-Means Clustering:

K-Means clustering is a very popular clustering algorithm which applied when we have a dataset with labels unknown. The goal is to find certain groups based on some kind of similarity in the data with the number of groups represented by K. This algorithm is generally used in areas like market segmentation, customer segmentation, etc. But it can also be used to segment different objects in the images on the basis of the pixel values.

The algorithm for image segmentation works as follows:

1. First, we need to select the value of in K-means clustering.
2. Select a feature vector for every pixel (Colour values such as RGB value, texture, etc.).
3. Define a similarity measure B/W feature vectors such as Euclidean distance to measure the similarity B/W any two points or pixel.
4. Apply K-Means algorithm to the cluster centers.
5. Apply connected component's algorithm.
6. Any component smaller than the threshold may be combined with a similar component until no further combinations are possible.

Following are the steps for applying the K-means clustering algorithm:

1. Select points and assign them one cluster center each.
2. Until the cluster center won't change, perform the following steps:
   a. Allocate each point to the nearest cluster center and ensure that each cluster center has one point.
   b. Replace the cluster center with the mean of the point assigned to it.
3. End

How to determine the value of K?

The objective function of the K-Means algorithm is WCSS

$$C = \sum i = 1M \sum k = 1N|xik - Ci|2C = \sum i = 1M \sum k = 1N|xki - Ci|2$$

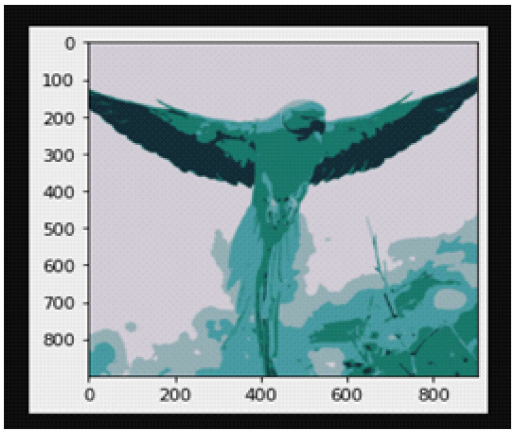
A graph plotted between WCSS and K popularly known as the Elbow method. The point where the graph makes a sharp bend is noted and the

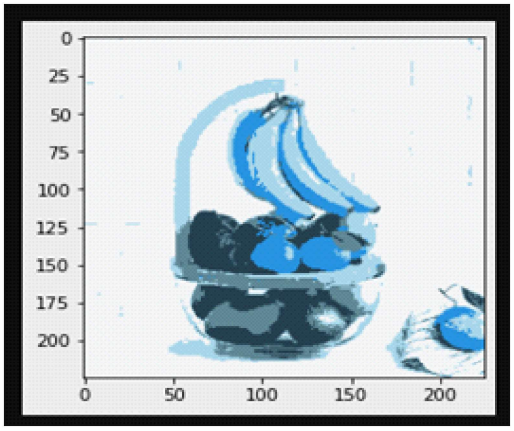
corresponding value of is taken. Determination of is done using the Elbow method, a rather brute-force approach.

Example: Python Implementation K-Means image segmentation

```
import numpy as np
import cv2
import matplotlib. pyplot as plt
from sklearn. cluster import KMeans
%matplotlib inline
image_1 = cv2.imread("/content/image_1.jpg",
cv2.IMREAD_UNCHANGED)
image_2 = cv2.imread("/content/image_2.jpg",
cv2.IMREAD_UNCHANGED)
vector_1 = image_1. reshape ((-1,3))
vector_2 = image_2. reshape ((-1,3))
kmeans_1 = KMeans (n_clusters=5, random_state = 0, n_init=5).
fit(vector_1)
c = np. uint8 (kmeans_1. cluster_centers_)
seg_data= c [kmeans_1. labels. Flatten ()]
seg_image = seg_data. reshape ((image_1. shape))
plt. Imshow (seg_image)
plt. pause (1)
kmeans_2 = KMeans (n_clusters=5, random_state = 0, n_init=5).
fit(vector_2)
c = np. uint8(kmeans_2. cluster_centers_)
seg_data= c [kmeans_2. labels. Flatten ()]
seg_image = seg_data. reshape ((image_2. shape))
plt. imshow (seg_image)
plt. pause (1)
```

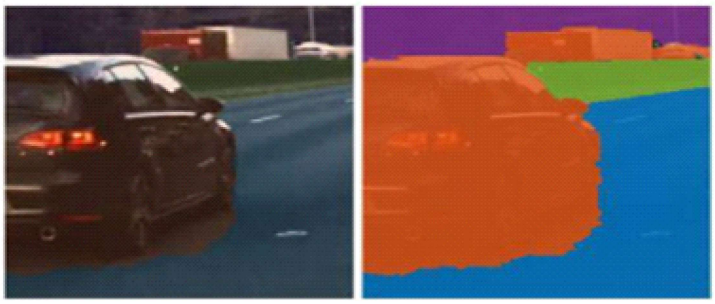
Output:



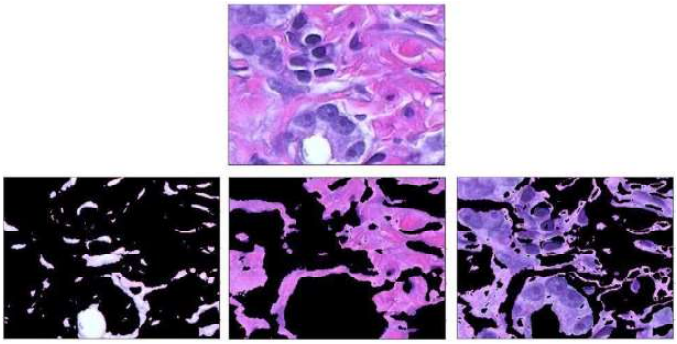


Application of Image Segmentation using K-Means clustering

1. ***Autonomous vehicles:*** During the design of the perception system of self-driving vehicles semantic segmentation is used to separate objects from the surrounding in the vicinity of the car.



2. ***Imaging in Medical Field:*** Clustering is used to separate and identify types of tissues in cancerous cell images.



3. **3-D Scene Segmentation:** In Robotic vision technology, systems are required to identify surrounding and localize objects. The Scene generated is 3D in nature. Clustering Algorithms are used for this purpose.

11.7 IMAGE SEGMENTATION BASED ON THRESHOLDING

Image segmentation based on thresholding is a fundamental technique used to partition an image into different regions based on pixel intensity values. It is commonly employed in various image processing tasks such as object detection, image analysis, and medical imaging. In thresholding, pixel values in an image are compared to a threshold value. Depending on whether the pixel's intensity is above or below this threshold, it is assigned to one of two categories (typically foreground or background). This process creates a binary image, where each pixel is either part of an object or the background.

Steps for thresholding-based image segmentation:

1. **Grayscale conversion:** If the image is in colour, it is first converted to grayscale, as thresholding generally works on single-channel images (intensity values).
2. **Choosing a threshold:** The threshold value T is chosen, which could be:
   - A global threshold:A single threshold value applied to the entire image.
   - Adaptive thresholding:The threshold value changes based on local regions of the image.
3. **Segmentation Process:** After selecting the threshold , the pixel values are compared to it:
   - If the pixel intensity at a location is greater than the threshold , it is assigned to one class (e.g., object).
   - If the pixel intensity is less than or equal to the threshold , it is assigned to another class (e.g., background)
4. **Output of the segmented image:** The result of the thresholding operation is a binary image, often represented with values 1 for the foreground and 0 for the background.

Types of thresholding

There are several methods of thresholding which are as follows:

1. Global thresholding:

- The same threshold value is applied to the entire image.
- Simple method but may not work well if lighting conditions vary across the image.

For a grayscale image, the thresholding operation can be expressed as:

$$I_{output}(x,y) = \begin{cases} 1 \; if \; I(x,y) \geq T \\ 0 \; if \; I(x,y) < T \end{cases}$$

Where:

$I(x, y)$ is the intensity of the pixel at position (x, y).

T is the threshold value

$I_{output}(x, y)$ is the binary image (1 for background, 0 for background)

2. Adaptive thresholding:

- The threshold value T is computed locally for different regions of the image. This method is helpful when there is uneven lighting or shadows in the image.
- In this case, the threshold value varies based on local regions:

$$I_{output}(x,y) = \begin{cases} 1 \; if \; I(x,y) \geq Local\ Threshold(x,y) \\ 0 \; if \; I(x,y) < Local\ Threshold(x,y) \end{cases}$$

Where $Local\ Threshold\ (x, y)$ is determined based on the local pixel intensities in the neighbourhood of (x, y).

3. Otsu's thresholding:

- An automated method that finds the optimal threshold by maximizing the variance between the foreground and background pixel values. This technique is often used when the image has bimodal (two distinct intensity peaks) histograms.

4. Multiple thresholding:

- More than one threshold value is used to segment the image into multiple regions (e.g. background, object, and other objects).

Practical Applications:

- **Object Detection**: Thresholding can help in identifying objects by isolating them from the background.
- **Medical Imaging**: Segmentation of different tissues or organs based on intensity values.

- ❖ **Image Analysis**: Identifying regions of interest, such as bright or dark spots in an image.

Example of Global Thresholding (in Python):

