INDEX

Chapter 1	:	Introduction to Image Processing
Chapter 2	:	Image Sensing and Acquisition
Chapter 3	:	Sampling and Quantization
Chapter 4	:	Image Enhancement in Spatial Domain
Chapter 5	:	Histogram Modelling
Chapter 6	:	Image Enhancement in Frequency Domain
Chapter 7	:	Image Segmentation
Chapter 8	:	Image Morphology, Representation and Description
Chapter 9	:	Image Transforms
Chapter 10	:	Wavelet Transform
Chapter 11	:	Image Compression
Chapter 12	:	Colour Image Processing
Chapter 13	:	Introduction to Digital Video

Table of Contents

- **Index**
- **Syllabus**
- **Dec. 2015** — **D(15)-1 to D(15)-22**
- **May 2016** — **M(16)-1 to M(16)-12**
- **Dec. 2016** — **D(16)-1 to D(16)-24**
- **May 2017** — **M(17)-1 to M(17)-25**
- **University Question Papers** — **Q-1 to Q-6**

❑❑❑

Image Processing

Statistical Analysis

Chapter No.	Dec. 2015	May 2016
Chapter 1	-	-
Chapter 2	-	08 Marks
Chapter 3	-	05 Marks
Chapter 4	15 Marks	12 Marks
Chapter 5	-	10 Marks
Chapter 6	10 Marks	05 Marks
Chapter 7	25 Marks	15 Marks
Chapter 8	20 Marks	15 Marks
Chapter 9	30 Marks	25 Marks
Chapter 10	-	-
Chapter 11	20 Marks	25 Marks
Chapter 12	10 Marks	-
Chapter 13	-	-
Repeated Questions	-	30 Marks

Dec. 2015

Chapter 4 : Image Enhancement in Spatial Domain

[Total Marks : 15]

Q. 1(a) What do you understand by zero memory operation. **(5 Marks)**

Ans. :

In point processing, we work with single pixels i.e. T is 1 × 1 operator. It means that the new value g(x, y) depends on the operator T and the present f(x, y).

To implement point processing techniques, we do not need to store previous values. Hence no memory is required for point processing operations. Because of this, point processing techniques are also called zero memory operations.

Some of the common examples of point processing are

(1) Digital negative
(2) Contrast stretching
(3) Thresholding
(4) Grey level slicing
(5) Bit plane slicing
(6) Dynamic range compression
(7) Power law transformation

Q. 2(b) For the given 3 bits per pixel, 4 × 4 size image perform following operations.

(i) Intensity level slicing with background, $r_1 = 3$ and $r_2 = 5$.

(ii) Bit plane slicing.

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

(10 Marks)

Ans. :

(i) Intensity level slicing with background

The transformation for grey level slicing with background is shown in Fig. Q. 2(b)

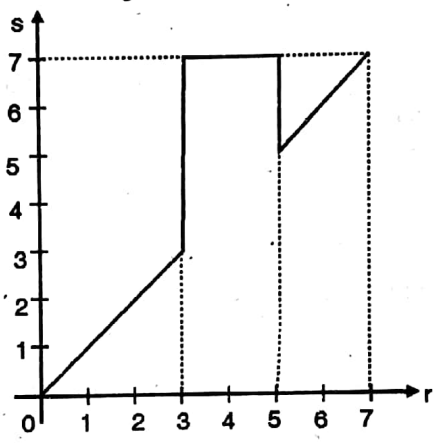


Fig. Q. 2(b)

$$s = \begin{cases} L-1 & r_1 \le r \le r_2 \\ r & \text{otherwise} \end{cases}$$

Since the image is 3 bit, we have

$$L = 2^3 = 8; \therefore L - 1 = 7$$

$$s = \begin{cases} 7 & 3 \le r \le 5 \\ r & \text{otherwise} \end{cases}$$

Hence the output image is as shown

6	2	7	2
1	7	0	7
7	7	2	1
2	7	7	6

(ii) Bit plane slicing :

We begin with converting the image into a 3-bit binary image.

110	010	011	010
001	101	000	111
100	011	010	001
010	101	111	110

We now create 3 planes from the bit positions.

1	0	0	0
0	1	0	1
1	0	0	0
0	1	1	1
MSB plane			

1	1	1	1
0	0	0	1
0	1	1	0
1	0	1	1
Middle plane			

0	0	1	0
1	1	0	1
0	1	0	1
0	1	1	0
LSB plane			

Chapter 6 : Image Enhancement in Frequency Domain

[Total Marks : 10]

Q. 3(b) Explain Homomorphic filtering with the help of block diagram. **(10 Marks)**

Ans. : Homomorphic Filtering :

Homomorphic filter uses a slightly different approach compared to other frequency domain filters. Homomorphic filtering uses the illumination-reflectance model to filter images in the frequency domain. In homomorphic filtering we seperate the illumination and reflectance components and work with them seperately.

$$f(x, y) = i(x, y) \cdot r(x, y)$$

We cannot directly take the Fourier transform since

$$F\{f(x, y)\} \neq F\{i(x, y)\}F\{r(x, y)\}$$

To separate the i and the r components, we use the log function.

$$Z(x, y) = \ln f(x, y)$$

$$Z(x, y) = \ln i(x, y) + \ln r(x, y)$$

We now take the Fourier transform to move into the frequency domain

$$F\{Z(x, y)\} = F\{\ln i(x, y)\} + F\{\ln r(x, y)\}$$

$$Z(u, v) = F_i(u, v) + F_r(u, v)$$

$$\text{Where } F_i(u, v) = F\{\ln i(x, y)\}$$

$$F_r(u, v) = F\{\ln r(x, y)\}$$

We now use a filter H(u, v)

$$\therefore S(u, v) = Z(u, v) \cdot H(u, v)$$

$$S(u, v) = F_i(u, v) \cdot H(u, v) + F_r(u, v) \cdot H(u, v)$$

The key approach in homomorphic filtering is to seperate the illumination and reflectance components. Between them i(x, y) contributes to the low frequency since illumination is more or less uniform and r(x, y) is the high frequency component since r(x, y) tends to vary abruptly at junctions of dissimilar objects.

A typical homomorphic filter H(u, v) is shown in Fig. 2-Q. 3(b).

Generally $Y_L < 1$ and $Y_H > 1$. H (u, v) tends to decrease the contribution made by low frequencies (illumination) and amplify the contribution made by high frequencies (reflectance).

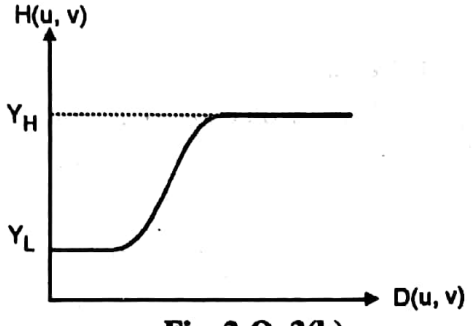


Fig. 2-Q. 3(b)

How do we get back the modified image ?

$$S(u, v) = H(u, v) F_i(u, v) + H(u, v) F_r(u, v)$$

To come back to the spatial domain, we take the inverse Fourier transform.

$$s(x, y) = F^{-1}\{S(u, v)\}$$
$$= F^{-1}\{H(u, v) F_i(u, v)\} + F^{-1}\{H(u, v) F_r(u, v)\}$$
$$s(x, y) = i'(x, y) + r'(x, y)$$

Finally remember, the first step in homomorphic filtering was to take the log, hence here we need to take the inverse operation i.e., the exponential operation.

$$\therefore g(x, y) = e^{s(x, y)} = e^{i'(x, y)} \cdot e^{r'(x, y)}$$
$$= i_0(x, y) \cdot r_0(x, y)$$

where $i_0(x, y) = e^{i'(x, y)}$

$$r_0(x, y) = e^{r'(x, y)}$$

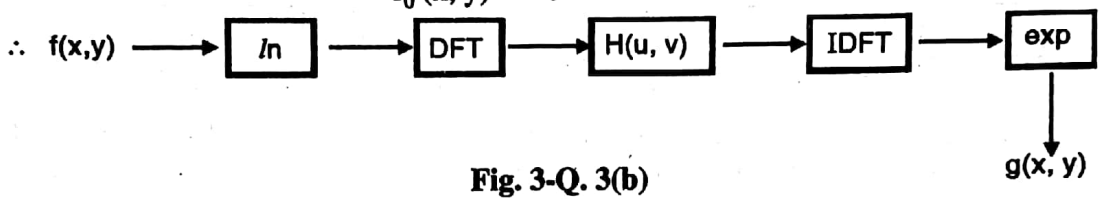


Fig. 3-Q. 3(b)

The filter H(u, v) can be implemented by a simple formula,

$$H(u, v) = (Y_H - Y_L)\ [1 - e^{-C(D^2(u, v)/D_0^2)}] + Y_L \quad ...(2)$$

C is a constant, used to control the sharpness.

Chapter 7 : Image Segmentation [Total Marks : 25]

Q. 1(b) Discuss different discontinuities in image. **(5 Marks)**

Ans. : Image Segmentation based on discontinuities :

When we look at an image, we immediately observe 3 basic types of discontinuities in the grey level viz. points, lines and edges. The easiest way is to use spatial masks which have properties to detect these discontinuities.

Point detection :

Remember one thing, points, lines and edges are all high frequency components and hence we need masks which are basically high pass. Hence the masks that we present here for point, line and edge detection have the properties of a high mask mainly, the sum of the coefficients of the mask has to be equal to zero in all the three cases.

Detecting points is fairly simple. We use the standard high pass mask for it. And so that this mask detects only points and not lines, we set a threshold value i.e., we say a point has been detected at the location on which the mask is centered only if

−1	−1	−1
−1	8	−1
−1	−1	−1

$$|R| \geq T \quad ...(1)$$

Where R is derived from the standard convolution formula

$$R = w_1 z_1 + w_2 z_2 + \ldots\ldots\ldots + w_9 z_9$$

$$R = \sum_{i=1}^{9} w_i z_i$$

We take | R | because we want to detect both the kinds of points i.e. white points on a black background as well as black points on a white background.

T is a non negative threshold which is defined by the user.

Line detection :

Detection of lines can be done using the masks shown below. In an image, lines can be in any direction and detecting these lines would need different masks.

−1	−1	−1
2	2	2
−1	−1	−1

Horizontal

−1	−1	2
−1	2	−1
2	−1	−1

+ 45°

−1	2	−1
−1	2	−1
−1	2	−1

Vertical

2	−1	−1
−1	2	−1
−1	−1	2

− 45°

We notice that all these masks have a sum equal to zero, and hence all of them are high pass masks. The first mask would respond strongly to lines that are oriented horizontally. The second mask would respond to lines at an angle of + 45°, the third mask would respond strongly to vertical lines and the last mask would respond strongly to lines at an angle of − 45°.

Edge detection :

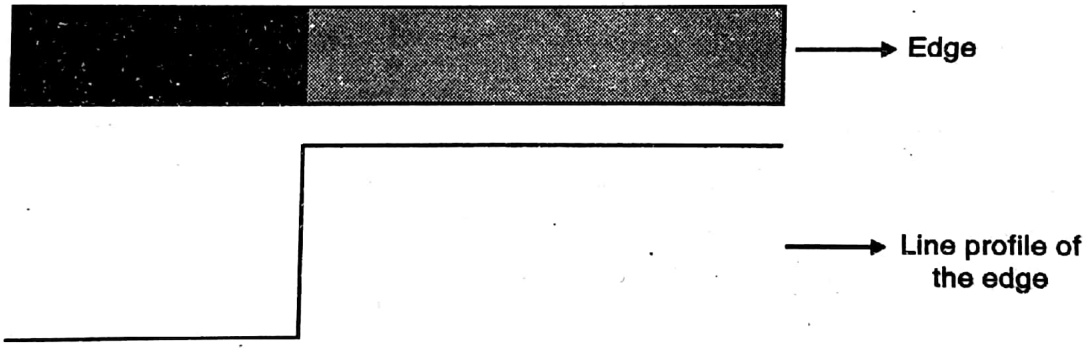


Fig. Q. 1(b)

More than isolated points and lines, it is the detection of edges that form an important part of image segmentation. An edge can be defined as a set of connected pixels that form a boundary between two disjoint regions. A typical example of an edge is shown in Fig. Q. 1(b).

Here the set of pixels that separate the black region from the grey region is called an *edge*. The line profile of such an edge is shown along with edge.

Q. 6(a) Write detail notes on : Edge Linking using Hough transform. **(10 Marks)**

Ans. :

Edge linking :

It was seen that gradient operators like Roberts, Sobel, Prewitts as well as the Compass operators enhance the edges. When we implement these filters practically, there are usually breaks in lines. Due to noise, there are spurious intensity discontinuties because of which the output of a gradient filtered image would have pixels that lie on the edge as well as pixels that represent noise.

Due to this fact, edge detection algorithms are generally followed by edge linking procedures which form edges (lines) from edge pixels. This is done by joining the edge pixels.

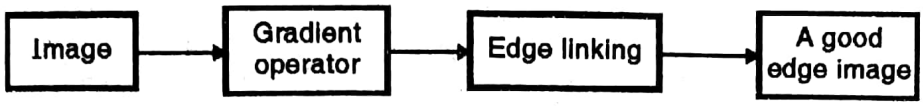


Fig. 1-Q. 6(a)

Hough transform :

Hough transform has emerged over the past decade as a method of choice for pixel linking and curve detection. Hough presented this transform in the year 1962.

Given a set of discrete pixels, the Hough transform checks if these points lie on a straight line and if yes, it draws a line joining all these points.

Hough transform is best explained by an example. Consider a point (x_1, y_1). A line passing through this point can be written in the slope-intercept form as $y_1 = ax_1 + b$.

Using this equation and varying the values of a and b, infinite number of lines pass through this point (x_1, y_1).

However, if we write this equation as

$$b = -ax_1 + y_1$$

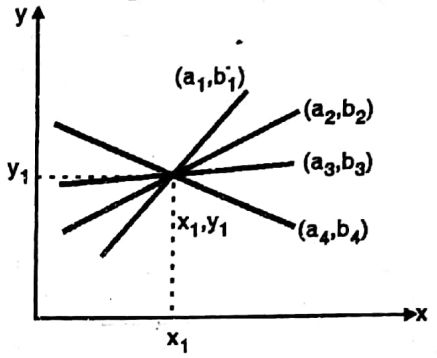


Fig. 2-Q. 6(a)

and consider the ab plane instead of the xy plane, we get a single line for a point (x_1, y_1)

This entire line in the ab plane is due to a single point in the xy plane and different values of a and b.

Now consider another point (x_2, y_2) in the xy plane.

The slope intercept equation of this line is

$$y_2 = ax_2 + b$$

Writing this equation in terms of the ab plane we get

$$b = -ax_2 + y_2$$

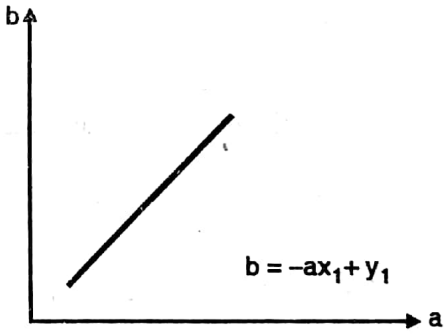


Fig. 3-Q. 6(a)

This is another line in the ab plane. These two line will intersect each other somewhere in the ab plane only if they are a part of a straight line in the xy plane.

The point of intersection in the ab plane is noted (a′, b′).

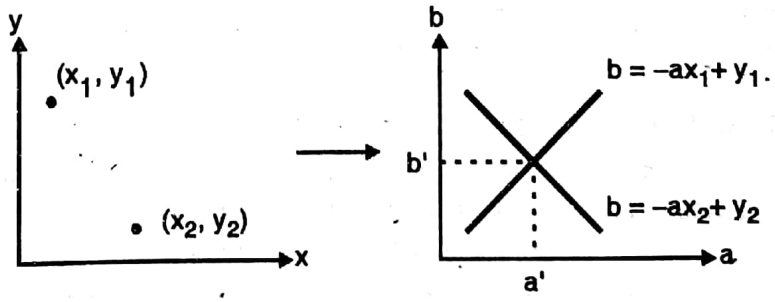


Fig. 4-Q. 6(a)

Using this (a′, b′) in the standard slope-intercept form i.e., $y = a'x + b'$, we get a line that passes through points (x_1, y_1) and (x_2, y_2) in the xy plane!!!!

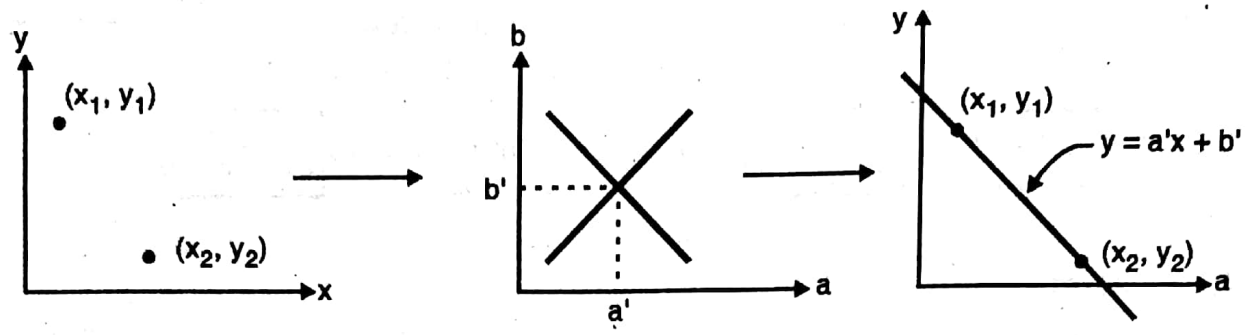


Fig. 5-Q. 6(a)

In fact if we have n points that lie on a straight line, we get n lines (each line representing a point) in the ab plane and all these lines would intersect at a single point (a′, b′) in the ab plane. Using these values of a′ and b′ in equation y = a′x + b′, we would get a line that would pass through all these points in the xy plane.

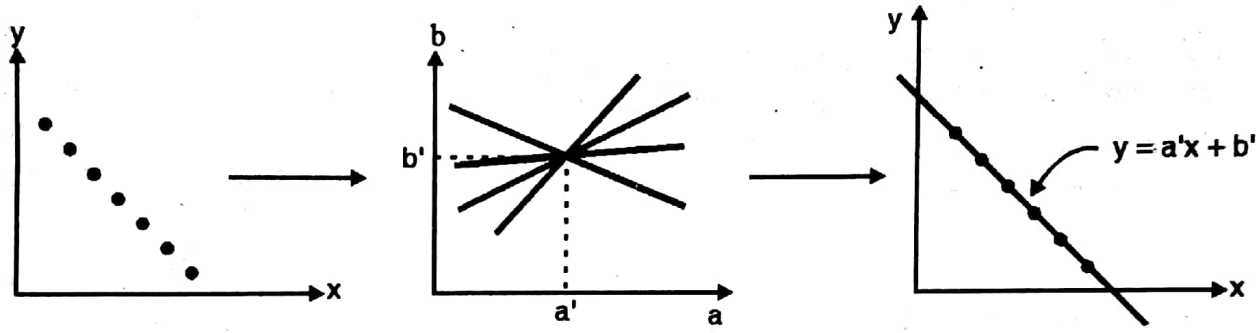


Fig. 6-Q. 6(a)

The ab plane is called the parameter space.

This parameter space is subdivided into accumulator cells as shown.

Here (a_{max}, a_{min}) and (b_{max}, b_{min}) are the expected ranges of slope and intercept values.

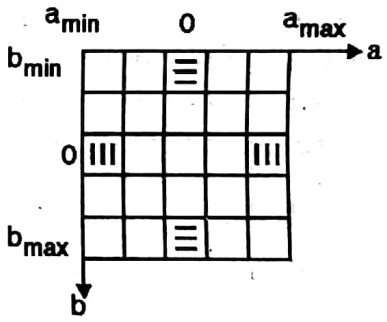


Fig. 7-Q. 6(a)

Q. 6(d) Write detail notes on : Segmentation techniques. Region growing and split and merge. **(10 Marks)**

Ans. : Segmentation :

It was stated that segmentation could be carried out in two separate ways, one based on discontinuities and the other based on similarities. Region based segmentation is a technique in which segmentation is carried out based on the similarities in the given image. The region based approach to segmentation seeks to create regions directly by grouping together pixels which share common features into areas or regions of uniformity. The regions that are formed using the Region based segmentation have the following properties.

Let R represent the entire image.

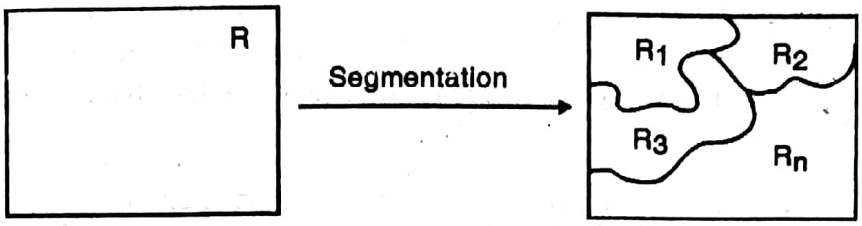


Fig. 1-Q. 6(d)

We can view segmentation as per process that partitions R into sub regions R_1, R_2, ..., R_n such that,

(1) $\bigcup_{i=0}^{n} R_i = R$

(2) R_i is a connected region, i = 1, 2, ...n

(3) $R_i \cap R_j = \phi$ for all i and j; $i \neq j$

(4) $P(R_i)$ = True for i = 1, 2, ...n

(5) $P(R_i \cup R_j)$ = False for $i \neq j$

Where $P(R_i)$ is the logical predicate over the points in set R and ϕ is the null set.

By predicate we mean some condition that is set.

The five conditions stated can be interpreted as follows :

(1) The union (sum) of all the regions equals the whole image. In other words all pixels in the image must be assigned to a region.

(2) Each region is contiguous and connected, (it could be either 4-connectivity or 8-connectivity or m-connectivity).

(3) The interaction of any pair of adjacent regions equals the empty set. i.e., each pixel belongs to a single region only and hence there is no overlap between regions.

(4) For each region, the uniformity predicate is true. Each region must satisfy some particular uniformity condition.

(5) For any two adjacent regions, the uniformity predicate is false. It simply means two adjacent regions do not have anything in common.

Region based segmentation can be carried out in four different ways.

(1) Region growing
(2) Region splitting
(3) Region merging
(4) Split and merge

Region growing :

As the name implies, region growing is a procedure in which pixels are grouped into larger regions based on some predefined condition. The basic approach is to select a seed point (a pixel from where we begin) and grow regions from this seed pixel.

Let us pick up an arbitrary pixel (x_1, y_1) from the image that needs to be segmented. This pixel is called the seed pixel. We now examine the nearest neighbours (4 or 8 neighbours depending on whether we assume 4-connectivity or 8-connectivity) of (x_1, y_1) one by one. The neighbouring pixel is accepted in the same region as (x_1, y_1) if they together satisfy the homogeneity property of a region i.e., if both of them satisfy the predefined condition.

Once a new pixel (x_2, y_2) is accepted as a member of the current region, the neighbours of this new pixel are examined (again, 4 or 8 neighbours, depending on the connectivity assumed).

This procedure goes on recursively until no new pixel is accepted. All the pixels of the current region are given a unique label. Now a new seed pixel is chosen and the same procedure is repeated. We continue doing this until all the pixels are assigned to some region or the other.

Split and merge :

As the name suggest, both split and merge are used. Region splitting starts with the whole image as one big region and splits this region into four quadrants. We continue splitting until all the sub-regions satisfy the predefined homogeneity property (predicate). In region merging each pixel is taken as a small region. We merge small regions into larger regions if they satisfy the homogeneity property. So when do we use region splitting and region merging ?

If homogeneous regions are small, the region merging technique is superior to the region growing technique. If most of the regions in an image are homogeneous, the region splitting technique is preferred to the region merging method. In most applications, a combination of split and merge techniques are used. Together they are called the split and merge technique.

In the split and merge technique we start with a rectangular regions of size a × b pixels. We check the homogeneity property for each region. If the homogeneity property fails, we split up the region into four quadrants each of size a/2 × b/2. If the region satisfies the homogeneity property, we merge it with the adjacent region forming a 2a × 2b size region.

Chapter 8 : Image Morphology, Representation and Description
[Total Marks : 20]

Q. 3(a) Explain the first difference make the chain code invariant to rotation. **(10 Marks)**

Ans. : Consider a simple boundary.

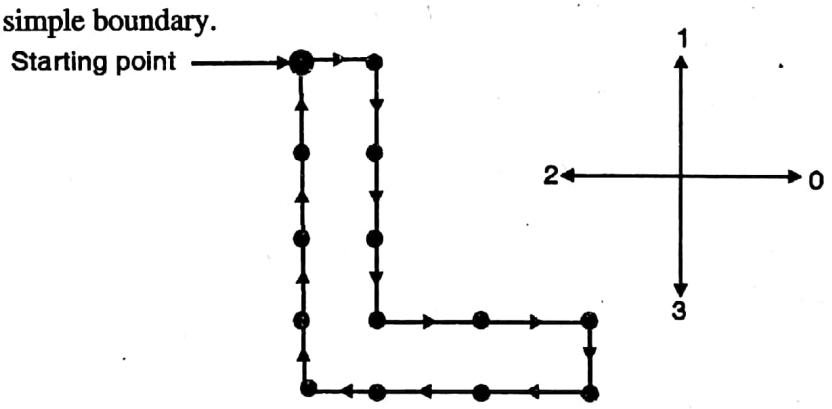


Fig. 1-Q. 3(a)

The 4-directional chain code (clockwise direction) is

0 3 3 3 0 0 3 2 2 2 1 1 1 1

Its first difference is

3 0 0 1 0 3 3 0 0 3 0 0 0 ...(1)

We shall now rotate the boundary

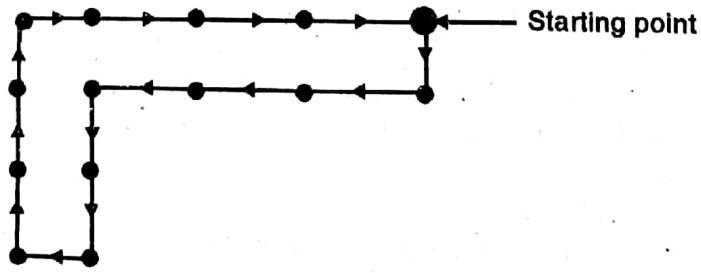


Fig. 1(a)-Q. 3(a)

The 4-directional chain code is

3 2 2 2 3 3 2 1 1 1 0 0 0 0

It's first difference is

3 0 0 1 0 3 3 0 0 0 0 0 ...(2)

We see that 1 = 2

We can thus conclude that the first difference of the original object and the first difference of the object after rotation remain the same.

Hence the first difference of the chain code makes it invariant to rotation.

Q. 6(b) Write detail notes on : Thinning with example. **(10 Marks)**

Ans. :

Thinning is derived from the Hit-or Miss transform. For a image *A* and a Composite structuring element $B = (B_1, B_2)$, Thinning can be defined as

$$A \oslash B = A - (A \otimes B)$$

Where "–" is the set difference. This equation can also be written as

$$A \oslash B = A \cap (A \otimes B)^c \quad ...(1)$$

In thinning, a part of the boundary of the object is subtracted from the object.

The concept of thinning of an object can be explained using an example.

Consider image A,

$A =$

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

Let the structuring element be,

$B_1 =$

0	0	0
0	①	0
0	1	0

$B_2 =$

0	1	
0	⓪	0
0	0	0

$A \otimes B =$

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

$$A \oslash B = A - (A \otimes B) =$$

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

We observe that the top row of five pixels is removed producing a 4×5 rectangle. Repeating the thinning operation using a 90° rotated version of the structuring elements (B_1 and B_2) removes the left hand vertical column of pixels yielding a 4×4 square. This procedure can be repeated till the required thinning is achieved.

A more useful expression for thinning image *A*, symmetrically, is based on a sequence of structuring elements. Thinning transformation is very often used sequentially.

Let $B = \{B_1, B_2, ...B_n\}$ denote a sequence of composite structuring elements. Here B_n is the rotated version of B_{n-1}. Sequential Thinning can then be expressed as

$$A \oslash \{B\} = (((A \oslash B_1) \oslash B_2) \oslash B_3) \oslash .....B_n)$$

Given below is a set of 8 structuring elements that are commonly used for thinning.

$B_1 =$

0	0	0
x	1	x
1	1	1

$B_2 =$

x	0	0
1	1	0
1	1	x

$B_3 =$

1	x	0
1	1	0
1	x	0

$B_4 =$

1	1	x
1	1	0
x	0	0

$B_5 =$

1	1	1
x	1	x
0	0	0

$B_6 =$

x	1	1
0	1	1
0	0	x

$B_7 =$

0	x	1
0	1	1
0	x	1

$B_8 =$

0	0	x
0	1	1
x	1	1

The process is to thin image A by B_1 then thin the result with B_2 and so on until A is thinned with B_n. The entire procedure is repeated until no further change occurs.

