→ 2021 fall, 29 Assessment

Pradip Dhungana
2017/01

(1) Perform histogram equalization on the following 8-bit gray level image & also scale the intensity from 1 to 20.

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

⇒ Here,
max^m grey level value = 7

applying 2^k rule:- $2^0 = 1$
$2^1 = 2$
$2^2 = 4$
$2^3 = 8$

level

So, L=8 & grey range will be 0 to L-1.
So in this case,
grey level range = 0 to 7.

Now,

→

grey level (r_k)	No. of pixels with r_k (n_k)
0	0
1	3
2	4
3	2
4	2
5	1
6	1
7	3

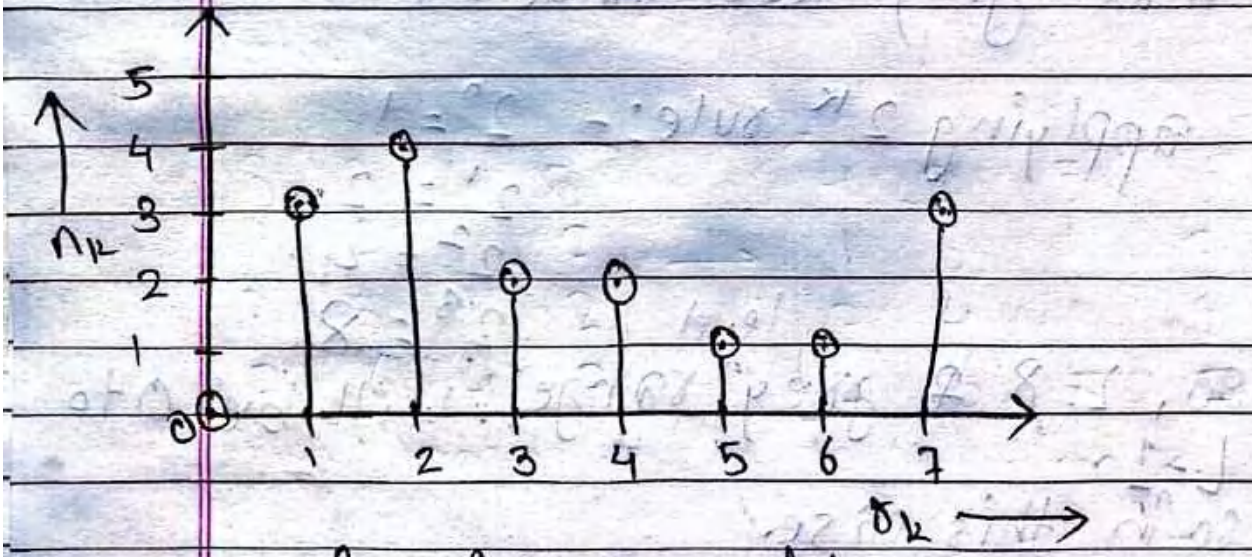


fig: histogram of Input image.

Now,

CDF = Cumulative Distribution function.

PDF = Probability Distribution function

n = total no. of Pixels.

since intensity gray level

Grey level (r_k)	no. of Pixel (n_k)	$P(r_k) = n_k/n$ (PDF)	S_k (CDF)	$(L-1)+1$	Histogram equalization level
				$1+(S_k \times 19)$	
0	0	0	0	0	0
1	3	0.1875	0.1875	1.3125	1
2	4	0.25	0.4375	3.0625	3
3	2	0.125	0.5625	3.9375	4
4	2	0.125	0.6875	4.8125	5
5	1	0.0625	0.75	5.25	5
6	1	0.0625	0.8125	5.6875	6
7	3	0.1875	1	7	7
	n=16				

After the Image is processed above

grey level (r_k)	no. of Pixels (n_k)
0	0
1	3
2	4
3	4
4	2
5	2+1=3
6	1
7	3

repeated

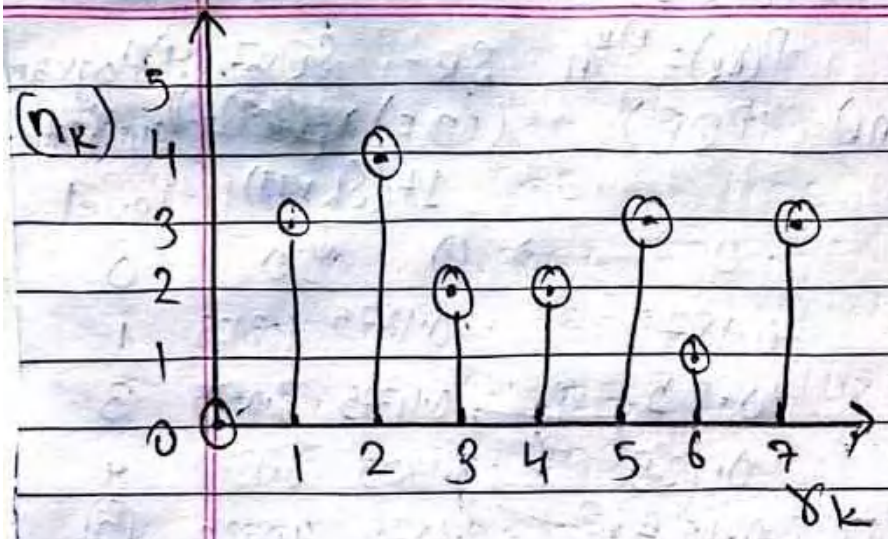


fig : Histogram of processed Image.

Now,

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

Input Image

→

4	3	5	5
1	3	1	3
7	4	1	3
7	6	5	7

output Image.

-> To scale the Intensity values from 1-20 ~~करने~~ करने,

~~New value = (old value) x 19 + 1~~

$(S_k \times (20-1)) + 1$

~~=~~

2023 spring: 1 (b)

Q) Compute the Histogram of Equilization of Given Data

Grey level (r_k)	no. of Pixels (n_k)
0	5320
1	1000
2	500
3	525
4	1236
5	956
6	856
7	128