Here's a simple Python code using OpenCV for thresholding an image:

```
import cv2
import numpy as np
# Load the image in grayscale
image = cv2.imread('image.jpg', cv2.IMREAD_GRAYSCALE)
# Apply a global threshold
threshold_value = 127 # You can change this value
_, segmented_image = cv2.threshold(image, threshold_value, 255, cv2.THRESH_BINARY)
# Show the original and segmented images
cv2.imshow('Original Image', image)
cv2.imshow('Segmented Image', segmented_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Considerations:

- **Threshold Selection**: The performance of thresholding depends on the proper choice of threshold value. If the threshold is too high or too low, important features may be missed, or the image may be over-segmented.
- **Noise**: Thresholding is sensitive to noise. Pre-processing techniques like smoothing may be needed to improve results.

11.8 LIMITATIONS OF IMAGE SEGMENTATION

Image segmentation techniques have a lot to offer users. But they also have certain challenges. Some of the major limitations are listed as follows:

a) Data quality and diversity

The quality and diversity of the training data significantly influence the performance of image segmentation models. Limited or biased training data may lead to poor generalization and suboptimal segmentation

results. Therefore, gathering a diverse range of high-quality images that represent various scenarios, viewpoints, lighting conditions, and object classes relevant to your application is advised. Also, you must use augmentation techniques. These include rotation, flipping, scaling, cropping, and colour jittering. They make the training dataset more diverse and improve model strength. Lastly, cleaning and preprocessing the training data to remove noise, artifacts, and inconsistencies is essential. Techniques like normalization, histogram equalization, and image registration can enhance data quality.

b) Algorithm selection and tuning

As there is a wide range of algorithms available, choosing the right segmentation algorithm and optimizing its parameters for a specific task can be challenging. To overcome this challenge, you must compare and evaluate different segmentation algorithms (e.g., U-Net, Mask R-CNN, FCN) based on their performance metrics, computational efficiency, and suitability for the application domain. Additionally, techniques like grid search, random search, or Bayesian optimization can be used to conduct systematic experiments and architecture configurations of the selected algorithms. You can also combine multiple segmentation models or algorithms using ensemble techniques to leverage their complementary strengths and improve overall performance.

c) Computational requirements

Image segmentation often requires significant computational resources, especially when processing large datasets or deploying models on resource-constrained devices. To optimize segmentation algorithms for efficiency, you can use parallel processing, GPU acceleration, and model compression techniques such as pruning, quantization, and knowledge distillation. Additionally, you can also utilize cloud computing services to offload computational tasks, scale resources dynamically, and use specialized hardware accelerators for faster inference.

d) Integration with existing systems

Adding image segmentation to existing software is complex. This is true especially when dealing with many data formats. These include protocols and environments. To get over these issues, you can design segmentation systems as modular components with well-defined interfaces and APIs to facilitate seamless integration with other software systems and platforms. Additionally, you can provide comprehensive documentation, code samples, and developer support to assist integration efforts and address potential challenges.

e) Accuracy and precision

Achieving accurate and precise segmentation results is essential for many applications, but it can be challenging due to factors such as complex object shapes, occlusions, and variations in image quality. It can evaluate segmentation models using appropriate performance metrics such as Intersection over Union (IoU), Dice coefficient, pixel accuracy, and class-wise F1 score to assess accuracy and precision. You can further continuously refine segmentation models through iterative training, validation, and fine-tuning processes to improve accuracy and reduce errors.

11.9 EDGE LINKING

Edge Linking: In this step, we try to refine the edge detection by linking the adjacent edges and combine to form the whole object. The edge linking can be performed using any of the two methods below:

- *Local Processing*: In this method, we used gradient and direction to link the neighbourhood edges. If two edges have a similar direction vector then they can be linked.
- *Global processing*: This method can be done using HOG transformation

Pros:

- This approach is similar to how the humans brain approaches the segmentation task.

- Works well in images with good contrast between object and background.

Limitations:

- Does not work well on images with smooth transitions and low contrast.
- Sensitive to noise.
- Robust edge linking is not trivial and easy to perform.

11.10 WATERSHED TRANSFORMATION

In the watershed-based image segmentation, the image is splitted into different areas. Normally watershed-based segmentation will lead to an over-segmentation of the image, especially for noisy image such as medical CT images. To avoid this over segmentation, either the input image must be pre-processed or the different regions of the image must be merged based on similarity.

It is easy to understand the watershed technique by consider the image as a plain surface where high pixel values correspond to peaks and low pixel values correspond to valleys. Same as actual watersheds in irrigation, if a drop of water falls on any particular point of the contour, automatically it will flow to lower ground till it reaches a local minimum which is catchment basin and the collected points in a particular basin is called as a member of that watershed. Expressions (1) and (2) represent the watershed transform.

$$B = G * \nabla I \qquad (1)$$

Where ∇I is defined such that

$$\nabla I(i,j) = [I(i,j+1 - I(i,j)]^2 + [I(i,j+1 - I(i+1,j)]^2 \qquad (2)$$

The next step is to find the local minima of B. An element m (i, j) is said to be a local minimum if $m(i,j) < p(i_0,j_0) \forall (i_0,j_0) \in N(i,j)$ where, $N(i,j)$ represents spatial neighbourhood of the element at row i and column j in eight connectivity. The local minima m_i represents the catchment basins of the image and are each assigned with a unique label greater than zero. The final step is to follow each element in B towards its lowest valued neighbour until it merges into one of the catchment basins $m_{i.}$ Equations

(3) and (4) give the watershed and an edge map equation respectively. Once an element merges with catchment basin $m_{i,}$ it assumes the label of that basin. The above function can be represented as WS. So, the watershed W is defined as in the equation (3).

$$W = WS(G * \nabla I) \quad (3)$$

The resulting edge map E is defined in the expression (4).

$$E(i,j) = \begin{cases} 1 & \nabla W(i,j) > 0 \\ 0 & therwise \end{cases} \quad (4)$$

Where $W(i,j)$ is the label of watershed in which element (i,j) is a member after all of the watersheds have been labelled.

11.10.1 Limitations of watershed transform-based image segmentation

Noise is a great challenge in the segmentation process of an image with watershed-based technique and over segmentation is another important drawback with watershed- based image segmentation. These drawbacks can be avoided by using region margin method. A combination of wavelet and watershed-based image segmentation method can also be used to address the above problems.

11.10.2 Steps of the watershed algorithm-based image segmentation

Step 1: Read the original image (I0).

Step 2: Take L level wavelet transform of image (IL).

Step 3: Smooth the image and remove noise particles using wavelet denoising.

Step 4: Apply watershed segmentation.

Step 5: Resulting merged region image is projected onto the IL-1 layer by an Inverse wavelet transform.

Step 6: Go to step 3 and repeat the above procedure until L equal 0 (initial stage).

Acquisition the source image: Get the MRI image, which is the source image for the segmentation process obtained from a clinic.

Contrast maximization: Maximization of contrast of an image is adopted to identify and minimize number of valley point. Wavelet transform is preferred for contrast enhancement.

To maximize the contrast, first compute $\alpha_{n,s}$ using the Equation (5) and (6).

Let n be the decomposition level,

$$E_{n,s} = \frac{\sum_{i,j} |x_{n,s}(x,j)|}{N} \quad (5)$$

$$E_{n,s} = \frac{\sum_{i,j} |x_{n,s}(x,j)|}{N} \quad (6)$$

Where s is the sub band which indicates $LH, HL, \text{and } HH$. $E_{n,s}$ is the average absolute value of a specific sub bands s in level n. N is the number of coefficients of each sub band, $x_{n,s}(i,j)$ is enhanced sub band from each sub band $x_{n,s}(i,j)$ and can be computed by equation (7).

$$x_{n,s}(i,j) = \alpha_{n,s} x_{n,s}(i,j) \quad (7)$$

Enhanced image is computed for each sub band and reconstructed the image using inverse wavelet transform. Resulting image is the contrast-enhanced image.

Conversion of objects of interest: The watershed transform detects intensity valleys in the image. The enhanced image is converted into the objects of interest with high intensity valleys.

Detection of intensity valleys: All the intensity valleys are below a particular threshold. The location rather than the size of the regions is important. The min function modifies the image to contain only those valleys found by the extended min function. The min function will also change a valleys pixel value to zero (deepest possible valley for until 8 images). All regions containing an imposed minimum will be detected by the watershed transform.

Watershed segmentation: Watershed segmentation of the imposed minima image is accomplished with the watershed function. The watershed function returns a label matrix containing non-negative numbers that correspond to watershed regions. Pixels that do not fall into any watershed region are given a pixel value of zero.

However, the image is degraded by noise and so cause over-segmentation. Therefore, over-segmentation images are further merged in some regions. The region to be merged is based on homogeneity and similarity criteria based on the wavlet coefficients. Mean, second and third order central moments values of the wavelet coefficients of each of the regions are calculated.

Mergin over-segmentation regions: The watershed transform is sensitive to the change of the data therefore the noise objective brim and then changes inside objectives can lead to over-segmentation. Over segmentation is caused by the watershed transform, and the image background is divided into several parts and merged using the following conditions:

a) The regions that will be combined are neighbours.
b) The characteristics of the region, which will be combined, should be similar.
c) The big region after combining is useful.

The regions which are neighbours and similar pixel can be combined. After combination, the background and the target image are separated. Usually, similitude can be defind according to gray, texture, and so on.

If B_i is set of boundary points of catchment basin $I(CB)$, N_i is the number of B_i, then the definition of $B\ mean\ (i)$ is given by equation (8).

$$B\ mean(i) = f(pj)/Ni \qquad (8)$$

During projection from I^{th} layer to $(I-1)^{th}$ layer, a parent-child spatial relationship between the image elements of two successive

layers is defined. This relationship is evaluated by means of a similarity measure. The children of a layer may belong to different patterns in the upper layer. The similarity between a child image element and its possible parents describes the similarities.



11.11 LET US SUM UP

In this unit, we have discussed the following:

1. Image segmentation techniques
2. Edge based segmentation
3. Line based segmentation
4. Various region-based segmentation techniques, and
5. Boundary detection algorithm.



11.12 FURTHER READING

- "Digital Image Processing" by Rafael C. Gonzalez and Richard E. Woods, Pearson, 4th Edition (2018).
- "Mathematical Morphology and Its Applications to Image Processing" by Jean Serra, Kluwer Academic Publishers, 1st Edition (1992).
- "Image Processing, Analysis, and Machine Vision" by Milan Sonka, Vaclav Hlavac, Roger Boyle, Cengage Learning, 4th Edition (2014).
- "Morphological Image Analysis: Principles and Applications" by Pierre Soille, Springer, 1st Edition (2003).
- "Fundamentals of Digital Image Processing" by Anil K. Jain, Prentice-Hall, 1st Edition (1989).



11.13 ANSWERS TO CHECK YOUR PROGRESS

CHECK YOUR PROGRESS A

1. Image segmentation partitions an image into set of regions. In many applications, regions represent meaningful areas in an image. In other

application, regions might be set of border pixels into structures such as line segments, edges, etc. The level of portioning of an image depends on the problem to be solved. Segmentation should stop when the objects of interests have been isolated. Segmentation has two objectives:

a. To decompose an image into regions for further analysis.
b. To perform a change of representation of an image for faster analysis.

2. Segmentation is the most important step in automated recognition system which has numerous applications and some of them are discussed below:

a. Medical imaging.
b. Satellite.
c. Movement detection.
d. Security and surveillance.
e. License plate recognition (LPR).
f. Industrial inspection and automation.
g. Robot navigation.

CHECK YOUR PROGRESS B

1. Generally, all segmentation algorithms are based on one of the two properties of gray level values of pixels.

a. Discontinuity: In this approach, images are portioned based on the difference or discontinuity of the gray level values. Edge based segmentation methods fall in this category.

b. Similarity: Images are partitioned based on the similarity of the gray level values of the pixels according to a pre-defined criterion. Thresholding, region-based clustering and matching-based segmentation techniques fall in this category.

CHECK YOUR PROGRESS C

1. A typical edge may be the border between a block of red colour and a block of yellow. Edge can also be the boundary between two regions with relatively distinct gray-level properties.

2. 'Sign' of second derivative is used to determine whether edge pixel lies on dark side or on light side of edge. 'Zero crossing' is at the midpoint of a transition in gray level.
3. Gradient Operator: Gradient is a first order derivative and is defined for an image $f(x, y)$ as

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

If points in the direction of maximum rate of change of f at a point (x, y).

Magnitude of the gradient is $mag(\nabla) = \left[G_x^2 + G_y^2\right]^{1/2}$.

This can be approximated as $(\nabla f) = |G_x| + |G_y| mag$

and the direction of the gradient is given by $\alpha(x, y) = \tan^{-1}\left[\frac{G_x}{G_y}\right]$,

where α is measured with respect to x-axis.

Laplacian is hardly used in practice for edge detection because it is very sensitive to noise, and produced double edges. Edge direction is not detected by Laplacian. It is used to find the location of edge using zero crossing detection.

4. **5 x 5 LOG mask**

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

CHECK PROGRESS D

1. Image segmentation based on thresholding applies a single fixed criterion to all pixels in the image simultaneously. Hence, it is grid. On the other hand, image segmentation based on the region-based approach is more flexible. Hence it is possible to adjust the acceptance

criteria in the course of region-growing process so that they can depend on the shape of the growing regions if desired.

2. Step 1: Computation of the histogram of the input image.

   The histogram of the image gives the frequency of occurrence of the gray level. The histogram threshold is fixed as 32. Now the input image is divided into two regions as follows:

   Region 1: Gray level < 32

   Region 2: Gray level > 32

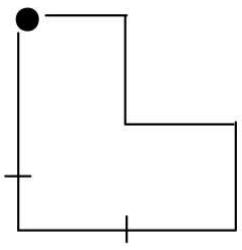
   The input image after this decision is given as

2	2	2	2	2	1	1	1
2	2	2	2	2	1	1	1
1	1	2	2	2	2	2	2
1	2	2	2	2	1	2	2
2	2	2	2	2	2	1	1
2	2	2	2	2	2	1	1
1	2	1	2	2	2	2	2
1	1	2	2	2	2	2	2

3. The advantage of using more than one seed is that better segmentation of the image can be expected, since more seeds lead to more homogeneous regions. The drawback of using more than one seed is that the probability of splitting a homogeneous region in two or more segments increases.

CHECK PROGRESS E

1.



Chain code : 0 3 0 3 2 2 1 1

Difference : 3 3 1 3 3 0 3 0



11.14 MODEL QUESTIONS

Q1: What is the formal definition of segmentation?

Q2: What are the different ways in which segmentation algorithms can be classified?

Q3: List the edge operators.

Q4: What is meant by first and second order derivatives of an image?

Q5: What is he problem of second-order derivatives?

Q6: What is meant by edge linking?

Q7: What is meant by crack edge?

Q8: What is the objective of edge relaxation?

Q9: What is a Hough transform?

Q10: Why and how are edges combined?

Q11: What is meant by a contour?

Q12: How can segmentation algorithm be evaluated?

Q13: How are isolated points and lines recognized?

Q14: Explain in detail the stages of edge detection algorithms. How are they present in edge operators?

Q15: Explain in detail the edge linking algorithms.

Q16: Explain in detail the active contour models.

Q17: Explain watershed segmentation.

Q18: Explain how the edge segmentation algorithms are evaluated?

Q19: Consider one dimensional image $f(x) = [10\ 10\ 10\ 10\ 40\ 40\ 40\ 40\ 20\ 20]$. What are the first and second derivatives? Locate the position of edge.

Q20: Consider an image $F = \begin{pmatrix} 1 & 2 & 5 \\ 5 & 5 & 5 \\ 5 & 3 & 2 \end{pmatrix}$. Show the output of any edge detection algorithm.

Q21: A black object in a white background is known to occupy 20%, 40%, and 80% of the histogram. What is the threshold value? If the colours

of the object and the background are reversed, what is the threshold value?

Q22: Consider the following image:

8	4	5
8	4	1
7	5	1

Show the crack image.

Q23: For the given image, apply graph theoretic algorithm and show the flow of the edges.

9	7	9
2	4	5
7	5	1

Q24: Explain in detail the graph theoretic algorithm, with respect to the following image:

4	5	6
11	9	38
3	11	13

Q25: Consider the following image:

1	1	9	8	7
0	1	8	8	8
0	0	7	9	8
0	1	8	8	9
1	2	8	8	9