Chapter 9 : Image Transforms [Total Marks : 30]

Q. 1(c) What is an Unitary matrix ? **(5 Marks)**

Ans. : A matrix is said to be unitary is it complex conjugate is the same as its inverse. Hence a matrix T is unitary if.

$$T^{*'} = T^{-1}$$

Multiplying both sides by T, we get,

$$T \cdot T^{*'} = T \cdot T^{-1} \quad ...(1)$$

$\therefore T \cdot T^{*\prime} = I$ ...(2)

Where * indicates complex conjugate of T.

Hence to verify whether T is a unitary matrix we need to use the relationship $TT^{*\prime} = I$.

We shall now check if the DFT matrix is unitary or not.

We know that the Fourier transform can be represented as F = Wf where f is the input and W is the DFT matrix. Taking N = 4, we form a DFT matrix.

$$W = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

To check whether W is unitary or not, we need to check the relationship.

$W \cdot W^{*\prime} = I.$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} = \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}$$

W $W^{*\prime} = 4[I]$

Hence the DFT is an example of a unitary transform. 4 indicates that the DFT matrix is not normalised.

Q. 1(d) Define morphological operations Erosion and Dilation. **(5 Marks)**

Ans. :

(i) **Dilation :** Let A(x, y) be a binary image and C(x, y) be the resulting image obtained after A(x, y) has been dilated with a $m \times n$ template B(i, j). B is called the structuring element where $0 \le i \le m - 1, 0 \le j \le n - 1$

B is usually a binary structure. For dilation

$$C(x, y) = \text{Maximum } \{A(x - i, y - j) \times B(i, j)\} \quad \ldots(2)$$

This simply means that if B is a binary structuring element, the output pixel is the maximum value of all the pixels in the input pixels neighbourhood multiplied by the corresponding coefficients of the structuring element. Since the structuring element is usually a binary mask, for a binary image, if any of the pixels has a value 1, the output pixel is set to 1.

Fig. 1-Q. 1(d) illustrates the dilation of a binary image. The structuring element that we use is

Consider the image shown.

1	1	1

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

Dilation is achieved as shown.

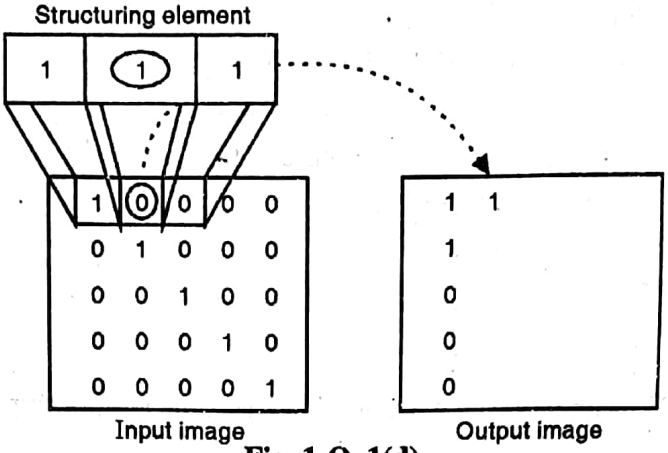


Fig. 1-Q. 1(d)

(ii) Erosion : In a similar fashion, we can achieve erosion by using a simple formula which could be used in programming.

If A(x, y) is the image and B(i, j) is the structuring element, then C(x, y) which is the eroded image is given by

$$C(x, y) = \text{Minimum } \{A(x - i, y - j) \times B(i, j)\} \quad \ldots(1)$$

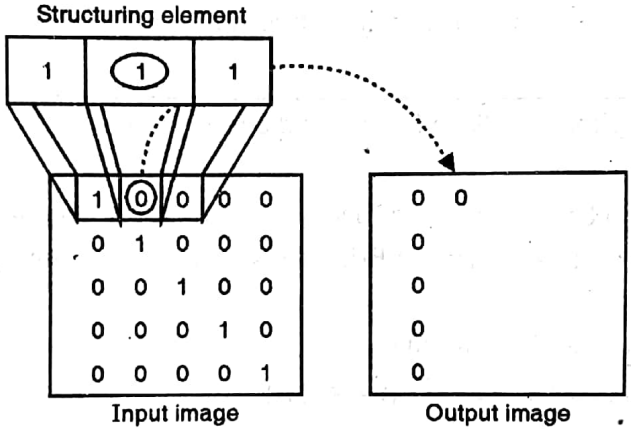


Fig. 2-Q. 1(d)

The Fig. 2-Q. 1(d) shows the erosion operation. These formulae can be used only if all the coefficients of the structuring element are equal to 1. Dilation and Erosion can also be achieved on grey level images. Fig. 3-Q. 1(d) and Fig. 4-Q. 1(d) explain dilation and erosion on grey level images.

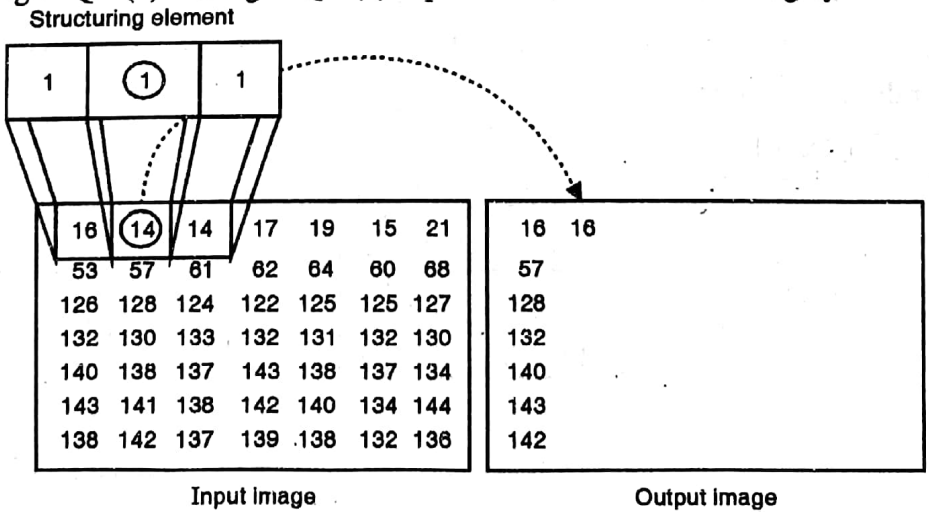


Fig. 3-Q. 1(d) : Dilation

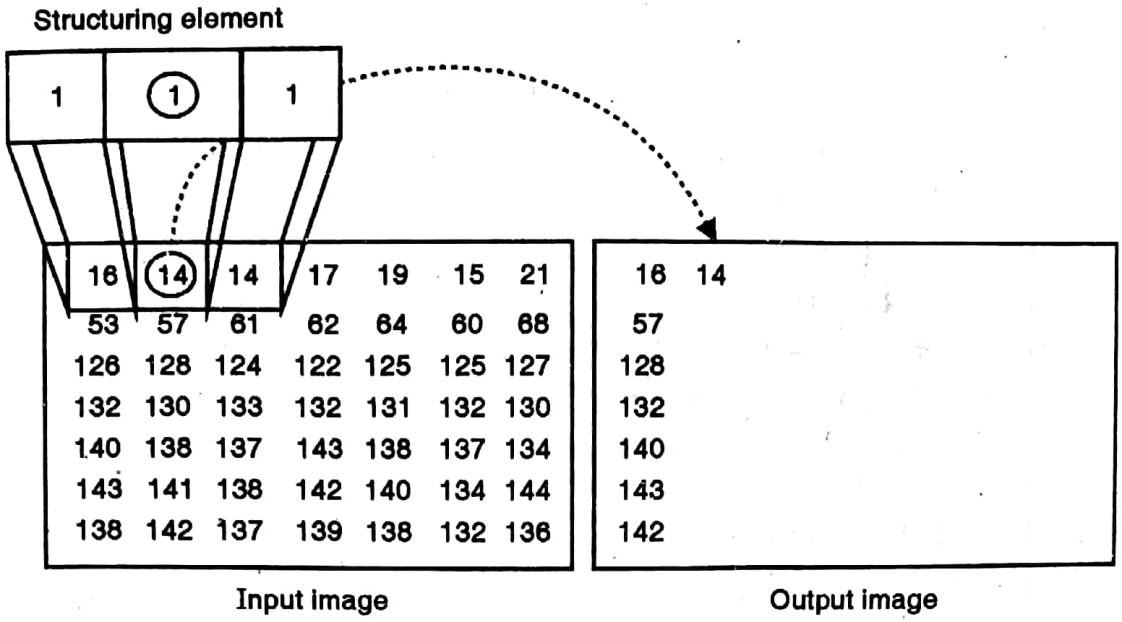


Fig. 4-Q. 1(d) : Erosion

Q. 4(a) Write 8 × 8 Hadamard transform matrix and its signal flow graph for fast hadamard transform. Using this butterfly diagram. (Signal flow graph) compute Hadamard transform for x (n) = {1, 2, 1, 1, 3, 2, 1, 2}. **(10 Marks)**

Ans. :

Since N = 8, we use 8 point Butterfly diagram.

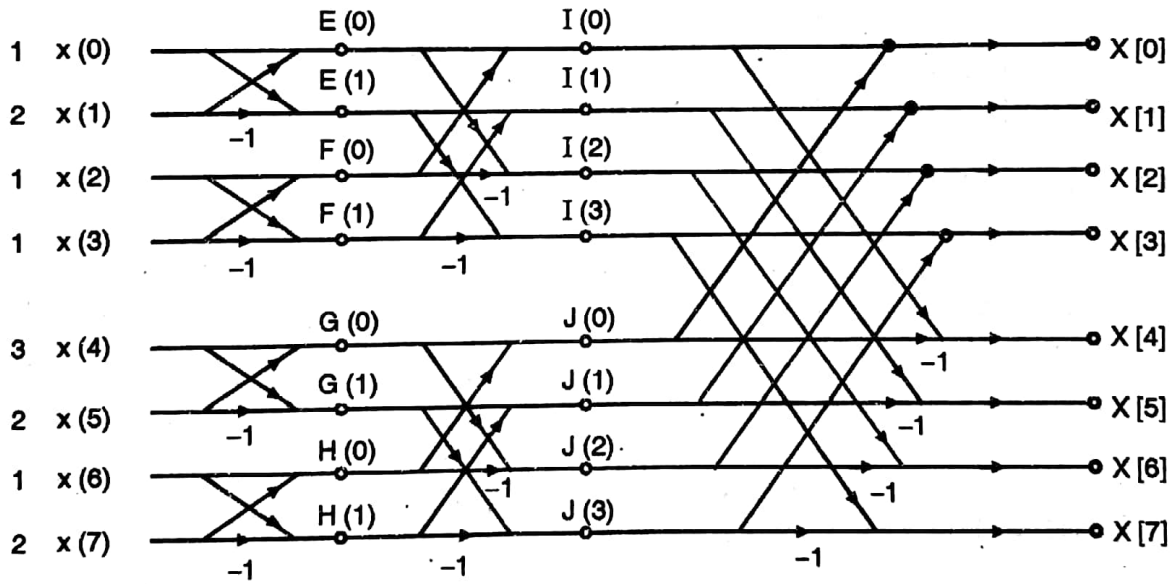


Fig. 1-Q. 4(a)

$$E(0) = x(0) + x(1) = 1 + 2 = 3$$
$$E(1) = x(0) - x(1) = 1 - 2 = -1$$
$$F(0) = x(2) + x(3) = 1 + 1 = 2$$
$$F(1) = x(2) - x(3) = 1 - 1 = 0$$
$$G(0) = x(4) + x(5) = 3 + 2 = 5$$
$$G(1) = x(4) - x(5) = 3 - 2 = 1$$

$$
\begin{aligned}
H(0) &= x(6) + x(7) = 1 + 2 = 3 \\
H(1) &= x(6) - x(7) = 1 - 2 = -1 \\
I(0) &= E(0) + F(0) = 3 + 2 = 5 \\
I(1) &= E(1) + F(1) = -1 + 0 = -1 \\
I(2) &= E(0) - F(0) = 3 - 2 = 1 \\
I(3) &= E(1) - F(1) = -1 - 0 = -1 \\
J(0) &= G(0) + H(0) = 5 + 3 = 8 \\
J(1) &= G(1) + H(1) = 1 - 1 = 0 \\
J(2) &= G(0) - H(0) = 5 - 3 = 2 \\
J(3) &= G(1) - H(1) = 1 - (-1) = 2 \\
X[0] &= [I(0) + J(0)] = (5 + 8) = 13 \\
X[1] &= [I(1) + J(1)] = (-1 + 0) = -1 \\
X[2] &= [I(2) + J(2)] = (1 + 2) = 3 \\
X[3] &= [I(3) + J(3)] = (-1 + 2) = 1 \\
X[4] &= [I(0) - J(0)] = (5 - 8) = -3 \\
X[5] &= [I(1) - J(1)] = (-1 - 0) = -1 \\
X[6] &= [I(2) - J(2)] = (1 - 2) = -1 \\
X[7] &= [I(3) - J(3)] = (-1 - 2) = -3 \\
\therefore\ X[n] &= \{13, -1, 3, 1, -3, -1, -1, -3\}
\end{aligned}
$$

Q. 4(b) Find the DCT of the given image using matrix multiplication method. **(10 Marks)**

$$
f(x, y) = \begin{bmatrix} 2 & 4 & 4 & 2 \\ 4 & 6 & 8 & 3 \\ 2 & 8 & 10 & 4 \\ 3 & 8 & 6 & 2 \end{bmatrix}
$$

Ans. :

The size of the image is 4×4

We use the matrix notation

$$F = C \cdot f \cdot C'$$

where c is a 4×4 DCT matrix

$$
C = \begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ 0.653 & 0.2705 & -0.2705 & -0.653 \\ 0.5 & -0.5 & -0.5 & 0.5 \\ 0.2705 & -0.653 & 0.653 & -0.2705 \end{bmatrix}
$$

$$
\therefore F = \begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ 0.653 & 0.2705 & -0.2705 & -0.653 \\ 0.5 & -0.5 & -0.5 & 0.5 \\ 0.2705 & -0.653 & 0.653 & -0.2705 \end{bmatrix} \begin{bmatrix} 2 & 4 & 4 & 2 \\ 4 & 6 & 8 & 3 \\ 2 & 8 & 10 & 4 \\ 3 & 8 & 6 & 2 \end{bmatrix}
$$

Solving this, we obtain the DCT of the given image.

$$F = \begin{bmatrix} 19 & -0.27 & -8.0 & 0.65 \\ -2.69 & -0.25 & 2.30 & 0.89 \\ -3.5 & 1.46 & 1.5 & -1.68 \\ 0.03 & -1.60 & -0.95 & -0.25 \end{bmatrix}$$

Chapter 11 : Image Compression [Total Marks : 20]

Q. 5(a) Discuss the different type of redundancies in images with examples. **(10 Marks)**

Ans. : Images in their raw from occupy a lot of memory space and hence data compression becomes necessity while dealing with them. Data compression basically tries to reduce the overall size of data. This reduction is possible when the original dataset contains some type of unwanted data redundancy. Digital images contain a lot of data which provide no relevant information, also called redundant data which can be exploited to reduce the size of the image.

In general, three basic redundancies exist in digital images that follow.

1. **Inter-pixel redundancy :** It is a redundancy corresponding to statistical dependencies or high correlation among neighbouring pixels. Hence in images, there exists large area having the same grey level. This is known as inter-pixel redundancy and can be reduced using Run Length encoding.
2. **Coding Redundancy :** In an uncompressed image, every pixel is coded with a fixed number of bits known as fixed length coding. For example, an image with 256 gray scales is represented by an array of 8-bit integers. However this type of coding does not take into account the fact that some grey level occur more often than the others. This is known as coding redundancy. Huffman coding and arithmetic coding are used to reduce coding redundancy.
3. **Psycho-visual Redundancy :** Even though an image could have 256 grey levels, the Human eye can resolve very few grey levels locally. Hence Psycho-visual redundancy deals with the way humans perceive grey levels. Psycho-visual redundancy can be reduced by using IGS coding.

Run Length Encoding (RLE) :

It is the simplest dictionary-based data compression technique. Image files frequently contain the same character repeated many times in a row. Images, particularly those having very few grey levels, often contain regions of adjacent pixels, all with the same grey level. Each row of such images can have long runs of the same grey value. In, such cases, one can store a code specifying the value of the grey level, followed by the length of the run, rather than storing the same value many times over.

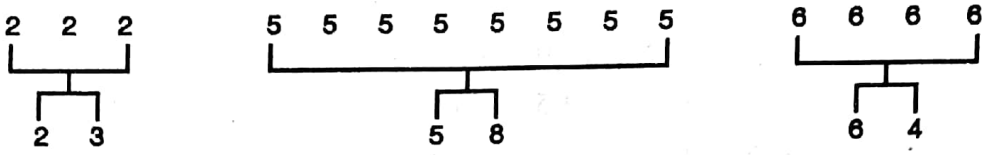
Let us take an example.

Consider the first row of an image which has the following grey values.

2	2	2	5	5	5	5	5	5	5	5	6	6	6	6
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•

The first row of the image has 15 grey values.

We could store the above row in a compact manner using RLE.



The first code specifies the grey value, followed by the length of the run.

Hence the RLE of the first row is simply.

2 3 5 8 6 4

As is evident, run length encoding achieves considerable compaction in images which have a fairly constant background. The RLE eliminates *Inter-pixel redundancies.*

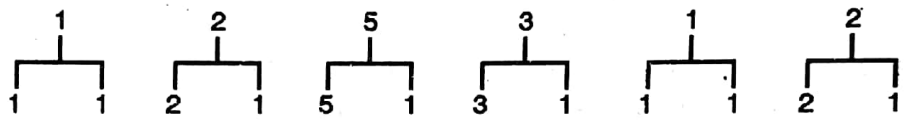
The run length encoding also has a serious problem. At times, the RLE doubles the size of the file.

Consider the following example.

Perform RLE on the following data.

1 2 5 3 1 2

In RLE, the first value specifies the grey value while the second value specifies the run. as is evident, run



∴ The RLE code is

1 1 2 1 5 1 3 1 1 1 2 1

The RLE code in this case is double that of the original sequence. Hence RLE should only be used if we have the same character grey value repeated many times in a row.

Statistical coding :

The images dealt with so far have been coded and saved in the natural code. That is for a 256 grey level image, grey level at f(x, y) is coded with its 8-bits binary equivalent. A grey level of a value 4 is coded as 0000 0100. Similarly for a 8 grey level image, grey value at f (x, y) is coded using 3-bits. The table that shows codes for a sampled image having grey levels varying from 0 to 7 (3-bit image) is given.

These codes are referred to as Equal Length Codes.

Input	Natural code
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Equal length codes do not make use of the statistics of data. The assumption is made that all grey levels have equal occurrence in the image. Since this possibility is unlikely, this is not an optimum for o

coding. If we can develop a code such that fewer bits are assigned to grey levels having higher probability of occurrence and vice versa, we could reduce the number of bits required for transmission. Such a coding is known as Variable Length Coding or Entropy Coding and it eliminates *Coding redundancies.*

Here, we are not discarding any information as would be done in lossy compression, but simply represent more frequently occurring values with shorter codes and vice versa.

Consider an image containing L grey levels i = 0, 1 ..., L – 1. Let n_i denote the number of pixels having grey level i.

The probability of occurrence of grey level i in the image is

$$p_i = \frac{n_i}{\sum_{i=0}^{L-1} n_i} \quad \ldots(1)$$

Let us define entropy H associated with the grey level as

$$H = -\sum_{i=0}^{L-1} p_i \log_2 p_i \quad \ldots(2)$$

Entropy is the measure of randomness in the set of random variables. Entropy is never negative since p_i lies in the range [0,1].

Details of the above equation can be found in any book on Information theory. When all variables are equally likely i.e., $p_0 = p_1 = p_2 ..... = p_{L-1} = 1/L$, then the randomness is maximized and the entropy attains its greatest value.

Improved Grey Scale (IGS) Quantization :

Brightness of a region as perceived by the eye does not depend on the absolute grey level values. This is because the human eye does not respond with equal sensitivity to all visual information. Some information is visually more important than the others.

Information which is not visually important is called *Psychovisual Redundancy*. Psychovisual redundancies exist in all images and can be eliminated without hampering the subjective quality of the image. We have seen that simply reducing the quantization (number of bits for representation), compresses the image but also produces false contouring. I.G.S. coding reduces the quantization but also reduces false contouring.

Steps involved in I.G.S. coding are,

(1) Lower 4-bits of the preceding modified pixel are added to the present pixel.
(2) The new 4 MSB's of the present pixel are taken as the I.G.S. code.
(3) Repeat step 1 and 2 after moving on to a new pixel.
(4) If MSB is 1111, then add 0000 instead of the 4 LSB's of the previous sum

These steps will be clear with a simple example.

Q. 6(c) Write detail notes on : Differential Pulse Code Modulation (DPCM). **(10 Marks)**

Ans. : Differential Pulse Code Modulation (DPCM) :

A strong correlation exists between pixels spatially located in proximity in any given image. Predictive coding exploits this correlation by trying to predict the pixel to be encoded. This prediction is made on the basis of previously transmitted pixel values in the encoded form. Once this is done, only the prediction error is transmitted. Predictive coding is also called Differential Pulse-Code Modulation (DPCM).

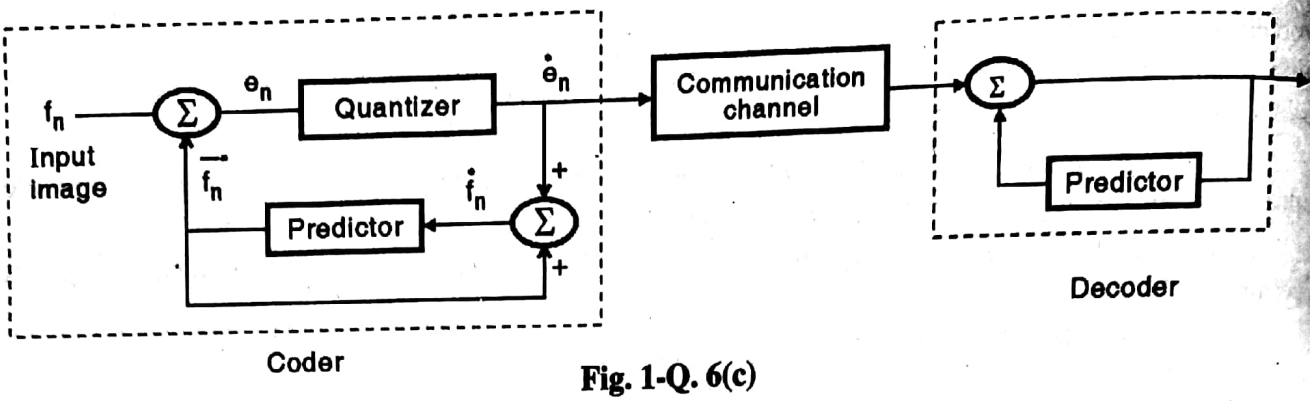


Fig. 1-Q. 6(c)

As started earlier, the philosophy underlying predictive coding is to remove mutual redundancies between successive pixels and encode only the new information. Consider the image f_n and let $\dot{f}_n$ be the values of the reproduced image.

When f_n arrives, a quantity $\overline{\dot{f}_n}$, an estimate of f_n is predicted from the previously decoded samples f_{n-1}, f_{n-2} .... i.e.,

$$\overline{\dot{f}_n} = \psi \{ \dot{f}_{n-1}, \dot{f}_{n-2} \ldots\}$$

Where ψ is the prediction rule which is set right in the beginning.

From the diagram

$$e_n \underset{=}{\Delta} f_n - \overline{\dot{f}_n} \quad \ldots(1)$$

Let $\dot{e}_n$ be the quantized value of e_n. The reproduced value of f_n is now taken as

$$\dot{f}_n = \overline{\dot{f}_n} + \dot{e}_n \quad \ldots(2)$$

The coding process continues in a recursive manner. This is what Differential Pulse Code Modulation (DPCM) is all about.

In this, the coder has to calculate the reproduced sequence $\dot{f}_n$. The decoder is simply the predictor loop from the coder.

Rewriting Equation (2) as,

$$f_n = \overline{\dot{f}_n} + e_n \quad \ldots(3)$$

Subtracting Equation (2) from Equation (3) we get,

$$\delta f_n = f_n - \dot{f}_n = e_n - \dot{e}_n = q_n$$

This simply means that the error in the input sequence is equal to the quantized error in e_n i.e. q_n. An example at this stage will make things clearer.

Chapter 12 : Colour Image Processing [Total Marks : 10]

Q. 2(a) Discuss color models for a digital image. **(10 Marks)**

Ans. : Colour models :

Colour model is also known as colour space. Colour models are different ways in which colour information is stored. The most common and obvious colour model in the RGB colour model. The

reason for having more than one colour model is that some operations are easier to implement if we move away from the RGB model.

1. RGB colour model :

As stated earlier, this is the most common format used in image processing. The RGB colour space is shown in Fig. 1-Q. 2(a). Each point in the image (pixel) is represented by a point in the first quadrant of the three-space as shown.

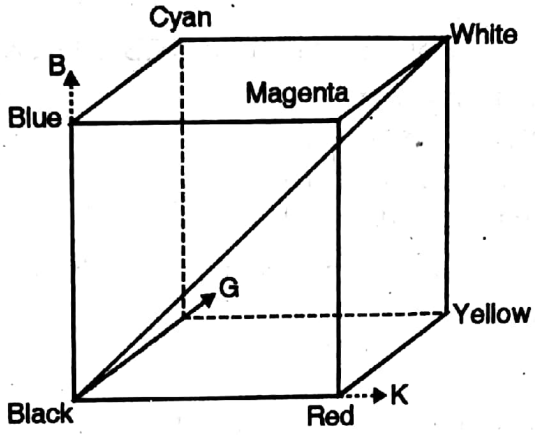


Fig. 1-Q. 2(a)

The origin of the RGB space (Red = 0, Green = 0, Blue = 0) represents black while the opposite corner (Red = max, Green = max, Blue = max) represents white.

A typical way of representing a RGB colour image is M × N × 3 where M × N is the physical size of the image and 3 represents the R, G, B planes i.e. an RGB image can be visualised as stack of 3 planes of size M × N each.

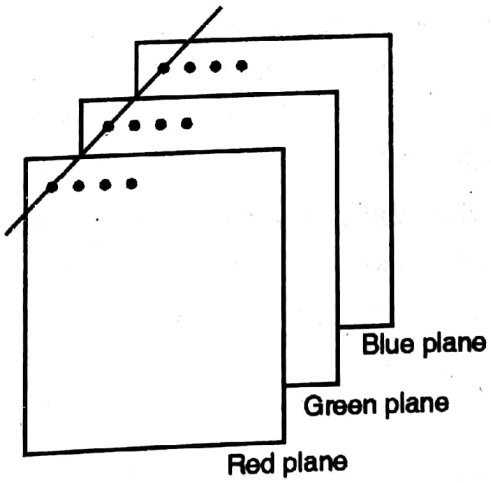


Fig. 2-Q. 2(a)

2. NTSC Colour model :

The NTSC colour model is used in television in the United States. That is the reason why their VCD's do not normally work here, as we use the PAL format.

Similar to the RGB model, the NTSC model consists of three components viz; Luminance (Y), Hue (I) and Saturation (Q). In this model, the Luminance component represents the grey scale information while the Hue and the Saturation carry the colour information.

Computing the YIQ components from the RGB model is a simple task, Given below is the Transformation.

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.211 & -0.523 & 0.312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad \ldots(1)$$

It is important to note that the sum of the first row is equal to 1, while the sum of the remaining two rows is 0.

In a similar fashion, it is a simple task to get the RGB components from the YIQ components. The transformation that achieves this is given below :

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0.956 & 0.621 \\ 1 & -0.272 & -0.647 \\ 1 & -0.106 & 1.703 \end{bmatrix} \begin{bmatrix} Y \\ I \\ Q \end{bmatrix} \quad \ldots(2)$$

Matlab has an inbuilt function for the transformation kindly go through the *rgb2ntsc* and *ntsc2rgb* commands in Matlab.

3. YCbCr colour model :

This is another model which is common. It is widely used in digital video. Here Y stands for illumination and is represented by a single component. Colour information is stored as two-colour difference components, Cb and Cr.

Here Cb is the difference between the blue component and a reference value while Cr is the difference component between the red component and a reference value.

4. CMY and CMYK Models :

Here C stands for cyan, M for magenta and Y for yellow. Cyan, Magenta and Yellow are secondary colours of light. While television, cameras, scanners, computer monitors work an the RGB colour model, which is a additive colour system, offset printing, digital printing, paint, photographic prints to are based on the CMY or CMYK system which is the subtractive system of colour.