Then,

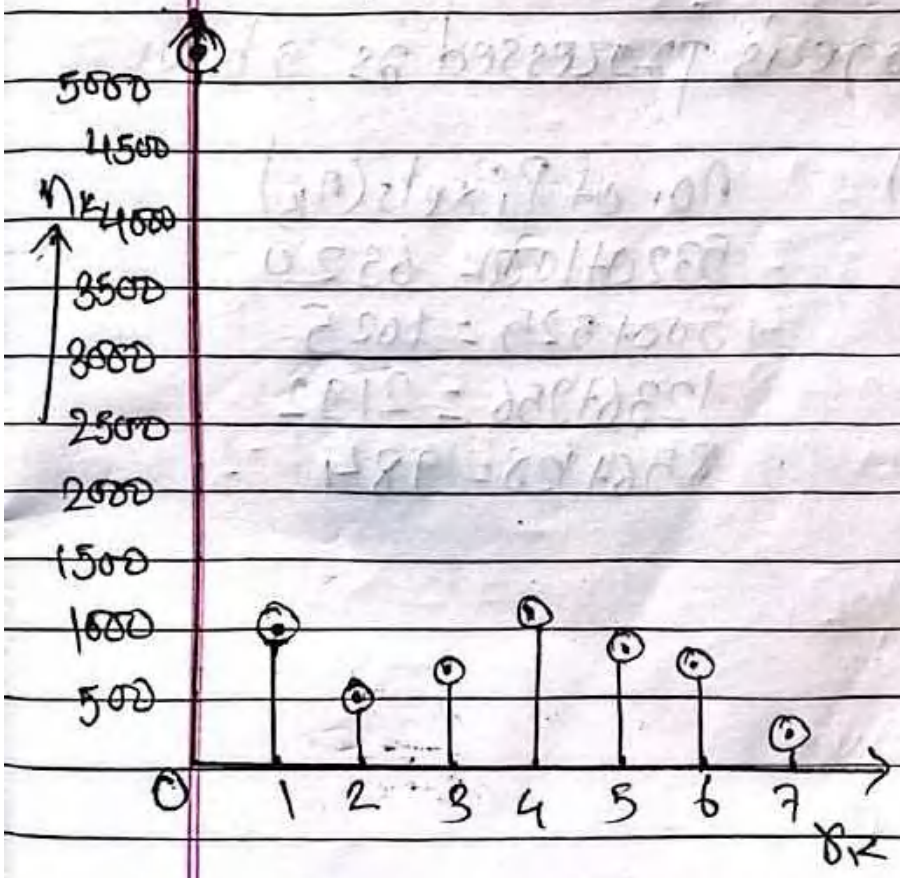


fig. Histogram of Input Image.

Now,

Grey level (r_k)	No. of Pixels (n_k)	$P(r_k) = n_k/n$ PDF	S_k (CDF)	$S_k \times 7$	Histogram Equalization level
0	5320	0.5056 →	0.5056	3.5392	4
1	1000	0.0950 ← →	0.6006	4.2042	4
2	500	0.0475 ← →	0.6481	4.5367	5
3	525	0.0499 ← →	0.698	4.886	5
4	1236	0.1174 ← →	0.8154	5.7078	6
5	956	0.0908 ← →	0.9062	6.3434	6
6	856	0.0813 ← →	0.3875	6.9125	7
7	128	0.0121 ← →	0.9996	6.9972	7
	n=10521				

After the image is processed as above,

Grey level (r_k)	No. of Pixels (n_k)
4	5320+1000 = 6320
5	500+525 = 1025
6	1236+956 = 2192
7	856+128 = 984

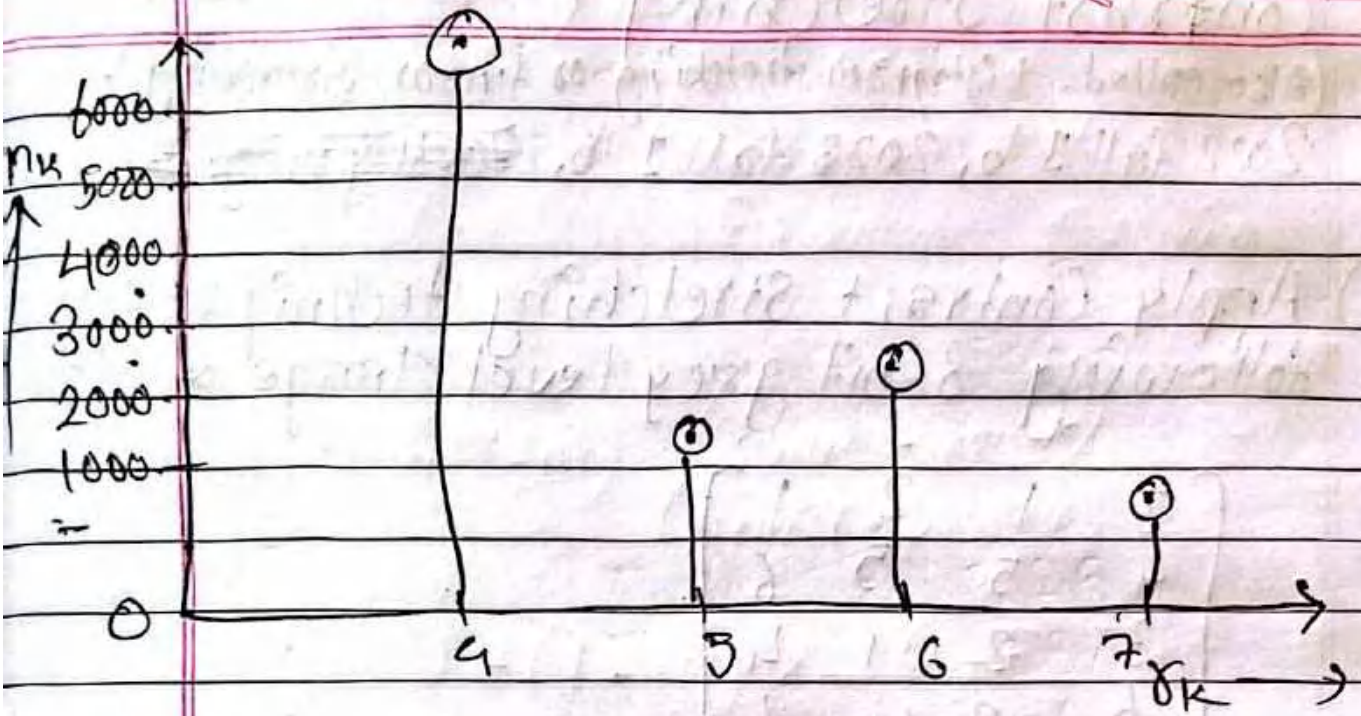


fig: Histogram of processed Image

(contrast stretching of given histogram)

Contrast Stretching

(Also called, histogram stretching or linear stretching).

2022 fall 1 b, 2023 fall 1 b, ~~2024 spring 2 a~~

Q) Apply Contrast Stretching technique on following 3-bit grey level Image of Size

$$\begin{bmatrix} 2 & 1 & 2 & 1 \\ 4 & 5 & 5 & 6 \\ 3 & 2 & 1 & 4 \\ 6 & 2 & 1 & 6 \end{bmatrix}$$

⇒ Here,

grey level value = 6

Applying 2^K rule:-

$$2^0 = 1$$
$$2^1 = 2$$
$$2^2 = 4$$
$$2^3 = 8$$

So, L=8 & Grey level range will be 0 to L-1.