What is the result if the threshold is 3? If the threshold condition is changed as (pixel-seed) < 0.1 (maximum – minimum) of 8-neighbourhood, what is the resultant image?

Q26: For the given image,

$$F = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 9 & 9 & 9 & 9 & 2 & 2 & 2 & 2 \\ 9 & 9 & 9 & 9 & 2 & 2 & 2 & 2 \\ 9 & 9 & 9 & 9 & 2 & 2 & 2 & 2 \\ 9 & 9 & 9 & 9 & 2 & 2 & 2 & 2 \\ 9 & 9 & 9 & 9 & 2 & 2 & 2 & 2 \\ 9 & 9 & 9 & 9 & 2 & 2 & 2 & 2 \\ 9 & 9 & 9 & 9 & 2 & 2 & 2 & 2 \end{pmatrix}$$

Show the result of the split-and-merge algorithm.

28. The contingency table of a segmentation algorithm is given as follows:

Expert human	**Present**	**Absent**
ROI present	8	1
Absent	1	1

What is the efficiency of the segmentation algorithm?

*****

UNIT 12: IMAGE COMPRESSION TECHNIQUES

UNIT STRUCTURE

12.1 Learning Objective

12.2 Introduction

12.3 Requirement for Image Compression

12.4 Redundancy in Images

12.5 Image Compression Scheme

12.6 Huffman Coding

12.7 Arithmetic Coding

12.8 Transformed Based Coding

12.9 Vector Quantisation

12.10 Let Sum Up

12.11 Further Reading

12.12 Answers to Check Your Progress

12.13 Model Questions

12.1 LEARNING OBJECTIVES

After going through this unit, you will be able to

- understand the concept and importance of image compression.
- explore the types of redundancies in images.
- learn about various image compression schemes and their applications.
- understand the principles and algorithms behind popular image compression techniques: Huffman coding, Arithmetic coding, Transform-based coding, and Vector quantization.

12.2 INTRODUCTION

In the digital age, high-resolution imaging devices produce large image files, leading to increased demands for storage space and bandwidth during transmission. Image compression addresses these challenges by reducing file sizes while maintaining acceptable visual quality. This unit

explores the fundamental concepts and techniques of image compression, including the identification and elimination of redundancies in image data. You will learn about both lossless compression methods, such as Huffman coding and arithmetic coding, which preserve all original data, and lossy compression techniques, like transform-based coding and vector quantization, which sacrifice some data for greater efficiency. By understanding these methods, you will gain insights into how compression optimizes storage, enhances transmission speed, and improves compatibility across devices, making it indispensable in modern digital applications.

12.3 REQUIREMENT FOR IMAGE COMPRESSION

The size of image files has grown dramatically since the introduction of high-resolution imaging equipment. However, with growing resolution storage space and bandwidth requirement also increases. Thus, image compression has become important in many aspects. The following are some points highlighting the advantages of image compression.

- **Storage Efficiency**: The amount of storage space needed for the images as their size gets reduced due to compression.
- **Enhanced Transmission Speed:** Lowering Image size also lowers the bandwidth demand and transmission takes a shorter time.
- **Faster Loading:** Long files take time to load. By compressing images, file sizes can be reduced significantly shortening the loading time for the page.
- **Mobile Optimisation:** Mobile users do not require very high resolution images since mobile devices operate with small screen size and lower resolutions. Therefore, using compression reduce the images and make them more mobile friendly while improving the experience and speeds of loading time.
- **Compatibility:** Makes image sharing and distribution across different platforms more practical.
- **Cost Reduction:** Reduces storage and transmission costs.

12.4 REDUNDANCY IN IMAGES

The data representing an image often may have many redundancies which may be categorised into the following types-

- Spatial Redundancy: Occurs when neighbouring pixels have similar values. For example, large areas of uniform colour in an image.
- Spectral Redundancy: Occurs due to correlation between different colour channels in an image (e.g., RGB channels).
- Temporal Redundancy: Present in image sequences or video frames where changes between consecutive frames are minimal.
- Perceptual Redundancy: Certain details in an image may not be perceptible to the human eye and can be safely eliminated.

12.5 IMAGE COMPRESSION SCHEMES

There are various schemes available for image compression. Most of the compression schemes generally consists of the following steps:

- **Transformation:** Transfers image data to a domain where redundancies are more detectable.
- **Quantization:** Reduces the precision of transformed coefficients to eliminate less significant information.
- **Encoding:** the quantised image is then encoded using different coding schemes to reduce the size.

Compression schemes can broadly be classified into the following two categories -

Lossless Compression: It a compression scheme where when the encoded image is decoded, we get back the original image (such as PNG or GIF).

Lossy Compression: Some loss is incurred during this type of compression techniques (Such as JPEG or MP3). We cannot reconstruct the exact original image.



CHECK YOUR PROGRESS

State True /False

1. High-resolution imaging devices lead to increased storage and bandwidth requirements for image files.
2. Lossless compression allows the exact original image to be reconstructed after decoding.
3. Temporal redundancy occurs when neighbouring pixels in an image have similar values.
4. Quantization in image compression reduces the precision of transformed coefficients to eliminate less significant information.

Fill in the Blanks:

5. _____ redundancy occurs when changes between consecutive frames in a video sequence are minimal.
6. _____ compression schemes ensure no data is lost during the compression process.
7. Quantization is a step in image compression that reduces the _____ of transformed coefficients.

12.6 HUFFMAN CODING

It is a simple and efficient lossless data compression scheme, where codes are assigned to the input characters. The codes are of variable-length, which depends on the frequency of the characters. More frequently appearing characters receive shorter codes, while infrequently appearing characters receive longer codes.

Steps:

- Calculate the frequency of each symbol in the image data.
- Build a binary tree, combining symbols with the lowest frequencies.
- Assign binary codes to each symbol based on their position in the tree.

Example 12.1:

The process can be illustrated with the following example. Let us consider the string “AAABBCCDDDDD”. First we need to construct a frequency table, where we store the frequency count of each symbol as shown in table 12.1, and then sort the table in ascending order. We then repeatedly combine the symbols with smaller frequencies.

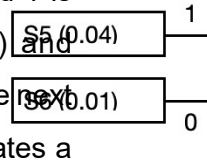
Table 12.2: Sorted frequency table

Symbol	Frequency	Code
B	2	110
C	2	111
A	3	10
D	5	0

Table 12.1: Frequency table

Symbol	Frequency
A	3
B	2
C	2
D	5

We first combine B (frequency 2) and C (frequency 2) (lowest frequencies) to create a new node BC with frequency 4(2+2). While combining two nodes, we make the one with the lower frequency as the left child and the other one as the right child. Label 0 is then assigned to the left edge and 1 is assigned to the right edge. Next, we combine A(frequency 3) and BC(frequency 4) and create a node A(BC) with frequency 7 (3+4). The next combination is -A(BC)(frequency 7) and D (frequency 5) which creates a node D(A(BC)) (frequency 12). The final tree is depicted in figure 12.1.



The next step is to assign codes to the symbols. We traverse the tree from the root node till we encounter the symbol in one of the leaves.

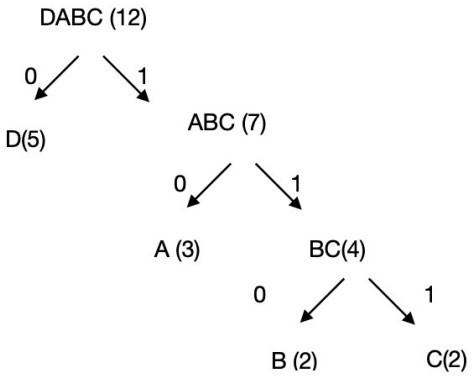


Figure 12.1: Huffman Tree for "AAABBCCDDDDD"

As we traverse, we note down the labels of each edge which finally is assigned as the Huffman code for the symbol. The final codes for the symbols are shown in table 12.2.

Example 12.2: The process can also be applied to images, where we consider the intensity levels of the pixels. Consider intensity levels of pixels in an image and their corresponding probability distributions. By applying Huffman coding algorithm, we can construct a tree as shown in figure 12.2. The final Huffman codes are shown in table 12.3. As we can see from the table we need 8 bits to represent the given intensity levels of each pixel. However, with Huffman coding, we can reprint the intensities with code lengths that varies from 1to 5.

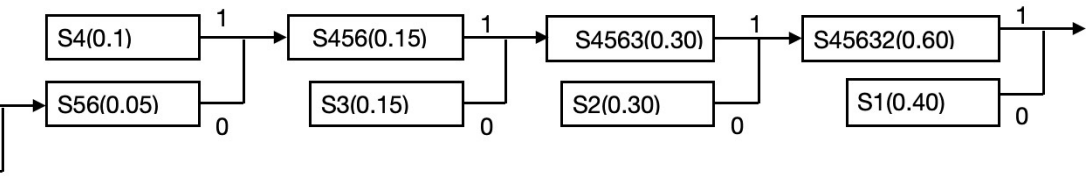


Figure 12.2: Huffman tree for the symbols in table 12.3

Table 12.3: Huffman coding for image pixels

Pixels (with intensity value gray scale)	Probability (arranged in decreasing order)	Binary Code	Huffman Code	Length of Huffman Code
S1 - 50	0.4	00110010	0	1
S2 - 100	0.3	01100100	10	2
S3 - 200	0.15	11001000	110	3
S4 - 150	0.1	10010110	1111	4
S5 - 255	0.04	11111111	11101	5
S6 - 175	0.01	10101111	11100	5

12.7 ARITHMETIC CODING

Arithmetic coding is a form of entropy encoding in lossless data compression. Arithmetic coding differs from Huffman coding because it encodes the whole message into a single number: a fraction between 0 and 1. The steps to perform in the arithmetic coding scheme are as follows:

Steps:

- **Input Data and Probabilities:** Prepare the input data into a format with data point and corresponding probability of appearance.
- **Divide the Range [0, 1]:** Divide the range of probability [0,1] into subintervals. Assign a subinterval to a symbols proportional the probability of owcuurance.
- **Encode the Message:** Start with the first symbol. As a symbol is processed, extract the range of probability distribution of that symbol and then divide that range again into subintervals as was done in the previous step. Repeat the step until all the symbols are processed.
- **Generate the Final Code:** Once all the symbols are processed select any value from the final interval. This value is used to represent the whole image.

- **Decode:** In the decoding process, the same process is repeated in the reverse order to contract the original image.

Example 12.3: Encode the message "ABBA"

Given probabilities:

- Symbol A: 0.4
- Symbol B: 0.3
- Symbol C: 0.2
- Symbol D: 0.1

Step 1: Divide the Range [0, 1]

Step 2: Encode Each Symbol

Start with the full range [0, 1]. For each symbol, narrow the range:

- **First Symbol: A**

  Range for A: [0.0, 0.4).

Table 12.4: Probability Table

Symbol	Probability	Range
A	0.4	[0.0, 0.4)
B	0.3	[0.4, 0.7)
C	0.2	[0.7, 0.9)
D	0.1	[0.9, 1.0)

Narrowed range:

- **Second Symbol: B**

  Range for B in the current range [0.0, 0.4):

Table 12.5 Arithmetic Coding for “ABBA”

A	B	B	A
0	0	0.16	0.208
0.1	0.04	0.172	0.2116
0.2	0.08	0.184	0.2152
0.3	0.12	0.196	0.2188
0.4	0.16	0.208	0.2224
0.5	0.2	0.22	0.226
0.6	0.24	0.232	0.2296
0.7	0.28	0.244	0.2332
0.8	0.32	0.256	0.2368
0.9	0.36	0.268	0.2404
1	0.4	0.28	0.244

Subdivide [0.0, 0.4) into proportions based on probabilities.

New range for B: [0.16, 0.28).

- **Third Symbol: B**

  Range for B in the current range [0.16, 0.28):

  Subdivide [0.16, 0.28) into proportions.

  New range for B: [0.208, 0.244).

- **Fourth Symbol: A**

  Range for A in the current range [0.208, 0.244):

  Subdivide [0.208, 0.244) into proportions.

  New range for A: [0.208, 0.2224).

Step 3: Final Encoded Value

Choose any value in the final range [0.208, 0.2224) as the encoded value. For example, **0.213**.

12.8 TRANSFORM- BASED CODING

Transform coding converts spatial domain image data into the frequency domain using mathematical transforms, such as the Discrete Cosine

Transform (DCT) or Discrete Wavelet Transform (DWT). High-frequency components (less perceptible details) are often discarded to achieve compression. Effective for lossy compression with good visual quality. Forms the basis of popular formats like JPEG.

Steps:

- Divide the image into smaller blocks.
- Apply a transform (e.g., DCT) to each block.
- Quantize and encode the transformed coefficients.

12.9 VECTOR QUANTIZATION (VQ)

It a lossy compression technique, where the input data is considered as a collection of large number of input vectors. These vectors are then mapped with a finite set of representative vectors and in the compressed image, the index of these representative vectors are mentioned in place of the original vectors. The representative set of vectors is called as **"Codebook".** The following are some of the important concepts involved in vector quantization.

- **Input Vectors**: The input high dimensional data like fracture vectors, pixels intensities etc.
- **Codebook**: The set of representative vectors. Each vector from the input vector set is finally represented with one of the vectors from codebook.
- **Quantization**: In the compressed image, each input vector of the original image is replaced with the index of the closest representative vector from the codebook.
- **Reconstruction**: In this process, the image is reconstructed with the help of the indices stored in the compressed image and the code book. Each index is replaced by the corresponding vector from the code book. As, the codebook has only representative vectors not the original vectors, it is not possible to get back the original image and thus the process is lossy.

The block diagram of the vector quantisation processis given in figure 12.3. An important subprocess of this procedure is the construction of the

Codebook. For this K-means clustering algorithm may be use, where the set of input vectors are divided into k clusters. The centroid vector of each closer can be used a the representative vector for that cluster in the codebook. Thus in the codebook, there will be only k vectors. Each input vector is then replaced with the index of the nearest codebook vector using a distance metric, such as the **Euclidean distance.** During reconstruction, the codebook vectors corresponding to the indices are used to approximate the original data.

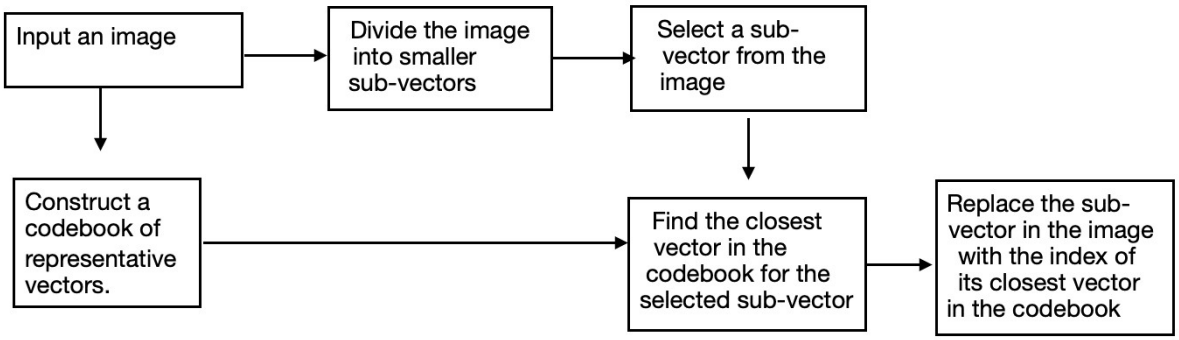


Figure 12.2: Huffman tree for the symbols in table 12.3



CHECK YOUR PROGRESS

State True /False

8. Huffman coding assigns variable-length codes based on the frequency of symbols, with more frequent symbols receiving longer codes.
9. Arithmetic coding encodes the entire message into a single fractional number.
10. Transform-based coding converts spatial domain image data into the frequency domain using methods like DCT or DWT.

Fill in the Blanks:

11. Huffman coding is a _____ compression technique where codes are assigned based on the frequency of symbols.
12. In arithmetic coding, the final encoded value is a _____ in the range [0, 1].

13. In vector quantization, the _____ contains the representative vectors used to approximate the original image.
14. The _____ algorithm can be used to construct the codebook in vector quantization.



12.10 LET SUM UP

- Image compression reduces the storage space and bandwidth requirements for high-resolution images. It also improves transmission speed, page loading time, and compatibility across devices.
- An image data can have redundancies, which can be categorised into spatial, spectral, temporal, or perceptual redundancy. Removal of these redundancies optimize storage and bandwidth requirements.
- Compression schemes are classified into lossless (e.g., PNG) and lossy (e.g., JPEG) based on data reconstruction.
- Lossless compression retains the original information, thus it is possible to get back the original image.
- Lossy compression techniques results in some information loss as it discard the non-essential data.
- Huffman coding assigns shorter binary codes to frequent symbols for efficient lossless compression.
- Arithmetic coding encodes an entire message into a single fractional value between 0 and 1.
- Transform-based coding (e.g., DCT) shifts image data to the frequency domain, discarding high-frequency components for lossy compression.
- Vector quantization maps input data to representative vectors in a codebook for lossy compression.



12.11 FURTHER READING

1) Gonzalez, R. C., & Woods, R. E. (2008). *Digital Image Processing* (3rd ed.). Pearson.

2) Pratt, W. K. (2007). Digital Image Processing: PIKS Inside (4th ed.). Wiley.

3) Szeliski, R. (2010). Computer Vision: Algorithms and Applications. Springer.



12.12 ANSWER TO CHECK YOUR PROGRESS

1. True
2. True
3. False
4. True
5. Temporal
6. Lossless
7. Precision
8. False
9. True
10. True
11. Lossless
12. Fraction
13. Codebook
14. K-means Clustering



12.13 MODEL QUESTIONS

Q1: Why has image compression become important in the era of high-resolution imaging devices?

Q2: List the key needs for image compression.

Q3: Explain how image compression enhances transmission speed, mobile optimization and increase storage efficiency.

Q4: Define the following types of redundancies in images- Spatial redundancy, Spectral redundancy, Temporal redundancy, Perceptual redundancy. Give example of each.

Q5: What are the main steps involved in most image compression schemes?

Q6: Discuss the Huffman encoding scheme. Assign Huffman codes to the symbols in the string "BBBCCCDDDAAABCADDDACDBADDDDCCC".

Q7: Using the probabilities A: 0.4, B: 0.3, C: 0.2, D: 0.1, calculate the range for encoding the message "BACCD".

Q8: What is transform-based coding, and how does it contribute to image compression? Name two common transforms used in transform-based coding. What happens to high-frequency components during transform-based compression?

Q9: What is a codebook in vector quantization, and how is it constructed?

Q10: With a block diagram Explain the steps involved in vector quantization. Why is vector quantization considered a lossy compression technique?

*****

UNIT 13: BINARY IMAGE PROCESSING

UNIT STRUCTURE

13.1 Learning Objectives

13.2 Introduction

13.3 Binarization

13.4 Mathematical Morphology

13.5 Structuring Element

13.6 Morphological Image Processing

- 13.6.1 Basic Set Theory
- 13.6.2 Logical Operations

13.7 Standard Binary Morphological Operations

- 13.7.1 Dilation
- 13.7.2 Erosion
- 13.7.3 Opening
- 13.7.4 Closing
- 13.7.5 Properties of Morphological Operations

13.8 Application of Morphological Operations

13.9 Let Us Sum Up

13.10 Further Reading

13.11 Answers to Check Your Progress

13.12 Model Questions

13.1 LEARNING OBJECTIVES

After going through this unit, you will be able to

- define mathematical morphology and its basis in set theory.
- identify the role of structuring elements in morphological operations.
- recognize the difference between binary and grayscale morphological operations.
- understand and apply basic morphological operations such as erosion, dilation, opening, and closing.

- comprehend the effects of each operation on an image, including their impact on object boundaries and shape.