C, M and Y mix to form Black (k).

When a surface coated with cyan pigment is illuminated with white light, no red light is in reflected from the surface, in other words cyan pigment subtracts red light from the reflected white light.

Transformation from RGB to CMY is a trivial task.

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad \ldots(3)$$

Here R, G and B values are assumed to be normalised. (for normalisation, divide the R, G, B planes by 256).

5. HSI colour model :

HSI is an important colour model and it's design reflects the way humans see colour. For example, one does not refer to the colour of a dress by giving the percentage of the red, green and blue components!!

When we see colour, we describe it in terms of its Hue, Saturation and Intensity (Brightness).

In the HSI model, I stands for intensity and for our purpose it is simply the average of the R, G and B components. The I component specifies the brightness irrespective of the colour H stands for Hue. Hue is an attribute that describes pure colour. Hue is what the artist refers to as "pigment" it is what we think as colour. Yellow, orange, cyan etc. are examples of Hue. S stands for Saturation. This gives us the measure of the degree to which pure colour is diluted by white light.

❑❑❑

May 2016

Chapter 2 : Image Sensing and Acquisition [Total Marks : 08]

Q. 2(a) Explain in detail any two types of image file formats. **(8 Marks)**

Ans. : Image file formats :

Images obtained from the camera are stored on the host computer using different formats. A file format is a structure which defines how information is stored in the file and how that information is displayed on the monitor. There are numerous image file formats which are available. Of these only a few of them can be used universally. By universally it means, they can be understood by different operating systems.

Some of the image formats that can be used on either Macintosh or PC platforms are;

1. BMP (Bit Mapped Graphic Image)
2. JPEG (Joint Photographic Expert Group)
3. TIFF (Tagged Image File Format)
4. EPS Encapsulated Post Script)
5. GIF (Graphic Interchange Format)
6. PICT (Macintosh Supported)

TIFF, which is one of the most well known formats was developed in 1986 and in its many versions is a standard image format for a bit-mapped graphics image. The TIFF format is also a data compression technique for monochrome as well as colour images. Images seen on the Internet sites are normally TIFF/GIF images simply because they occupy less space. GIF uses a form of Huffman coding.

BMP images developed by Microsoft can save both monochrome as well as colour images. All the wall papers that your computer has are BMP images. Similarly when we work in Paint Brush, we can save our work as a BMP image. The quality of BMP files is very good but they occupy a lot of memory space. PICT is a general purpose format supported by Macintosh and used for storing bit-mapped images. EPS is file format of the post script page description language and is device independent. This simply means that images can readily be transferred from one application to another. However EPS images can only be printed on post script compatible printers.

JPEG (Joint Photographic Expert Group) is the name of the committee that developed an image format which used compression algorithms.

JPEG images are compressed images which occupy very little space. It is based on the Discrete Cosine Transform-DCT (explained in chapter of Image Transforms). JPEG images use lossy compression algorithms which result in some loss of original data and hence the quality of JPEG images is not as good as BMP images.

Although these formats differ in technical details, they share structural similarities.

The Fig. 1-Q. 2(a) shows the typical organisation of information encoded in an image file.

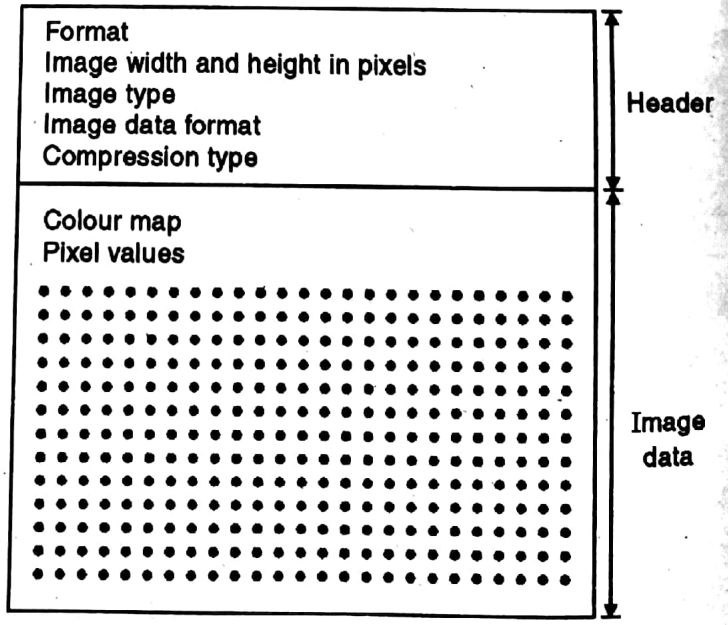


Fig. 1-Q. 2(a)

The image file consists of two parts;

(1) Header (2) Image data

The header file gives us the information about the kind of image. The header file, begins with a binary code or ASCII string which identifies the format being used. The width and height of the image are given in number of pixels. Common image types include binary images, 8-bit grey scale images, 8-bit colour and 24-bit colour images. The image data format specifies the order in which pixel values are stored in the image data section. A commonly used order is left to right and top to bottom. Image data format also specifies if the RGB values in the image are interlaced. By interlaced we mean that the RGB values of each pixel stay together consecutively followed by the three colour components for the next pixel i.e.,

$$\underbrace{R\ G\ B}_{1^{st}\text{ Pixel}}\quad \underbrace{R\ G\ B}_{2^{nd}\text{ Pixel}}\quad \underbrace{R\ G\ B}\ldots\ldots$$

If the RGB values are not interlaced, the values of one primary colour for all pixels appear first, then the values of the next primary colour followed by the values of the third primary colour. Thus the image data is in the form.

R R R R..... G G G G..... B B B B.....

Chapter 3 : Sampling and Quantization [Total Marks : 05]

Q. 1(b) Explain in short sampling and quantization method for digital image. **(5 Marks)**

Ans. : Sampling :

To understand sampling, it is needed to understand convolution involving impulse functions). If f(t) is a step signal shown in Fig. 1-Q.1(b) and g(t) is a train of impulses (g(t) is also known as a COMB function), then the convolution of the two is given by f(t) * g(t). Let us see what happens when we convolve f(t) with g(t).

Proof : We know that the convolution of f(t) and g(t) is,

$$y(t) = \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau = \int_{-\infty}^{\infty} g(\tau)\, f(t-\tau)\, d\tau$$

$$y(t) = \int_{-\infty}^{\infty} [\delta(t-T) + \delta(t) + \delta(t+T)]\, f(t-\tau) d\tau$$

$$y(t) = \int_{-\infty}^{\infty} \delta(t-T)\, f(t-\tau)\, d\tau + \int_{-\infty}^{\infty} \delta(t)\, f(t-\tau)\, d\tau + \int_{-\infty}^{\infty} \delta(t+T)\, f(t-\tau)\, d\tau$$

But, $\int_{-\infty}^{\infty} \delta(t-T)\, f(t-\tau)\, d\tau = \delta(t-T) * f(t)$ [This is impulse at t = –T convolved with f(t)]

And, $\int_{-\infty}^{\infty} \delta(t+T)\, f(t-\tau)\, d\tau = \delta(t+T) * f(t)$ [This is impulse at t = +T convolved with f(t)]

This gives us the diagrams as shown in Fig. 1.

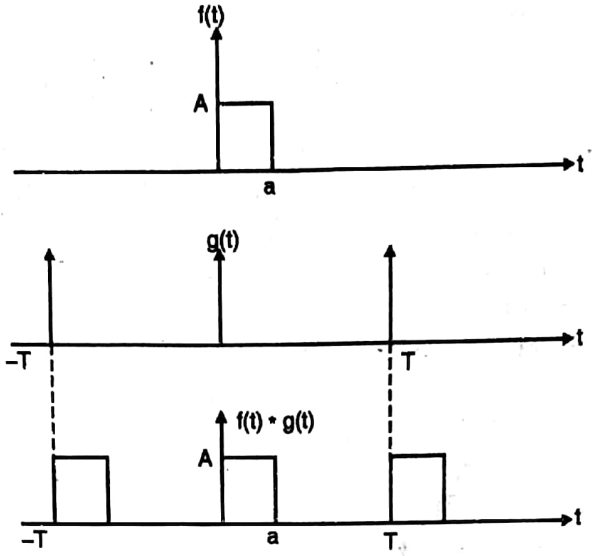


Fig. 1-Q.1(b)

The convolution replicates the signal f(t) at every impulse g(t). Another important property that one needs to know is that convolution in one domain is equal to multiplication in the other i.e. convolution in the time domain is equal to multiplication in the frequency domain and similarly multiplication in the time domain is equal to convolution in the frequency domain.

Quantization :

The values obtained by sampling a continuous function usually comprise of an infinite set of real numbers ranging from a minimum to a maximum depending upon the sensors calibration. These values must be represented by a finite number of bits usually used by a computer to store or process any data. In practice, the sampled signal values are represented by a finite set of integer values. This is known as **quantization**. Rounding of a number is a simple example of quantization. With these concepts of sampling and quantization, we now need to understand what these terms mean when we look at an image on the computer monitor. Higher the spatial resolution of the image, greater is the sampling rate i.e. lower is the image area $\Delta x\ \Delta y$ represented by each sampled point. Similarly, higher the grey level resolution (tonal resolution) more are the number of quantized levels. Hence spatial resolution gives us an indication of the sampling while grey level resolution (tonal resolution) gives us an indication of the quantization.

Spatial resolution $\longrightarrow$ Sampling

Grey level resolution $\longrightarrow$ Quantization

We have already stated that an image can be considered as a 2-D array. Image f(x,y) is arranged in the form of $N \times M$ array

$$f(x, y) = \begin{bmatrix} f(0,0) & f(0,1) & \ldots & f(0, M-1) \\ f(1,0) & f(1,1) & \ldots & f(1, M-1) \\ f(2,0) & f(2,1) & \ldots & f(2, M-1) \\ \vdots & & & \vdots \\ f(N-1,0) & f(N-1,1) & \ldots & f(N-1, M-1) \end{bmatrix}_{N \times M}$$

Hence every image that is seen on the monitor, is actually this matrix. Each element of the matrix is called a *pixel*. Whenever we see an image on the screen of the computer it is actually a matrix which consists of $N \times M$ pixels and each pixel is considered to be a sample. Hence more the pixels, more the samples, higher the sampling rate and hence better the spatial resolution. The value of each pixel is known as the grey level.

The computer understands only ones and zeros. Hence these grey levels need to be represented in terms of zeros and ones. If we have two bits to represent the grey levels, only 4 different grey levels (2^2) can be identified viz. 00, 01 10, 11, where 00 is black, 11 is white and the other two are different shades of grey. Similarly, if we have 8 bits to represent the grey levels, we will have 256 grey levels (2^8). Hence more the bits, more are the grey levels and better is the tonal clarity (quantization). The total size of the image is N × M × m, where m is the number of bits used.

Chapter 4 : Image Enhancement in Spatial Domain
[Total Marks : 12]

Q. 2(b) For the 3 bit 4 × 4 size image perform following operations. **(12 Marks)**

(i) Thresholding T = 3

(ii) Intensity level slicing with background, $r_1 = 3$ and $r_2 = 5$.

(iii) Bit plane slicing for MSB and LSB planes.

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

Ans. :

(i) Thresholding (T = 3) :

Since the given image is 3-bit, $L = 2^3 = 8$

The transformation for thresholding is shown in Fig. 1-Q. 2(b).

Here, $s = L - 1 = 7 \quad r \geq 3$

$s = 0 \quad r < 3$

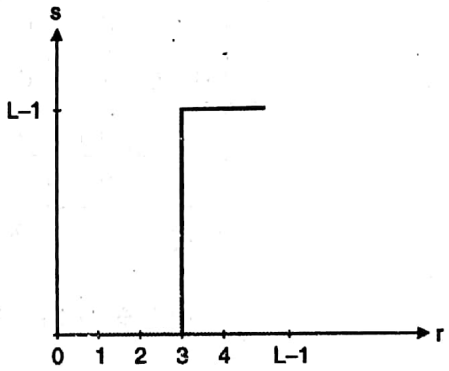


Fig. 1-Q. 2(b)

∴ The final image would be,

7	7	0	0
0	7	0	7
7	7	0	7
0	7	7	7

(ii) Intensity level slicing with background for $r_1 = 3, r_2 = 5$:

The transformation for intensity level slicing is shown in Fig. 2-Q. 2(b).

Here,

$s = L - 1 = 7 \quad 3 \leq r \leq 5$

$s = r \quad$ otherwise

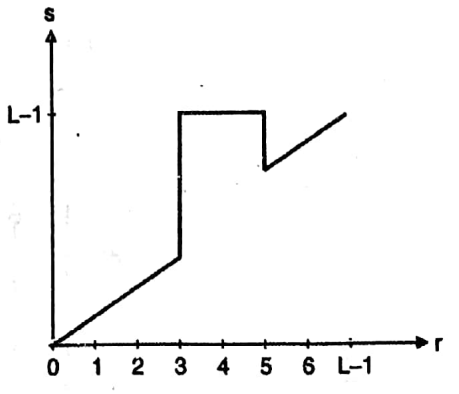


Fig. 2-Q. 2(b)

Hence all values between 2 and 5 are made equal to 7 and the remaining values remain unchanged. Hence the final result is,

7	7	1	2
1	7	0	7
7	7	2	6
2	7	6	7

(iii) Bit plane slicing :

We convert the image to binary. Since there are 8 grey levels (0 to 7), we require 3 bits to represent each pixel.

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

Binary ⟶

011	011	001	010
001	100	000	111
011	100	010	110
010	100	110	100

The LSB and MSB planes are shown below.

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

LSB plane

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

MSB plane

Chapter 5 : Histogram Modelling [Total Marks : 10]

Q. 3(a) Perform histogram equalization and draw new equalized histogram of the following image data. **(10 Marks)**

Gray level	0	1	2	3	4	5	6	7
No. of pixels	400	700	1350	2400	3000	1500	650	0

Ans. :

Here L = 8

We first plot the original histogram is shown in Fig. 1-Q. 3(a).

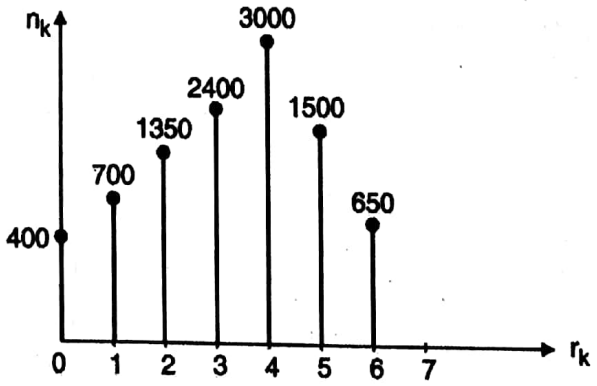


Fig. 1-Q. 3(a)

Grey level	n_k	PDF = $\frac{n_k}{n}$ $p_r(r_k)$	CDF $s_k = \Sigma p_r(r_k)$	$(L-1) \times s_k$ i.e. $7 \times s_k$	Round off
0	400	0.04	0.04	0.28	0
1	700	0.07	0.11	0.77	1
2	1350	0.135	0.245	1.715	2
3	2400	0.24	0.485	3.395	3
4	3000	0.3	0.785	5.495	5
5	1500	0.15	0.935	6.545	7
6	650	0.065	1	7	7
7	0	0	1	7	7
	N = 10000				

We consider the 1st, 2nd and last column.

Grey level	n_k	New grey level Round off
0	400	0
1	700	1
2	1350	2
3	2400	3
4	3000	5
5	1500	7
6	650	7 } 1500 + 650 + 0 = 2150
7	0	7

Hence the equalized frequency table is shown below.

Grey level	0	1	2	3	4	5	6	7
Frequency	400	700	1350	2400	-	3000	-	2150

The equalized histogram is shown in Fig. 2-Q.3(a).

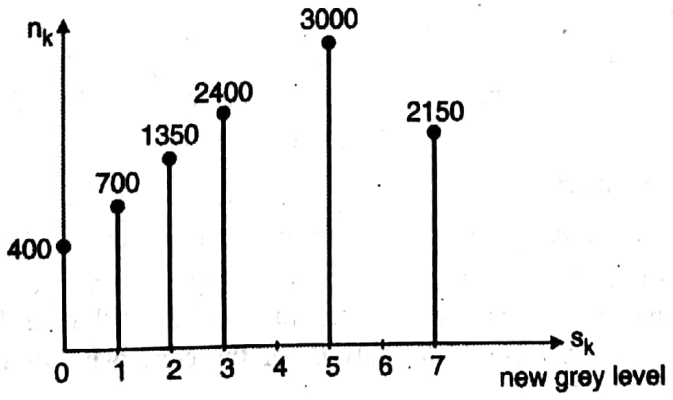


Fig. 2-Q.3(a)

Chapter 6 : Image Enhancement in Frequency Domain [Total Marks : 05]

Q. 6 (ii) Write detail note on : Homomorphic filter. **(5 Marks)**

Ans. : **Please refer Q. 3(b) of Dec. 2015.**

Chapter 7 : Image Segmentation [Total Marks : 15]

Q. 5(b) What is image segmentation ? Explain the following methods of image segmentation.

(i) Region growing

(ii) Split and merge . **(10 Marks)**

Ans. : **Please refer Q. 6(d) of Dec. 2015.**

Q. 6 (i) Write detail note on : Hough Transform **(5 Marks)**

Ans. : **Please refer Q. 6(a) of Dec. 2015.**

Chapter 8 : Image Morphology, Representation and Description [Total Marks : 15]

Q. 1(c) Explain in short morphological operations Dilation and Erosion. **(5 Marks)**

Ans. : **Please refer Q. 1(d) of Dec. 2015.**

Q. 6(iii) Write detail note on : Hit or Miss Transform. **(5 Marks)**

Ans. :

HIT or MISS transformation :

The Hit or Miss transformation is the morphological operator used for finding local patterns of pixels. By local, we mean the size of the structuring element. The concept is quite simple. A small odd sized mask (structuring element), typically 3×3, is scanned over a binary image. If the binary-valued pattern of the structuring element matches the state of the pixels under the structuring element (HIT), the output pixel in spatial correspondence to the center pixel of the structuring element is set to some desired binary state (normally 1). If the binary pattern of the structuring element does not match the state of the pixels under the structuring element (MISS), the output pixel in spatial correspondence to the center pixel of the structuring element is set to the opposite binary state (normally 0).

Let $B = (B_1, B_2)$ be the structuring element. Here B_1 is a set formed from elements of B associated with the object and B_2 is a set of elements of B associated with the corresponding background. $B = (B_1, B_2)$ is called a composite structuring element. The HIT-or-MISS transformation of set A with structuring element B is given by the equation

$$A \otimes B = \{a | B_1 \in A \text{ and } B_2 \in A^c\}$$

Where A is the image set, A^c is the complement of the image set and B is the structuring element.

This means that for a point 'a' to be in the resulting set, two conditions must be fulfilled simultaneously. First, the part B_1 of the composite structuring element that has its representative point at 'a' must be contained in A, and second, the part B_2 of the composite structuring element must be contained in A^c.

If this sounds very confusing, let us try to simplify what we just said. We are saying that the structuring element in the HIT-or-MISS transform is a slight extension to the type used in dilation and

erosion, in that it can contain both the foreground and background pixels rather than just foreground pixels.

An example of a composite structuring element used in a Hit-or-Miss operation is shown below.

$$B = \begin{array}{|c|c|c|}\hline \times & 1 & \times \\ \hline 0 & 1 & 1 \\ \hline 0 & \times & \times \\ \hline \end{array}$$

Here 1 : Foreground (object), 0 : Background, × : Don't care.

The Hit-or-Miss transformation operates as a binary matching between image A and the structuring element B.

If the foreground and background pixels in the structuring element exactly match the foreground and background pixels in the image, then the pixel underneath the origin of the structuring element in set to the foreground colour. If it doesn't match, that pixel is set to background colour.

The Hit-or-Miss transform can also be expressed in terms of dilation and erosion.

$A \otimes B = (A \ominus B_1) \cap (A^c \ominus B_2)$

The Hit-or-Miss transformation is used for thinning and thickening of objects. This is what the next section is all about.

Q. 6(iv) Write detail note on : Chain code. **(5 Marks)**

Ans. :

Chain codes :

Chain codes are used to represent a boundary by a connected sequence of straight-line segments. This representation is based on 4-connectivity or 8-connectivity of the segments. The chain code works best with binary images and is a concise way of representing a shape contour. The chain code direction convention is shown in Fig. 1-Q. 6(iv).

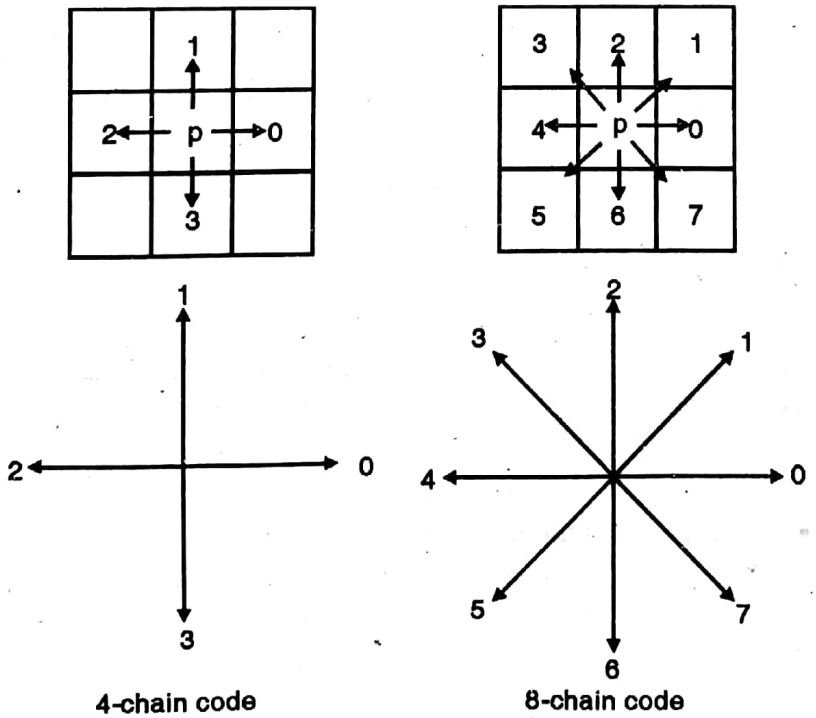


Fig. 1-Q. 6(iv)

As an edge is traced from its beginning point to the end point, the direction that must be taken to move from one pixel to the next is given by the number present in either the 4-chain code or the 8-chain code. An edge can be completely described in terms of its starting coordinate and its sequence of chain code descriptors. Of the two chain codes, the 4-chain code is easier requiring only four different code values.

Chapter 9 : Image Transforms [Total Marks : 25]

Q. 1(a) What is unitary transform matrix ? Explain with example. **(5 Marks)**

Ans. : **Please refer Q. 1(c) of Dec. 2015.**

Q. 4(a) Using matrix multiplication method calculate 2-D DFT of **(10 Marks)**

$$f(x, y) = \begin{bmatrix} 1 & 0 & 3 & 1 \\ 1 & 1 & 2 & 2 \\ 2 & 0 & 1 & 3 \\ 1 & 2 & 2 & 4 \end{bmatrix}$$

Ans. :

The 2-D DFT can be calculated using the formula

$$F = T f T'$$

Here T is the N × N DFT matrix.

Since the given image is of size 4 × 4, the DFT matrix in this case will also be 4 × 4.

$$\therefore T = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

Since T is symmetric, the 2 - D DFT can be calculated using the formula,

$$F = T f T$$

$$\therefore F = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 & 0 & 3 & 1 \\ 1 & 1 & 2 & 2 \\ 2 & 0 & 1 & 3 \\ 1 & 2 & 2 & 4 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

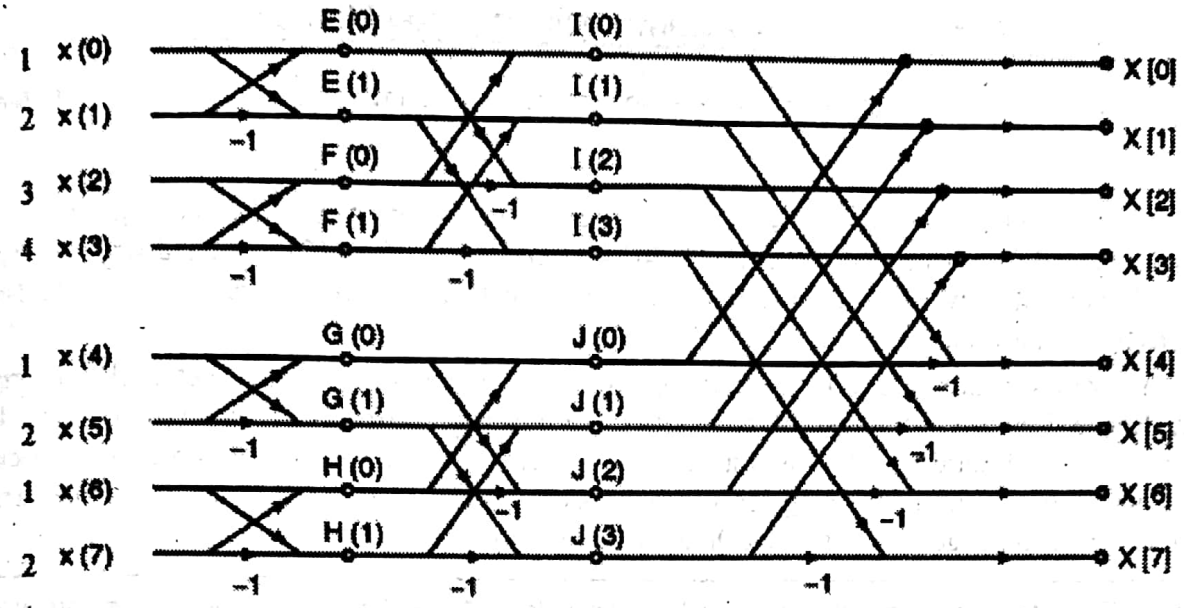
Solving the above matrix multiplication, we obtain the final 2-D DFT

$$\therefore F = \begin{bmatrix} 26 & -3+7j & 0 & -3-7j \\ -1+3j & -4-2j & 3-3j & -2+2j \\ -4 & 1+j & 6 & 1-j \\ -1-3j & -2-2j & -3+3j & -4+2j \end{bmatrix}$$

Q. 4(b) Using the butterfly diagram, compute Hadamard transform for $x(n) = \{1, 2, 3, 4, 1, 2, 1, 2\}$. **(10 Marks)**

Ans. :

Since N = 8, we use a 8 point Butterfly diagram.



$E(0) = x(0) + x(1) = 1 + 2 = 3$

$E(1) = x(0) - x(1) = 1 - 2 = -1$

$F(0) = x(2) + x(3) = 3 + 4 = 7$

$F(1) = x(2) - x(3) = 3 - 4 = -1$

$G(0) = x(4) + x(5) = +2 = 3$

$G(1) = x(4) - x(5) = 1 - 2 = -1$

$H(0) = x(6) + x(7) = 1 + 2 = 3$

$H(1) = x(6) - x(7) = 1 - 2 = -1$

$I(0) = E(0) + F(0) = 3 + 7 = 10$

$I(1) = E(1) + F(1) = -1 + (-1) = -2$

$I(2) = E(0) - F(0) = 3 - 7 = -4$

$I(3) = E(1) - F(1) = -1 - (-1) = 0$

$J(0) = G(0) + H(0) = 3 + 3 = 6$

$J(1) = G(1) + H(1) = -1 + (-1) = -2$

$J(2) = G(0) - H(0) = 3 - 3 = 0$

$J(3) = G(1) - H(1) = -1 - (-1) = 0$

$X[0] = [I(0) + J(0)] = 10 + 6 = 16$

$X[1] = [I(1) + J(1)] = -2 + (-2) = -4$

$X[2] = [I(2) + J(2)] = -4 + 0 = -4$

$X[3] = [I(3) + J(3)] = 0 + 0 = 0$

$X[4] = [I(0) - J(0)] = 10 - 6 = 4$

$X[5] = [I(1) - J(1)] = -2 - (-2) = 0$

$X[6] = [I(2) - J(2)] = -4 - 0 = -4$

$X[7] = [I(3) - J(3)] = 0 - 0 = 0$

$\therefore X[n] = \{16, -4, -4, 0, 4, 0, -4, 0\}$

Chapter 11 : Image Compression [Total Marks : 25]

Q. 1(d) Justify/contradict : All image compression techniques are invertible. **(5 Marks)**

Ans. : False

If an input x(n), transformed by a function T produces an output y(n), then putting y(n) into the inverse function T^{-1} produces the output x(n), and vice versa. i.e., $y(n) = T[x(n)]$, and $x(n) = T^{-1}[y(n)]$. Here T^{-1} is called the inverse of T. Such a transformation that has an inverse is called Invertible. Not all functions have an inverse.