Here,

minimum value,

$m_{min} = 1$

maximum value,

$m_{max} = 6$

Now, for a 3-bit Image, range of grey level = 0 to L-1, so $2^3 = 8$
i.e. 0 to 7

Now,
New minimum (l_{min}) = 0
New maximum (l_{max}) = 7

Grey level (r_k)	No. of Pixels (n_k)
0	0
1	4
2	4
3	1
4	2
5	2
6	3
7	0

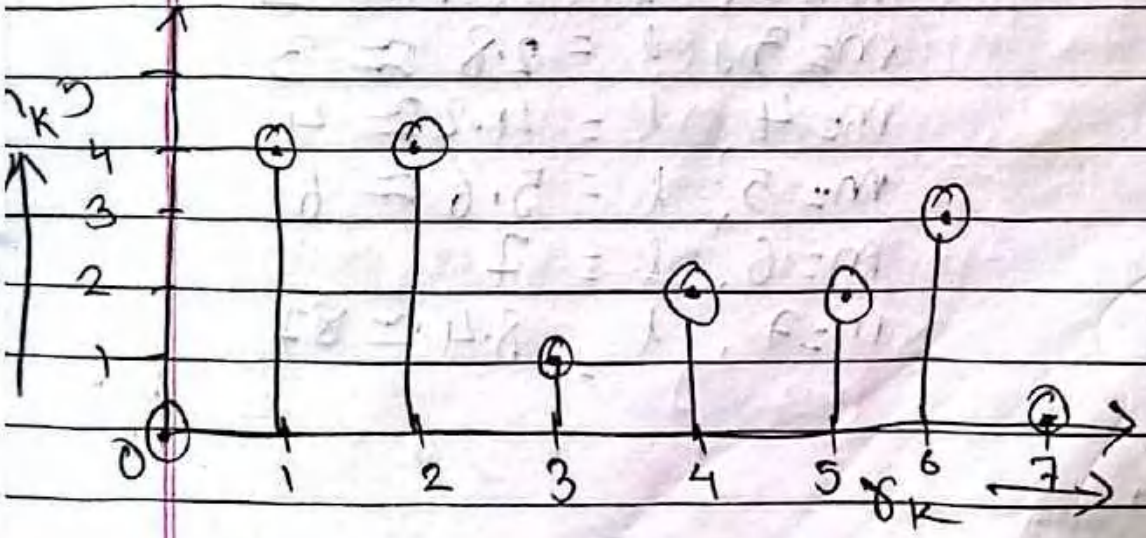


fig: Histogram of Input Image.

Now,

$m_{min} = 1 \quad m_{max} = 6$

$l_{min} = 0 \quad l_{max} = 7$

Then,

$$l = \frac{(l_{max} - l_{min})}{(m_{max} - m_{min})}(m - m_{min}) + l_{min}$$

$$= \frac{7-0}{6-1} \times (m-1) + 0$$

$$\therefore l = \frac{7(m-1)}{5}$$

Also, when $m = 0,\ l = -1.4 \simeq 0$

$m = 1,\ l = 0$

$m = 2,\ l = 1.4 \simeq 1$

$m = 3,\ l = 2.8 \simeq 3$

$m = 4,\ l = 4.2 \simeq 4$

$m = 5,\ l = 5.6 \simeq 6$

$m = 6,\ l = 7$

$m = 7,\ l = 8.4 \simeq 8$

Now,

With these transformation,

r_k	n_k → अगाडि को value of pixel हरु अनि कुन कुन no. को repeat भाको छ भनेर।
0	0+4=4
1	4
2	0 (यो नलिखदा हिन्छ।)
3	1
4	2
6	3
7	3+0=3
8	

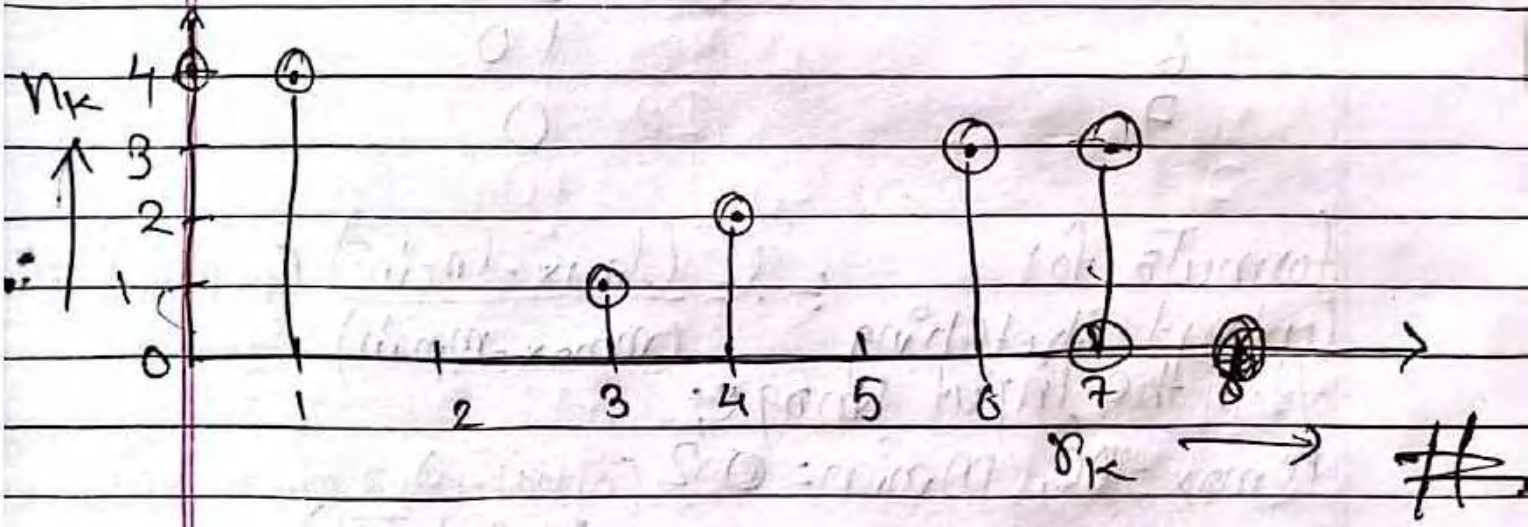


fig: Histogram of processed image.

2024 spring 2a)

(b) Gray level histogram of an image represented in 3 bit system is given below. stretch the contrast of histogram over the entire range.

=> Gray level (r_k) | frequency (n_k)

r_k	n_k
0	0
1	0
2	50
3	60
4	50
5	20
6	10
7	0

formula for Contrast Stretching : $s = \frac{(l_{max} - l_{min})}{(m_{max} - m_{min})} (m - m_{min}) + l_{min}$

from the input image:

$m_{max} = 6$, $m_{min} = 2$ (0 गर्ने max र min का पर्दैन, n_k बाट लिने)