- apply morphological operations for noise removal, object isolation, and enhancement in binary and grayscale images.
- use opening and closing for removing small noise elements and closing small holes in images.
- explore applications of morphological processing in noise removal, object detection, and segmentation.

13.2 INTRODUCTION

In this unit we will discuss about binary image processing, its significance, applications, and various techniques used to process binary images. The content begins with the concept of binarization, the basic process of converting grayscale or color images into binary formats, followed by efficient and simple image analysis and then goes into detail in mathematical equations in, a powerful framework based on set theory to analyze the shape, structure, and spatial relationships of two images. An important element in shape processing is a predefined structuring element, dimension or kernel that interacts with the image in shape transformation Understanding how classification objects work will give you an insight into the different approaches they take so two images are processed and analyzed. The content will then introduce basic set theory and logical operations, which form the mathematical backbone of morphological image processing.We will explore the basic binary morphology operations, including erosion, dilation, opening and closing, and their properties and practical applications. These operations are the building blocks of complex image analysis tasks such as edge detection, noise filtering, and object segmentation.

13.3 BINARIZATION

Binarization is a process that transforms a grayscale or color image into a binary image. Binary images are also known as *black and white (BW)*

images as the pixels are either black or white. The black pixels are represented as 0 while white pixels are represented as 1. Usually, the black pixels in a binary image represent background and white pixels represent foreground. Thresholding is the most common technique used to convert a grayscale image into binary image. Thresholdingis a technique of image processing in which intensity value of a pixel is set as 1 if its value is greater or equal to a given threshold, otherwise it is set as 0. Thresholding can be categorized into global thresholding and local thresholding. In global thresholding like Otsu method, one threshold is used for the entire image while in local thresholding thresholds are chosen locally for a region or part of an image. The thresholding operation is defined by equation 13.1.

$$BW(i,j) = \begin{cases} 1 & if\ I(i,j) \geq thresh \\ 0 & otherwise \end{cases}$$ ——————-Eq. 13.1

where *BW* is the binary image and *I* is the grayscale image, *(i,j)* represents a pixel position.

Binary images consume less memory and computational power as they have only 1-bitplane. Thus, they are easy to handle and process. Although, high-end machines are available now at an affordable rate, the use of binary images are still relevant. They are useful in analysing the shapes, structures and geometry of the objects present within the image. The objects and structures present in color or grayscale images are often represented using masks. These masks are nothing but binary images, where the objects of interests are appeared as white and rest as black.



CHECK YOUR PROGRESS

Q1. Which of the following techniques is commonly used for image binarization?

a) Gaussian Blur
b) Otsu's Method
c) Sobel Edge Detection
d) Hough Transform

Q2. In image binarization, what does the threshold value determine?

a) The intensity range of colors in the image

b) The separation of foreground from the background

c) The direction of edge gradients

d) The connectivity of pixels

Q3. Adaptive thresholding is particularly useful for:

a) High-resolution images

b) Noisy images

c) Images with varying illumination

d) Edge detection

MATHEMATICAL MORPHOLOGY

Mathematical morphology is a theory used torepresent and process structures and geometry present in images based on set theory, topology, and lattice algebra. In mathematical morphology the sets are used to represents the objects present in images. The pixels belonging to objects represented as one set (foreground) and rest belong to another set (background).It provides a set of operations known as morphological operations to process shapes, sizes, and structures of objects present in images. The basic operations are erosion and dilation.

Mathematical morphological operations are applicable to various image processing tasks including image enhancement, segmentation, edge detection, and feature extraction. They are widely used in medical image analysis, pattern recognition, document processing.

13.4 STRUCTURING ELEMENT

A *structuring element* is the most integral part of mathematical morphology which interacts with an image to perform morphological operations. It is a predefined small matrix which has binary values (either 0 or 1). It defines the size and shape of the neighborhood area used to perform the morphological operations. It determines the impact of any morphological operations on an image. The shape of a structuring element can be round, rectangular, oval, line etc; size can vary depending on the requirements. Figure 13.1 presents 3 examples of structuring elements.

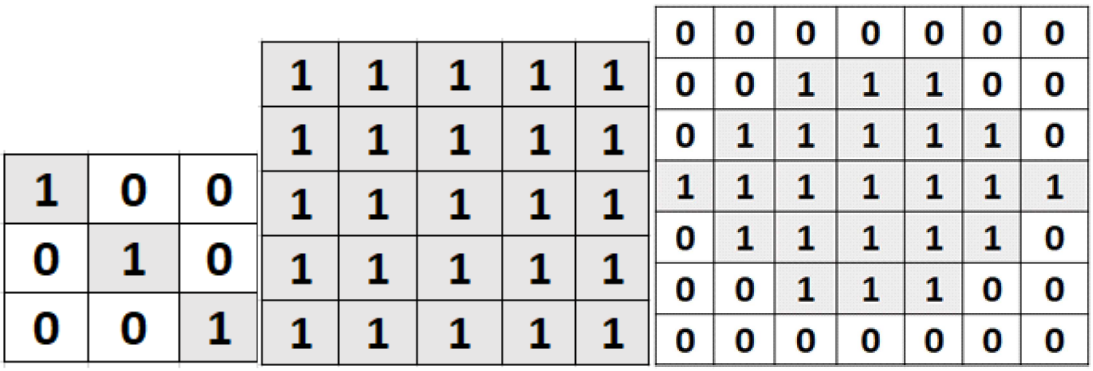


Figure 1.1: Structuring elements with various shape and size. (a) Line structuring element of size 3x3, (b) Rectangular structuring element of size 5x5 and (c) Eliptical structuring element of size 7x7.

During a morphological operation, the structuring element is swamped over the image from left-top corner to right-bottom corner. At every step the central point of the structuring element is placed over a pixel, P of the image currently being processing and the intensity of each pixel in the neighbourhood of P is compared with the value of the corresponding point of the structuring element. When the intensities of all neighbourhood pixels of P match the values of the structuring element, it is called a *fit*. If none matches it is called a *miss* and when there is a partial match it is termed a *hit*. Figure 13. 2 shows a sample structuring element and sample binary image in matrix form. The results of scanning the sample image by the structuring element as given be figure 13.2 is demonstrated in figure 13.4.

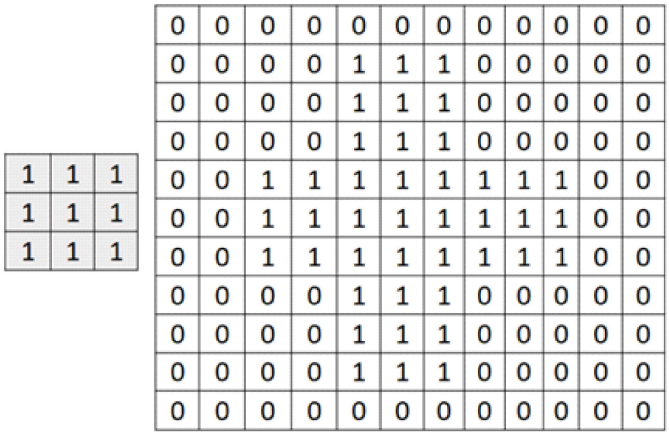


Figure 13.2 (a) 3x3 rectangular structuring element, (b) a sample black and white image

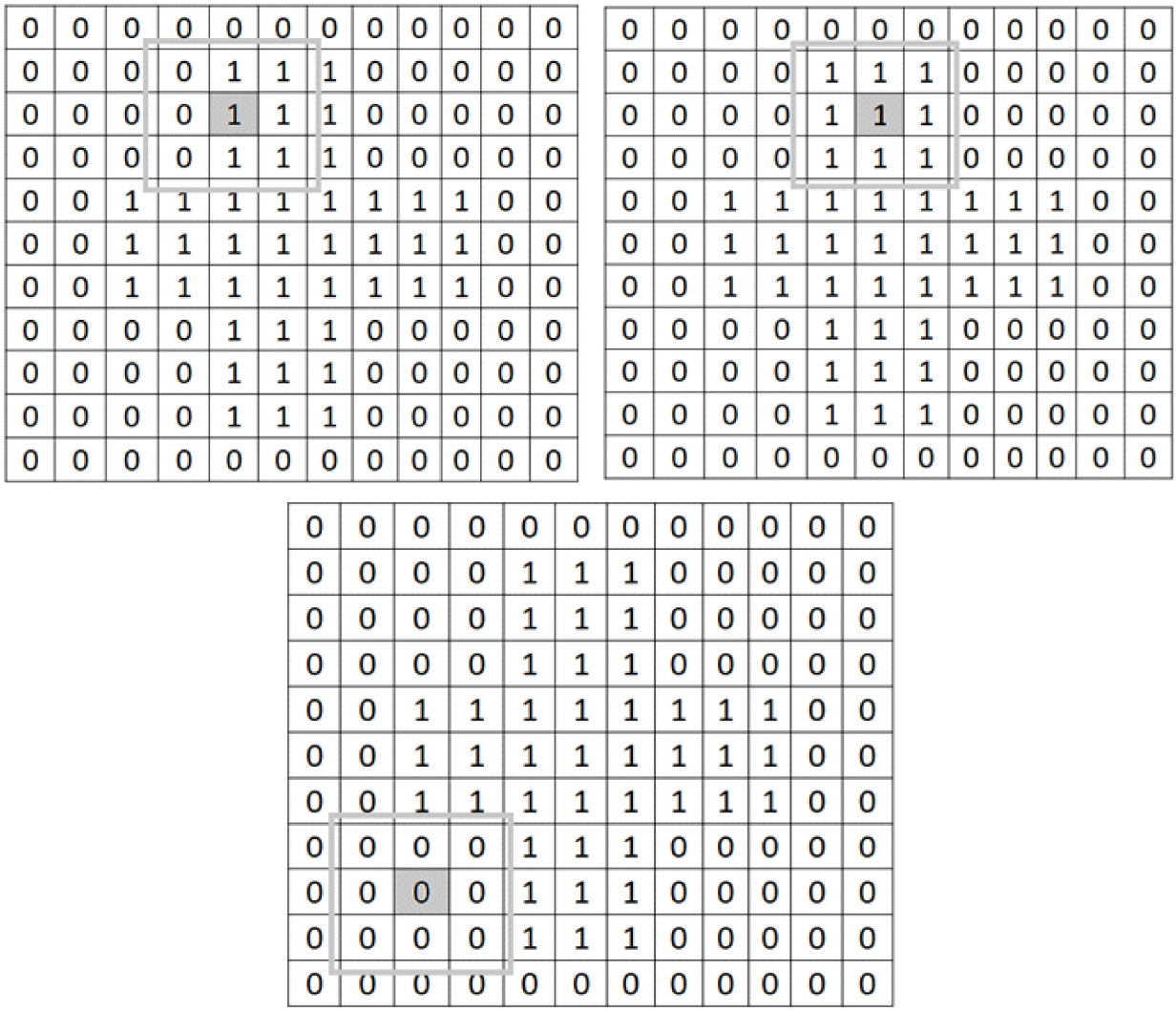


Figure 13.3 Result of scanning the binary image in 13.2 (b) by structuring element in 13.1 (a). (1) Hit, (b) Fit, and (c) Miss



CHECK YOUR PROGRESS

Q4. Mathematical morphology is based on which mathematical theory?

a) Linear Algebra

b) Set Theory

c) Differential Equations

d) Fourier Analysis

Q5. The structuring element in morphological operations is:

a) Always a square matrix

b) A geometric shape used for interaction with the image

c) A gradient mask

d) A histogram of pixel intensities

Q6. What does the “hit” in hit-or-miss transform refer to?

a) The structuring element partially overlapping with the image

b) The structuring element perfectly matching the foreground of the image

c) The detection of edges in the image

d) The removal of noise from the image

Q7. What does the “miss” in hit-or-miss transform refer to?

a) The structuring element matching the background of the image

b) A failed detection of an object

c) Noise left after the operation

d) Partial overlap of the structuring element with the image

Q8. The term “fit” in morphological operations means:

a) The structuring element aligns perfectly within the region of interest

b) The structuring element partially covers the region of interest

c) The structuring element enhances object boundaries

d) The structuring element detects intensity gradients

13.5 MORPHOLOGICAL IMAGE PROCESSING

Morphological image processing allows manipulation of objects and structures found within images. Basic goal is to modify the shapes, boundaries, and connectivity of the geometric structures with the help of a structuring element. Fundamental morphological image processing techniques perform tasks like, shrinkage, expansion, segmentation, noise removal, boundary detection etc. The basis of all morphological operations is basic set theory. Most of the morphological operations applies set

operations along with logical operations. The context of basic set theory and logical operation in relation to morphological image processing explained in the following subsections.

13.5.1 Basic Set Theory

Basic set theory provides the mathematical foundation for most of the image processing techniques including morphological image processing. A set is a collection of distinct elements that represents objects, pixels, or features in an image. In binary images, objects are treated as sets of pixels for the coordinates of their pixels represent an element in the set. Set theory defines various operations that can be used directly in image processing. For instance, **union** and **intersection** are used to form or find the common elements in the sets, just like merging or overlapping regions in an image, as well as to find the pixels that belong only to one of the two sets. **Difference** operations find pixels affecting only one set and not the other; this produces a result from subtracting objects or isolating boundaries. **Complementation** can identify all elements not in each set; this is related to inversion of an image.

Basic morphological operations, erosion and dilation, can be defined using set theory; erosion represents Minkowski subtraction of a structuring element from the image set, while dilation corresponds to Minkowski addition. These set-theoretic principles form the basis for understanding the application of image morphological techniques. Set theory provides such a logical and algebraic foundation that one could describe a part of a requirement to implement various kinds of image processing tasks in this structure. That makes it possible to minimize effort in analysing and manipulating the data's spatial and structural relationships.

13.5.2 Logical Operations

Logical image processing naturally represents the data to be manipulated through pixel-wise operations using Boolean logic

and thus can be easily used to access and extend the power of typical image processing. Logical operations are elementary in image processing, used to manipulate and analyse binary images by performing pixel-wise operations based on Boolean logic. Common logical operations include AND, OR, NOT, and XOR; these relatively simple formulations operate directly on binary pixel values (0 or 1) from one or more images. These operations prove to be an extensive methodology for numerous tasks involved in processing images. The AND operation uses a pixel to be set as 1, only if corresponding pixels of each input image are ones. This can be used to specify intersection areas or masking portions of the image. The OR operation sets a pixel to 1 if any corresponding pixels are 1 in the input images, combining areas. NOT inverts the pixel binary values of the image and switches black to white pixels. XOR finds differences between input images and produces them as 1. It is a powerful tool in preprocessing, segmentation, and image analysis-simple for computation purposes. Its use could involve combining results of different algorithms, isolating objects of interest, or deleting unwanted regions. Thus, logical operations are some of the major tools that make possible various functionalities in image-processing workflows.

13.6 STANDARD BINARY MORPHOLOGICAL OPERATIONS

In this section we will discuss about four basic morphological operations: erosion, dilation, opening and closing and their properties. As mentioned earlier, these operations are performed by placing a structuring element over each pixel of an image.

13.6.1 Erosion

Erosion helps in shrinking object size by removing pixels from object boundaries. Erosion records the pixels whose

neighbourhood fit with the structuring element. Mathematically erosion can be expressed as:

$$I \ominus S = z \in E | S_z \subseteq I \}$$

where S_z is the translation of S_z by a vector *z*, and *E* stands for Euclidean space. The following condition determines the pixel value of the eroded picture *E* at (x, y) when the structural element *S* is positioned at the neighbourhood $N(x, y)$ of image *I* centred at pixel position (x, y):

$$E(x,y) = \begin{cases} 1 & if\ S\ fits\ with\ N(x,y) \\ 0 & otherwise \end{cases}$$



Figure 13.5 (a) Original Image, (b) Result of Erosion applied on image (a) and Result of Dilation applied on image

13.6.2 Dilation

Dilation expands objects by adding pixels to the boundaries. It records the pixels whose neighbourhood hit with the structuring element. Mathematically dilation can be expressed as:

$$I \oplus S = \{z \in E | S_z \cap I / = \Phi\}$$

where S_z is the translation of S_z by a vector *z*, and *E* stands for Euclidean space. The following condition determines the pixel value of the dilated picture *D* at (x, y) when the structural element *S* is positioned at the neighbourhood $N(x, y)$ of image *I* centred at pixel position (x, y):

$$D(x,y) = \begin{cases} 1 & if\ S\ hits\ with\ N(x,y) \\ 0 & otherwise \end{cases}$$

Figure 13.5 shows the results of erosion and dialation.

13.6.3. OPENING

The opening is the result of erosion operation followed by dilation. It is useful in removing noise and separating objects that are close. It can be expressed as:

$$I \quad S = (I \ominus S) \oplus S$$

Figure 13.6 demonstrate the result of opening.



Figure 13.6 (a) Original Image, (b) Result of Opening applied on image (a)

13.6.4 Closing

The closing is does the opposite of opening. In, the resultant image is obtained by performing dilation after erosion. It can be used to fill small gaps between boundary pixels and connecting disjoint components. Mathematically it can be expressed as:

$$I \bullet S = (I \oplus S) \ominus S$$

Figure 13.7Shows the result of closingoperation on a binary image.

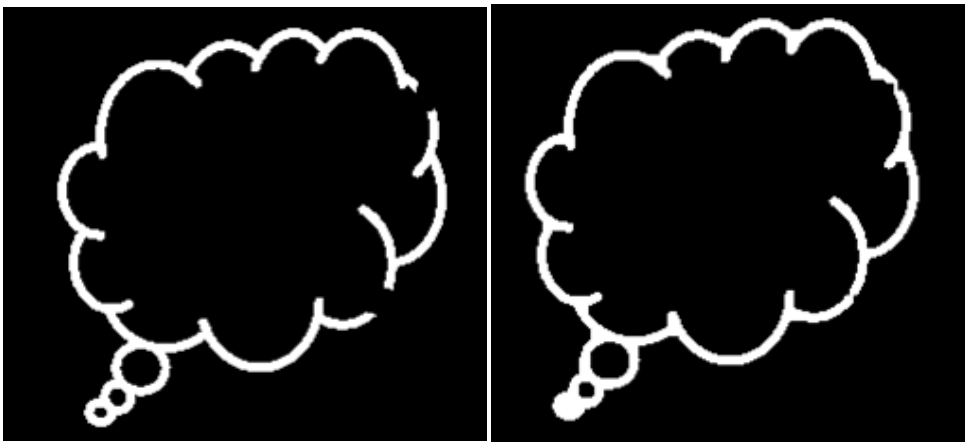


Figure 13.7 (a) Original Image, (b) Result of Closing applied on image (a)

13.7 PROPERTIES OF MORPHOLOGICAL OPERATIONS

The key properties of erosion, dilation, opening and closing are listed below.

i. They are translation invariant.

ii. The dilation is commutative.

iii. The dilation is associative.

iv. Erosion and dilation satisfy the duality.

v. Opening and closing satisfy the duality.

vi. The dilation is distributive over set union.

vii. The erosion is distributive over set intersection.

viii. Opening and closing are idempotent.

ix. Opening is anti-extensive.

13.8 APPLICATION OF MORPHOLOGICAL OPERATIONS

Morphological Operations were widely used for image processing tasks like noise removal, image segmentation, background correction. Few application scenarios of morphological operations are briefly discussed below:

A. Removal of Noise

One of the more popular applications would be in the removal of noise from binary or grayscale images. Erosion can help remove small specks of noise, while dilation can help fill tiny holes within objects. In an efficient way, for example, morphological operations can be used to remove noise from the scanned documents in Optical Character Recognition (OCR) systems so that text recognition can be performed more cleanly and accurately. Figure 13.8 shows the result of noise removal using morphological operators.



Figure 13.8 (a) Original Image, (b) After Noise removal

B. Detection and Segmentation of Objects

Morphological techniques like opening and closing are fundamental in the detection and isolation of objects enclosed in an image. In image segmentation, these operations improve the results of thresholding techniques by eliminating pruned minor details such as a small noise or even connecting broken parts of objects. One possible example can be found in satellite imaging, wherein morphological operations aid in feature detection and segmentation of roads, buildings, and vegetation.

C. Preprocessing Medical Imaging

The principle of morphological processing is important in the field of medical image analysis, wherein it enhances important features while suppressing unwanted information. For instance: Erosion and dilation assist in the segmentation of structures, with cells, tumors, and blood vessels being the most sought-after in segments. Opening can remove small artifacts in X-rays or MRIs. Closing helps to fill in gaps in regions of interest, increasing the visibility of continuous structures such as veins or bones.

D. Shape and Boundary Analysis

Morphological operations can be efficiently applied for shape and boundary analysis. The boundary of objects can be highlighted with basic operations like dilation and erosion. This is extremely important in the case of industrial quality control, where the shape of manufactured components is scrutinized for defects.An example of boundary detectionis shown in figure 13.9.



Figure 13.9 (a) Original Image, (b) Detected Boundary



CHECK YOUR PROGRESS

Q9. Erosion in morphological operations is primarily used to:

a) Remove noise

b) Shrink objects in an image

c) Enhance object boundaries

d) Dilate small objects

Q10. What happens during the erosion operation?

a) The structuring element adds pixels to object boundaries

b) The structuring element removes pixels from object boundaries

c) The image is converted to grayscale

d) Noise is amplified

Q11. Dilation is used to:

a) Fill gaps in objects

b) Reduce object size

c) Remove edges from objects

d) Apply a Gaussian blur

Q12. What is the effect of applying dilation on a binary image?

a) Expands the boundaries of foreground objects

b) Shrinks the boundaries of background objects

c) Converts binary to grayscale

d) Enhances noise in the image

Q13. The opening operation is a combination of:

a) Dilation followed by erosion

b) Erosion followed by dilation

c) Two successive erosions

d) Dilation followed by thresholding

Q14. Which operation is most effective for removing small objects or noise from an image?

a) Closing

b) Opening

c) Erosion

d) Dilation

Q15. Which of the following is an application of morphological operations?

a) Edge detection in binary images

b) Noise removal in grayscale images

c) Feature extraction in medical imaging

d) All of the above



13.9 LET US SUM UP

- Mathematical morphology provides a framework to analyse and process geometric structures in images using set theory, topology, and lattice algebra.
- Mathematical morphology is applied to binary or grayscale images. Binary images are 1-bit images in which 0 represents black and 1represent white.