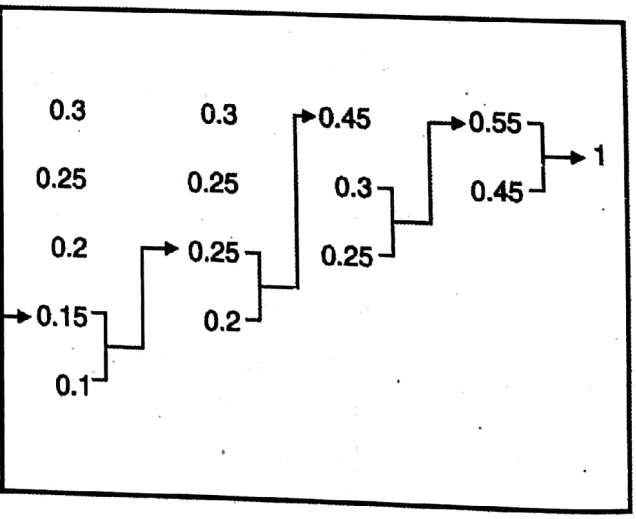
In image compression, after performing Lossless compression, we can get back the original image. Hence Lossless compression techniques are invertible. However, while performing lossy Compression, we discard data that is not visually significant. Hence it is impossible to get back the original image back after performing lossy compression. Hence lossy compression techniques are not invertible hence all image compression techniques are not invertible.

Q. 3(b) Find Huffman code for the symbols given below. Which kind of redundancy is removed by Huffman code ? Explain the term compression ratio. **(10 Marks)**

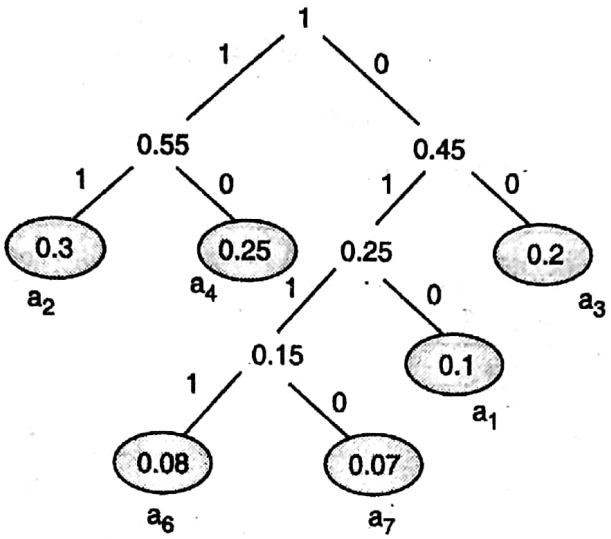
Symbols	Probability
a_1	0.1
a_2	0.3
a_3	0.2
a_4	0.25
a_5	0.07
a_6	0.08

Ans. : We begin with creating a Huffman table.

Symbol	Probability	Descending order
a_1	0.1	0.3
a_2	0.3	0.25
a_3	0.2	0.2
a_4	0.25	0.1
a_5	0.07	0.08
a_6	0.08	0.07



We now form the Huffman tree.



We lable the left side branches as 1 and right half branches as zero.

The encircled values are the original probabilities. The Huffman codes thus obtained are:

Symbol	Huffman code
a_1	010
a_2	11
a_3	00
a_4	10
a_5	0110
a_6	0111

Huffman code removes coding redundancies.

Compression ratio : Compression ratio is defined as

$$C_R = \frac{\text{Number of bits in output image}}{\text{Number of bits in input image}}$$

Q. 5(a) What are the different types of redundancies in digital image ? Explain in detail giving example of each. **(10 Marks)**

Ans. : Please refer Q. 5(a) of Dec. 2015.

❑❑❑

Image Processing

Statistical Analysis

Chapter No.	Dec. 2016	May 2017
Chapter 1	-	-
Chapter 2	10 Marks	-
Chapter 3	-	-
Chapter 4	10 Marks	15 Marks
Chapter 5	10 Marks	10 Marks
Chapter 6	-	05 Marks
Chapter 7	20 Marks	15 Marks
Chapter 8	20 Marks	20 Marks
Chapter 9	25 Marks	25 Marks
Chapter 10	-	-
Chapter 11	20 Marks	25 Marks
Chapter 12	05 Marks	05 Marks
Chapter 13	-	-
Repeated Questions	-	-

Dec. 2016

Chapter 2 : Image Sensing and Acquisition [Total Marks : 10]

Q. 4(b) What are various file formats ? Explain each in brief. **(10 Marks)**

Ans. :

Image file formats

Images obtained from the camera are stored on the host computer using different formats. A file format is a structure which defines how information is stored in the file and how that information is displayed on the monitor. There are numerous image file formats which are available. Of these only a few of them can be used universally. By universally it means, they can be understood by different operating systems.

Some of the image formats that can be used on either Macintosh or PC platforms are;

BMP (Bit Mapped Graphic Image) JPEG (Joint Photographic Expert Group)

TIFF (Tagged Image File Format) EPS (Encapsulated Post Script)

GIF (Graphic Interchange Format) PICT (Macintosh Supported)

TIFF, which is one of the most well known formats was developed in 1986 and in its many versions is a standard image format for a bit-mapped graphics image. The TIFF format is also a data compression technique for monochrome as well as colour images. Images seen on the Internet sites are

normally TIFF/GIF images simply because they occupy less space. GIF uses a form of Huffman coding.

BMP images developed by Microsoft can save both monochrome as well as colour images. All the wall papers that your computer has are BMP images. Similarly when we work in Paint Brush, we can save our work as a BMP image. The quality of BMP files is very good but they occupy a lot of memory space. PICT is a general purpose format supported by Macintosh and used for storing bit-mapped images. EPS is file format of the post script page description language and is device independent. This simply means that images can readily be transferred from one application to another. However EPS images can only be printed on post script compatible printers.

JPEG (Joint Photographic Expert Group) is the name of the committee that developed an image format which used compression algorithms.

JPEG images are compressed images which occupy very little space. It is based on the Discrete Cosine Transform-DCT (explained in chapter of Image Transforms). JPEG images use lossy compression algorithms which result in some loss of original data and hence the quality of JPEG images is not as good as BMP images.

Although these formats differ in technical details, they share structural similarities.

The Fig. 1-Q. 4(b) shows the typical organisation of information encoded in an image file.

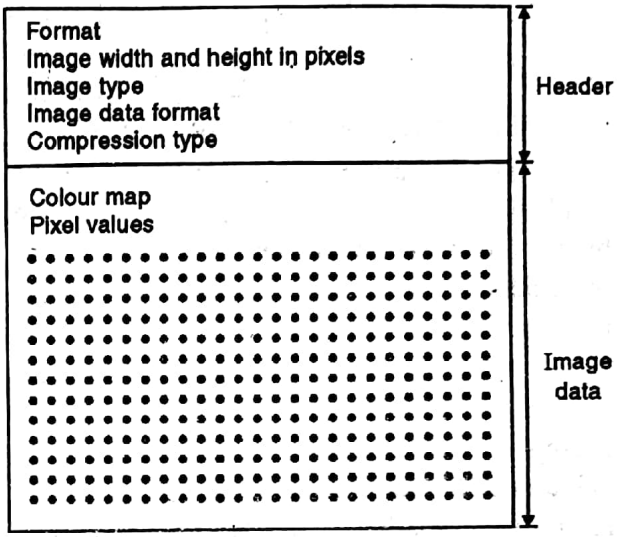


Fig. 1-Q. 4(b)

The image file consists of two parts;

(1) Header (2) Image data

The header file gives us the information about the kind of image. The header file, begins with a binary code or ASCII string which identifies the format being used. The width and height of the image are given in number of pixels. Common image types include binary images, 8-bit grey scale images, 8-bit colour and 24-bit colour images. The image data format specifies the order in which pixel values are stored in the image data section. A commonly used order is left to right and top to bottom. Image data format also specifies if the RGB values in the image are interlaced. By interlaced we mean that the RGB values of each pixel stay together consecutively followed by the three colour components for the next pixel i.e.,

$$\underbrace{R\ G\ B}_{1^{st}\text{ Pixel}}\quad\underbrace{R\ G\ B}_{2^{nd}\text{ Pixel}}\quad\underbrace{R\ G\ B}_{\cdots\cdots}\ \ldots\ldots$$

If the RGB values are not interlaced, the values of one primary colour for all pixels appear first, then the values of the next primary colour followed by the values of the third primary colour. Thus the image data is in the form.

R RR R..... G GG G..... B BB B.....

The compression type in the header indicates whether the image data is compressed using algorithms such as Run length encoding . Apart from these, there are a few more formats which we come across while dealing with images.

PGM (Portable Grey Map) – PGM is used to represent grey level images.

PBM (Portable Bit Map) – PBM is used to represent binary images

PPM (Portable Pixel Map) – PPM is used to represent RGB colour images

These formats are distinguished by 2 character signatures as shown below.

Signature	Image type	Storage type
P1	Binary (PBM)	ASCII
P2	Grey scale (PGM)	ASCII
P3	RGB (PPM)	ASCII
P4	Binary (PBM)	Raw bytes
P5	Grey scale (PGM)	Raw bytes
P6	RGB (PPM)	Raw bytes

Most often the storage type is ASCII.

The header of a PGM, PBM or PPM file begins with the appropriate signature followed by a blank line. There is a comment line after the signature hich starts with the # character. Next in the header are the width and height of the image as ASCII decimal values. PBM files have no further header information. PGM and PPM files contain a third integer value, again in ASCII decimal form, representing the maximum allowable pixel value.

From this value, we could find out the number of bits allocated for each pixel. For example, if the maximum allowable pixel value is 255 then we know that each bit is represented by 8 bits ($2^8 = 256$) Fig. 2-Q. 4(b) shows a 8 × 8 grey scale image and its representation as a ASCII PGM file

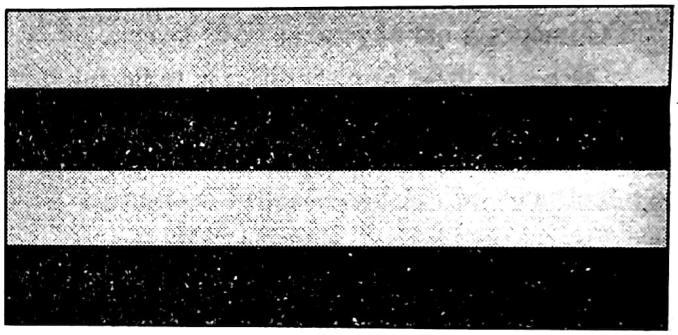


Fig. 2-Q. 4(b)

Header

```
P2
#A PGM image
8 8 255
120 120 120 120 120 120 120 120
120 120 120 120 120 120 120 120
33 33 33 33 33 33 33 33
33 33 33 33 33 33 33 33
120 120 120 120 120 120 120 120
120 120 120 120 120 120 120 120
33 33 33 33 33 33 33 33
33 33 33 33 33 33 33 33
```

Just by observing the header, we know that it is a PGM file. P2 indicates that it is a grey level image stored is ASCII. The 8 8 255 below the comment line indicates that the size of the image is 8 × 8 and can have 256 grey levels i.e., each value is represented by 8-bits.

Chapter 4 : Image Enhancement In Spatial Domain[Total Marks-10]

Q. 1(a) Explain any five zero memory operations. **(10 Marks)**

Ans. : Zero memory operations

In point processing, we work with single pixels i.e. T is 1 × 1 operator. It means that the new value g(x, y) depends on the operator T and the present f(x, y). To implement point processing techniques, we do not need to store previous values. Hence no memory is required for point processing operations. Because of this, point processing techniques are also called zero memory operations.

Some of the common examples of point processing are

(1) Digital negative (2) Contrast stretching
(3) Thresholding (4) Grey level slicing
(5) Bit plane slicing (6) Dynamic range compression
(7) Power law transformation

5 zero memory operations

(1) **Digital negative :** Digital negatives are useful in a lot of applications. A common example of digital negative is the displaying of an X-ray image. As the name suggests, negative simply means inverting the grey levels i.e. black in the original image will now look white and vice versa. The Fig. 1-Q. 1(a) is the digital negative transformation for a 8-bit image.

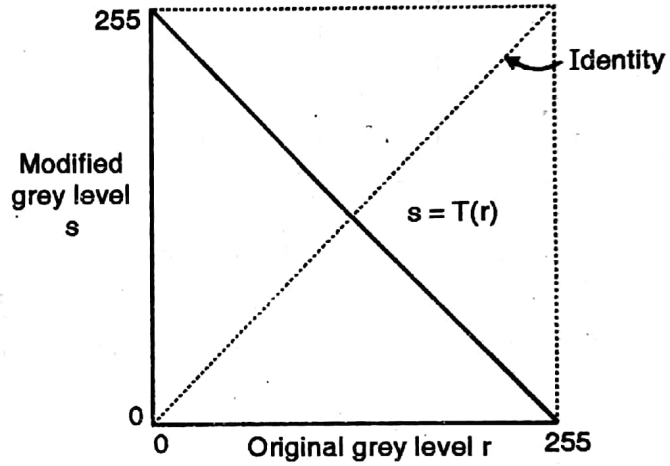


Fig. 1-Q. 1(a)

The digital negative image can be obtained by using a simple transformation given by,

$$s = 255 - r \ (r_{max} = 255)$$

Hence when

$$r = 0, s = 255 \text{ and when } r = 255, s = 0.$$

In general,

$$s = (L - 1) - r \quad \ldots(1)$$

Here L is the number of grey levels. (256 in this case)

(2) **Thresholding :** Extreme contrast stretching yields thresholding. If we observe the contrast stretching diagram closely we notice that if the first and the last slope are made zero, and the centre slope is increased, we would get a thresholding transformation i.e. if $r_1 = r_2$, $s_1 = 0$ and $s_2 = L - 1$, we get a thresholding function. It is shown in Fig. 2-Q. 1(a).

The formula for achieving thresholding is as follows.

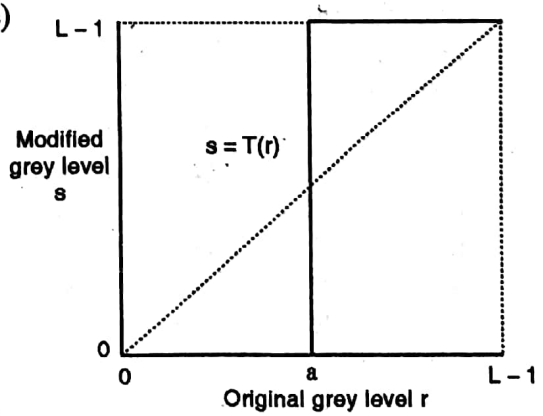
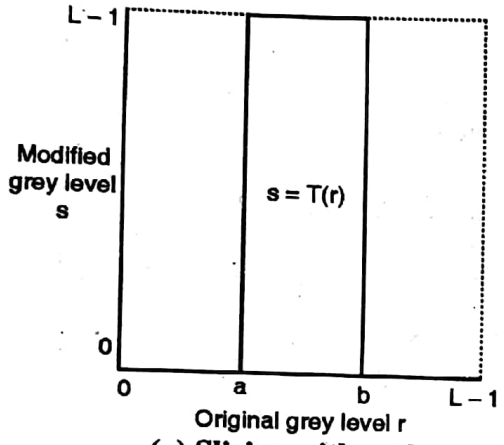


Fig. 2-Q. 1(a) : Original grey level r

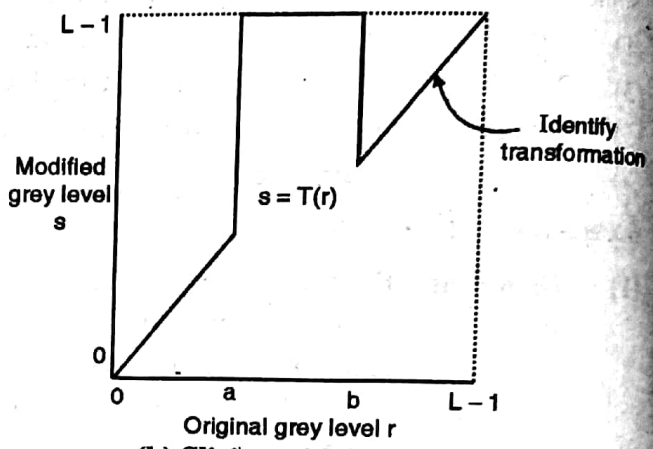
$$\left.\begin{array}{ll} s = 0; & \text{if } r \le a \\ s = L-1; & \text{if } r > a \end{array}\right\} \quad \ldots(2)$$

Where L is the number of grey levels.

(3) **Grey level slicing (Intensity slicing) :** What thresholding does is it splits the grey level into two parts. At times, we need to highlight a specific range of grey values like for example enhancing the flaws in an X-ray or a CT image. In such circumstances, we use a transformation known as grey level slicing. The transformation is shown in the Fig. 3(a)-Q. 1(a). It looks similar to the thresholding function except that here we select a band of grey level values.



(a) Slicing without background



(b) Slicing with background

Fig. 3-Q. 1(a)

This can be implemented using the formulation

$$\left.\begin{array}{ll} s = L-1; & \text{if } a \le r \le b \\ s = 0; & \text{otherwise} \end{array}\right\} \quad \ldots(3)$$

This method is known as Grey level slicing without background. This is because in this process, we have completely lost the background. In some applications, we not only need to enhance a band of grey levels but also need to retain the background. This technique of retaining the background is called as Grey-level slicing with background. The transformation is as shown in the Fig. 3(b)-Q. 1(a).The formulation for this is

$$\left.\begin{array}{ll} s = L-1; & \text{if } a \le r \le b \\ s = r; & \text{otherwise} \end{array}\right\} \quad \ldots(4)$$

(4) **Dynamic range compression (Log transformation) :** At times, the dynamic range of the image exceeds the capability of the display device. What happens is that some pixel values are so large that the other low value pixels get obscured. A simple day-to-day example of such a phenomena is that during daytime, we cannot see the stars. The reason behind this is that the intensity of the sun is so large and that of the stars is so low that the eye cannot adjust to such a large dynamic range.

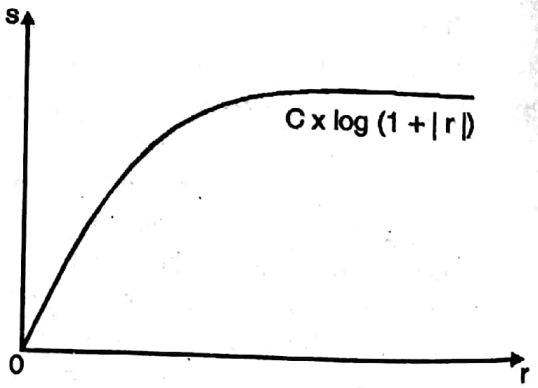


Fig. 4-Q. 1(a)

In image processing, a classic example of such large differences in grey levels is the Fourier spectrum. In the Fourier spectrum only some of the values are very large while most of the values are too small.

The dynamic range of pixels is of the order of 10^6. Hence, when we plot the Fourier spectrum, we see only small dots, which represent the large values. Something needs to be done to be able to see the small values as well. This technique of compressing the dynamic range is known as dynamic range compression. We all know that the log operator is an excellent compressing function. Hence the dynamic range compression is achieved by using a log operator. C is the normalisation constant.

(5) Power law transformation

The basic formula for power-law transformation is

$$g(x, y) = c \times f(x, y)^{\gamma}$$

It can also be written as

$$s = c\, r^{\gamma} \quad ...(5)$$

Here c and γ are positive constants. The transformation is shown below for different values of γ which is also called the gamma correction factor. We observe that by changing the value of gamma, we obtain a family of transformation curves.

Non-linearities encountered during image capturing, printing and displaying can be corrected using gamma correction. Hence gamma correction is important if the image needs to be displayed on the computer. The power law transformation can also be used to improve the dynamic range of an image.

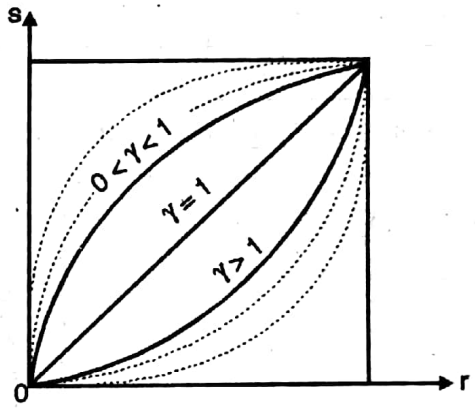


Fig. 5-Q. 1(a) : Gamma correction factor

Chapter 5 : Histogram Modelling [Total Marks-10]

Q. 1(b) Perform histogram equalization and draw new equalized histogram of the following image data. **(10 Marks)**

Grey level	0	1	2	3	4	5	6	7
Number of pixels	800	1000	850	650	300	250	100	150

Ans. :

We draw the original histogram as shown in Fig. 1-Q. 1(b).

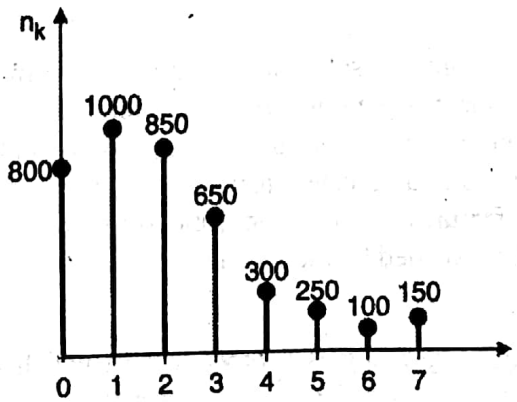


Fig. 1-Q. 1(b)

Grey level	n_k	PDF $p_r(r_k) = \frac{n_k}{\Sigma n}$	CDF $s_k = \Sigma p_r(r_k)$	$(L-1) \times s_k$ i.e. $7 \times s_k$	Round off New grey levels	No. of pixels
0	800	0.195	0.195	1.365	1	800
1	1000	0.244	0.439	3.073	3	1000
2	850	0.207	0.646	4.522	5	850
3	650	0.159	0.805	5.635	6	650 + 300 = 950
4	300	0.073	0.878	6.146	6	
5	250	0.061	0.939	6.573	7	250 + 100 + 150 = 500
6	100	0.024	0.963	6.741	7	
7	150	0.037	1	7	7	
	$\Sigma n_k = 4100$					

The equalized histogram is shown in Fig. 2-Q. 1(b).

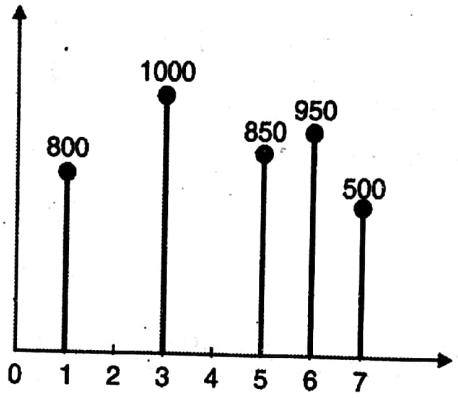


Fig. 2-Q. 1(b)

Chapter 7 : Image Segmentation [Total Marks 20]

Q. 2(b) What is segmentation ? Explain (i) Region Growing (ii) Region splitting (iii) Thresholding **(10 Marks)**

Ans. :

Segmentation

Segmentation as the name suggests, subdivides (segments) an image into its constituent regions or objects. Unlike image enhancement, were our primary concern was to improve the subjective quality of images for display, in segmentation, we address some aspects of analyzing the content of an image. Segmentation is used when we need to automate a particular activity. The ultimate aim in an automated system is to extract important features from image data, from which description, interpretation, or understanding of the scene can be provided by the machine.

(i) Region Growing

As the name implies, region growing is a procedure in which pixels are grouped into larger regions based on some predefined condition. The basic approach is to select a seed point (a pixel from where we begin) and grow regions from this seed pixel.

Let us pick up an arbitrary pixel (x_1, y_1) from the image that needs to be segmented. This pixel is called the seed pixel. We now examine the nearest neighbours (4 or 8 neighbours depending on whether we assume 4-connectivity or 8-connectivity) of (x_1, y_1) one by one. The neighbouring pixel is accepted in the same region as (x_1, y_1) if they together satisfy the homogeneity property of a region i.e., if both of them satisfy the predefined condition.

Once a new pixel (x_2, y_2) is accepted as a member of the current region, the neighbours of this new pixel are examined (again, 4 or 8 neighbours, depending on the connectivity assumed).

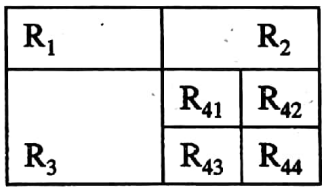
This procedure goes on recursively until no new pixel is accepted. All the pixels of the current region are given a unique label. Now a new seed pixel is chosen and the same procedure is repeated. We continue doing this until all the pixels are assigned to some region or the other.

(ii) Region Splitting

In this case we try to satisfy the homogeneity property where pixels that are similar are grouped together. If the grey levels present in the region do not satisfy the property, we divide the region into four equal quadrants. If the property is satisfied, we leave the region as it is. This is done recursively until all the regions satisfy the property. To explain this in terms of graph theory, we call each region a node. This node (parent node) is split into its four children (leaf nodes) if it does not satisfy the given property. If the node satisfies the property, it is left as it is. We now check each leaf node and see if these leaf nodes satisfy the given property. If yes, they are left unaltered and if no, there are further split.

This particular splitting technique has a convenient representation in the form of a quad tree. (That is, a tree in which a node has exactly four descendents).

Consider the figure shown below



In this, image R represents the entire image and hence R is the parent node. This parent node is split up into four leaf nodes, R_1, R_2, R_3 and R_4. Of these leaf nodes only R_4 does not contain pixels which satisfy some common property and hence is split again i.e., R_{41}, R_{42}, R_{43}, R_{44}. The quad tree for this division is shown in Fig. 1-Q. 2(b).

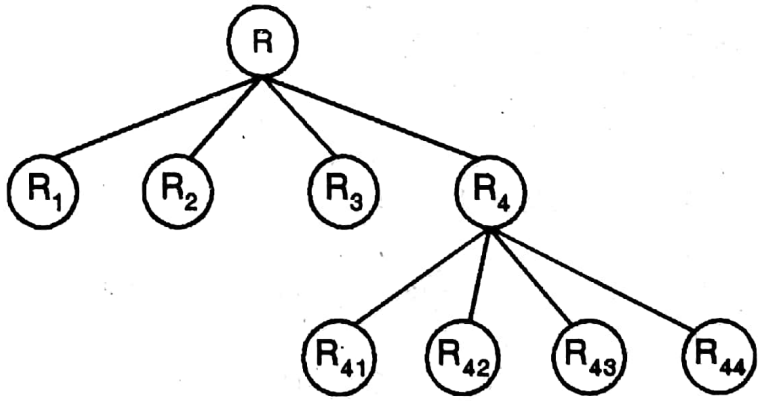


Fig. 1-Q. 2(b)

This method is applicable to images whose number of rows and number of columns are some integer power of 2.

(iii) Thresholding

Thresholding is one of the most important techniques for segmentation and because of its simplicity, is widely used. Thresholding has already been discussed in the chapter of image enhancement. We shall define thresholding in a more formal way here. In segmentation,

thresholding is used to produce regions of uniformity within the given image based on some threshold criteria T.

Thresholding is used to produce regions of uniformity within the given image based on some threshold criteria T.

Suppose we have an image which is composed of dark objects of varying grey levels against a light background of varying grey levels as shown in Fig. 2-Q. 2(b).

Fig. 2-Q. 2(b)

The histogram of such an image would appear more or less as shown in Fig. 3-Q. 2(b).

An obvious way to extract the objects from the background is to select a threshold T that separates the two regions. Then a point (x, y) for which $f(x, y) < T$ is called an object point and for $f(x, y) > T$, it is called a background point.

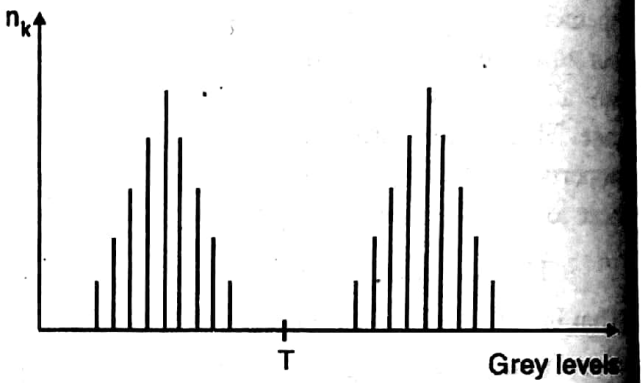


Fig. 3-Q. 2(b)

Similarly we could have images whose histograms look like the one shown in Fig. 4-Q. 2(b).

i.e., an image with two different types of dark objects (large variation in their grey levels) against a bright background. In such a case, we would require two threshold values, T_1 and T_2 to separate the two objects from their background. This is known as multilevel thresholding.