Since this is a 3-bit system,

so, $2^3 = 8$

range = 0 to 7

$\therefore l_{min} = 0$

$l_{max} = 7$

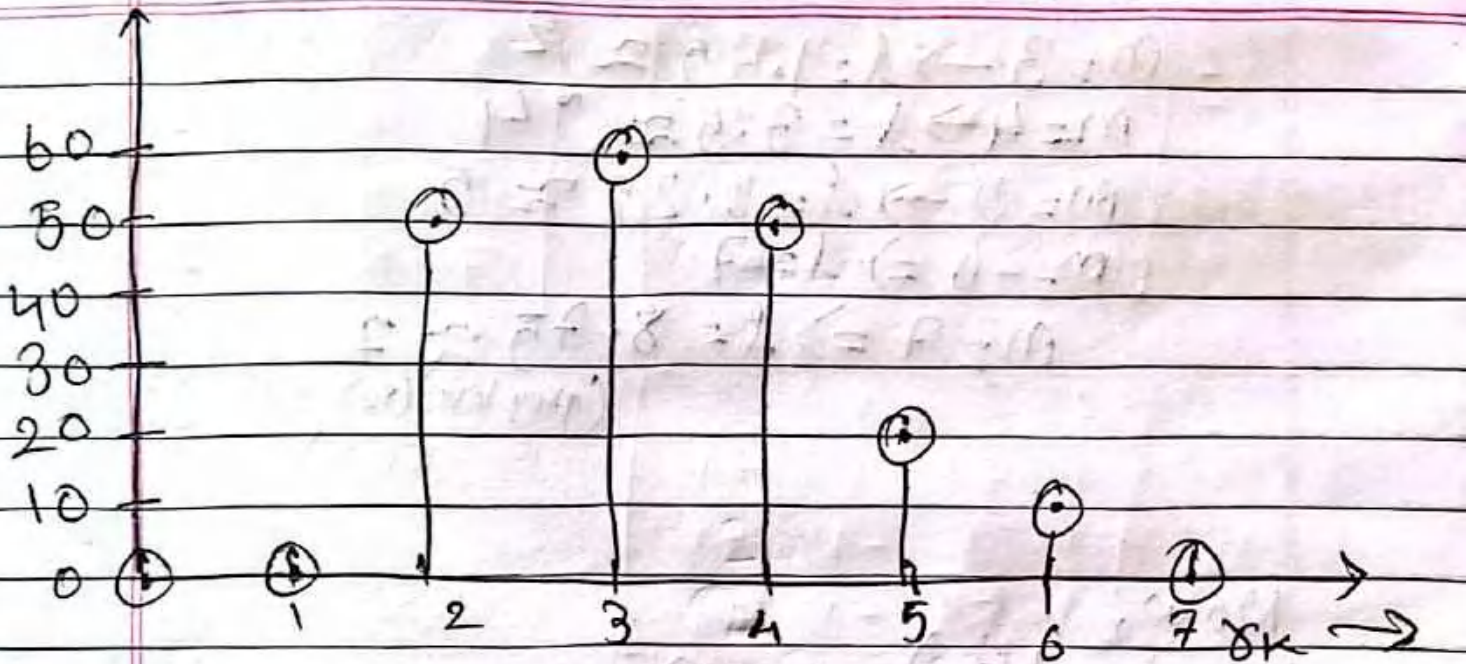


fig: Histogram of Input Image.

Now,

$$l = \frac{(l_{max} - l_{min})}{(m_{max} - m_{min})}(m - m_{min}) + l_{min}$$

$$= \frac{7-0}{(6-2)}(m-2) + 0$$

$$= \frac{7(m-2)}{4}$$

Now,

when, $m = 0 \Rightarrow l = -3.5 \simeq 0$

$m = 1 \Rightarrow l = -1.75 \simeq 0$

$m = 2 \Rightarrow l = 0$

$m=3 \rightarrow l=1.75 \simeq 2$

$m=4 \rightarrow l=3.5 \simeq 4$

$m=5 \rightarrow l=5.25 \simeq 5$

$m=6 \Rightarrow l=7$

$m=7 \Rightarrow l=8.75 \simeq 7$

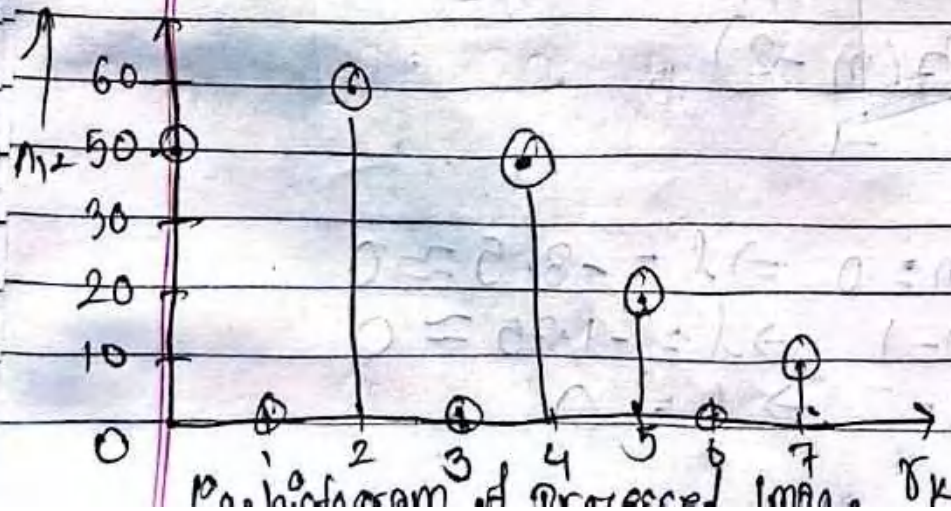
(grey level(s_k) 0 देखी 7 सम्म हुन्छ)

Now,

Grey level (s_k)	n_k
0	0+0+50=50
2	60 (m=3 को 60 हो,)
4	50
5	20
7	10+0=10

i.e.

s_k	0	1	2	3	4	5	6	7
n_k	50	0	60	0	50	20	0	10



Histogram of Processed Image.

2-D discrete Walsh Hadamar Transform:

2023 fall 2b, 2025 fall Assessment 2b.

a) Compute the 2D discrete walsh hadamard transform of the given Image block below:

$$\begin{bmatrix} 5 & 6 & 8 & 10 \\ 6 & 6 & 6 & 7 \\ 3 & 5 & 3 & 6 \\ 7 & 8 & 6 & 7 \end{bmatrix}$$

$$H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

⇒ Since this is a 4×4 form. so, walsh hadamard transformation is given by:-

$$H(N) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

where $N=4$

Then, 2-D discrete walsh Hadamard transformation is given by:-

$$X[N] = H[N] \times X(n) \times H[N]^T$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ -1 & -1 & -1 & 1 \end{bmatrix} \times \begin{bmatrix} 5 & 6 & 8 & 10 \\ 6 & 6 & 6 & 7 \\ 8 & 5 & 3 & 6 \\ 7 & 8 & 6 & 7 \end{bmatrix} \times$$

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

$$= \begin{bmatrix} 4 & 0 & 0 & -2 \\ 0 & 4 & -2 & 0 \\ 0 & -2 & 4 & 0 \\ -2 & 0 & 0 & 4 \end{bmatrix} \times \begin{bmatrix} 5 & 6 & 8 & 10 \\ 6 & 6 & 6 & 7 \\ 3 & 5 & 3 & 6 \\ 7 & 8 & 6 & 7 \end{bmatrix}$$

$$= \begin{bmatrix} 6 & 8 & 20 & 26 \\ 18 & 20 & 6 & 14 \\ -8 & -6 & 18 & 20 \\ 20 & 22 & 0 & 8 \end{bmatrix} \#\#$$

2021 spring 2a

Q) Consider the following image with the new pixel at (2,2) if the smoothing is done using a 3×3 bit neighbourhood. Find:

(i) Mean Filter
(ii) Weighted Average filter
(iii) Median Filter
(iv) Min filter
(v) Max filter

	Col0	Col1	Col2	Col3	Col4
Row 0	1	2	1	3	8
Row 1	3	8	8	5	1
Row 2	5	7	5	4	4
Row 3	2	2	1	8	6
Row 4	6	5	3	3	6

⇒ Here,

New pixel position = (2,2) since we have to use: 3×3 bit neighbourhood, centered around the pixel (2,2) = 5 the neighbourhood will be;

$$\begin{bmatrix} 8 & 8 & 5 \\ 7 & 5 & 4 \\ 2 & 1 & 8 \end{bmatrix}$$

(1) Mean Filter

$= \frac{1}{9}[8+8+5+7+5+4+2+1+8]$

$= 5.33$ #

(2) Weighted Average Filter
we need a scaling of matrix

$$\begin{bmatrix} 8 & 8 & 5 \\ 7 & 5 & 4 \\ 2 & 1 & 8 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

So, $\frac{1}{9}[(8\times1)+(8\times1)+(5\times1)+(7\times1)+(5\times2)+(4\times1)+(2\times1)+(1\times1)+(8\times1)]$

$= 5.3$

(111) Median Filter:
sort all 9 values in the 3x3 neighbourhood & pick the middle value:

ascending order:-

1 2 4 5 5 7 8 8 8

∴ median = 5.