- Images are treated as sets, wherein objects are represented by one set of pixels and background is represented by another set.
- Erosion, dilation, opening, and closing are the fundamental morphological operations.
- These operations are based on basic set operations like union, intersection, difference and logical AND, OR operations.

- Erosion shrinks objects and dilation expands. Closing helps in separating supposed to be disconnected objects whereas opening helps in reducing gap in the boundary.
- Morphological operations use a predefined binary matrix called as structuring element. A structuring element can be circular, rectangular, line, elliptical etc.
- Morphological image processing is used in image enhancement, segmentation, edge



13.10 FURTHER READING

- "Digital Image Processing" by Rafael C. Gonzalez and Richard E. Woods, Pearson, 4th Edition (2018).
- "Mathematical Morphology and Its Applications to Image Processing" by Jean Serra, Kluwer Academic Publishers, 1st Edition (1992).
- "Image Processing, Analysis, and Machine Vision" by Milan Sonka, Vaclav Hlavac, Roger Boyle, Cengage Learning, 4th Edition (2014).
- "Morphological Image Analysis: Principles and Applications" by Pierre Soille, Springer, 1st Edition (2003).
- "Fundamentals of Digital Image Processing" by Anil K. Jain, Prentice-Hall, 1st Edition (1989).



13.11 ANSWERS TO CHECK YOUR PROGRESS

Q 1: b) Otsu's Method

Q 2: b) The separation of foreground from the background

Q 3: c) Images with varying illumination

Q 4: b) Set Theory

Q 5: b) A geometric shape used for interaction with the image

Q 6: b) The structuring element perfectly matching the foreground of the image

Q 7: a) The structuring element matching the background of the image

Q 8: a) The structuring element aligns perfectly within the region of interest

Q 9: b) Shrink objects in an image

Q10: b) The structuring element removes pixels from object boundaries

Q11: a) Fill gaps in objects

Q12: a) Expands the boundaries of foreground objects

Q13: b) Erosion followed by dilation

Q14: b) Opening

Q15: d) All of the above



13.12 MODEL QUESTIONS

Q1: Explain the importance of morphological image processing in computer vision and image analysis.

Q2: What is image binarization? Discuss its significance and methods used for thresholding, including Otsu's method and adaptive thresholding.

Q3: Define mathematical morphology. How does it differ from other image processing techniques?

Q4: What is a structuring element in mathematical morphology? Explain its role and types used in morphological operations.

Q5: Discuss the concept of morphological operations and how they are applied to binary images for noise removal and shape analysis.

Q6: Explain how basic set theory concepts, such as union, intersection, and complement, are used in morphological operations.

Q7: Describe the role of logical operations like AND, OR, and NOT in the context of binary image processing.

Q8: Provide a detailed explanation of standard binary morphological operations and their impact on image structure.

Q9: What is dilation in morphological image processing? Explain its working and applications with suitable examples.

Q10: Define erosion and explain how it is used to refine object boundaries or shrink objects in binary images.

Q11: What is the opening operation in mathematical morphology? Discuss its advantages in removing small objects or noise.

Q12: Define the closing operation and explain its application in filling small gaps or holes in binary images.

Q13: Discuss the key properties of morphological operations, including idempotence, duality, and invariance under translation.

Q14: Enumerate and explain the applications of morphological operations in fields like medical imaging, object detection, and character recognition.

Q15: How do morphological operations contribute to feature extraction and preprocessing in machine learning and deep learning pipelines?

*****

UNIT 14: COLOR IMAGE PROCESSING

UNIT STRUCTURE

14.1 Learning Outcomes

14.2 Introduction

14.3 Color Formation

14.4 Human Perception of Color

14.5 Color Model

- 14.5.1 RGB Model
- 14.5.2 CMY Color Model
- 14.5.3 HIS Color Model
- 14.5.4 YIQ Color Model

14.6 Histogram Equalization of ColorImage

14.7 Color Image Filtering

14.8 Let Us Sum Up

14.9 Further Reading

14.10 Answers to Check Your Progress

14.11 Model Questions

14.1 LEARNING OBJECTIVES

After going through this unit, you will be able to

- understand color Models and Representations
- understand how color images are captured, stored, and represented digitally in various file formats.
- perform transformations between different color spaces and understand the purpose of such conversions.
- analyzecolor histograms for each channel to assess intensity distribution and apply histogram equalization.
- apply image enhancement techniques (e.g., contrast adjustment, saturation control, filtering) specific to color images.

14.2 INTRODUCTION

Color image processing is focuses on analysing images based oncolor information. Grayscale images represent intensity in terms of one channel, while color images capture pixel intensity in terms of three channels to represent colors as combinations of primary components, such as red, green, and blue (RGB), or other color spaces like HIS, CMY, and YIQ. Color information carries very crucial information required for image analysis for fields like computer vision, human computer interactions.

Color image processing involves various techniques, including color transformation, enhancement, segmentation, and feature extraction, tailored to handle the multidimensional nature of color images. These techniques enable the analysis of images for tasks like object detection, scene understanding, and pattern recognition, where color plays a pivotal role. Moreover, converting between color spaces can simplify complex tasks, such as identifying objects under varying lighting conditions or enhancing image details.

14.3 COLOR FORMATION

The human visual system and the way light interacts with an item combine to create the fascinating phenomena of color. Light is made up electromagnetic waves. Electromagnetic waves have a range of wavelengths that correlate to various hues. When light strikes an item, certain wavelengths are absorbed by the surface while others are reflected. The wavelength reflected determines our perception of an object's color. For an instance, an object appears red because it reflects red light while absorbing the other wavelengths.

Forming colors can also take place through additive or subtractive mixing. Additive color mixing is the combination of the primary colors- red, green, and blue (or RGB)- to see some colors, as in a digital screen or display. When mixed, all three colors produce a white light in equal intensity. In contrast, a subtractive color mix involves the pigments cyan, magenta, and yellow (or CMY). This subtractive process applies mostly in printing,

where certain wavelengths are absorbed and reflect the rest, producing a new color.

Environmental factors play other roles in the formation of colors, which include the intensity of the light source and its composition. For instance, seeing the color of an object can vary depending on the type of artificial light it is illuminated with, such as daylight, fluorescent, or incandescent. Humans see color because of the cones in their eyes: these cones respond to certain wavelengths, and thus the physical phenomenon of color gets translated into vivid, meaningful visual experiences.

14.4 HUMAN PERCEPTION OF COLOR

Color is perceived by humans through a complex interplay of light, the eye and the brain. Light enters through the eye through the cornea and lens; they focus the incoming light on the retina, where two very special types of photoreceptor cells are found, called cones and rods. Rods are sensitive to light and dark, allowing one to see in dim light, while cones are important for color perception. Cones are sensitive to three specific wavelengths, which correspond with receptors for red, green, and blue light.

The signals from cones then go through the different neural pathways in order to perceive many colors at the time. The process involves a trichromatic theory that states that there are three kinds of cones: red, green, and blue which work for different stimuli to produce all the visible colors. The second model is called the opponent process theory, which explains the visual system's responses in terms of opposing color pairs, for example, red-green and blue-yellow.

According to conditioning, color perception is also affected by other adaptive influences such as the unique effects of variations in lighting and nearby colors and individual differences like color blindness. The extremely stable color perceived for all visible objects, under the various conditions of illumination, is the color constancy. Factors beyond physiological mechanisms, such as culture, emotion, and psychology, too, shape the way humans interpret and ascribe meaning to colors.

14.5 COLOR MODEL

Color models are mathematical representations which describe and manipulate colors in a standardized way. These models define how colors are represented as combinations of primary components, making it easier to process, store, and reproduce colors digitally or physically. Different models serve specific applications, such as image processing, printing, and broadcasting. Below are some commonly used color models:

14.5.1 RGB Model

The RGB color model has three primary components: red, green, and blue. Each color component has an intensity value, typically ranging from 0 to 255 in an 8-bit system. The number of pixels used to represent a pixel of an RGB image is called as a pixel depth. As each color component is if 8-bit, the pixel depth of an RGB image is of 24-bit. Thus, this combination of these three components allows for the representation of over 16 million (2^{24}=16,777,216)unique colors. In this model, pure black is represented with (0,0,0), pure red with (255, 0, 0), pure green with (0, 255, 0), pure blue with (0, 0, 255) and pure white is represented with (255, 255, 255). Any intermediate value represents a color based on the contribution of these three primary colors. It is an additive model in which light added to these color components to create hue. It directly correlates to the way human vision perceives light and is particularly suited for devices like screens that emit light.Figure 14.1 demonstrate the RGB color cube and RGB image generation using three color components.

The RGB model is useful mainly for hardware. Electronic displays like televisions, computer screens, smartphones, digital cameras, and projectors use RGB models.

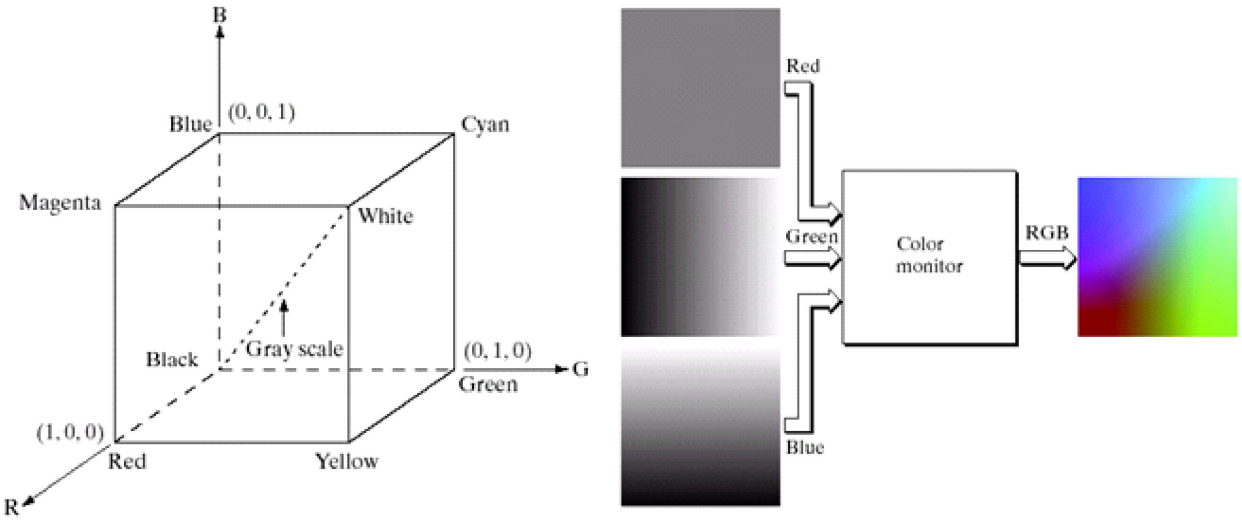


Figure 14.1. (a)The RGB Color cube, (b) Generating RGB image using three color planes (image source [1])

The main advantage of RGB model is ability to produce a huge range of colors making it suitable for applications that require detailed and accurate color production. Also, it is highly compatible with hardware ensuring faster processing and rendering of graphics. However, RGB model is not intuitive for human interpretation as human eyes perceive color in terms of hue, saturation, and brightness rather than red, green, blue. Another limitation comes with inconsistencies of across devices. Different devices come with different calibration for color production leading to difference in color output by different devices for the same amount of red, green, blue values.

In summary, RGB model is essential for all light-based systems, offering excellent precision and compatibility with digital devices. However, for certain tasks like color analysis or manipulation, transformations to more perceptually uniform color models may be necessary.

14.5.2 CMY Color Model

CMY, A subtractive color space which is popularly used for color reproducing in printed and other physical media. Unlike the

CMY model which subtracts the light to reproduce colors, the RGB model adds lights to create colors. In this model for example, cyan is made by reflecting green and blue while absorbing red light, magenta is made by reflected red and blue while absorbing green light, and yellow is made by reflected red and green while absorbing blue light. Forming black by mixing the three combines inks of red, green and blue can depict how every visible colorcan be produced by taking out the red light. Alas the mix creates a dark brownish color which could mean that the other two inks are filled instead of red because the cuff will make Beagle (or purple). To adjust this shortcoming the CMYK model includes black 'K' letter which has been said to stand for Key color black because it is used as the key plate in printing. This detail cut down the volume of ink used as well as increased the contrast of dark shades and details. The transformation from RGB to CMY can be expressed as equation 14.1.

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix} \qquad \text{————equation (14.1)}$$

The CMY/CMYK model's operational method relies on the premise that each color comprises its own set of absorbed light wavelengths by ink layers. As an illustration, cyan ink soaks up red light and reflects green and blue, indicating that the color is of a cyan tint. Different colors are produced by adding pixels of various inks and, through that process, subtracting light in some form. For example, blue pigments are formed after cyan and red green rays are subtracted when cyan and magenta are combined.

The CMY/CMYK pattern is important in a variety of applications, especially in printing, where it is the standard for accurate reproduction of characters on paper and other media Widely used in packaging and painting for the home, office, industrial printing systems, physical advertising, packaging and brochures

The strength of the model is its optimization for printing, where it better renders physical characters from digital characters, and reduces ink consumption and maintains high fidelity If you include a black (K) component a, it also increases black tone accuracy, improves contrast and texture for texture. However, the CMY/CMYK model has its limitations. This is not suitable for screens and other light displays, as it relies on subtractive color mixing rather than the additive approach used in RGB. Additionally, converting a digital RGB format to CMYK for printing can cause some color discrepancies due to differences in the color environment.

In summary, the CMY/CMYK model plays an important role in the printing industry, enabling efficient ink handling and accurate color reproduction on physical media even though it is not designed for digital presentation but supports RGB models is brought about by the distinction between digital structures and their physical manifestations.

14.5.3 HIS Color Model

The HSI color model has been formulated to suit the intuitive nature of humans when perceiving and describing colors. Three dimensional spaces of hue, saturation and intensity represent colors whereas chromatic elements are segregated from brightness: achromatic ones. This separation makes the model more comprehensible and functional in respect of many spheres related to image processing.

Hue refers to the type of color such as red, green, or blue. It is often described as an angle along a color wheel which extends from 0° to 360°. Consequently, the red hue corresponds to 0 degrees while the green and blue hues correspond to 120 and 240 degrees respectively. Saturation is the strength of the color, with a value of zero representing a tone of gray while a value higher than zero represents a more vivid color. Intensity, however, is defined as the average brightness of a color which may have been created from a

mixture of all colors, which means all the RGB values in this case as well. The HSI model has a wide range of applications such as in image segmentation, it is used to colorize certain regions of an image to identify several objects in an image. Also, some of the colors are enhanced to improve the images for diagnostic purposes in medical imaging. Furthermore, the model has numerous application in image enhancement where the lightness and contrast of an image can be adjusted separate to the hue and saturation of the image. Figure 14.2 shows the conceptual relationship between RGB and HSI model.

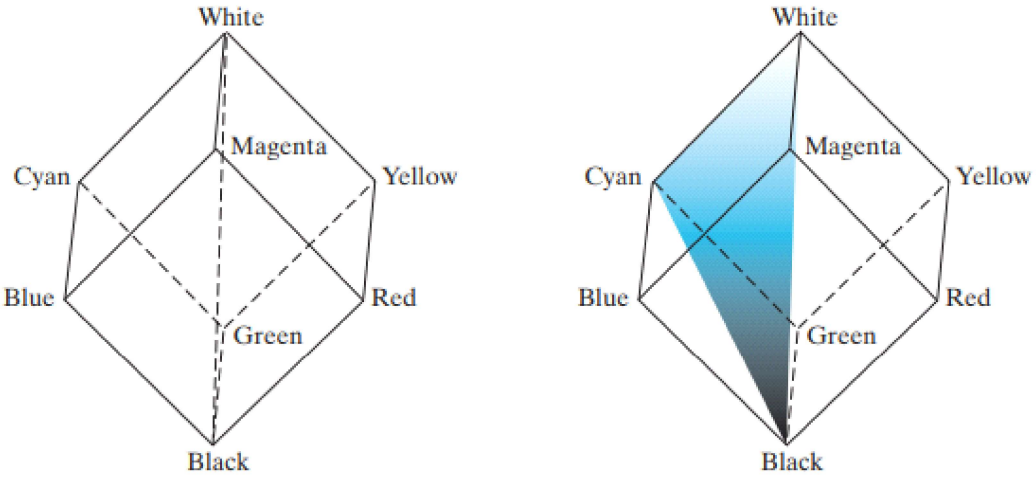


Figure 14.2 Conceptual relationship between RGB and HSI model (image source [1])

The HSI model has one of its best features in the ability to explain color concepts in an intuitive manner. If brightness is separated from the color components, it is possible to change brightness without changing the hue and saturation. This property can be very important in analysis work, like the one done in object recognition, where brightness as well ascolor information can also be processed independently.

The HSI model, however, has its own drawbacks. The conversion from RGB to HSI needs non-linear equations, which are computation intensive. Also, the model explains color non-uniformity,

wherein the visual differences of the color do not vary proportionately with the differences in the HSI values, and this can result in errors in use for some applications.

In summary, the HSI color model offers a simple and efficient means to represent and alter colors. It is a great advantage in almost all cases as it is possible to differentiate the luminance from the colors. Its application in image processing and in medical imaging is going to be most valuable. Its application is however weighed down by its complexity and non-uniformity of its application.

14.5.4 YIQ Color Model

The YIQ model is a color space historically used in television broadcasting, especially under NTSC (National Television Systems Commission) standards. It divides the image into three distinct categories: brightness (Y), in-phase (I), and quadrature (Q). The Y axis represents the bright or gray area of the image, ensuring compatibility with a black-and-white television. The I phase captures color information along the orange-to-teal axis, while the Q phase transmits color information along the purple-to-green axis. Together, the I and Q components produce chrominance, which carries details of the color, while the Y component brightens the image. The transformation from RGB to YIQ can be performed using equation 14.2.

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix} \text{———equation (14.2)}$$

This color model had the necessary features for high-definition use. Analog was a key feature of NTSC systems, enabling the transmission of video signals with color and brightness information. Moreover, the separation of brightness and chrominance improved the bandwidth used. This design took advantage of the fact that human vision is more sensitive to light than color details, allowing for better compression without noticeable loss of quality