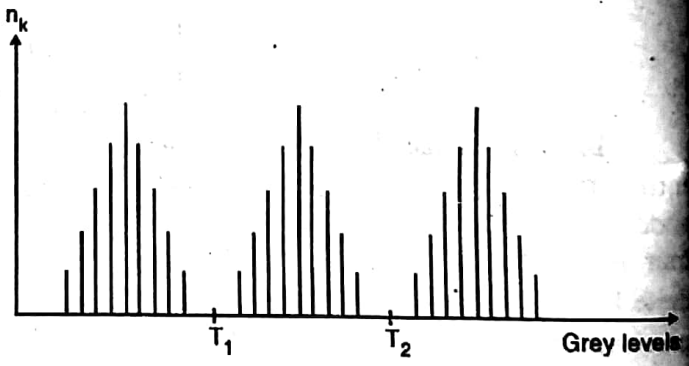


Fig. 4-Q. 2(b)

The thresholding operation can thus be thought of as a test involving the function T and can be defined as

$$T = T\{x, y, A(x, y), f(x, y)\}$$

Here, f(x, y) is the grey level of the pixel at (x, y) and A(x, y) denotes some local property in the neighbourhood of this image.

Q. 6(b) Write short note on : Hough transform **(5 Marks)**

Ans. :

Hough transform

Hough transform has emerged over the past decade as a method of choice for pixel linking and curve detection. Hough presented this transform in the year 1962.

Given a set of discrete pixels, the Hough transform checks if these points lie on a straight line and if yes, it draws a line joining all these points.

Hough transform is best explained by an example. Consider a point (x_1, y_1). A line passing through this point can be written in the slope-intercept form as $y_1 = ax_1 + b$.

Using this equation and varying the values of a and b, infinite number of lines pass through this point (x_1, y_1).

However, if we write this equation as

$$b = -ax_1 + y_1$$

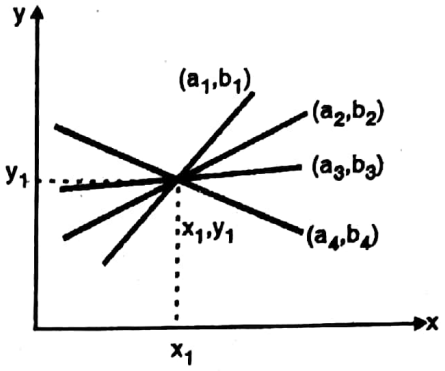


Fig. 1-Q. 6(b)

and consider the ab plane instead of the xy plane, we get a single line for a point (x_1, y_1)

This entire line in the ab plane is due to a single point in the xy plane and different values of a and b.

Now consider another point (x_2, y_2) in the xy plane.

The slope intercept equation of this line is

$$y_2 = ax_2 + b$$

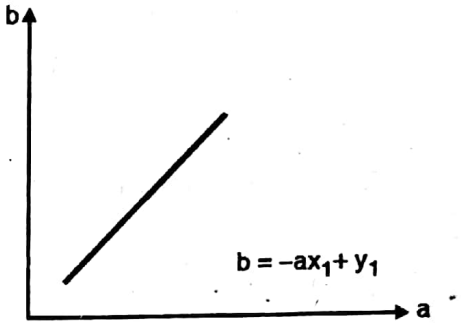


Fig. 2-Q. 6(b)

Writing this equation in terms of the ab plane we get

$$b = -ax_2 + y_2$$

This is another line in the ab plane. These two line will intersect each other somewhere in the ab plane only if they are a part of a straight line in the xy plane.

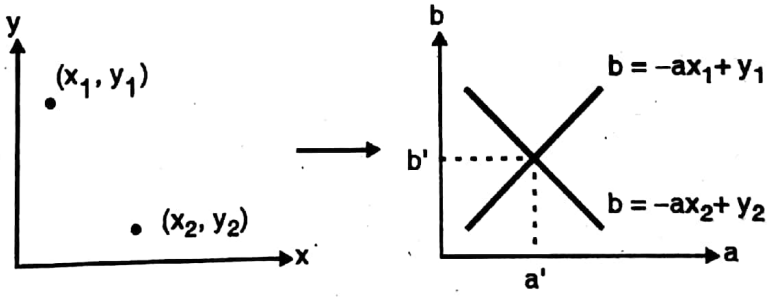


Fig. 3-Q. 6(b)

The point of intersection in the ab plane is noted (a', b').

Using this (a', b') in the standard slope-intercept form i.e., $y = a'x + b'$, we get a line that passes through points (x_1, y_1) and (x_2, y_2) in the xy plane!!!!

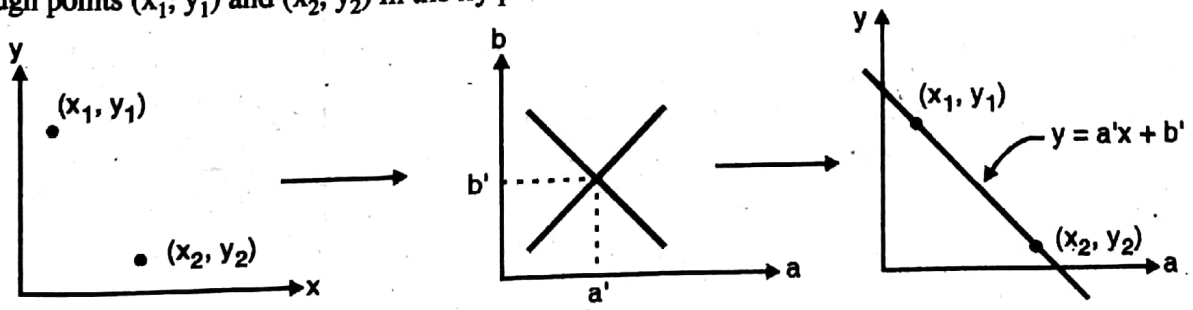


Fig. 4-Q. 6(b)

In fact if we have n points that lie on a straight line, we get n lines (each line representing a point) in the ab plane and all these lines would intersect at a single point (a′, b′) in the ab plane. Using these values of a′ and b′ in equation y = a′x + b′, we would get a line that would pass through all these points in the xy plane.

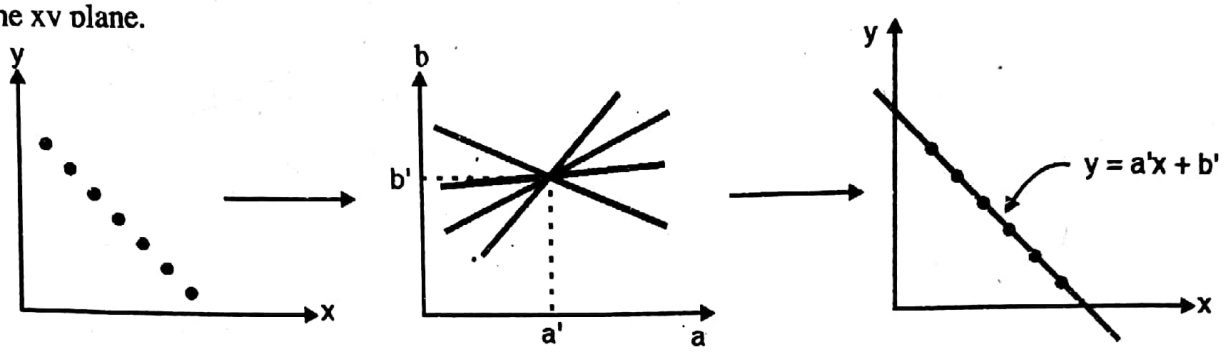


Fig. 5-Q. 6(b)

The ab plane is called the parameter space.

This parameter space is subdivided into accumulator cells as shown.

Here (a_{max}, a_{min}) and (b_{max}, b_{min}) are the expected ranges of slope and intercept values.

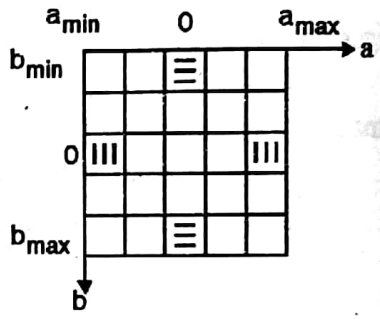


Fig. 6-Q. 6(b)

Q. 6(d) Write short note on : 4, 8 and m-connectivity **(5 Marks)**

Ans. :

4- Connectivity

Two pixels p and q with some common criteria are said to be 4-connected if they share a side. i.e., two pixels p and q with some common criteria are 4-connected if q is in the set $N_4(p)$.

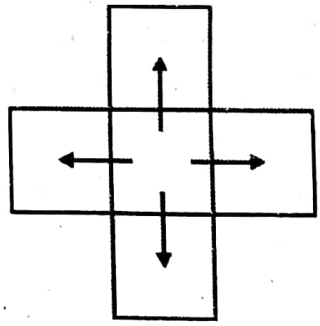


Fig. 1-Q. 6(d)

8-Connectiviy

Two pixels p and q with some common criteria are said to be 8-connected if they share either a side or a corner. i.e, two pixels p and q with some common criteria are 8-connected if q is in the set $N_8(p)$.

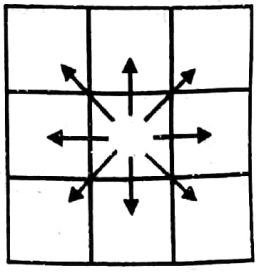


Fig. 2-Q. 6(d)

m-Connectivity

Two pixels p and q are said to be m-connected if

(i) q is in $N_4(p)$ or

(ii) q is in $N_D(p)$ and the set $N_4(p) \cap N_4(q)$ is empty and p and q share some common criteria.

Chapter 8 : Image Morphology, Representation and Description
[Total Marks-20]

Q. 3(a) Explain with an example that the first difference of a chain code normalizes it to rotation. **(10 Marks)**

Ans. :

Consider a simple boundary

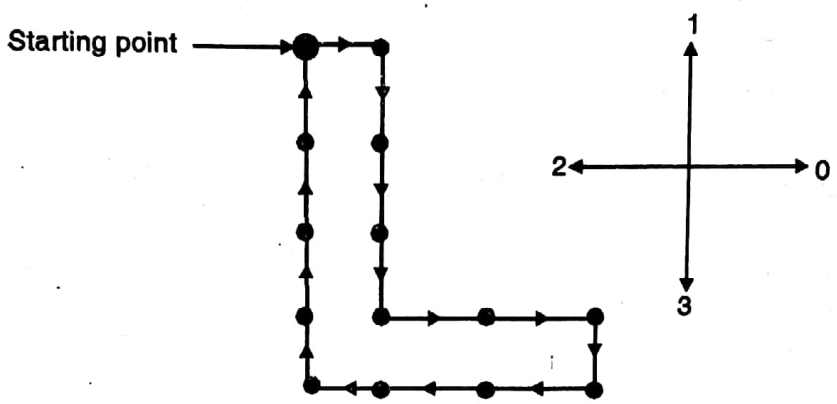


Fig. 1-Q. 3(a)

The 4-directional chain code (clockwise direction) is

0 3 3 3 0 0 3 2 2 2 1 1 1 1

Its first difference is

3 0 0 1 0 3 3 0 0 3 0 0 0 ...(1)

We shall now rotate the boundary

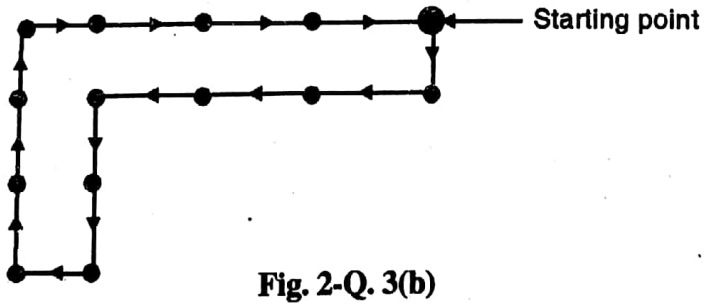


Fig. 2-Q. 3(b)

The 4-directional chain code is

3 2 2 2 3 3 2 1 1 1 0 0 0 0

It's first difference is

3 0 0 1 0 3 3 0 0 3 0 0 0 ...(2)

Here 1 = 2

We can thus conclude that the first difference of the original object and the first difference of the object after rotation remain the same.

Hence the first difference of the chain code normalises it is rotation.

Q. 3(b) Explain the following morphological operations : (i) Opening (ii) Closing **(10 Marks)**

Ans. :

Dilation and Erosion operations are fundamental to Image Morphology. In fact, many of th morphological algorithms that would be discussed now are based on these two operations.

Opening

Morphological opening of an image is basically Erosion followed by a Dilation, using the sam structuring element.

If A is the image and B is the structuring element then, opening of A by B is given as

$$OPEN(A, B) = D(E(A)) \rightarrow D \rightarrow \text{Dilation} \quad \ldots(1$$

$$E \rightarrow \text{Erosion}$$

It is also written as

$$A \circ B = (A \ominus B) \oplus B$$

Opening generally smoothes the contours of the image breaks down narrow bridges an eliminates thin protrusions. Thus opening isolates objects which may be just touching one another Opening is therefore used in analysis of wear particles in engine oil, ink particles in recycled paper an most importantly in study of cells in cytology.

Example: Perform the opening operation on the image shown. Use the structuring elemen Image is of size 10×10

①
1

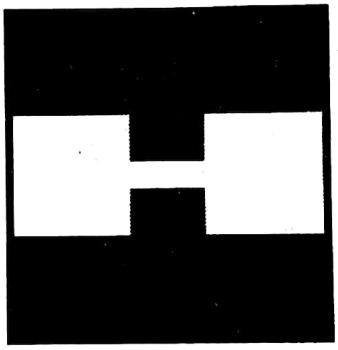


Fig. 1-Q. 3(b)

The image raw data is shown

Opening is basically Erosion followed by Dilation

①
1

Open = D (E(A))

$A \circ B = (A \ominus B) \oplus B$

A =

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

B =

①
1

Step 1 : Remember, Erosion is minimum

$$\{A(x - i, y - j) \times B(i, j)\}$$

Eroding A with B, we get

$A \ominus B =$

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

We now dilating this image

$Open = (A \ominus B) \oplus B =$

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

The modified image looks like

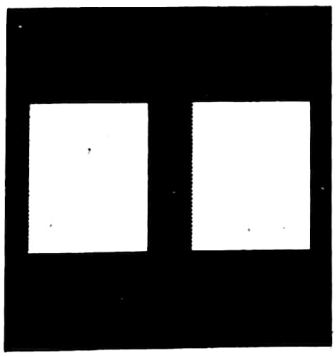


Fig. 2-Q. 3(b)

The original image had two white blocks which were connected by a thin white strip. Opening this image got rid of this strip. The size of the white blocks remain unchanged. Hence we see that opening breaks down narrow bridges or isolates objects which may just be touching one another.

(ii) Closing

Morphological closing of an image is basically Dilation followed by Erosion, using the same structuring element.

$$CLOSE(A, B) = E(D(A)) \quad \ldots(2)$$

It is also written as

$$A \bullet B = (A \oplus B) \ominus B$$

Closing generally tends to fuse narrow breaks and eliminates small holes. This simplifies the process of assessing the separation of particles.

Example : Perform closing operation on the image shown below. The size of the image is 10×11. The two white blocks are not connected if we consider 4-connectivity.

Use the same structuring element

B =

①
1

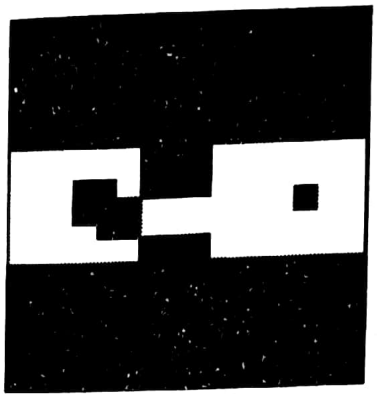


Fig. 3-Q. 3(b)

The image raw data is as shown. Note the two big blocks are not connected if we consider 4-connectivity.

A =

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

We know that closing is basically Dilation followed by

Close = E (D(A)) $\qquad A \bullet B = (A \oplus B) \ominus B$

Remember,

Dilation= Maximum $\{A(x - i, y - j) \times B(i, j)\}$

Dilating A, we get

$A \oplus B =$

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

Eroding $(A \oplus B)$,

Closing $= (A \oplus B) \ominus B$

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

The modified image looks like Fig. 4-Q. 3(b).

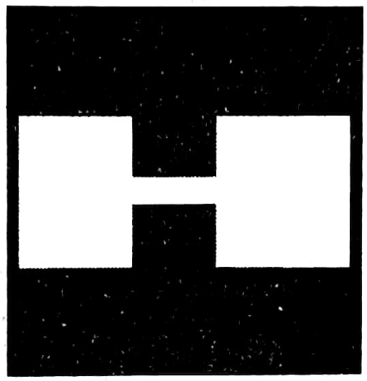


Fig. 4-Q. 3(b)

Comparing this image with the original image, we conclude that the closing operation tends to fuse narrow breaks and also eliminates small holes.

Chapter 9 : Image Transforms [Total Marks-25]

Q. 2(a) Find the DFT of the given image : **(10 Marks)**

0	3	3	1
3	1	2	1
3	2	4	2
1	1	2	1

Ans. : The 2-D DFT can be calculated using the formula

$$F = T f T'$$

Here T is the N ×N DFT matrix.

Since the given image is of size 4×4, the DFT matrix in this case will also be 4×4.

$$\therefore T = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

Since T is symmetric, the 2 - D DFT can be calculated using the formula,

$$F = T f T$$

$$\therefore F = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 0 & 3 & 3 & 1 \\ 3 & 1 & 2 & 1 \\ 3 & 2 & 4 & 2 \\ 1 & 1 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

Solving the above matrix multiplication, we obtain the final 2-D DFT

$$\therefore F = \begin{bmatrix} 30 & -4-2j & 6 & -4+2j \\ -4-2j & -2-4j & -4-2j & -2 \\ 6 & -4-2j & -2 & -4+2j \\ -4+2j & -2 & -4+2j & -2+4j \end{bmatrix}$$

Q. 5(b) Write 8 × 8 HADAMARD transform matrix and its signal flow graph. Using Butterfly diagram, compute HADAMARD transform for x(n) = {1,2,1,2,1,2,3,4}. **(10 Marks)**

Ans. : 8 × 8 Hadamard transform matrix is shown below

$$H(8) = \left[\begin{array}{cccc:cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ \hdashline 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{array}\right]$$

Sign changes: 0, 7, 3, 4, 1, 6, 2, 5

Since N = 8, we use 8 point Butterfly diagram,

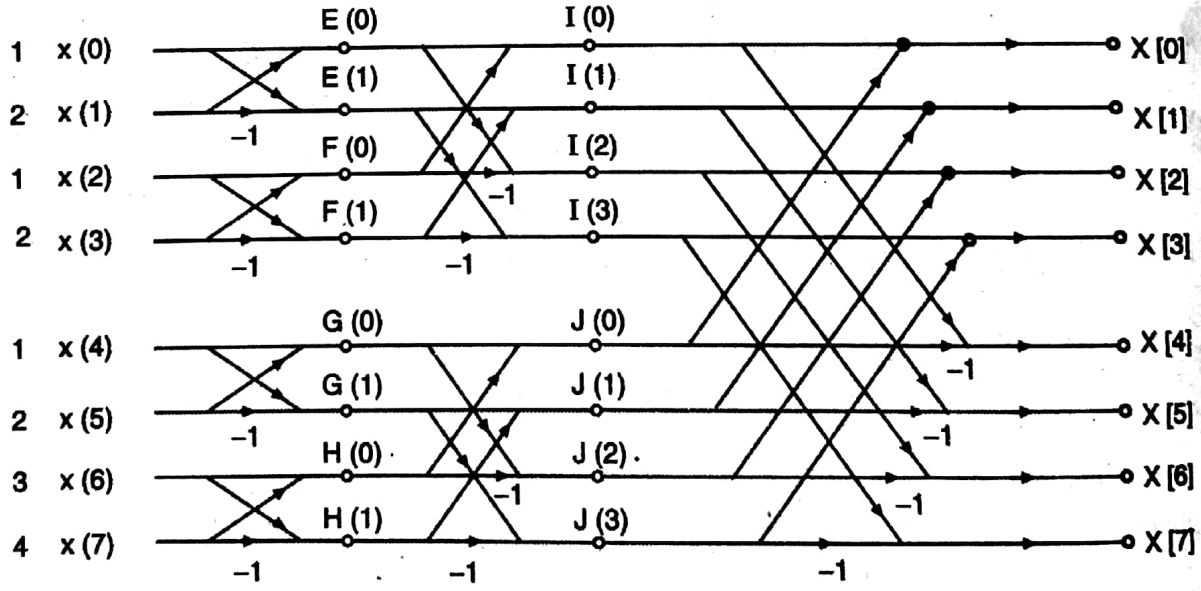


Fig. 1-Q. 5(b)

$$E(0) = x(0) + x(1) = 1 + 2 = 3$$
$$E(1) = x(0) - x(1) = 1 - 2 = -1$$
$$F(0) = x(2) + x(3) = 1 + 2 = 3$$
$$F(1) = x(2) - x(3) = 1 - 2 = -1$$

$$G(0) = x(4) + x(5) = 1 + 2 = 3$$
$$G(1) = x(4) - x(5) = 1 - 2 = -1$$
$$H(0) = x(6) + x(7) = 3 + 4 = 7$$
$$H(1) = x(6) - x(7) = 3 - 4 = -1$$
$$I(0) = E(0) + F(0) = 3 + 3 = 6$$
$$I(1) = E(1) + F(1) = -1 - 1 = -2$$
$$I(2) = E(0) - F(0) = 3 - 3 = 0$$
$$I(3) = E(1) - F(1) = -1 - (-1) = 0$$
$$J(0) = G(0) + H(0) = 3 + 7 = 10$$
$$J(1) = G(1) + H(1) = -1 - 1 = -2$$
$$J(2) = G(0) - H(0) = 3 - 7 = -4$$
$$J(3) = G(1) - H(1) = -1 - (-1) = 0$$
$$X[0] = [I(0) + J(0)] = (6 + 10) = 16$$
$$X[1] = [I(1) + J(1)] = (-2 - 2) = -4$$
$$X[2] = [I(2) + J(2)] = (0 - 4) = -4$$
$$X[3] = [I(3) + J(3)] = (0 + 0) = 0$$
$$X[4] = [I(0) - J(0)] = (6 - 10) = -4$$
$$X[5] = [I(1) - J(1)] = (-2 - (-2)) = 0$$
$$X[6] = [I(2) - J(2)] = (0 - (-4)) = 4$$
$$X[7] = [I(3) - J(3)] = (0 - 0) = 0$$

$\therefore$ $X[n] = \{16, -4, -4, 0, -4, 0, 4, 0\}$

Q. 6(a) Write short note on : Discrete Cosine Transform **(5 Marks)**

Ans. :

Discrete Cosine Transform (DCT)

The Cosine Transform published by N. Ahmed, T. Natrajan and K. R. Rao has found applications in image compression. It has excellent energy compactness. All JPEG images use the Discrete Cosine Transform as the initial stage for compression.

Just as the Fourier transform uses sines and cosines waves to represent a signal, the DCT uses only cosine waves. Hence the DCT is purely real unlike the DFT which is complex (has magnitude and phase).

The one dimensional DCT of a sequence $f(x)$, $0 \le x \le N-1$ is defined as,

$$F(u) = \alpha(u) \sum_{x=0}^{N-1} f(x) \cos\left[\frac{\pi (2x+1) u}{2N}\right]; \quad 0 \le u \le N-1$$

Where,

$$\alpha(0) = \sqrt{\frac{1}{N}}; \quad \alpha(u) = \sqrt{\frac{2}{N}} \quad \text{for } 1 \le u \le N-1$$

The inverse transformation is given by,

$$f(x) = \sum_{u=0}^{N-1} \alpha(u) F(u) \cos\left[\frac{\pi (2x+1) u}{2N}\right]; \quad 0 \le x \le N-1$$

. The corresponding 2-D DCT pair is,

$$F(u, v) = \alpha(u)\,\alpha(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) \cos\left[\frac{(2x+1)\,u\,\pi}{2N}\right] \cos\left[\frac{(2y+1)\,v\,\pi}{2N}\right]$$

for u, v = 0, 1, 2, ...N – 1 and

$$f(x, y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \alpha(u)\,\alpha(v)\,F(u, v) \cos\left[\frac{(2x+1)\,u\pi}{2N}\right] \cos\left[\frac{(2y+1)\,v\,\pi}{2N}\right]$$

for x, y = 0, 1, 2, ...N – 1.

The N × N Cosine Transform Matrix C = {C (u, v)} is defined as,

$$C(u, v) = \begin{cases} \sqrt{\frac{1}{N}} & u = 0,\ 0 \le v \le N-1 \\ \sqrt{\frac{2}{N}} \cos\left[\frac{\pi\,(2v+1)\,u}{2N}\right] & 1 \le u \le N-1,\ 0 \le v \le N-1 \end{cases} \quad \ldots(1)$$

It is this equation that we shall use to compute the DCT of a given sequence.

The cosine transform matrix is real and orthogonal but not symmetric.

$$\therefore\ C = C^* \Rightarrow C^{-1} = C'$$

$$\therefore\ C \cdot C' = I$$

The DCT of a column sequence f(x); $0 \le x \le N-1$ can be computed.

$$F = C \cdot f \quad \ldots(2)$$

Similarly, since the discrete cosine transform is not symmetric, the 2-dimensional - DCT of an image can be generated.

$$\therefore\ F = C\,f\,C' \quad \ldots(3)$$

Chapter 11 : Image Compression [Total Marks-20]

Q. 4(a) Classify image compression methods in detail along with the different redundancies that can be present in digital images. **(10 Marks)**

Ans. :

Images in their raw from occupy a lot of memory space and hence data compression becomes necessity while dealing with them. Data compression basically tries to reduce the overall size of data. This reduction is possible when the original dataset contains some type of unwanted data redundancy. Digital images contain a lot of data which provide no relevant information, also called redundant data which can be exploited to reduce the size of the image.

In general, three basic redundancies exist in digital images that follows :

1. **Inter-pixel Redundancy :** It is a redundancy corresponding to statistical dependencies or high correlation among neighbouring pixels. Hence in images, there exists large area having the same grey level. This is known as inter-pixel redundancy and can be reduced using Run Length encoding.
2. **Coding Redundancy :** In an uncompressed image, every pixel is coded with a fixed number of bits known as fixed length coding. For example, an image with 256 gray scales is represented by an array of 8-bit integers. However this type of coding does not take into account the fact that some grey level occur more often than the others. This is known as coding redundancy. Huffman coding and arithmetic coding are used to reduce coding redundancy.

3. **Psycho-visual Redundancy :** Even though an image could have 256 grey levels, the Human eye can resolve very few grey levels locally. Hence Psycho-visual redundancy deals with the way humans perceive grey levels. Psycho-visual redundancy can be reduced by using IGS coding.

Run Length Encoding (RLE)

It is the simplest dictionary-based data compression technique. Image files frequently contain the same character repeated many times in a row. Images, particularly those having very few grey levels, often contain regions of adjacent pixels, all with the same grey level. Each row of such images can have long runs of the same grey value. In, such cases, one can store a code specifying the value of the grey level, followed by the length of the run, rather than storing the same value many times over.

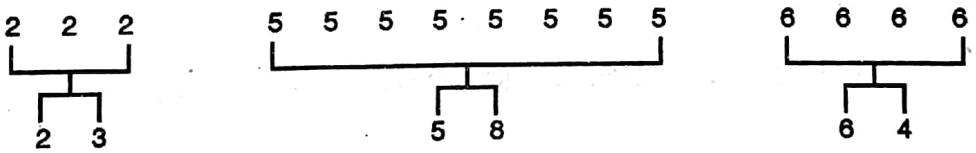
Let us take an example.

Consider the first row of an image which has the following grey values.

2	2	2	5	5	5	5	5	5	5	5	6	6	6	6
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•

The first row of the image has 15 grey values.

We could store the above row in a compact manner using RLE.



The first code specifies the grey value, followed by the length of the run.

Hence the RLE of the first row is simply.

2 3 5 8 6 4

As is evident, run length encoding achieves considerable compaction in images which have a fairly constant background. The RLE eliminates *Inter-pixel redundancies.*

Statistical coding

The images dealt with so far have been coded and saved in the natural code. That is for a 256 grey level image, grey level at f(x, y) is coded with its 8-bits binary equivalent. A grey level of a value 4 is coded as 0000 0100. Similarly for a 8 grey level image, grey value at f (x, y) is coded using 3-bits. The table that shows codes for a sampled image having grey levels varying from 0 to 7 (3-bit image) is given. These codes are referred to as Equal Length Codes.