(iv) Min Filter:
Take the smallest value in the 3×3 neighbourhood = 1.

(v) Max Filter:
largest value in 3×3 neighbourhood
= 8 //

Image compression is minimizing the size in bytes of a graphics file without degrading the quality of the image to an unexpectable level.
Apps:
(i) Progressive transmission of Image (Internet)
(ii) Video coding (HDTV Technologies).
(iii) Digital libraries & Image databases.
(iv) Medical Imaging.

Huffman Coding:

- an algorithm for lossless compression of files
- Popular technique for removing coding redundancy.

2021 spring 4(b)

(a) Employ Huffman coding Algorithm to determine the bits required for encoding the message 'mississipi'.

=> Here,
First count the frequency of each characters:

m: 1
i: 4
s: 4
p: 1

Now, Arranging in decreasing order,

Symbol (grey level) (r_k)	Freq. (n_k)	$P(r_k) = n_k/n$	1st Instance	2nd Instance
i	4	0.4	0.4	0.6 0
s	4	0.4 00	0.4 00	0.4 1
m	1	0.1 010	0.2 01	
p	1	0.1 011		
	n = 10			

Now,

r_k	$P(r_k)$	Huffman code	length
i	0.4	1	1
s	0.4	00	2
m	0.1	010	3
p	0.1	011	3

Total Bits (length × $P(r_k)$)

$1 \times 0.4 = 0.4$

$2 \times 0.4 = 0.8$

$3 \times 0.1 = 0.3$

$3 \times 0.1 = 0.3$

Hence,

The total bits from the table:

Total $= 0.4 + 0.8 + 0.3 + 0.3 = 1.8 \times 10 = 18$ bits.

Hence,

Encoded Message:

m = 010

i = 1

s = 00

s = 00

i = 1

s = 00

s = 00

i = 1

p = 011

Encoded: 010100001000011011 (18 bits).

2022 fall 4a, 2024 spring 4a, 2024 fall 4b, 2023 spring 4b, 2023 fall 4b).

Q) Find the average length of following 3 bit image after using Huffman coding technique.

r_k	0	1	2	3	4	5	6	7
count	113	139	142	145	181	105	52	1325

⇒ Arranging...

Symbol (r_k)	freq. (n_k)	Probability $P_r(r_k) = \frac{n_k}{n}$			
7	1325	0.6382	0.6382	0.6382	0.6382
4	181	0.0871	0.0871	0.1213	0.1382
3	145	0.0698	0.0755	0.0871	0.1213
2	142	0.0684	0.0698	0.0755	0.0871
1	139	0.0669	0.0684	0.0698	0.0755
0	113	0.0544	0.0669	0.0684	
5	105	0.0505	0.0544		
6	52	0.0250			
	0.2078				

Now,

r_k	$P(r_k)$	Huffman code	length
7	0.6382	0	1
4	0.0871	110	3
3	0.0698	1000	4
2	0.0684	1001	4
1	0.0669	1010	4
0	0.0544	1011	4
5	0.0505	1110	4
6	0.0250	1111	4

Now,

0.6382 → 0.6382 → 0.6382 → 0.4221

0.1620 → 0.2595 → 0.1620

0.1382 → 0.1620

0.1213

Average length $\bar{L}$ =

$$L_{avg} = \sum_{i=0}^{7} P(r_k) \times \text{length}$$

$= 0.6382 \times 1 + 0.0871 \times 3 + 0.0698 \times 4 + 0.0684 \times 4 + 0.0669 \times 4 + 0.0544 \times 4 + 0.0505 \times 4 + 0.0250 \times 4$

$= 2.2359$ bits/symbol.

Now,

Compression Ratio = $\frac{\text{original Bit length}}{L_{avg}(\text{compress})}$

$= \frac{3}{2.2559}$ → (Total 7, we can represent using 000 to 110 = 3 bits so)

C.R = 1.341 bits/symbol

Now,

Coding Efficiency (η) = $\frac{H}{L_{avg}} \times 100\%$ → Entropy

$$H = \sum_{i=0}^{7} P_s(s_k) \times \log_2 P_s(s_k)$$

$= 0.6382 \times \log_2(0.6382) + 0.0871 \times \log_2(0.0871) + 0.0698 \times \log_2(0.0698) + 0.0684 \times \log_2(0.0684) + 0.0669 \times \log_2(0.0669) + 0.0544 \times \log_2(0.0544) + 0.0505 \times \log_2(0.0505) + 0.0250 \times \log_2(0.0250)$

$= -0.4135 + (-0.3066) + (-0.2680) + (-0.2646) + (-0.2610) + (-0.2284) + (-0.2175) + (-0.1330)$

H = 2.0926 bits/symbol

Hence,

Coding efficiency (η) $= \frac{H}{L_{avg}} \times 100\%$

$= \frac{2.0926}{2.2359} \times 100\%$

$= 93.59\%$ //

Psychovisual Redundancy:

- Psychovisual Redundancy refers to the concept that certain information in an image is less important to the human visual system & can be eliminated without significantly affecting perceived image quality.
- This allows for more efficient image compression by removing data that humans are less likely to notice.
- It is commonly referred to as quantization.
- It performs lossy data compression.

Dilation, Erosion, Opening, Closing

2016 fall 5(a).

Q) Given an Image A & a structuring element B below, calculate Erosion A-B, dilation A⊕B, Opening A°B & closing A·B.

⇒ Given, A =

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

& B =

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

origin of structuring element (generally considered to be the central pixel)

classmate
Date
Page

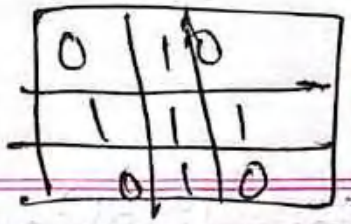
(I) Performing dilation [i.e. $A \oplus B$]:

Conditions to check:
all match → 1
some match → 1
no match → 0

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

(II) Performing Erosion [i.e. $A \ominus B$]

Conditions to check:
all match → 1
some match → 0
no match → 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	1	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

(iii) Performing Opening [A ∘ B]

Opening → First perform erosion & then perform dilation of the eroded image.