The YIQ model has several strengths. Its efficiently distributed bandwidth ensures that more data is loaded into light, which is critical for human perception, and reduces the amount of data needed for chrominance Furthermore, the model was designed to match old black-and-white televisions and white supporting color TV will move backwards towards A smooth transition is ensured in the early days

However, the YIQ model has limitations that have limited its usefulness in modern applications. It has been largely replaced by more advanced color zones such as YUV and YCbCr which are better suited for digital systems.In addition, it is mainly limited to the use of analog video systems, making it mostly irrelevant in modern digital image processing and broadcasting.

In summary, the YIQ model was a leading color space that played an essential role in the television industry. It balanced luminance with chrominance for optimum usage of bandwidth and showed reverse compatibility. However, due to its obsoletion by newer technologies with enhanced performance in digital systems, it has been supplanted.

14.6 HISTOGRAM EQUALIZATION OF COLOR IMAGE

Histogram equalization is a popular image processing technique used to enhance the contrast of an image by redistributing its intensity levels. For color images, this technique should be used carefully to produce naturally balanced colors while enhancing the visual effect. Since histogram equalization applied directly to any color channel (RGB) can distort the color balance of an image, the process often involves converting the image into a color space that separates intensity (or brightness) from chromatic components such as HSV (Hue, Saturation) inside , Value)) or YCbCr. The histogram equalization is then applied only to the intensity component (e.g., the V channel of HSV or the Y channel of YCbCr), excluding the chromatic components (hue and saturation).

After matching, the intensified component is recombined with the original chromatic information, and the image is converted back to the original RGB color space This keeps the colors visually the same, while difference is all significantly increased. Histogram equalization is particularly useful in applications where detail is required, such as medical imaging, satellite imagery, and photography It can help reveal hidden information in areas that are less obvious or overexposed, and making it a valuable tool for both system design and hands-on image analysis. Figure 14.3 shows the result of histogram equalization when applied to a low contrast image.

14.7 COLOR IMAGE FILTERING

Color image filtering is an important technique in image processing. This technique is applied to improve, alter, or even extract features from color images. In grayscale images, filtering is performed on one intensity channel; however, color images contain multiple channels, such as red, green, and blue. Every channel is associated with a particular color component, and filtering can be applied directly to these channels or to derived components in other color spaces, such as HSV or YCbCr. Filtering may be used for smoothing textures, enhancing edges, or even to make features stand out in an image. One of the most common applications of color image filtering is noise reduction, where Gaussian, median, or bilateral filters are used to smooth out irregularities while preserving the most important details.

Gaussian filters apply a weighted averaging process to reduce high-frequency noise, whereas median filters are effective at removing salt-and-pepper noise by replacing each pixel with the median of its neighbors. Bilateral filters go a step further by smoothing an image while retaining the details of edges, hence highly useful in color images in which edges are boundaries to objects. Another important application of color image filtering is edge detection. This can be done by employing Sobel, Prewitt, or Laplacian filters to detect intensity or color transitions and emphasize the boundaries between objects within the image. In general, when working with color

images, these filters are applied to the intensity component of a color space such as HSV because edge information is naturally represented in brightness rather than in chromatic channels. Color-based segmentation or enhancement can also be applied using filtering. For instance, filters can be devised to extract specific color bands or to emphasize

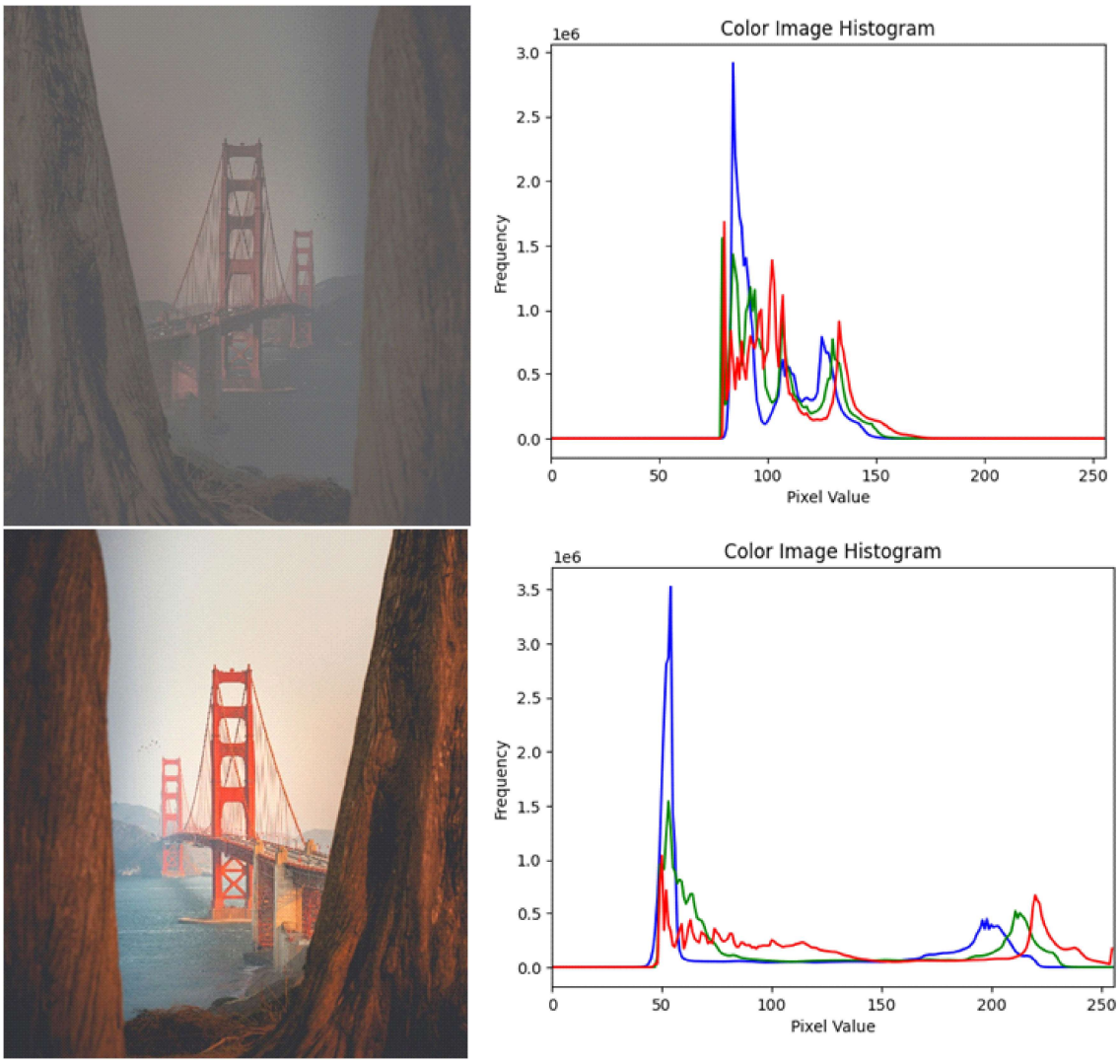


Figure 14.3. (a) A low contrast image (b) Histogram of the image (a), (c) Image (a) after histogram equalization and (d) Histogram of the image (c)

certain colors in an image, and this is helpful in object recognition, medical imaging, and remote sensing applications where specific features may not easily be seen by emphasizing the same. Figure 14.4 shows the results of image smoothing and sharpening filters.

In summary, color image filtering is a versatile tool that is important in tasks such as noise reduction, edge detection, feature enhancement, and segmentation. The choice of filtering method and its application to specific color channels or intensity components depends on the requirements of the task. Properly applied, filtering can significantly improve the quality and usability of color images in many domains.



Figure 14.4 (a) Original Lena Image, (b) image (a) after applying image smooting using a 5x5 average filter and (c) image (a) after image sharpening using the Laplacian (image source [1])



CHECK YOUR PROGRESS

Q1. Which of the following tasks is NOT typically a part of color image processing?

a) Color transformation
b) Image filtering
c) Edge detection
d) Audio signal processing

Q2. How does the human eye perceive color?

a) Through rods sensitive to all wavelengths of light
b) Through cones sensitive to red, green, and blue wavelengths
c) Through a combination of rods and cones sensitive to ultraviolet light
d) Through the optic nerve directly detecting RGB signals

Q3. What type of model is the RGB color model?

a) Subtractive

b) Additive

c) Multiplicative

d) Perceptual

Q4. Why is the CMYK model preferred in printing over the RGB model?

a) It is an additive model better suited for displays.

b) It separates brightness and color components.

c) It includes a black component for better contrast and efficiency.

d) It uses only primary colors for a broader color gamut.

Q5. In the HSI model, what does the “Saturation” component represent?

a) The brightness of the color.

b) The type of color, such as red or green.

c) The purity or vividness of the color.

d) The intensity of red, green, and blue combined.

Q6. What is the primary advantage of the YIQ color model in broadcasting?

a) High compatibility with digital displays.

b) Efficient bandwidth utilization by separating brightness and color details.

c) Superior color accuracy for printing purposes.

d) Backward compatibility with modern RGB systems.

Q7. Which channel is typically used for histogram equalization in the YIQ model?

a) Y (Luminance)

b) I (In-phase)

c) Q (Quadrature)

d) All three channels simultaneously

Q8. What is the main purpose of histogram equalization in color image processing?

a) To reduce noise in color images.

b) To enhance contrast by redistributing pixel intensities.

c) To convert between color models.

d) To sharpen edges in a color image.

Q9. What is the key difference between additive and subtractive color models?

a) Additive models are used for printing, while subtractive models are for displays.

b) Additive models mix light to form colors, while subtractive models mix pigments to absorb light.

c) Additive models use CMY components, while subtractive models use RGB components.

d) Subtractive models are hardware-based, while additive models are perceptual.

Q10. In color image filtering, why might one process the luminance component separately?

a) To reduce computational complexity.

b) To ensure color accuracy is maintained.

c) Because luminance is more critical to human perception than chrominance.

d) To apply histogram equalization directly on chrominance components.



14.8 LET US SUM UP

- Human vision perceives color through cones in the retina, sensitive to red, green, and blue wavelengths.
- Colors are interpreted based on hue, saturation, and brightness.
- RGB is an additive color model combining red, green, and blue light to form colors.

- Widely used in digital displays like monitors, TVs, and cameras.
- Subtractive color model like CMY used in printing, where inks absorb specific wavelengths to create colors.
- CMYK adds a black component for better contrast and efficiency.
- Optimized for physical media but requires conversion from RGB for digital designs.
- Cylindrical color model HSI is based on how humans perceive color and useful in image editing and segmentation for its intuitive representation.
- YIQ model is optimized for human sensitivity to brightness but less relevant in modern digital systems.
- Histogram Equalization (HE) of color images enhances contrast in images by redistributing pixel intensities.
- Color image filtering has a range of applications including noise reduction, edge enhancement, and feature extraction.



14.9 FURTHER READING

- "Digital Image Processing" by Rafael C. Gonzalez and Richard E. Woods, Pearson, 4th Edition, 2018.
- "Color Image Processing: Methods and Applications" by Rastislav Lukac and Konstantinos N. Plataniotis, CRC Press, 1st Edition, 2006.
- "Image Processing, Analysis, and Machine Vision" by Milan Sonka, Vaclav Hlavac, and Roger Boyle, Cengage, 4th Edition, 2014.
- "Fundamentals of Multimedia" by Ze-Nian Li, Mark S. Drew, and Jiangchuan Liu, Springer, 2nd Edition, 2014.



14.10 ANSWERS TO CHECK YOUR PROGRESS

Q1. d) Audio signal processing

Q2. b) Through cones sensitive to red, green, and blue wavelengths

Q3. b) Additive

Q4. c) It includes a black component for better contrast and efficiency.

Q5. c) The purity or vividness of the color.

Q6. b) Efficient bandwidth utilization by separating brightness and color details.

Q7. a) Y (Luminance)

Q8. b) To enhance contrast by redistributing pixel intensities.

Q9. b) Additive models mix light to form colors, while subtractive models mix pigments to absorb light.

Q10. c) Because luminance is more critical to human perception than chrominance.



14.11 MODEL QUESTIONS

Q 1: Explain the significance of color image processing and its key applications in various domains.

Q 2: Describe how humans perceive color and the role of cones in the human visual system.

Q 3: What are the primary differences between additive and subtractive color models? Provide examples of each.

Q 4: Discuss the RGB color model in detail, including its working principle, applications, and limitations.

Q 5: Explain the CMY/CMYK color model. Why is the black (K) component important in printing?

Q 6: Compare the RGB and CMYK color models. Highlight their advantages and disadvantages in practical applications.

Q 7: What is the HSI color model? Describe its components and explain how it aligns with human perception of color.

Q 8: Discuss the YIQ color model, its components, and why it is suitable for analog television broadcasting.

Q 9: Illustrate how histogram equalization is applied to color images and explain its importance in enhancing image contrast.

Q10: Describe the process and benefits of filtering color images. How is it different from filtering grayscale images?

Q11: Explain the transformations required to convert an image from the RGB color space to the HSI color space.

Q12: Discuss the strengths and limitations of the YIQ model compared to modern color spaces like YUV or YCbCr.

Q13: What challenges are involved in applying histogram equalization to color images compared to grayscale images?

Q14: Describe the role of luminance and chrominance in color models like YIQ and explain why they are separated.

Q15: How does color image filtering enhance image quality? Provide examples of techniques like smoothing and sharpening.

*****

UNIT 15: WAVELET-BASED IMAGE PROCESSING

UNIT STRUCTURE

15.1 Learning Objectives

15.2 Introduction

15.3 Wavelet

15.4 Discrete Wavelet Transform

15.5 Haar Wavelet

15.6 Filter Bank

15.7 Embedded Image Coding

15.8 Let Us Sum Up

15.9 Further Reading

15.9 Answers to Check Your Pogress

15.10 Model questions

15.1 LEARNING OBJECTIVES

After going through this unit, you will be able to

- explain the concept of wavelet
- explain the theoretical background of continuous wavelet transform
- describe the significance of discrete wavelet transform
- describe the usefulness of Haar Wavelet Transform
- explain the significance of filter bank
- explain the significance of embedded image encoding

15.2 INTRODUCTION

The Fourier Transform has long been a fundamental mathematical tool in signal and image processing, enabling the analysis and optimization of spectral characteristics in the frequency domain. However, its global nature can limit its effectiveness when handling localized or transient features in images. Wavelet transforms provide a multi-resolution approach that captures both frequency and spatial information, addressing this limitation. This capability has contributed to the widespread adoption of wavelet-based

techniques across a variety of image processing applications. Wavelet transforms decompose an image into a series of scaled and shifted versions of a base function referred to as a wavelet. In contrast to the infinite sinusoidal waves of the Fourier Transform, wavelets are localized waveforms that offer a powerful mechanism for analyzing image features at different scales and locations. This multi-resolution approach allows for the efficient representation of both smooth regions and sharp transitions, rendering wavelets particularly well-suited for tasks such as image compression, denoising, and feature extraction. The adaptability of wavelets to local image characteristics has driven the development of numerous wavelet families, each tailored to specific applications and image properties.

15.3 WAVELET

Wavelets are waveforms that are localized and confined within specific, well-defined time and frequency ranges, in contrast to the sinusoidal waves employed in Fourier analysis, which are confined solely to the frequency domain but extend infinitely in time. This unique localization property of wavelets, in both time and frequency, makes them well-suited and advantageous for the analysis of signals and images containing transient features or discontinuities. The ability to localize in both the time and frequency domains allows wavelets to effectively capture and characterize the temporal and spectral properties of the input data, which is crucial for applications such as feature extraction, signal denoising, and image compression.

A wavelet is a function $\psi(t)$ (often called the “mother wavelet”) that satisfies the admissibility condition:

$$C_\psi = \int_{-\infty}^{\infty} \frac{|\hat{\psi}(f)|^2}{|f|} df < \infty$$

where $\hat{\psi}(f)$ is the Fourier transform of $\psi(t)$. This condition essentially ensures that the wavelet has zero mean and sufficient decay in both time and frequency domains.

From the mother wavelet, a family of wavelets is generated through scaling and shifting:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi(\frac{t-b}{a})$$

where:

- a is the scaling parameter, which controls the width or frequency of the wavelet. Larger values of a correspond to lower frequencies (wider wavelets) and vice-versa.
- b is the translation parameter, which shifts the wavelet in time.

Continuous Wavelet Transform

The *Continuous Wavelet Transform (CWT)* of $x(t)$ a signal is then defined as the inner product of the signal with the scaled and shifted wavelet:

$$CWT(a,b) = \int_{-\infty}^{\infty} f(t)\frac{1}{\sqrt{|a|}}\psi^*(\frac{t-b}{a})dt$$

where:

- $\psi(t)$ is the mother wavelet, a localized waveform.
- a is the scaling parameter, controlling the wavelet's width (frequency).
- b is the translation parameter, shifting the wavelet in time.
- $\psi^*(t)$ denotes the complex conjugate of $\psi(t)$.

The CWT provides a time-frequency representation of the signal, showing how the frequency content changes over time.

15.4 DISCRETE WAVELET TRANSFORM

The Discrete Wavelet Transform is a powerful mathematical tool used for analyzing signals and images at different scales and resolutions. It's derived from the Continuous Wavelet Transform but differs in how it samples the scale and translation parameters. While the CWT uses continuous values for these parameters, the DWT restricts them to discrete values, typically powers of two. This discretization makes the DWT computationally efficient and suitable for digital implementation.

The DWT discretizes the scaling and translation parameters:

$$DWT(j,k) = \sum_{n=0}^{N-1} f(n)2^{-j/2}\psi(2^{-j}n - k)$$

where:

- j is the scale parameter (integer).

- k is the translation parameter (integer).
- N is the length of the discrete signal $f(n)$.

The CWT provides a highly redundant representation of the signal, meaning it contains more information than necessary for perfect reconstruction. This redundancy can be beneficial for certain applications but comes at the expense of increased computational complexity. In contrast, the DWT, by utilizing discrete scales and translations, achieves a non-redundant or critically sampled representation. This reduction in redundancy significantly decreases the computational burden, rendering the DWT more practical for many applications. In essence, the DWT offers a more practical and efficient approach to wavelet analysis compared to the CWT, while still retaining the ability to analyze signals at multiple resolutions.

The Discrete Wavelet Transform is extensively utilized across diverse applications due to its capacity to analyze signals at multiple resolutions, effectively capturing both frequency and spatial information. In the realm of image and video compression, such as JPEG2000, DWT plays a crucial role in reducing data redundancy by decomposing images into approximation and detail coefficients. Moreover, it finds application in denoising and signal processing, allowing the separation of noise from meaningful data while preserving critical features. Within the domain of medical imaging, DWT enhances images and compresses large datasets, facilitating efficient storage and transmission. Additionally, DWT is employed in biometric systems for feature extraction and in finance for time-series data analysis, establishing its versatility across numerous domains.

Implementation of Discrete Fourier Transform using Python's PyWavelets package:

```
import matplotlib.pyplot as plt

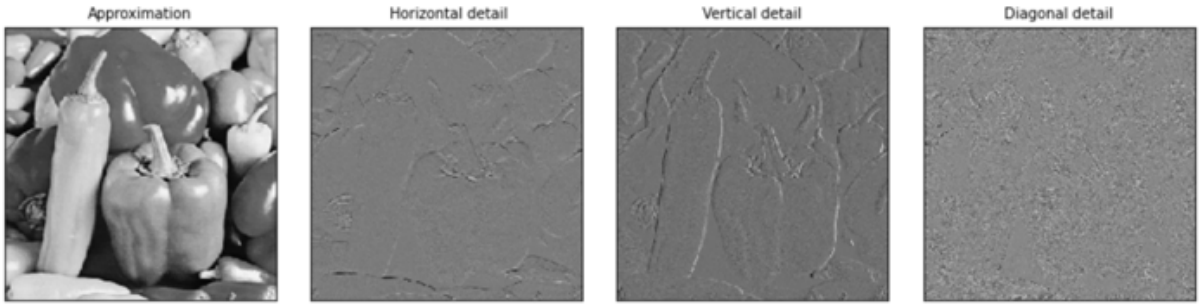
import pywt
import pywt.data

img = cv2.imread('peppers_gray.tif',0)