Input	Natural code
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Equal length codes do not make use of the statistics of data. The assumption is made that all grey levels have equal occurrence in the image. Since this possibility is unlikely, this is not an optimum for of coding. If we can develop a code such that fewer bits are assigned to grey levels having higher probability of occurrence and vice versa, we could reduce the number of bits required for transmission. Such a coding is known as Variable Length Coding or Entropy Coding and it eliminates *Coding redundancies.*

Here, we are not discarding any information as would be done in lossy compression, but simply represent more frequently occurring values with shorter codes and vice versa.

Consider an image containing L grey levels i = 0, 1 ..., L – 1. Let n_i denote the number of pixels having grey level i.

The probability of occurrence of grey level i in the image is

$$p_i = \frac{n_i}{\sum_{i=0}^{L-1} n_i} \quad \ldots(1)$$

Let us define entropy H associated with the grey level as

$$H = -\sum_{i=0}^{L-1} p_i \log_2 p_i \quad \ldots(2)$$

Entropy is the measure of randomness in the set of random variables. Entropy is never negative since p_i lies in the range [0,1].

Details of the above equation can be found in any book on Information theory. When all variables are equally likely i.e., $p_0 = p_1 = p_2 ..... = p_{L-1} = 1/L$, then the randomness is maximized and the entropy attains its greatest value.

Improved Grey Scale (IGS) Quantization

The brightness of a region as perceived by the eye does not depend on the absolute grey level values. This is because the human eye does not respond with equal sensitivity to all visual information. Some information is visually more important than the others.

Information which is not visually important is called *Psychovisual Redundancy*. Psychovisual redundancies exist in all images and can be eliminated without hampering the subjective quality of the image. We have seen that simply reducing the quantization (number of bits for representation), compresses the image but also produces false contouring. I.G.S. coding reduces the quantization but also reduces false contouring.

Steps involved in I.G.S. coding are,

(1) Lower 4-bits of the preceding modified pixel are added to the present pixel.

(2) The new 4 MSB's of the present pixel are taken as the I.G.S. code.

(3) Repeat step 1 and 2 after moving on to a new pixel.

(4) If MSB is 1111, then add 0000 instead of the 4 LSB's of the previous sum.

Q. 5(a) Given **(10 Marks)**

$$f = \begin{bmatrix} 10 & 44 & 16 \\ 10 & 14 & 48 \\ 11 & 10 & 22 \end{bmatrix}$$

(i) Find 3-bit IGS coded image and calculate compression factor and BPP.

(ii) Find the decoded image and calculate MSE and PSNR.

Ans. :

The given image is 6-bit image. We begin by converting the pixel values to their binary equivalent.

Grey level	Binary	Sum	3 bit IGS code
		000000	
10	001010	001010	001
44	101100	101110	101
16	010000	010110	010
10	001010	010000	010
14	001110	001110	001
48	110000	110110	110
11	001011	010001	010
10	001011	001100	001
22	010110	011010	011

IGS code gives us 3-bits/pixels (3-BPP).

The original data was 6-bit while the IGS code gave us a 3-bit code. Hence IGS coding by default gives 50% compression.

To obtain the MSE we need to obtain the decoded image. We multiply the IGS code with 2^5, 2^4 and 2^3.

Grey level	IGS code	Decimal equivalent
10	001	$(0\times2^5)+(0\times2^4)+(1\times2^3)=8$
44	101	$(1\times2^5)+(0\times2^4)+(1\times2^3)=40$
16	010	$(0\times2^5)+(1\times2^4)+(0\times2^3)=16$
10	010	$(0\times2^5)+(1\times2^4)+(0\times2^3)=16$
14	001	$(0\times2^5)+(0\times2^4)+(1\times2^3)=8$
48	110	$(1\times2^5)+(1\times2^4)+(0\times2^3)=48$

Grey level	IGS code	Decimal equivalent
11	010	$(0\times2^5)+(1\times2^4)+(0\times2^3)=16$
10	001	$(0\times2^5)+(0\times2^4)+(1\times2^3)=8$
22	011	$(0\times2^5)+(1\times2^4)+(1\times2^3)=24$

Hence the decoded image is shown below

$$f'(x,y) = \begin{array}{|c|c|c|}\hline 8 & 40 & 16 \\ \hline 16 & 8 & 48 \\ \hline 16 & 8 & 24 \\ \hline \end{array}$$

The MSE is given by the formula,

$$MSE = \frac{1}{M\cdot N}\sum\sum[f(x,y)-f'(x,y)]$$

$$= \frac{1}{9}[(10-8)^2+(44-40)^2+(16-16)^2+(10-16)^2+(14-8)^2$$

$$+(48-48)^2+(11-16)^2+(10-8)^2+(22-24)^2]$$

$$MSE = 13.89$$

Chapter 12 : Colour Image Processing [Total Marks-05]

Q. 6(c) Write short note on : HSI color model **(5 Marks)**

Ans. :

HSI colour model

HSI is an important colour model and it's design reflects the way humans see colour. For example, one does not refer to the colour of a dress by giving the percentage of the red, green and blue components!!

When we see colour, we describe it in terms of its Hue, Saturation and Intensity (Brightness).

In the HSI model, I stands for intensity and for our purpose it is simply the average of the R, G and B components. The I component specifies the brightness irrespective of the colour H stands for Hue. Hue is an attribute that describes pure colour. Hue is what the artist refers to as "pigment" it is what we think as colour. Yellow, orange, cyan etc. are examples of Hue. S stands for Saturation. This gives us the measure of the degree to which pure colour is diluted by white light. The HSI colour model decouples the Intensity component from Hue and Saturation which are the colour carrying components. As a result, the HSI model is an ideal tool for developing image processing algorithms. Hue and saturation which are the colour information parameters can be illustrated by the colour circle.

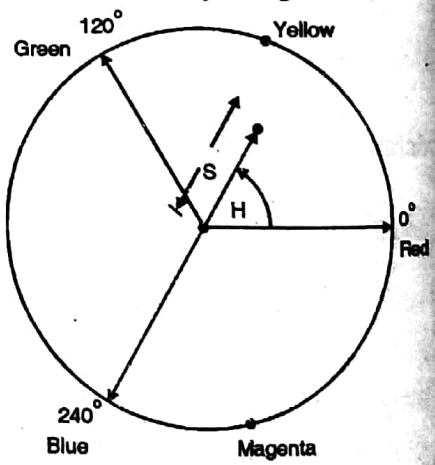


Fig. 1-Q. 6(c)

Hue is expressed as an angle. Generally, a Hue of 0° is red, 120° is green and 240° is blue, Hue traverses the colours of the visible spectrum as it goes from Red to Blue i.e. from 0° to 240°. Between 240° and 360° fall the non-spectral colours (purple) that the eye perceives. The saturation parameter is the length of the segment joining the point to the center of the colour circle.

The concept of saturation is fairly easy to understand. Imagine we want to paint our house and we have a bucket of pure Red paint. At this stage we are at 0° of the colour circle. Hence this pure red corresponds to a hue of 0° and a saturation of 1. If we mix white paint in the bucket, we are actually reducing its saturation. Red becomes pink. Pink would probably correspond to a saturation of about 0.5. As we add more and more white paint, eventually, the colour in the bucket would become white.

Similarly if we add some black to the original bucket of Red, we reduce its intensity (Making it dark red). If we keep on adding Black, eventually the entire colour in the bucket would become black.

Red and Pink are different saturations of the same Hue-Red. HSI is a great format to change from one colour to another. For example, to change cyan to magenta, only H needs to be changed. In the RGB format, we would have to change all the three R, G and B values.

❑❑❑

May 2017

Chapter 4 : Image Enhancement In Spatial Domain[Total Marks-15]

Q. 1(c) Explain low pass spatial filtering. **(5 Marks)**

Ans. :

Low pass spatial filtering

Low pass filtering as the name suggests removes the high frequency content from the image. It is used to remove noise present in the image.

Low pass averaging filter

If an image has Gaussian noise present in it, we use a low pass averaging filter to eliminate the noise. The frequency response of the low-pass filter along with its spatial response is shown in Fig. 1-Q. 1(c).

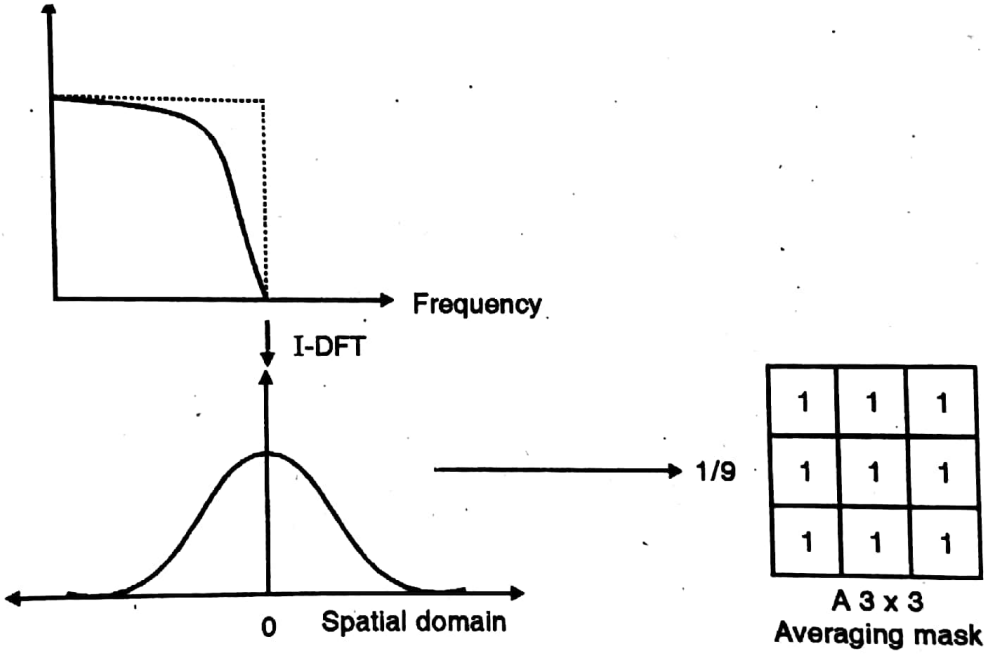


Fig. 1-Q. 1(c)

From the spatial response we generate the mask that would give us the low pass filtering operation. One important thing to note from the spatial response is that all the coefficients are positive. The standard low pass averaging mask (3 × 3) is given above. As the name suggests, each element of the mask is the average value.

We could also use a 5 × 5 or a 7 × 7 mask as per our requirement.

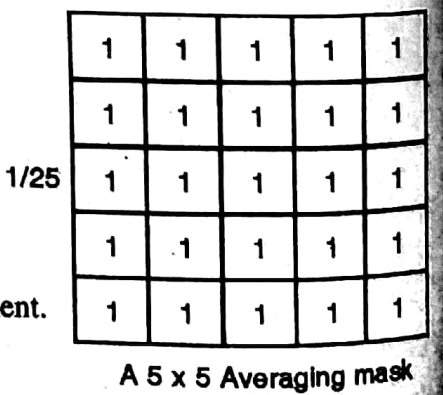


Fig. 2-Q. 1(c)

Low pass median filtering

The averaging filter removes the noise by blurring it till it is no longer seen. But in the process, it also blurs the edges. Bigger the averaging mask more is the blurring. There are times when the image contains salt and pepper noise. If we use an averaging filter to remove the same, it will blur the noise but it would also ruin the edges. Hence when we need to eliminate salt and pepper noise, we work with a non-linear filter known as the Median filter. They are also called order-statistic filters because their response is based on the ordering or ranking of the pixels contained within the mask.

In this case, we use a mask similar to the averaging filter except that the mask has no values. So it's like working directly with the 8 neighbours of the centre pixel.

The steps to perform median filtering are as follows :

(1) Assume a 3×3 empty mask.
(2) Place the empty mask at the left hand corner.
(3) Arrange the 9 pixels in ascending or descending order.
(4) Choose the median from these nine values.
(5) Place this median at the centre.
(6) Move the mask in a similar fashion to the averaging filter.

In median filtering, the grey level of the centre pixel is replaced by the median value of the neighbourhood.

Q. 2(a) Explain any five zero memory point operations. **(10 Marks)**

Ans. :

Zero memory point operations

In point processing, we work with single pixels i.e. T is 1×1 operator. It means that the new value g(x, y) depends on the operator T and the present f(x, y).

To implement point processing techniques, we do not need to store previous values. Hence no memory is required for point processing operations. Because of this, point processing techniques are also called zero memory operations.

Some of the common examples of point processing are

(1) Digital negative
(2) Contrast stretching
(3) Thresholding
(4) Grey level slicing
(5) Bit plane slicing
(6) Dynamic range compression
(7) Power law transformation

5 zero memory operations

(1) **Digital negative :** Digital negatives are useful in a lot of applications. A common example of digital negative is the displaying of an X-ray image. As the name suggests, negative simply means inverting the grey levels i.e. black in the original image will now look white and vice versa. The Fig. 1-Q. 2(a) is the digital negative transformation for a 8-bit image.

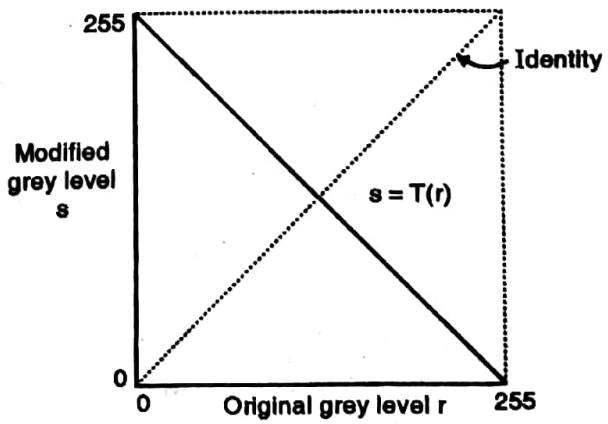


Fig. 1-Q. 2(a)

The digital negative image can be obtained by using a simple transformation given by,

$$s = 255 - r \ (r_{max} = 255)$$

Hence when

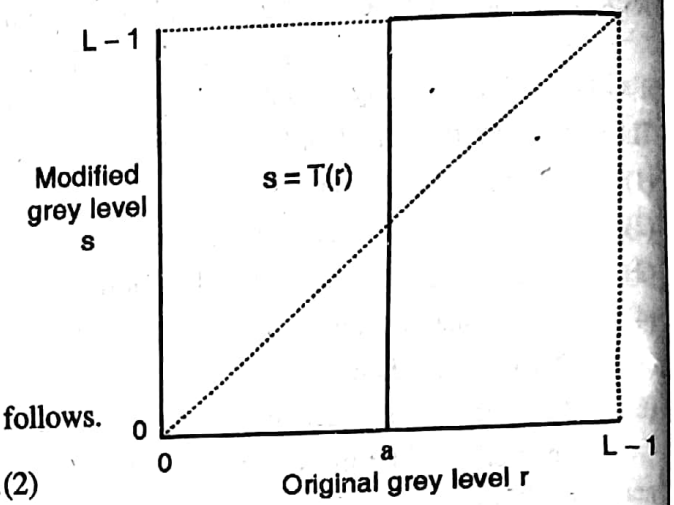
$$r = 0, s = 255 \text{ and when } r = 255, s = 0.$$

In general,

$$s = (L-1) - r \quad \ldots(1)$$

Here L is the number of grey levels. (256 in this case)

(2) **Thresholding** : Extreme contrast stretching yields thresholding. If we observe the contrast stretching diagram closely we notice that if the first and the last slope are made zero, and the centre slope is increased, we would get a thresholding transformation i.e. if $r_1 = r_2$, $s_1 = 0$ and $s_2 \neq L - 1$, we get a thresholding function. It is shown in Fig. 2-Q. 2(a).



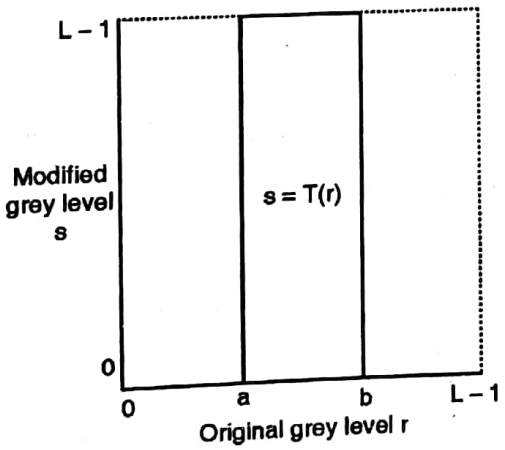
The formula for achieving thresholding is as follows.

$$\left.\begin{array}{ll} s = 0; & \text{if } r \leq a \\ s = L-1; & \text{if } r > a \end{array}\right\} \quad \ldots(2)$$

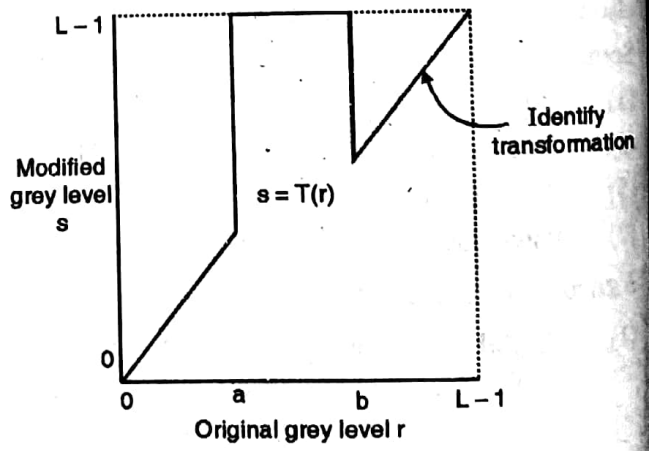
Fig. 2-Q. 2(a) : Original grey level r

Where L is the number of grey levels.

(3) **Grey level slicing (Intensity slicing) :** What thresholding does is it splits the grey level into two parts. At times, we need to highlight a specific range of grey values like for example enhancing the flaws in an X-ray or a CT image. In such circumstances, we use a transformation known as grey level slicing. The transformation is shown in the Fig. 3(a)-Q. 2(a). It looks similar to the thresholding function except that here we select a band of grey level values.



(a) Slicing without background



(b) Slicing with background

Fig. 3-Q. 2(a)

This can be implemented using the formulation

$$\left.\begin{array}{ll} s = L-1; & \text{if } a \leq r \leq b \\ s = 0; & \text{otherwise} \end{array}\right\} \quad \ldots(3)$$

This method is known as Grey level slicing without background. This is because in this process, we have completely lost the background. In some applications, we not only need to enhance a band of

grey levels but also need to retain the background. This technique of retaining the background is called as Grey-level slicing with background. The transformation is as shown in the Fig. 3(b)-Q. 2(a).The formulation for this is

$$\left.\begin{array}{ll} s = L-1; & \text{if } a \le r \le b \\ s = r; & \text{otherwise} \end{array}\right\}$$

(4) **Dynamic range compression (Log transformation) :** At times, the dynamic range of the image exceeds the capability of the display device. What happens is that some pixel values are so large that the other low value pixels get obscured. A simple day-to-day example of such a phenomena is that during daytime, we cannot see the stars. The reason behind this is that the intensity of the sun is so large and that of the stars is so low that the eye cannot adjust to such a large dynamic range. In image processing, a classic example of such large differences in grey levels is the Fourier spectrum. In the Fourier spectrum only some of the values are very large while most of the values are too small. The dynamic range of pixels is of the order of 10^6. Hence, when we plot the Fourier spectrum, we see only small dots, which represent the large values.

Something needs to be done to be able to see the small values as well. This technique of compressing the dynamic range is known as dynamic range compression. We all know that the log operator is an excellent compressing function. Hence the dynamic range compression is achieved by using a log operator. C is the normalisation constant.

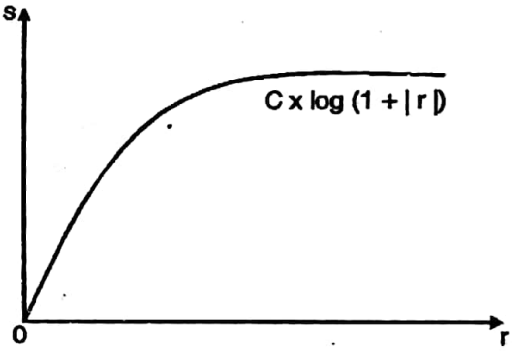


Fig. 4-Q. 2(a)

(5) **Power law transformation**

The basic formula for power-law transformation is

$$g(x, y) = c \times f(x, y)^{\gamma}$$

It can also be written as

$$s = c\, r^{\gamma} \qquad \ldots(4)$$

Here c and γ are positive constants. The transformation is shown below for different values of γ which is also called the gamma correction factor. We observe that by changing the value of gamma, we obtain a family of transformation curves.

Non-linearities encountered during image capturing, printing and displaying can be corrected using gamma correction. Hence gamma correction is important if the image needs to be displayed on the computer. The power law transformation can also be used to improve the dynamic range of an image. Given below is the MATLAB code for power transformation. The final image has been normalized to 0 - 255 range.

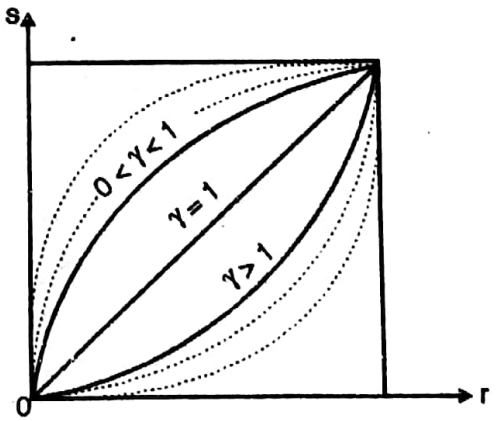


Fig. 5-Q. 2(a) : Gamma correction factor

Chapter 5 : Histogram Modelling [Total Marks-10]

Q. 2(b) Perform histogram equalization and draw new equalized histogram of the following image data. **(10 Marks)**

Grey level	0	1	2	3	4	5	6	7
Number of pixels	550	900	650	150	300	250	110	90

Ans. :

Grey level	0	1	2	3	4	5	6	7
Number of pixels	550	900	650	150	300	250	110	90

Draw the original histogram as shown in Fig. 1-Q. 2(b).

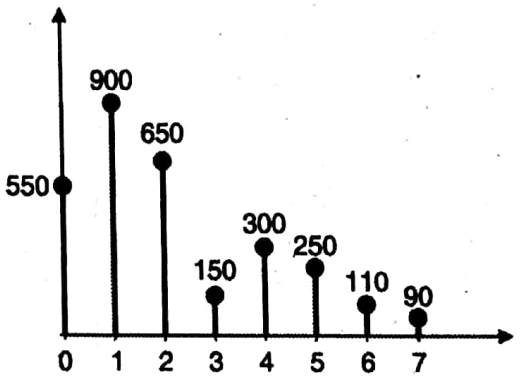


Fig. 1-Q. 2(b)

Now perform histogram equalization.

Grey level	n_k	PDF $p_r(r_k) = \frac{n_k}{\Sigma n}$	CDF $s_k = \Sigma p_r(r_k)$	$(L-1) \times s_k$ i.e. $7 \times s_k$	Round off New grey levels	No. of pixels
0	550	0.183	0.183	1.281	1	550
1	900	0.30	0.483	3.381	3	900
2	650	0.217	0.70	4.9	5	65 + 150 = 800
3	150	0.05	0.750	5.250	5	
4	300	0.10	0.850	5.950	6	300
5	250	0.083	0.933	6.531	7	250 + 110 + 90 = 450
6	110	0.037	0.970	6.790	7	
7	90	0.030	1	7	7	
	$\Sigma n_k = 3000$					

The equalized histogram is shown in Fig. 2-Q. 2(b).

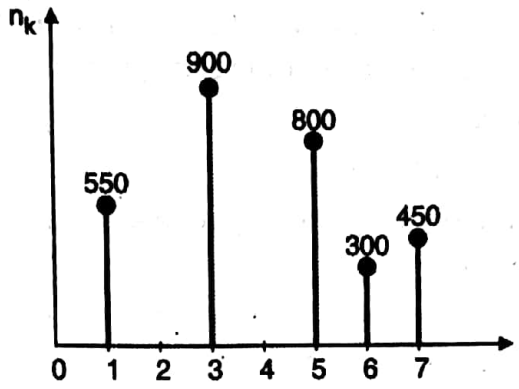


Fig. 2-Q. 2(b)

Chapter 6 : Image Enhancement in Frequency Domain[Total Marks-05]

Q. 6(d) Write short note on : Homomorphic filter **(5 Marks)**

Ans. :

Homomorphic filtering

Homomorphic filter uses a slightly different approach compared to other frequency domain filters. Homomorphic filtering uses the illumination-reflectance model to filter images in the frequency domain. In homomorphic filtering we separate the illumination and reflectance components and work with them separately.

$$f(x, y) = i(x, y) \cdot r(x, y)$$

We cannot directly take the Fourier transform since

$$F\{f(x, y)\} \neq F\{i(x, y)\}F\{r(x, y)\}$$

To separate the i and the r components, we use the log function.

$$Z(x, y) = \ln f(x, y)$$
$$Z(x, y) = \ln i(x, y) + \ln r(x, y)$$

We now take the Fourier transform to move into the frequency domain

$$F\{Z(x, y)\} = F\{\ln i(x, y)\} + F\{\ln r(x, y)\}$$
$$Z(u, v) = F_i(u, v) + F_r(u, v)$$

Where $F_i(u, v) = F\{\ln i(x, y)\}$

$F_r(u, v) = F\{\ln r(x, y)\}$

We now use a filter H(u, v)

$\therefore S(u, v) = Z(u, v) \cdot H(u, v)$

$S(u, v) = F_i(u, v) \cdot H(u, v) + F_r(u, v) \cdot H(u, v)$

The key approach in homomorphic filtering is to separate the illumination and reflectance components. Between them i(x, y) contributes to the low frequency since illumination is more or less uniform and r(x, y) is the high frequency component since r(x, y) tends to vary abruptly at junctions of dissimilar objects.

A typical homomorphic filter H(u, v) is shown in Fig. 1-Q. 6(d).

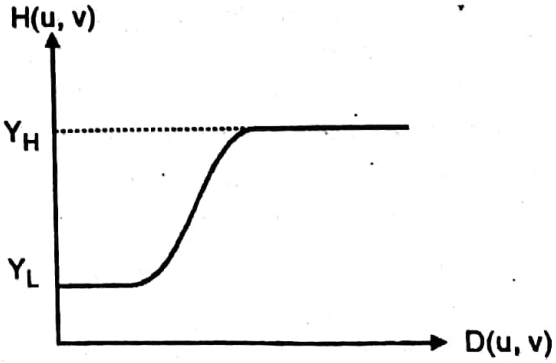


Fig. 1-Q. 6(d)

Generally $Y_L < 1$ and $Y_H > 1$. H(u, v) tends to decrease the contribution made by low frequencies (illumination) and amplify the contribution made by high frequencies (reflectance).

How do we get back the modified image ?

$$S(u, v) = H(u, v) F_i(u, v) + H(u, v) F_r(u, v)$$

To come back to the spatial domain, we take the inverse Fourier transform.

$$s(x, y) = F^{-1}\{S(u, v)\}$$
$$= F^{-1}\{H(u, v) F_i(u, v)\} + F^{-1}\{H(u, v) F_r(u, v)\}$$
$$s(x, y) = i'(x, y) + r'(x, y)$$

Finally remember, the first step in homomorphic filtering was to take the log, hence here we need to take the inverse operation i.e., the exponential operation.

$$\therefore g(x, y) = e^{s(x, y)} = e^{i'(x, y)} \cdot e^{r'(x, y)}$$
$$= i_0(x, y) \cdot r_0(x, y)$$

where $i_0(x, y) = e^{i'(x, y)}$

$$r_0(x, y) = e^{r'(x, y)}$$

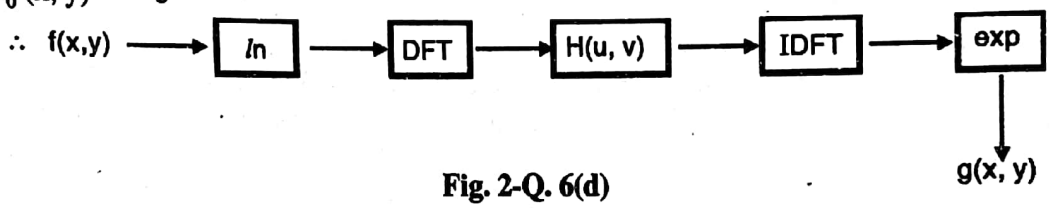


Fig. 2-Q. 6(d)

The filter H(u, v) can be implemented by a simple formula,

$$H(u, v) = (Y_H - Y_L)\ [1 - e^{-C\,(D^2(u, v)/D_0^2)}] + Y_L \quad ...(1)$$

C is a constant, used to control the sharpness.