From step (ii), our eroded image is,

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

Now,
Performing dilation,

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

(10) Performing Closing [A.B]

Closing → First Perform dilation & then Perform erosion of dilated image

Note: structuring will be same.

from step (9) our dilated image is:-

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

Now,

Performing erosion;

0 0 0 0 0 0 0 0

0 0 0 0 0 1 1 0

0 1 1 1 1 1 1 0

0 1 1 1 1 1 1 0

0 1 1 1 1 0 0 0

0 1 1 1 0 0 0 0

0 0 0 0 0 0 0 0 #

2022 fall 5(a)

Q) Find closing & opening of following image (A) for given structuring element (B).

$$A = \begin{matrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{matrix}$$

B :- 

⇒ To find out (i) Closing of A
(ii) opening of A.

(i) For closing:
Closing → dilation followed by erosion.

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

dilated Image

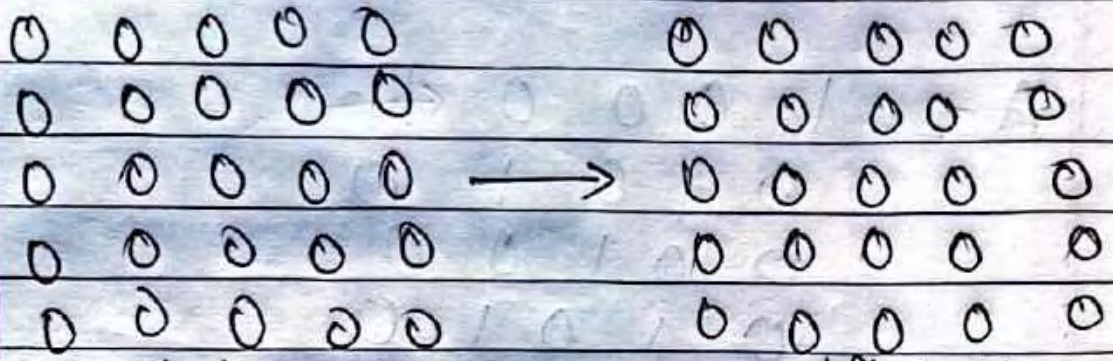
→

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

eroded Image

(11) For Opening:

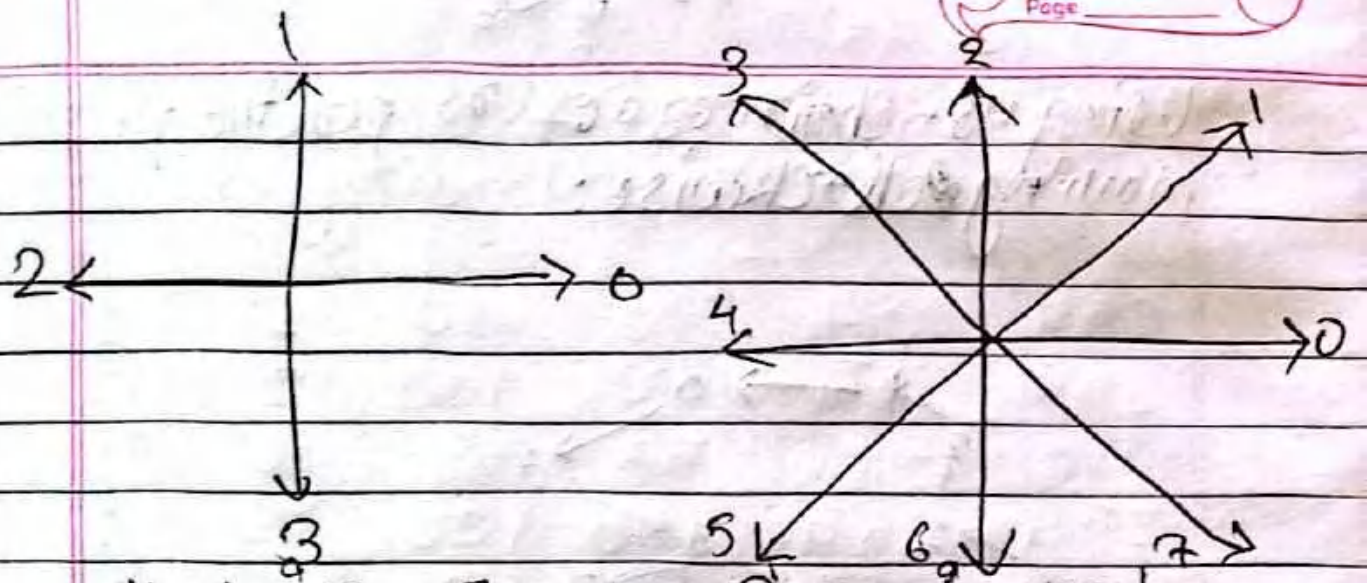
opening → erosion followed by dilation.



eroded Image

diluted Image.

Chain code & shape number



2019 Fall, 2021 fall, ~~2023 Fall~~.

(6a)

6) Given an Image "A" represents its pixel position. If A* is the starting pixel. write down the 8-chain code & find shape no. of it.

A* = A

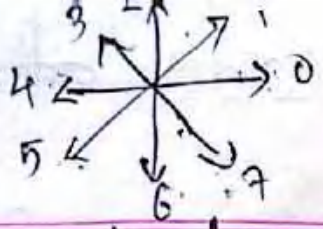
A A

A A

A A

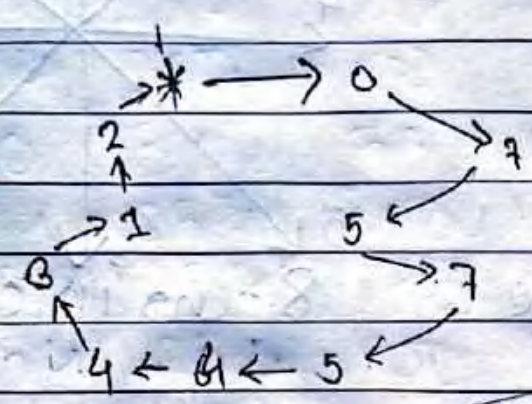
A A A

→



classmate
Date
Page

Using 8-chain code (as per the question moving clockwise:



1st difference

then

(i) ∴ chain code = 07575443121

For shape number, (1st difference मा लास्ट को no. 1st को no (0) मा जान लाउने step which is 7

1st difference = 7762670761 7

↳ min no.