titles = ['Approximation', ' Horizontal detail',
          'Vertical detail', 'Diagonal detail']
coeffs2 = pywt.dwt2(img, 'bior1.3')
LL, (LH, HL, HH) = coeffs2
fig = plt.figure(figsize=(12, 3))
for i, a in enumerate([LL, LH, HL, HH]):
    ax = fig.add_subplot(1, 4, i + 1)
    ax.imshow(a, interpolation="nearest", cmap=plt.cm.gray)
    ax.set_title(titles[i], fontsize=10)
    ax.set_xticks([])
    ax.set_yticks([])

fig.tight_layout()
plt.show()
```

Output:



15.5 HAAR WAVELET

The Haar wavelet is one of the simplest and most widely used wavelets in the domains of signal processing and numerical analysis. It serves as a foundational concept for understanding the fundamental principles of wavelet transforms and proves particularly advantageous for applications that require fast, computationally efficient transformations.

The Haar mother wavelet, $\psi(t)$ is defined as:

$$\psi(t) = \begin{cases} 1 & \text{if } 0 \leq t < 0.5, \\ -1 & \text{if } 0.5 \leq t < 1, \\ 0 & \text{otherwise.} \end{cases}$$

The corresponding scaling function, $\phi(t)$ is:

$$\phi(t) = \begin{cases} 1 & \text{if } 0 \leq t < 1, \\ 0 & \text{otherwise.} \end{cases}$$

The key features of Haar Wavelet are as follows:

- **Simplicity**: Its simple mathematical form, consisting of just constant values (+1, -1, and 0), makes it easy to understand, implement, and compute. This simplicity contributes to its computational efficiency, making it suitable for real-time applications.
- **Compact Support**: The Haar wavelet is only non-zero over a finite interval. This localized nature allows it to effectively capture sharp transitions and discontinuities in signals and images.
- **Orthogonality**: The Haar wavelet and its scaled and shifted versions are orthogonal to each other. This property is crucial for efficient signal decomposition and reconstruction, as it ensures that the wavelet coefficients represent independent pieces of information.
- **Fast Computation**: Due to its simple structure and orthogonality, the Haar wavelet transform can be computed very quickly using fast algorithms. This makes it attractive for applications where computational speed is critical.
- **Edge Detection**: The Haar wavelet's sensitivity to abrupt changes makes it well-suited for detecting edges and discontinuities in images. This is exploited in various image processing tasks, such as feature extraction and object recognition.

Haar Wavelet Transform

The Haar wavelet transform decomposes a signal into approximation (low-frequency components) and detail (high-frequency components) coefficients. It is performed iteratively, allowing multilevel decomposition.

15.6 FILTER BANK

Wavelet-based image processing relies on filter banks to perform the Discrete Wavelet Transform. These filter banks decompose an image into distinct frequency sub-bands, effectively separating the image's features at various scales. The image undergoes a series of high-pass and low-pass filtering operations. The high-pass filter extracts the high-frequency details (edges, textures), while the low-pass filter captures the low-frequency information (smooth areas). Following the filtering step, the output is down-sampled by a factor of 2, reducing the data size while maintaining essential information. A multi-resolution representation of the image is created by iteratively applying this process to the low-pass output from the previous step. To reconstruct the original image, the process is reversed using up-sampling and inverse filters.

Implementation of Filter Bank with Haar Wavelet using Python's PyWavelets package:

```
import pywt
import numpy as np
import cv2
import matplotlib.pyplot as plt

def apply_dwt_filter_bank(image, wavelet='haar', level=1):
    coeffs = pywt.wavedec2(image, wavelet, level=level)
    return coeffs

def reconstruct_from_dwt(coeffs, wavelet='haar'):
    return pywt.waverec2(coeffs, wavelet)

image_path = 'peppers_gray.tif'
image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)

# Apply DWT filter bank
wavelet_type = 'haar'
level = 2 # Decomposition level
coeffs = apply_dwt_filter_bank(image, wavelet=wavelet_type, level=level)

# Display approximation and detail coefficients
cA = coeffs[0]
details = coeffs[1:]  # List of (cH, cV, cD) at each level
```

Plotting the images based in the different coefficients:

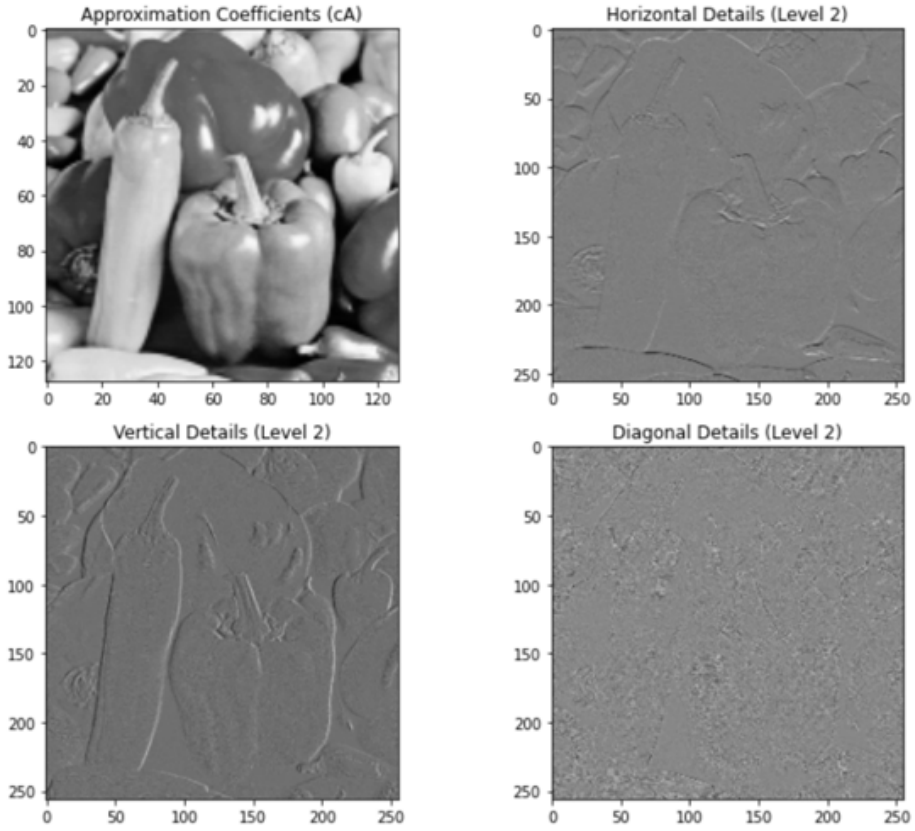
```
plt.figure(figsize=(10, 8))
plt.subplot(2, 2, 1)
plt.title("Approximation Coefficients (cA)")
plt.imshow(cA, cmap='gray')

for i, (cH, cV, cD) in enumerate(details):
    plt.subplot(2, 2, 2)
    plt.title(f"Horizontal Details (Level {i+1})")
    plt.imshow(cH, cmap='gray')

    plt.subplot(2, 2, 3)
    plt.title(f"Vertical Details (Level {i+1})")
    plt.imshow(cV, cmap='gray')

    plt.subplot(2, 2, 4)
    plt.title(f"Diagonal Details (Level {i+1})")
    plt.imshow(cD, cmap='gray')

plt.tight_layout()
plt.show()
```



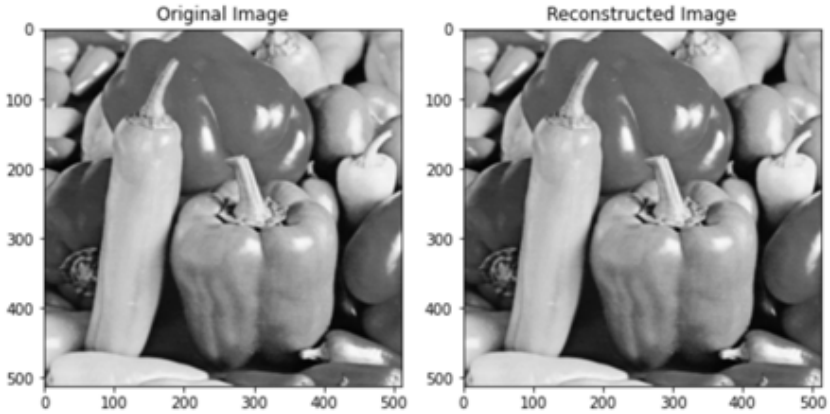
Plotting the original and the reconstructed image:

```
# Reconstruct the image
reconstructed_image = reconstruct_from_dwt(coeffs, wavelet=wavelet_type)

# Show original and reconstructed images
plt.figure(figsize=(8, 4))
plt.subplot(1, 2, 1)
plt.title("Original Image")
plt.imshow(image, cmap='gray')

plt.subplot(1, 2, 2)
plt.title("Reconstructed Image")
plt.imshow(reconstructed_image, cmap='gray')

plt.tight_layout()
plt.show()
```



15.7 EMBEDDED IMAGE CODING

Embedded image coding, also known as *embedded zerotree wavelet coding*, is a powerful compression technique used in wavelet-based image processing. It generates a bitstream that is progressively refinable, meaning that the image quality improves as more bits are received.

Embedded image coding's key advantage is its capacity for progressive image transmission. This feature proves valuable in bandwidth-constrained scenarios or when quick previews are required before downloading the full-resolution image. Additionally, embedded coders provide granular control over the bit rate, facilitating compression to target file sizes.

The embedded bitstream's ability to be truncated at any point further enables the generation of lower-resolution or lower-quality image versions as needed.



CHECK YOUR PROGRESS

1. Which of the following best describes a key advantage of wavelet transforms over Fourier transforms?
   A. Wavelets are easier to implement
   B. Wavelets are global in nature
   C. Wavelets capture both frequency and spatial information
   D. Fourier transforms are better for localized features

2. The Discrete Wavelet Transform (DWT) differs from the Continuous Wavelet Transform (CWT) in that DWT:
   A. Uses continuous scaling parameters
   B. Is computationally more complex
   C. Provides a redundant representation
   D. Uses discrete scales and translations

3. Which wavelet is the simplest and often used for introductory learning and fast computation?
   A. Daubechies wavelet
   B. Haar wavelet
   C. Morlet wavelet
   D. Biorthogonal wavelet

4. In wavelet decomposition, high-pass filters are used to extract:
   A. Low-frequency components (smooth areas)
   B. Noise components
   C. High-frequency components (edges and textures)
   D. Color information

5. What is the primary purpose of embedded image coding in wavelet-based image processing?
   A. To increase image resolution
   B. To create infinite compression
   C. To enable progressive image transmission
   D. To reduce color distortions

6. Which of the following properties is NOT associated with the Haar wavelet?
   A. Compact support
   B. Orthogonality
   C. Continuous smooth curve
   D. Fast computation



15.8 LET US SUM UP

- Wavelet transforms provide a multi-resolution approach to image processing by capturing both spatial and frequency information, making them highly effective for localized and transient feature analysis.
- Unlike the Fourier Transform, which uses global sine and cosine functions, wavelets are localized functions capable of analyzing signals at different scales and positions.
- A wavelet is a scaled and shifted version of a mother wavelet function, satisfying the admissibility condition, which ensures it has a zero mean and is suitable for signal decomposition.
- The Continuous Wavelet Transform (CWT) offers a redundant, high-resolution analysis of signals in time-frequency space, but it is computationally intensive.
- The Discrete Wavelet Transform (DWT) uses discrete values of scale and translation, resulting in a compact and efficient representation suitable for digital implementation.

- DWT is widely used in image compression (e.g., JPEG2000), denoising, biometric feature extraction, medical imaging, and financial data analysis due to its efficiency and multi-resolution capability.
- The Haar Wavelet is the simplest wavelet, offering fast computation, compact support, and orthogonality, making it suitable for edge detection, signal decomposition, and real-time applications.
- Filter banks are essential components in DWT, separating an image into high-frequency and low-frequency components using high-pass and low-pass filters, followed by down-sampling.
- The reconstruction process in wavelet-based systems uses inverse filters and up-sampling to recover the original image from its wavelet coefficients.
- Embedded Image Coding produces a progressive bitstream that allows scalable and efficient image compression, ideal for applications with bandwidth limitations or quality scalability requirements.
- Wavelet transforms are especially powerful for handling images with sharp edges, discontinuities, and localized patterns, outperforming traditional transforms in many practical scenarios.



15.9 FURTHER READING

- "Digital image processing and pattern recognition", Pakhiramalay K, PHI
- "Digital image processing", Sundaramurthy Sridhar, Oxford
- "Fundamentals of Digital Image Processing", S Annadurai, R Shanmugalakshmi, Pearson



15.10 ANSWERS TO CHECK YOUR POGRESS

1. C – Wavelets capture both frequency and spatial information.
2. D – DWT uses discrete scales and translations.
3. B – The Haar wavelet is the simplest and widely used.

4. C – High-pass filters extract high-frequency components like edges and textures.
5. C – Embedded image coding allows progressive transmission where quality improves as more data is received.
6. C – Haar wavelet is not a continuous smooth curve; it is piecewise constant.



15.11 MODEL QUESTIONS

Q 1: Explain the concept of wavelet transforms. How do they overcome the limitations of Fourier transforms in image processing?

Q 2: Describe the structure and mathematical formulation of the Continuous Wavelet Transform (CWT). What is its significance in signal and image analysis?

Q 3: What is the Discrete Wavelet Transform (DWT)? Discuss its advantages and applications in digital image processing.

Q 4: Write a detailed note on the Haar Wavelet. Discuss its mathematical properties and its role in efficient image analysis.

Q 5: What are filter banks in wavelet-based image processing? Explain how they function in the decomposition and reconstruction of images.

Q 6: Discuss the concept and advantages of embedded image coding. How does it facilitate progressive image transmission and scalable compression?

Q 7: Compare and contrast the Haar Wavelet with other wavelet families in terms of computational efficiency, support, and application.

Q 8: Illustrate with examples how wavelet transforms are applied in real-world scenarios such as medical imaging, biometric systems, and video compression.

*****