Chapter 7 : Image Segmentation [Total Marks-15]

Q. 3(b) What is segmentation ? Explain (i) Region Growing (ii) Region Splitting (iii) Thresholding **(10 Marks)**

Ans. :

Segmenatation

Segmentation, as the name suggests, subdivides (segments) an image into its constituent regions or objects. Unlike image enhancement, were our primary concern was to improve the subjective quality of images for display, in segmentation, we address some aspects of analyzing the content of an image. Segmentation is used when we need to automate a particular activity. The ultimate aim in an automated system is to extract important features from image data, from which description, interpretation, or understanding of the scene can be provided by the machine.

(i) Region Growing

As the name implies, region growing is a procedure in which pixels are grouped into larger regions based on some predefined condition. The basic approach is to select a seed point (a pixel from where we begin) and grow regions from this seed pixel.

Let us pick up an arbitrary pixel (x_1, y_1) from the image that needs to be segmented. This pixel is

called the seed pixel. We now examine the nearest neighbours (4 or 8 neighbours depending on whether we assume 4-connectivity or 8-connectivity) of (x_1, y_1) one by one. The neighbouring pixel is accepted in the same region as (x_1, y_1) if they together satisfy the homogeneity property of a region i.e., if both of them satisfy the predefined condition.

Once a new pixel (x_2, y_2) is accepted as a member of the current region, the neighbours of this new pixel are examined (again, 4 or 8 neighbours, depending on the connectivity assumed).

This procedure goes on recursively until no new pixel is accepted. All the pixels of the current region are given a unique label. Now a new seed pixel is chosen and the same procedure is repeated. We continue doing this until all the pixels are assigned to some region or the other.

(ii) Region splitting

In this case we try to satisfy the homogeneity property where pixels that are similar are grouped together. If the grey levels present in the region do not satisfy the property, we divide the region into four equal quadrants. If the property is satisfied, we leave the region as it is. This is done recursively until all the regions satisfy the property. To explain this in terms of graph theory, we call each region a node. This node (parent node) is split into its four children (leaf nodes) if it does not satisfy the given property. If the node satisfies the property, it is left as it is. We now check each leaf node and see if these leaf nodes satisfy the given property. If yes, they are left unaltered and if no, there are further split.

This particular splitting technique has a convenient representation in the form of a quad tree. (That is, a tree in which a node has exactly four descendents).

Consider the figure shown below

R_1		R_2
	R_{41}	R_{42}
R_3	R_{43}	R_{44}

In this, image R represents the entire image and hence R is the parent node. This parent node is split up into four leaf nodes, R_1, R_2, R_3 and R_4. Of these leaf nodes only R_4 does not contain pixels which satisfy some common property and hence is split again i.e., R_{41}, R_{42}, R_{43}, R_{44}. The quad tree for this division is shown in Fig. 1-Q. 3(b).

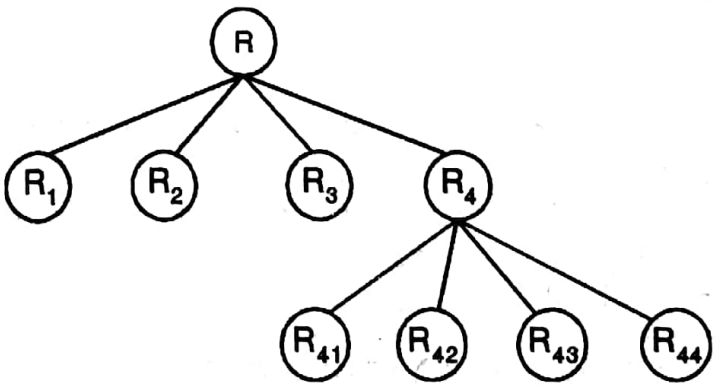


Fig. 1-Q. 3(b)

This method is applicable to images whose number of rows and number of columns are some integer power of 2.

(iii) Thresholding

Thresholding is one of the most important techniques for segmentation and because of its simplicity, is widely used. In segmentation, thresholding is used to produce regions of uniformity within the given image based on some threshold criteria T.

Thresholding is used to produce regions of uniformity within the given image based on some threshold criteria T.

Suppose we have an image which is composed of dark objects of varying grey levels against a light background of varying grey levels as shown in Fig. 2-Q. 3(b).

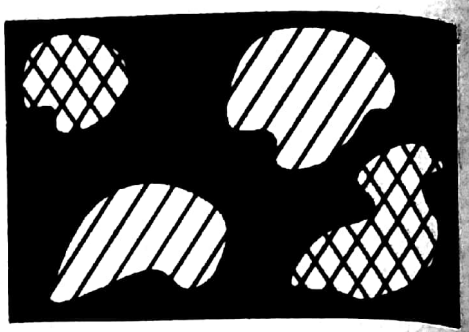


Fig. 2-Q. 3(b)

The histogram of such an image would appear more or less as shown in Fig. 3-Q. 3(b).

An obvious way to extract the objects from the background is to select a threshold T that separates the two regions. Then a point (x, y) for which $f(x, y) < T$ is called an object point and for $f(x, y) > T$, it is called a background point.

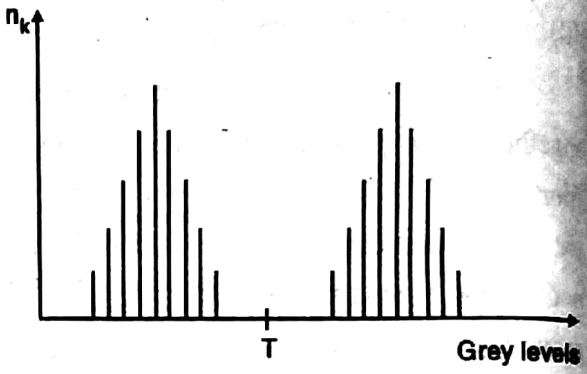


Fig. 3-Q. 3(b)

Similarly we could have images whose histograms look like the one shown in Fig. 4-Q. 3(b).

i.e., an image with two different types of dark objects (large variation in their grey levels) against a bright background. In such a case, we would require two threshold values, T_1 and T_2 to separate the two objects from their background. This is known as multilevel thresholding.

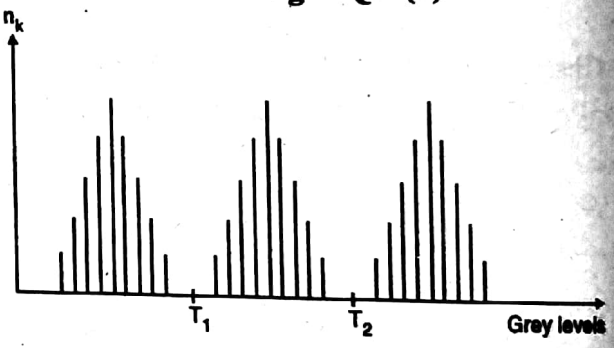


Fig. 4-Q. 3(b)

The thresholding operation can thus be thought of as a test involving the function T and can be defined as

$$T = T\{x, y, A(x, y), f(x, y)\}$$

Here, f(x, y) is the grey level of the pixel at (x, y) and A(x, y) denotes some local property in the neighbourhood of this image.

Q. 6(b) Write short note on : Edge linking using Hough transform. **(5 Marks)**

Ans. :

Edge linking using Hough transform

The gradient operators like Roberts, Sobel, Prewitts as well as the Compass operators enhance the edges. When we implement these filters practically, there are usually breaks in lines. Due to noise, there are spurious intensity discontinuties because of which the output of a gradient filtered image would have pixels that lie on the edge as well as pixels that represent noise.

Due to this fact, edge detection algorithms are generally followed by edge linking procedures which form edges (lines) from edge pixels. This is done by joining the edge pixels.

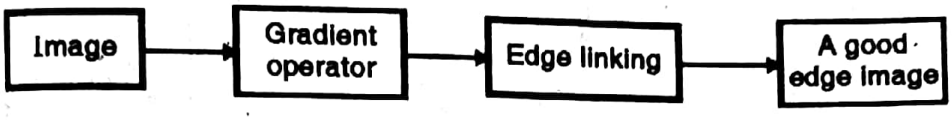


Fig. 1-Q. 6(b)

Hough transform has emerged over the past decade as a method of choice for pixel linking and curve detection. Hough presented this transform in the year 1962.

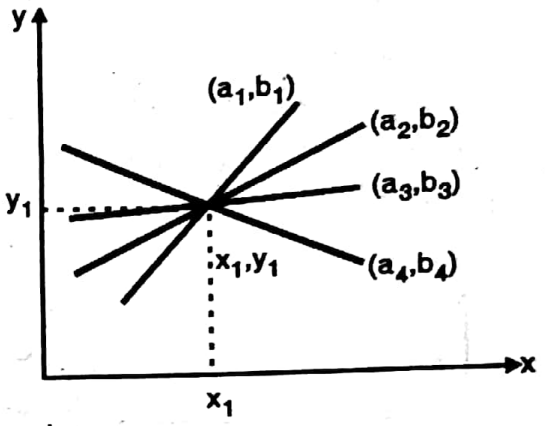
Given a set of discrete pixels, the Hough transform checks if these points lie on a straight line and if yes, it draws a line joining all these points.

Hough transform is best explained by an example. Consider a point (x_1, y_1). A line passing through this point can be written in the slope-intercept form as $y_1 = ax_1 + b$.

Using this equation and varying the values of a and b, infinite number of lines pass through this point (x_1, y_1).

However, if we write this equation as

$$b = -ax_1 + y_1$$



and consider the ab plane instead of the xy plane, we get a single line for a point (x_1, y_1)

This entire line in the ab plane is due to a single point in the xy plane and different values of a and b.

Now consider another point (x_2, y_2) in the xy plane.

The slope intercept equation of this line is

$$y_2 = ax_2 + b$$

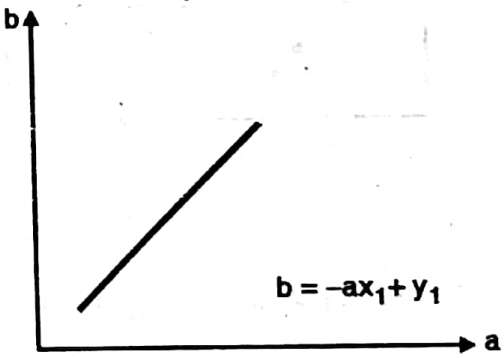


Fig. 3-Q. 6(b)

Writing this equation in terms of the ab plane we get

$$b = -ax_2 + y_2$$

This is another line in the ab plane. These two line will intersect each other somewhere in the ab plane only if they are a part of a straight line in the xy plane.

The point of intersection in the ab plane is noted (a′, b′).

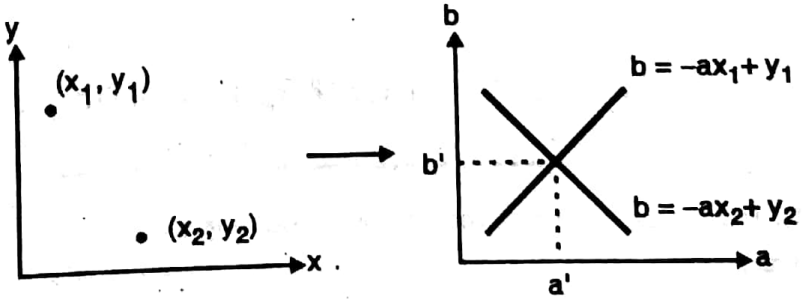


Fig. 4-Q. 6(b)

Using this (a′, b′) in the standard slope-intercept form i.e., $y = a'x + b'$, we get a line that passes through points (x_1, y_1) and (x_2, y_2) in the xy plane!!!!

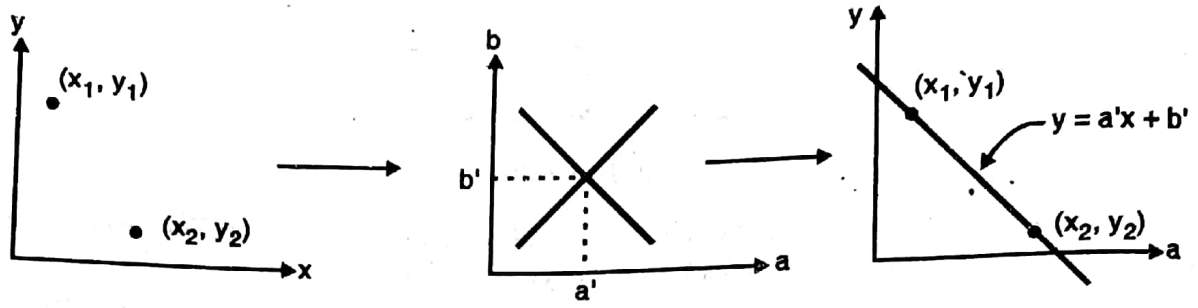


Fig. 5-Q. 6(b)

In fact if we have n points that lie on a straight line, we get n lines (each line representing a point) in the ab plane and all these lines would intersect at a single point (a′, b′) in the ab plane. Using these values of a′ and b′ in equation $y = a'x + b'$, we would get a line that would pass through all these points in the xy plane.

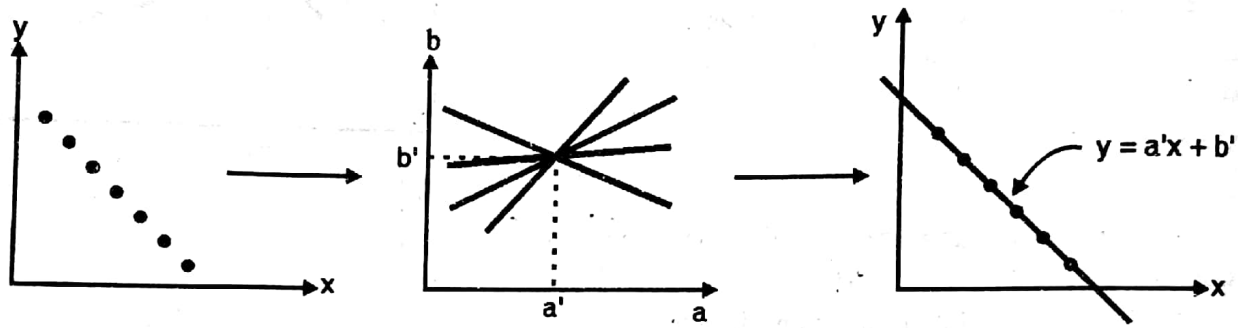


Fig. 6-Q. 6(b)

The ab plane is called the parameter space.

This parameter space is subdivided into accumulator cells as shown.

Here (a_{max}, a_{min}) and (b_{max}, b_{min}) are the expected ranges of slope and intercept values.

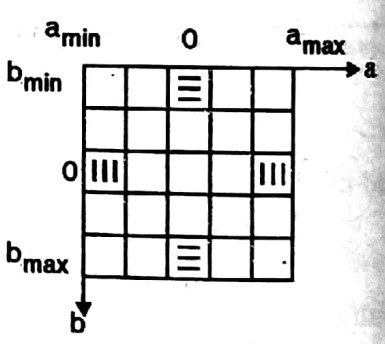


Fig. 7-Q. 6(b)

Chapter 8 : Image Morphology, Representation and Description [Total Marks-20]

Q. 1(d) Explain basic morphological operations. **(5 Marks)**

Ans. :

Basic morphological operations

(i) **Dilation :** Let A(x, y) be a binary image and C(x, y) be the resulting image obtained after A(x, y) has been dilated with a m × n template B(i, j). B is called the structuring element where $0 \le i \le m-1, 0 \le j \le n-1$

B is usually a binary structure. For dilation

$$C(x, y) = \text{Maximum } \{A(x - i, y - j) \times B(i, j)\} \quad \ldots(1)$$

This simply means that if B is a binary structuring element, the output pixel is the maximum value of all the pixels in the input pixels neighbourhood multiplied by the corresponding coefficients of the structuring element. Since the structuring element is usually a binary mask, for a binary image, if any of the pixels has a value 1, the output pixel is set to 1.

Fig. 1-Q. 1(d) illustrates the dilation of a binary image. The structuring element that we use is

1	(1)	1

Consider the image shown.

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

Dilation is achieved as shown.

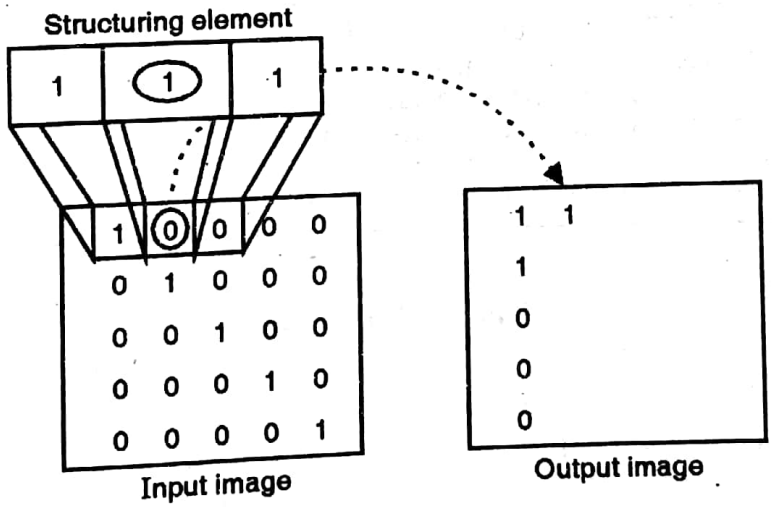


Fig. 1-Q. 1(d)

(ii) **Erosion :** In a similar fashion, we can achieve erosion by using a simple formula which could be used in programming.

If A(x, y) is the image and B(i, j) is the structuring element, then C(x, y) which is the eroded image is given by

$$C(x, y) = \text{Minimum } \{A(x - i, y - j) \times B(i, j)\} \quad \ldots(2)$$

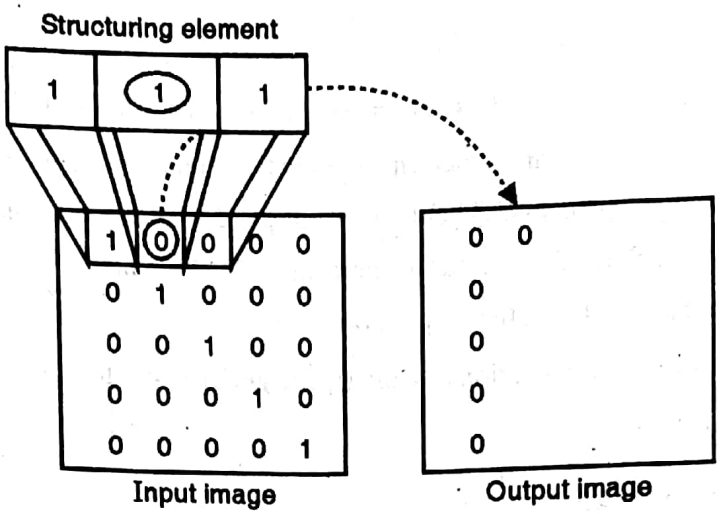


Fig. 2-Q. 1(d)

The Fig. 2-Q. 1(d) shows the erosion operation. These formulae can be used only if all the coefficients of the structuring element are equal to 1. Dilation and Erosion can also be achieved on grey level images. Fig. 3-Q. 1(d) and Fig. 4-Q. 1(d) explain dilation and erosion on grey level images.

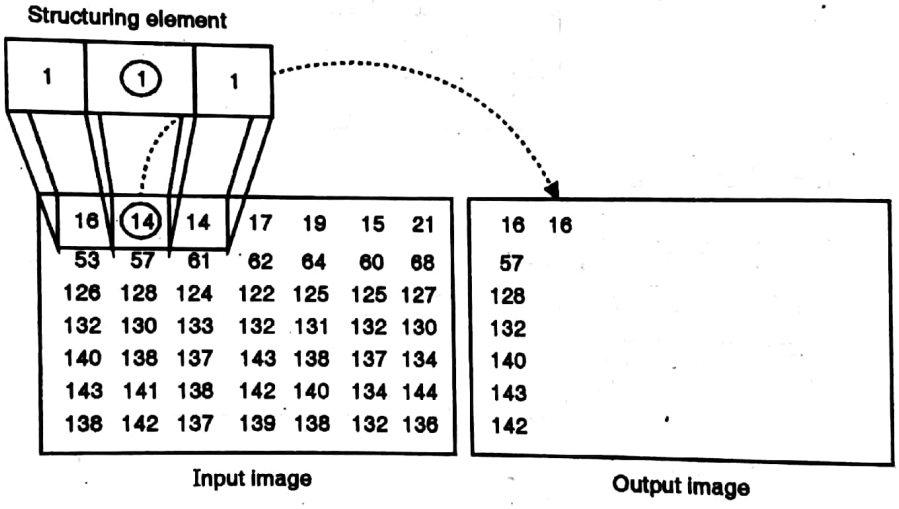


Fig. 3-Q. 1(d) : Dilation

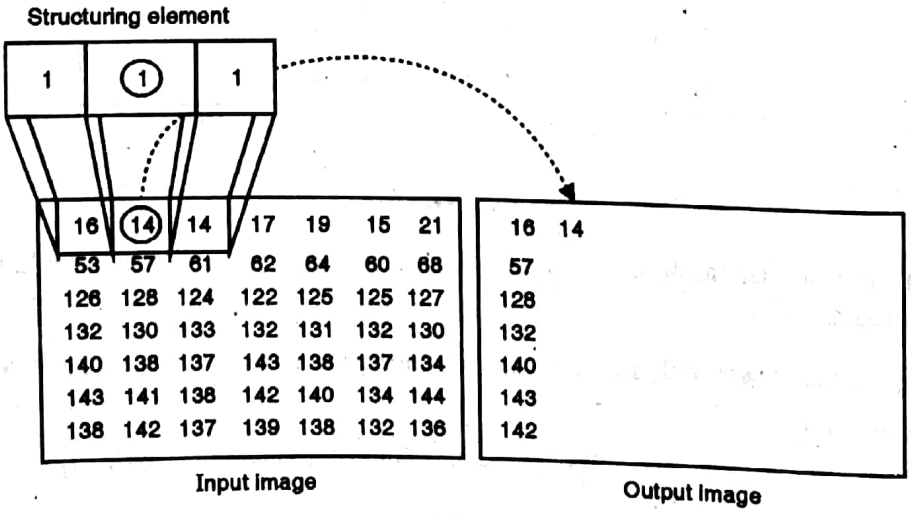


Fig. 4-Q. 1(d) : Erosion

Q. 4(a) Explain : The first difference makes the chain code invariant to rotation. **(10 Marks)**

Ans. :

Consider a simple boundary

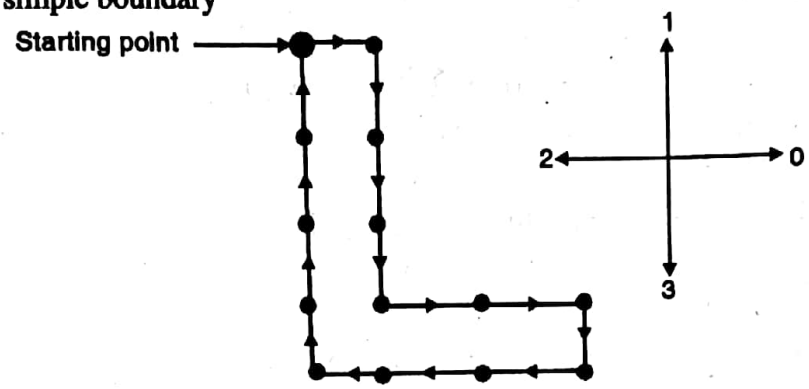


Fig. 1-Q. 4(a)

The 4-directional chain code (clockwise direction) is

0 3 3 3 0 0 3 2 2 2 1 1 1 1

Its first difference is

3 0 0 1 0 3 3 0 0 3 0 0 0 ...(1)

We shall now rotate the boundary

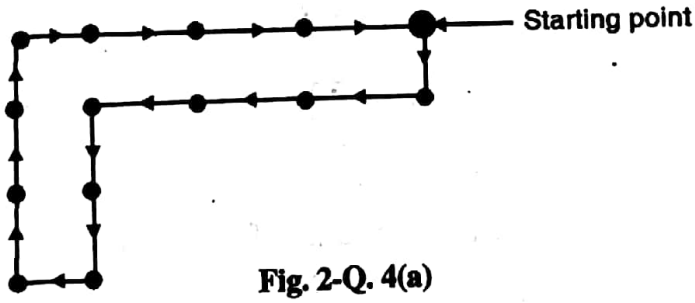


Fig. 2-Q. 4(a)

The 4-directional chain code is

3 2 2 2 3 3 2 1 1 1 0 0 0 0

It's first difference is

3 0 0 1 0 3 3 0 0 3 0 0 0 ...(2)

We see that 1 = 2

We can thus conclude that the first difference of the original object and the first difference of the object after rotation remain the same.

Hence the first difference of the chain code normalises it is rotation.

Q. 6(a) Write short note on : Moments with examples **(5 Marks)**

Ans. :

Moments

The theory of moments provides an interesting method of describing the properties of an object in terms of its area, position, and orientation parameters. The idea of moments is borrowed from physics.

Moments are area descriptors used to characterize the shape and size of the image. To do so we need to compute some sequence of numbers, which are called moments. These moments identify the

shape of the object such as area, centroid, moment of inertia, orientation etc.

Here the image is taken as an object and the grey level of a pixel is considered as the mass at a point of the object. Thus for a N × N image, the $(i, j)^{th}$ moment of the image f(x, y) is defined as

$$m(i, j) = \sum_x \sum_y x^i y^j f(x, y) \quad \ldots(1)$$

Now,

$$m(0, 0) = \sum_x \sum_y x^0 y^0 f(x, y)$$

$$m(0, 0) = \sum_x \sum_y f(x, y) \quad \ldots(2)$$

Hence moment m(0, 0) is the mass of the object. Similarly,

$$m(1, 0) = \sum_x \sum_y x^1 y^0 f(x, y)$$

$$m(1, 0) = \sum_x \sum_y x\, f(x, y) \quad \ldots(3)$$

$$m(0, 1) = \sum_x \sum_y x^0 y^1 f(x, y)$$

$$m(0, 1) = \sum_x \sum_y y\, f(x, y) \quad \ldots(4)$$

Here m(1, 0) and m(0, 1) are called the first order moments.

Any region-based feature requires a datum point from which further features can be derived. The centroid which is the centre of area, (the centre of mass) is a good parameter for specifying the location of the object.

Let the coordinates of the centroid be x′ and y′ such that the sum of the square of the distance from this point to all the other boundary points within the object is minimum.

Here,

$$x' = \frac{m(1, 0)}{m(0, 0)} \text{ and } y' = \frac{m(0, 1)}{m(0, 0)} \quad \ldots(5)$$

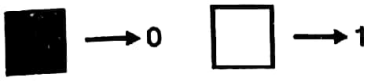
Hence (x′, y′) which are the coordinates of the centroid are $\left\{\frac{m(1, 0)}{m(0, 0)}, \frac{m(0, 1)}{m(0, 0)}\right\}$

Moments m(2, 0) and m(0, 2) are the moments of inertia of the object

$$m(2, 0) = \sum_x \sum_y x^2 f(x, y)$$

$$m(0, 2) = \sum_x \sum_y y^2 f(x, y)$$

Example : Compute all the moments upto second order



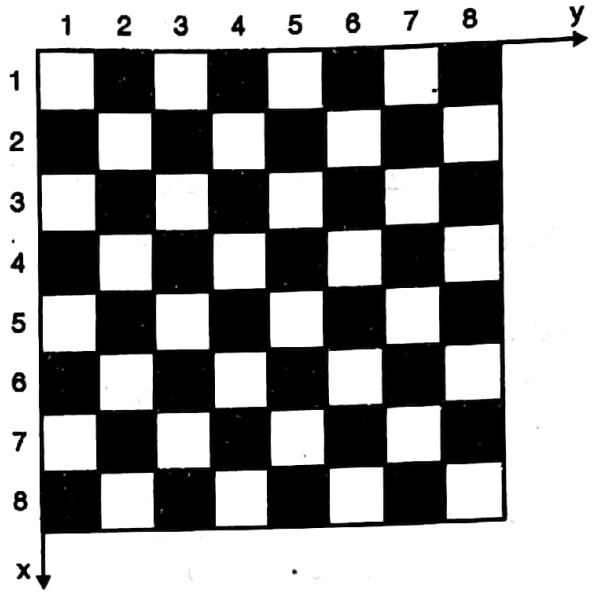


Fig. 1-Q. 6(a)

Moments are given by the formula

$$m(i, j) = \sum_x \sum_y x^i y^j f(x, y)$$

$$m(0, 0) = \sum_x \sum_y x^0 y^0 f(x, y) = 32$$

$$m(1, 0) = \sum_x \sum_y x^1 y^0 f(x, y) = 144$$

$$m(0, 1) = \sum_x \sum_y x^0 y^1 f(x, y) = 144$$

$$m(1, 1) = \sum_x \sum_y x^1 y^1 f(x, y) = 656$$

$$m(2, 0) = \sum_x \sum_y x^2 y^0 f(x, y) = 816$$

$$m(0, 2) = \sum_x \sum_y x^0 y^2 f(x, y) = 816$$

$$m(2,2) = \sum_x \sum_y x^2 y^2 f(x,y) = 21456$$

The coordinates of the centroid are given by,

$$(x', y') = \left\{\frac{m(1,0)}{m(0,0)}, \frac{m(1,0)}{m(0,0)}\right\}$$

$$(x', y') = \{4.5, 4.5\}$$

Chapter 9 : Image Transforms [Total Marks-25]

Q. 1(b) Discuss unitary matrix with example. **(5 Marks)**

Ans. :

Unitary matrix

A matrix is said to be unitary is it complex conjugate is the same as its inverse. Hence a matrix T is unitary if.

$$T^{*\prime} = T^{-1}$$

Multiplying both sides by T, we get,

$$T \cdot T^{*\prime} = T \cdot T^{-1} \quad \ldots(1)$$

$$\therefore T \cdot T^{*\prime} = I \quad \ldots(2)$$

Where * indicates complex conjugate of T.