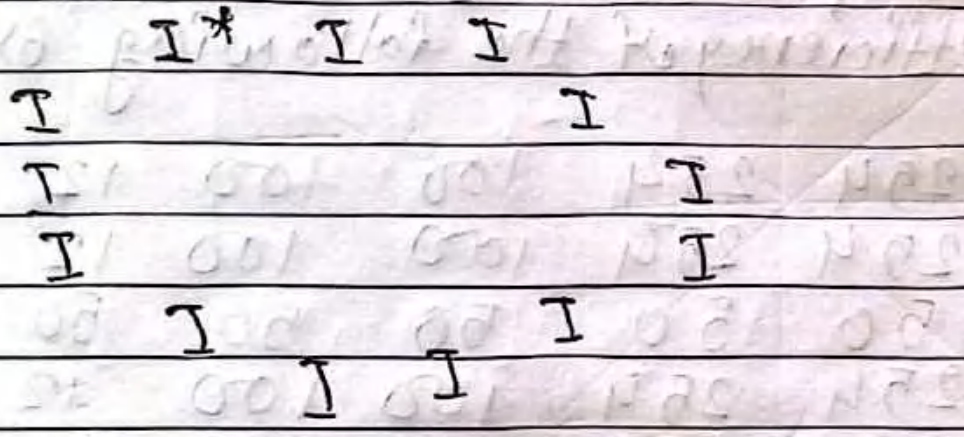
(ii) ∴ Shape number = 07617776267

(1st diff लाई arrange गरेर लेख्ने)

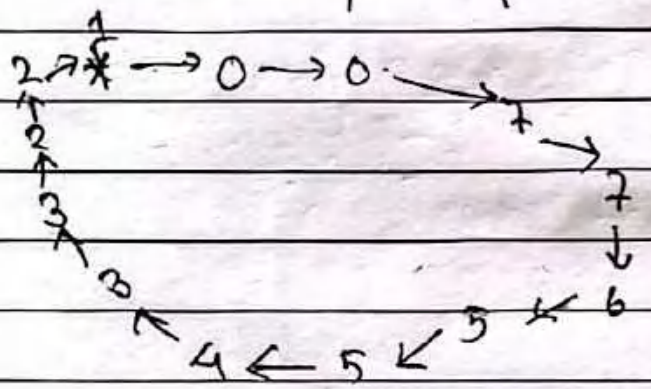
0 min हुने 0 पछाडिको सबै first मा लेख्ने त्यस पछि बाँकी लेख्ने।

2023 fall 6(b)

Q)



Using 8 chain code & clockwise rotation (as per question.)



∴ chain code = D07765543322 1

For shape number.

1st difference = 70707507707077

∴ Shape number = 07075077077 //

2023 fall 3 &, 2024 spring 25, 2021 fall 43

Q) Compute the Haar Transformation of the order N=2.

Step1:

⇒ Here, order = 2 so, N = 2, which represents order of the system or size of kernel.

Step2: Total no. of bits required based upon order of function is $n = \log_2 N = \log_2 2 = 1$

Step3: determine P & q:

(i) P ranges from 0 to n-1 i.e. $0 \le P \le 1-1$

$\therefore P = 0$

(ii) if P = 0 then q can be '0' or '1'.

Step4: Determine value of K which is total no. of rows in the kernel.

$K = 2^P + q - 1$ ⇒ for $p=0, q=0$, $K = 2^0 + 0 - 1 = 0$

for $p=0, q=1$ ⇒ $K = 2^0 + 1 - 1 = 1$

$\therefore K = 0, 1$

Step 5: Value of $z \in \left[0, \frac{1}{N}, \ldots \ldots \frac{N-1}{N}\right]$ (→ column)

i.e. 0 to $\frac{N-1}{N}$ i.e. 0 to $\frac{2-1}{2} = \frac{1}{2}$.

So: $z \in \left[0, \frac{1}{2}\right]$

Step 6: (i) If $K=0$, then, $H(z) = \frac{1}{\sqrt{N}}$ for all z

i.e. $H(z) = \frac{1}{\sqrt{2}}$ $\forall z \in \left[0, \frac{1}{2}\right]$

(ii) when $K = 1$

$$H_K(z) = H_{pq}(z) = \frac{1}{\sqrt{2}} \begin{cases} +2^{p/2}; \frac{q-1}{2^p} \le z < \frac{q-0.5}{2^p} \\ -2^{p/2}; \frac{q-0.5}{2^p} \le z \le \frac{q}{2^p} \\ 0; \text{otherwise} \end{cases}$$

Here, (→ step 4.)

for $K=1$, $P=0, q=1$

Then,

$$\frac{q-1}{2^p} \le z < \frac{q-0.5}{2^p} \Rightarrow \frac{1-1}{2^0} \le z < \frac{1-0.5}{2^0} \Rightarrow 0 \le z$$

Also,

$$\frac{q-0.5}{2^p} \leq z < \frac{q}{2^p} \Rightarrow \frac{+0.5}{2^0} \leq z < \frac{1}{2^0} \Rightarrow 0.5 \leq z < 1$$

So we get;

$$H_k(z) = H_{pq}(z) = \frac{1}{\sqrt{N}} \begin{cases} +2^{0/2}; & 0 < z < 0.5 \\ -2^{0/2}; & 0.5 \leq z < 1 \\ 0; & \text{else} \end{cases}$$

for, $z \in [0, 1/2]$

$$H_1(0) = H_{01}(0) = \frac{2^0}{\sqrt{2}} = \frac{1}{\sqrt{2}}$$

$$H_1\left(\frac{1}{2}\right) = H_{01}\left(\frac{1}{2}\right) = \frac{-2^0}{\sqrt{2}} = -\frac{1}{\sqrt{2}}$$

∴ we have kernel for the Haar transform:

	$z=0$	$z=1/2$
$k=0$	$1/\sqrt{2}$	$1/\sqrt{2}$
$k=1$	$1/\sqrt{2}$	$-1/\sqrt{2}$

$$\therefore H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

//

Hadamard Transformation:

- It is a 1D non-sinusodial signal.
- It is a transform for a complete set of orthogonal square functions.
- It has all its coefficients real & their values will be either +1 or -1 only.
- This has a very low computational complex
- It is similar to walsh transformatio

- The 1D hadamard Transform is represent by,

$$g(x,u) = \frac{1}{N}(-1)^{\sum_{i=0}^{P-1} b_i(x).b_i(u)}$$

where,

$$F(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)(-1)^{\sum_{i=0}^{P-1} b_i(x).b_i}$$

- The 2D hadamard Transform is:

$$g(x,y,u,v) = \frac{1}{N}(-1)^{\sum_{i=0}^{P-1} b_i(x)b_i(u)+b_i(y).b_i(}$$

2D function is,

$$F(u,v) = \frac{1}{N} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)(-1)^{\sum_{i=0}^{p-1} b_i(x).b_i(u) + b_i(y).b_i(v)}$$

where,

N = order of function.

x = time index

u = frequency index

p = no. of bits read to represent N

$p = \log_2 N$

2022 fall 2 b)

Q) Compute Hadamard transformation f the order N = 4.

=) Step 1: To determine the hadamard transf for N = 4 let us first consider the eq

$$g(x,u) = \frac{1}{N}(-1)^{\sum_{i=0}^{p-1} b_i(x)\cdot b_i(u)}$$

Step 2: Here N = 4.

Step 3: Since N = 4, x & u varies from 0 to N-1 i.e. Takes values [0, 1, 2, 3].

∴ Size of matrix = 4 × 4.

Step 4: No. of bits required to represent or der.

$$P = \log_2 N = \log_2 4 = 2$$

Step 5: for Hadamard transfer function when x = 0 & u = 0 then

$$g(x,u) = \frac{1}{N}(-1)^0 = \frac{1}{4} \quad \text{i.e. } g(0,0) = \frac{1}{4}$$

Determine the value of $b_i(x)$, as the value of $N=4$; $x=0,1,2,3$.