Hence to verify whether T is a unitary matrix we need to use the relationship $TT^{*\prime} = I$.

We shall now check if the DFT matrix is unitary or not.

We know that the Fourier transform can be represented as F = Wf where f is the input and W is the DFT matrix. Taking N = 4, we form a DFT matrix.

$$W = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

To check whether W is unitary or not, we need to check the relationship.

$$W \cdot W^{*\prime} = I$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} = \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}$$

$$W W^{*\prime} = 4[I]$$

Hence the DFT is an example of a unitary transform. 4 indicates that the DFT matrix is not normalised.

Q. 3(a) Find the DCT of the given image using matrix multiplication method : **(10 Marks)**

0	3	3	1
3	1	2	1
3	2	4	2
1	1	2	1

Ans. :

We use the matrix notation

$$F = C f C'$$

Where C is a 4×4 DCT matrix

$$F = \begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ 0.653 & 0.2705 & -0.2705 & -0.653 \\ 0.5 & -0.5 & -0.5 & 0.5 \\ 0.2705 & -0.653 & 0.653 & -0.2705 \end{bmatrix} \begin{bmatrix} 0 & 3 & 3 & 1 \\ 3 & 1 & 2 & 1 \\ 3 & 2 & 4 & 2 \\ 1 & 1 & 2 & 2 \end{bmatrix} \begin{bmatrix} 0.5 & 0.653 & 0.5 & 0.2705 \\ 0.5 & 0.2705 & -0.5 & -0.653 \\ 0.5 & -0.2705 & -0.5 & 0.653 \\ 0.5 & -0.653 & 0.5 & -0.2705 \end{bmatrix}$$

$$F = \begin{bmatrix} 7.5 & 0.11 & -1.5 & 1.57 \\ 0.11 & 0 & -1.03 & -0.71 \\ -1.5 & -1.03 & -1.5 & -1.19 \\ 1.57 & -0.71 & -1.19 & 0 \end{bmatrix}$$

Q. 4(b) Write 8 × 8 HADAMARD transform matrix and its signal flow graph. Using Butterfly diagram, compute HADAMARD transform for x(n) = {1,2,1,2,1,2,3,4}. **(10 Marks)**

Ans. :

8×8 Hadamard transform matrix is shown below

$$H(8) = \left[\begin{array}{cccc:cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ \hdashline 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{array}\right]$$

Sign changes
0
7
3
4
1
6
2
5

Since N = 8, we use 8 point Butterfly diagram,

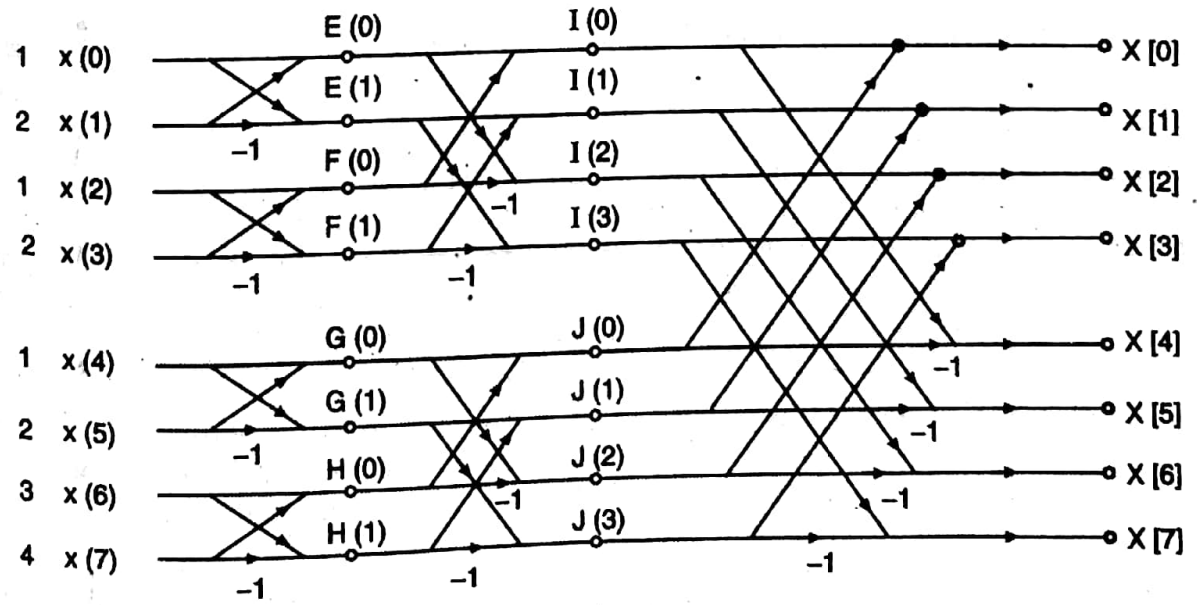


Fig. 1-Q. 4(a)

$$E(0) = x(0) + x(1) = 1 + 2 = 3$$
$$E(1) = x(0) - x(1) = 1 - 2 = -1$$
$$F(0) = x(2) + x(3) = 1 + 2 = 3$$
$$F(1) = x(2) - x(3) = 1 - 2 = -1$$
$$G(0) = x(4) + x(5) = 1 + 2 = 3$$
$$G(1) = x(4) - x(5) = 1 - 2 = -1$$
$$H(0) = x(6) + x(7) = 3 + 4 = 7$$
$$H(1) = x(6) - x(7) = 3 - 4 = -1$$
$$I(0) = E(0) + F(0) = 3 + 3 = 6$$
$$I(1) = E(1) + F(1) = -1 - 1 = -2$$
$$I(2) = E(0) - F(0) = 3 - 3 = 0$$
$$I(3) = E(1) - F(1) = -1 - (-1) = 0$$
$$J(0) = G(0) + H(0) = 3 + 7 = 10$$
$$J(1) = G(1) + H(1) = -1 - 1 = -2$$
$$J(2) = G(0) - H(0) = 3 - 7 = -4$$
$$J(3) = G(1) - H(1) = -1 - (-1) = 0$$
$$X[0] = [I(0) + J(0)] = (6 + 10) = 16$$
$$X[1] = [I(1) + J(1)] = (-2 - 2) = -4$$
$$X[2] = [I(2) + J(2)] = (0 - 4) = -4$$
$$X[3] = [I(3) + J(3)] = (0 + 0) = 0$$
$$X[4] = [I(0) - J(0)] = (6 - 10) = -4$$
$$X[5] = [I(1) - J(1)] = (-2 - (-2)) = 0$$
$$X[6] = [I(2) - J(2)] = (0 - (-4)) = 4$$
$$X[7] = [I(3) - J(3)] = (0 - 0) = 0$$
$$\therefore \quad X[n] = \{16, -4, -4, 0, -4, 0, 4, 0\}$$

Chapter 11 : Image Compression [Total Marks-25]

Q. 5(a) Classify image compression methods in detail along with the different redundancies that can be present in digital images. **(10 Marks)**

Ans. :

Images in their raw from occupy a lot of memory space and hence data compression becomes necessity while dealing with them. Data compression basically tries to reduce the overall size of data. This reduction is possible when the original dataset contains some type of unwanted data redundancy. Digital images contain a lot of data which provide no relevant information, also called redundant data which can be exploited to reduce the size of the image.

In general, three basic redundancies exist in digital images that follow.

1. **Inter-pixel Redundancy :** It is a redundancy corresponding to statistical dependencies or high correlation among neighbouring pixels. Hence in images, there exists large area having the same grey level. This is known as inter-pixel redundancy and can be reduced using Run Length encoding.
2. **Coding Redundancy :** In an uncompressed image, every pixel is coded with a fixed number of bits known as fixed length coding. For example, an image with 256 gray scales is represented by

an array of 8-bit integers. However this type of coding does not take into account the fact that some grey level occur more often than the others. This is known as coding redundancy. Huffman coding and arithmetic coding are used to reduce coding redundancy.

3. **Psycho-visual Redundancy :** Even though an image could have 256 grey levels, the Human eye can resolve very few grey levels locally. Hence Psycho-visual redundancy deals with the way humans perceive grey levels. Psycho-visual redundancy can be reduced by using IGS coding.

We shall explain one technique for each redundancy :

Run Length Encoding (RLE)

It is the simplest dictionary-based data compression technique. Image files frequently contain the same character repeated many times in a row. Images, particularly those having very few grey levels, often contain regions of adjacent pixels, all with the same grey level. Each row of such images can have long runs of the same grey value. In, such cases, one can store a code specifying the value of the grey level, followed by the length of the run, rather than storing the same value many times over.

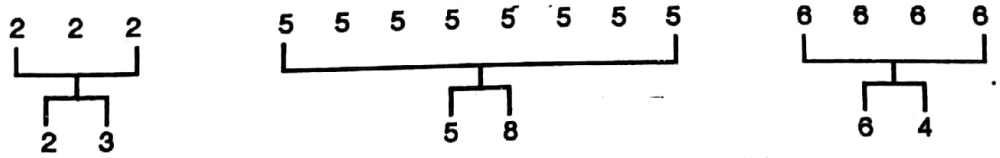
Let us take an example.

Consider the first row of an image which has the following grey values.

2	2	2	5	5	5	5	5	5	5	5	6	6	6	6
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•

The first row of the image has 15 grey values.

We could store the above row in a compact manner using RLE.



The first code specifies the grey value, followed by the length of the run.

Hence the RLE of the first row is simply.

2 3 5 8 6 4

As is evident, run length encoding achieves considerable compaction in images which have a fairly constant background. The RLE eliminates *Inter-pixel redundancies*.

Statistical coding

The images dealt with so far have been coded and saved in the natural code. That is for a 256 grey level image, grey level at f(x, y) is coded with its 8-bits binary equivalent. A grey level of a value 4 is coded as 0000 0100. Similarly for a 8 grey level image, grey value at f (x, y) is coded using 3-bits. The table that shows codes for a sampled image having grey levels varying from 0 to 7 (3-bit image) is given.

These codes are referred to as Equal Length Codes.

Input	Natural code
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Equal length codes do not make use of the statistics of data. The assumption is made that all grey levels have equal occurrence in the image. Since this possibility is unlikely, this is not an optimum for of coding. If we can develop a code such that fewer bits are assigned to grey levels having higher probability of occurrence and vice versa, we could reduce the number of bits required for transmission. Such a coding is known as Variable Length Codingor Entropy Coding and it eliminates *Coding redundancies.*

Here, we are not discarding any information as would be done in lossy compression, but simply represent more frequently occurring values with shorter codes and vice versa.

Consider an image containing L grey levels i = 0, 1 ..., L – 1. Let n_i denote the number of pixels having grey level i.

The probability of occurrence of grey level i in the image is

$$p_i = \frac{n_i}{\sum_{i=0}^{L-1} n_i} \quad \ldots(1)$$

Let us define entropy H associated with the grey level as

$$H = -\sum_{i=0}^{L-1} p_i \log_2 p_i \quad \ldots(2)$$

Entropy is the measure of randomness in the set of random variables. Entropy is never negative since p_i lies in the range [0, 1].

Details of the above equation can be found in any book on Information theory. When all variables are equally likely i.e., $p_0 = p_1 = p_2 ..... = p_{L-1} = 1/L$, then the randomness is maximized and the entropy attains its greatest value.

Improved Grey Scale (IGS) Quantization

The brightness of a region as perceived by the eye does not depend on the absolute grey level values (remember Mach Bands?). This is because the human eye does not respond with equal sensitivity to all visual information. Some information is visually more important than the others.

Information which is not visually important is called *Psychovisual Redundancy*. Psychovisual redundancies exist in all images and can be eliminated without hampering the subjective quality of the image. We have seen that simply reducing the quantization (number of bits for representation), compresses the image but also produces false contouring. I.G.S. coding reduces the quantization but also reduces false contouring.

Steps involved in I.G.S. coding are,

(1) Lower 4-bits of the preceding modified pixel are added to the present pixel.
(2) The new 4 MSB's of the present pixel are taken as the I.G.S. code.
(3) Repeat step 1 and 2 after moving on to a new pixel.
(4) If MSB is 1111, then add 0000 instead of the 4 LSB's of the previous sum.

Q. 5(b) Given **(10 Marks)**

$$f = \begin{bmatrix} 10 & 44 & 16 \\ 10 & 14 & 48 \\ 11 & 10 & 22 \end{bmatrix}$$

Find 3-bit IGS coded image and calculate compression factor, BPP and MSE.

Ans. :

The given image is 6-bit image. We begin by converting the pixel values to their binary equivalent.

Grey level	Binary	Sum	3 bit IGS code
		000000	
10	001010	001010	001
44	101100	101110	101
16	010000	010110	010
10	001010	010000	010
14	001110	001110	001
48	110000	110110	110
11	001011	010001	010
10	001011	001100	001
22	010110	011010	011

IGS code gives us 3-bits/pixels (3-BPP).

The original data was 6-bit while the IGS code gave us a 3-bit code. Hence IGS coding by default gives 50% compression.

To obtain the MSE we need to obtain the decoded image. We multiply the IGS code with 2^5, 2^4 and 2^3.

Grey level	IGS code	Decimal equivalent
10	001	$(0\times2^5)+(0\times2^4)+(1\times2^3)=8$
44	101	$(1\times2^5)+(0\times2^4)+(1\times2^3)=40$
16	010	$(0\times2^5)+(1\times2^4)+(0\times2^3)=16$
10	010	$(0\times2^5)+(1\times2^4)+(0\times2^3)=16$
14	001	$(0\times2^5)+(0\times2^4)+(1\times2^3)=8$
48	110	$(1\times2^5)+(1\times2^4)+(0\times2^3)=48$
11	010	$(0\times2^5)+(1\times2^4)+(0\times2^3)=16$

Grey level	IGS code	Decimal equivalent
10	001	$(0\times2^5)+(0\times2^4)+(1\times2^3)=8$
22	011	$(0\times2^5)+(1\times2^4)+(1\times2^3)=24$

Hence the decoded image is shown below

$$f'(x,y) = \begin{array}{|c|c|c|}\hline 8 & 40 & 16 \\ \hline 16 & 8 & 48 \\ \hline 16 & 8 & 24 \\ \hline \end{array}$$

The MSE is given by the formula,

$$MSE = \frac{1}{M\cdot N}\sum\sum[f(x,y)-f'(x,y)]$$

$$= \frac{1}{9}[(10-8)^2+(44-40)^2+(16-16)^2+(10-16)^2+(14-8)^2$$

$$+(48-48)^2+(11-16)^2+(10-8)^2+(22-24)^2]$$

$$MSE = 13.89$$

Q. 6(c) Write short note on : Fidelity criteria **(5 Marks)**

Ans. :

Error criteria

Error criteria is required to judge the performance of a lossy compression technique. It simply means how much of error is considered to be tolerable. The error criteria can be classified into two broad groups.

(1) Objective error (fidelity) criteria

(2) Subjective error (fidelity) criteria

Objective error criteria

This is a mathematical formulation to quantify the error introduced in Lossy compression. Let us denote the original image and the reconstructed image by f(x, y) and $\tilde{f}(x, y)$ respectively. Let the physical size of both the images be M × N.

The objective error criteria most often used is the mean-squared error.

This is defined as,

$$e_{ms} = \frac{1}{MN}\sum_{x=0}^{N-1}\sum_{y=0}^{M-1}\{f(x,y)-\tilde{f}(x,y)\}^2 \quad ...(1)$$

Sometimes the root mean squared error is preferred over the means squared error.

This defined as

$$e_{rms} = \left[\frac{1}{MN}\sum_{x=0}^{N-1}\sum_{y=0}^{M-1}\{f(x,y)-\tilde{f}(x,y)\}^2\right]^{1/2} \quad ...(2)$$

Since the value of the e_{rms} depends on the range of f(x, y), it is a practice to have the e_{rms} normalized. As a result of this normalisation,

$$e_{ms} = \frac{\sum_{x=0}^{N-1}\sum_{y=0}^{M-1}\{f(x,y) - \tilde{f}(x,y)\}^2}{\sum_{x=0}^{N-1}\sum_{y=0}^{M-1} f^2(x,y)} \quad \ldots(3)$$

A simple re-arrangement of terms gives us the Signal to Noise Ratio (SNR)

$$e_{SNR} \text{ (Mean squared error)} = \frac{\sum_{x=0}^{N-1}\sum_{y=0}^{M-1} f^2(x,y)}{\sum_{x=0}^{N-1}\sum_{y=0}^{M-1}\{f(x,y) - \bar{f}(x,y)\}^2} \quad \ldots(4)$$

$$e_{SNR} \text{ (Root mean squared error)} = \left[\frac{\sum_{x=0}^{N-1}\sum_{y=0}^{M-1} f^2(x,y)}{\sum_{x=0}^{N-1}\sum_{y=0}^{M-1}\{f(x,y) - \tilde{f}(x,y)\}^2}\right]^{1/2} \quad \ldots(5)$$

Here, $\{f(x, y) - \tilde{f}(x, y)\}$ is considered as noise introduced in the reconstructed signal. Though the fidelity criteria offers a simple mechanism to evaluate the error (information loss), it cannot distinguish between small error at many pixels and large error at few pixels. This consideration is important because the same value of error may produce a widely different visual effect. Since the compressed images are ultimately viewed by human beings, measuring image quality by subjective evaluations of a human observer is often more appropriate.

Subjective error criteria

The subjective error criteria is performed as follows. The original and the reconstructed images are shown to a large group of people. Each person assigns a grade to the reconstructed image. These grades are purely subjective. Finally, based on grades assigned by the entire group, an overall grade is assigned to the reconstructed image. Compliment of this grade gives an idea of the subjective error.

Chapter 12 : Colour Image Processing [Total Marks-05]

Q. 1(a) Explain any one color model. **(5 Marks)**

Ans. :

Colour models

Colour model is also known as colour space. Colour models are different ways in which colour information is stored. The most common and obvious colour model in the RGB colour model. The reason for having more than one colour model is that some operations are easier to implement if we move away from the RGB model. In the next section we discuss some important colour models

1. **RGB colour model**

As stated earlier, this is the most common format used in image processing.

The RGB colour space is shown in Fig. 1-Q. 1(a).

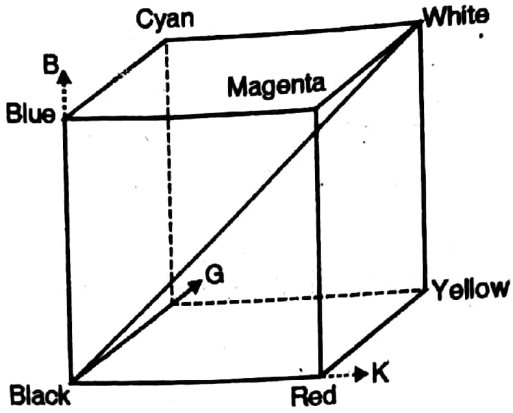


Fig. 1-Q. 1(a)

Each point in the image (pixel) is represented by a point in the first quadrant of the three-space as shown.

The origin of the RGB space (Red = 0, Green = 0, Blue = 0) represents black while the opposite corner (Red = max, Green = max, Blue = max) represents white.

A typical way of representing a RGB colour image is $M \times N \times 3$ where $M \times N$ is the physical size of the image and 3 represents the R, G, B planes i.e. an RGB image can be visualised as stack of 3 planes of size $M \times N$ each.

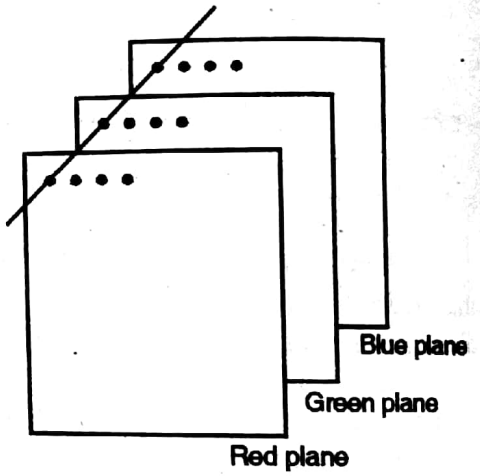


Fig. 2-Q. 1(a)

❑❑❑

Dec. 2015

Q. 1 Answer the following : **(20 Marks)**

(a) What do you understand by zero memory operation.

(b) Discuss different discontinuities in image.

(c) What is an Unitary matrix.

(d) Define Morphological operations Erosion and Dilation.

Q. 2 (a) Discuss color models for a digital image.

(b) For the given 3 bits per pixel, 4×4 size image perform following operations

(i) Intensity level slicing with background, $r_1 = 3$ and $r_2 = 5$

(ii) Bit plane slicing. **(10 Marks)**

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

Q. 3 (a) Explain : The first difference makes the chain code invaliant to rotation. **(10 Marks)**

(b) Explain Homomorphisc filtering with the help of block diagram. **(10 Marks)**

Q. 4 (a) Write 8×8 Hadamard transform matrix and its signal flow graph for fast Hadamard transform. Using this butterfly diagram (Signal flow graph) compute. Hadamard transform for $x(n) = \{1, 2, 1, 1, 3, 2, 1, 2\}$ **(10 Marks)**

(b) Find the DCT of the given Image using matrix multiplication method. **(10 Marks)**

$$f(x, y) \begin{bmatrix} 2 & 4 & 4 & 2 \\ 4 & 6 & 8 & 3 \\ 2 & 8 & 10 & 4 \\ 3 & 8 & 6 & 2 \end{bmatrix}$$

Q. 5 (a) Discuss the different types of redundancies in images with examples. **(10 Marks)**

(b) Construct Improved Gray Scale (IGS) quantization code for given gray scale data, {100, 110, 124, 124, 130, 200, 210}. Also compute e_{rms} (root mean square error). **(10 Marks)**

Q. 6 Write detail notes on (any two) **(20 Marks)**

(a) Edge Linking using Hough transform

(b) Thinning with example.

(c) Differential Pusle code Modulation (DPCM)

(d) Segmentation techniques : Region growing and split and merge.

❑❑❑

May 2016

Q. 1 (a) What is Unitary transform matrix? Explain with example. **(5 Marks)**

(b) Explain in short sampling and quantization method for digital image. **(5 Marks)**

(c) Explain in short morphological operations Dilation and Erosion. **(5 Marks)**

(d) Justify /contradict: All Image compression techniques are invertible. **(5 Marks)**

Q. 2 (a) Explain in detail any two types of Image File Formats. **(8 Marks)**

(b) For the 3 bit 4×4 size image perform following operations. **(12 Marks)**

i) Thresholding T = 3

ii) Intensity level slicing with background, $r_1 = 3$ and $r_2 = 5$

iii) Bit plane slicing for MSB and LB S planes

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

Q. 3 (a) Perform histogram equalization and draw new equalized histogram of the following image data. **(10 Marks)**

Gray level	0	1	2	3	4	5	6	7
No. of pixel	400	700	1350	2400	3000	1500	650	0

(b) Find Huffman code for the symbols given below. Which kind of redundancy is removed by Huffman code? Explain the term compression Ratio. **(10 Marks)**

Symbols	Probability
a_1	0.1
a_2	0.3
a_3	0.2
a_4	0.25
a_5	0.07
a_6	0.08

Q. 4 (a) Using matrix multiplication method calculate 2-D DFT of **(10 Marks)**

$$f(x, y) = \begin{bmatrix} 1 & 0 & 3 & 1 \\ 1 & 1 & 2 & 2 \\ 2 & 0 & 1 & 3 \\ 1 & 2 & 2 & 4 \end{bmatrix}$$

(b) Using the Butterfly diagram, compute Hadamard transform for

x(n) = { 1, 2, 3, 4, 1, 2, 1, 2 } **(10 Marks)**

Q. 5 (a) What are the different types of redundancies in digital image ? Explain in detail giving example of each. **(10 Marks)**

(b) What is image segmentation? Explain the following methods of image segmentation.

(i) Region growing

(ii) Split and Merge **(10 Marks)**

Q. 6 Write detail notes on (any two) **(20 Marks)**

(i) Hough Transform

(ii) Homomorphic filter

(iii) Hit or Miss Transform

(iv) Chain code

❑❑❑

Dec. 2016

Q. 1 (a) Explain any five zero memory operations. **(10 Marks)**

(b) Perform histogram equalization and draw new equalized histogram of the following image data. **(10 Marks)**

Grey level	0	1	2	3	4	5	6	7
Number of pixels	800	1000	850	650	300	250	100	150

Q. 2 (a) Find the DFT of the given image : **(10 Marks)**

0	3	3	1
3	1	2	1
3	2	4	2
1	1	2	1

(b) What is segmentation ? Explain (i) Region Growing (ii) Region splitting (iii) Thresholding **(10 Marks)**

Q. 3 (a) Explain with an example that the first difference of a chain code normalizes it to rotation. **(10 Marks)**

(b) Explain the following morphological operations : (i) Opening (ii) Closing **(10 Marks)**

Q. 4 (a) Classify image compression methods in detail along with the different redundancies that can be present in digital images. **(10 Marks)**

(b) What are various file formats ? Explain each in brief. **(10 Marks)**

Q. 5 (a) Given **(10 Marks)**

$$f = \begin{bmatrix} 10 & 44 & 16 \\ 10 & 14 & 48 \\ 11 & 10 & 22 \end{bmatrix}$$

(i) Find 3-bit IGS coded image and calculate compression factor and BPP.

(ii) Find the decoded image and calculate MSE and PSNR.

(b) Write 8 × 8 HADAMARD transform matrix and its signal flow graph. Using Butterfly diagram, compute HADAMARD transform for x(n) = {1,2,1,2,1,2,3,4}. **(10 Marks)**

Q. 6 (a) Write short note on : Discrete Cosine Transform **(20 Marks)**

(b) Write short note on : Hough transform

(c) Write short note on : HSI color model

(d) Write short note on : 4, 8 and m-connectivity

May 2017

Q. 1 Answer the following : **(20 Marks)**

(a) Explain any one color model.

(b) Discuss unitary matrix with example.

(c) Explain low pass spatial filtering.

(d) Explain basic morphological operations.

Q. 2 (a) Explain any five zero memory point operations. **(10 Marks)**

(b) Perform histogram equalization and draw new equalized histogram of the following image data. **(10 Marks)**

Grey level	0	1	2	3	4	5	6	7
Number of pixels	550	900	650	150	300	250	110	90

Q. 3(a) Find the DCT of the given image using matrix multiplication method : **(10 Marks)**

0	3	3	1
3	1	2	1
3	2	4	2
1	1	2	1

Q. 3(b) What is segmentation ? Explain (i) Region Growing (ii) Region Splitting (iii) Thresholding **(10 Marks)**

Q. 4 (a) Explain : The first difference makes the chain code invariant to rotation. **(10 Marks)**

(b) Write 8 × 8 HADAMARD transform matrix and its signal flow graph. Using Butterfly diagram, compute HADAMARD transform for x(n) = {1,2,1,2,1,2,3,4}. **(10 Marks)**

Q. 5 (a) Classify image compression methods in detail along with the different redundancies that can be present in digital images. **(10 Marks)**

(b) Given **(10 Marks)**

$$f = \begin{bmatrix} 10 & 44 & 16 \\ 10 & 14 & 48 \\ 11 & 10 & 22 \end{bmatrix}$$

Find 3-bit IGS coded image and calculate compression factor, BPP and MSE.

Q. 6 Write notes on (Any two) **(20 Marks)**

(a) Moments with examples

(b) Edge linking using Hough transform.

(c) Fidelity criteria

(d) Homomorphic filter

□□□