Step 6: Represent all $x=0,1,2,3$ values in binary form;

n	$b_1(n)$	$b_0(n)$
0	0	0
1	0	1
2	1	0
3	1	1

Step 7: Determine the coefficients for every point of transfer function $g(x,u)$ by expanding Hadamard fn & substituting the binary parameter values obtained in [6].

$$g(x,u) = \frac{1}{N}(-1)^{\sum_{i=0}^{P-1} b_i(x)\cdot b_i(u)}$$

$$= \frac{1}{4}(-1)^{\sum_{i=0}^{2-1} b_i(x)\cdot b_i(u)}$$

$$= \frac{1}{4}(-1)^{\sum_{i=0}^{1} b_i(x)\cdot b_i(u)}$$

$$g(x,u) = \frac{1}{4}(-1)^{\sum_{i=0}^{1} b_0(x)\cdot b_0(u) + b_1(x)\cdot b_1(u)}$$

n	$b_1(n)$	$b_0(n)$
0	0	0
1	0	1
2	1	0
3	1	1

→ Step8: Determining varies value from g(x,u) by substituting values of x & u.

$g(0,0)$:

$$g(0,u) = \frac{1}{4}(-1)^{b_0(x)b_0(u) + b_1(x)\cdot b_1(u)}$$

$$= \frac{1}{4}(-1)^{0\cdot 0 + 0\cdot 0}$$

$$= \frac{1}{4}$$

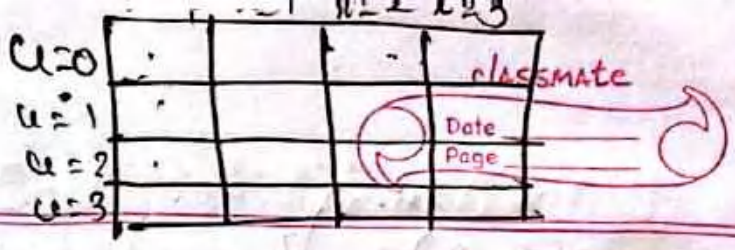
$$g(0,1) = \frac{1}{4}(-1)^{b_0(x)\cdot b_0(u) + b_0(x)\cdot b_1(u)}$$

$$= \frac{1}{4}(-1)^{0\cdot 1 + 0\cdot 0}$$

$$= \frac{1}{4}$$

$$g(0,2) = \frac{1}{4}(-1)^{1\cdot 0 + 0\cdot 1}$$

$$= \frac{1}{4}(-1)^{0}$$



$= \frac{1}{4}$

$g(0,3) = b_0(0)\cdot b_0(3) + b_1(0)\cdot b_1(3)$

$= \frac{1}{4}(-1)$

$0\cdot 1 + 0\cdot 1$

$= \frac{1}{4}(-1)$

$= \frac{1}{4}$

$g(1,0) = b_0(1)\cdot b_0(0) + b_1(1)\cdot b_1(0)$

$= \frac{1}{4}(-1)$

$1\cdot 0 + 0\cdot 0$

$= \frac{1}{4}(-1)$

$= \frac{1}{4}$

$g(1,1) = -1/4$

$g(1,2) = \frac{1}{4}$

$g(1,3) = -1/4$

$g(2,0) = 1/4$

$g(2,1) = 1/4$

$g(2,2) = -1/4$

$g(2,3) = -1/4$

$g(3,0) = 1/4$

$g(3,1) = -1/4$

$g(3,2) = -1/4$

$g(3,3) = 1/4$

Hence, Matrix:

	x=0	x=1	x=2	x=3
u=0	1/4	1/4	1/4	1/4
u=1	1/4	-1/4	1/4	-1/4
u=2	1/4	1/4	-1/4	1/4
u=3	1/4	-1/4	-1/4	1/4

$= \frac{1}{4}$

1	1	1	1
1	-1	1	-1
1	1	-1	1
1	-1	-1	1

Where,

$\frac{1}{N}$ = Sample value

$$\begin{bmatrix} H_{n-1} & H_{n-1} \\ H_{n-1} & -H_{(n-1)} \end{bmatrix} = \text{Indicate Kernel}$$

$H_0 = 1$

Hadamard matrix of 1: (H_1) i.e $n=1$

$$H_1 = \frac{1}{2}\begin{bmatrix} H_{1-1} & H_{1-1} \\ H_{1-1} & -(H_{1-1}) \end{bmatrix}$$

$$= \frac{1}{2}\begin{bmatrix} H_0 & H_0 \\ H_0 & -H_0 \end{bmatrix}$$

$$= \frac{1}{2}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Also, for $N=4$, $N = 2^n \Rightarrow 4 = 2^n \Rightarrow n = 2$

$$\therefore H_2 = \frac{1}{4}\begin{bmatrix} H_{2-1} & H_{2-1} \\ H_{2-1} & -(H_{2-1}) \end{bmatrix} \quad \left\{ H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \right.$$

$\left. \rightarrow -H_2 \right.$

$$= \frac{1}{4}\begin{bmatrix} H_2 & H_2 \\ H_2 & -H_2 \end{bmatrix} = \begin{bmatrix} H_2 & H_2 \\ H_2 & -H_2 \end{bmatrix}$$

$$= \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \#$$

Hadamard for N=2:

Step 1: To determine the hadamard trans for N=2, let us first consider the eqn,

$$g(x,u) = \frac{1}{N}(-1)^{\sum_{i=0}^{p-1} b_i(x).\, b_i(u)}$$

Step 2: Here N=2

Step 3: Since N=2, the values time & fre
Indexes of x & u are 0,1.
∴ Size of matrix = 2X2.

Step 4: No. of bits req. to represent ord

$$p = \log_2 N = \log_2 2 = 1$$

Step 5: For Hadamard f" when x=0 & u=
Then,

$$\therefore g(x,u) = \frac{1}{N} = \frac{1}{2} \text{ i.e. } g(0,0) = \frac{1}{2}$$

Step 6: Determine the value of $b_i(x)$ as the value of N=2, x=0,1. Represent in Binary form:

x	$b_0(x)$
0	0
1	1

only 1 bit so represented with 0.

Step 7: Determine the coefficients for every point of the transfer function $g(x,u)$, by expanding Hadamard fn & substituting the binary parameter values obtained in 6.

$$g(x,u) = \frac{1}{N}(-1)^{\sum_{i=0}^{p-1} b_i(x)\cdot b_i(u)}$$

$$= \frac{1}{2}(-1)^{\sum_{i=0}^{1-1} b_i(x)\cdot b_i(u)}$$

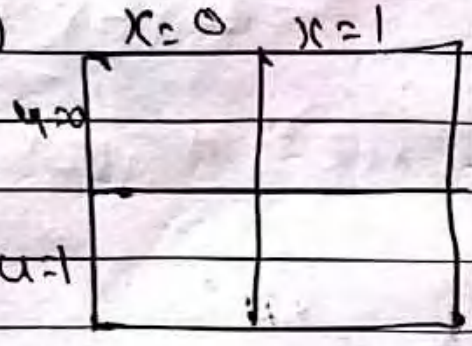
$$= \frac{1}{2}(-1)^{b_0(x)\cdot b_0(u)}$$

Step 8: Determine the varies values of $g(x,u)$ by substituting the values of x & u then, we get;

$$g(0,0) = \frac{1}{2}(-1)^{b_0(0)\cdot b_0(0)}$$

$$= \frac{1}{2}(-1)^{0\cdot 0}$$

$$= 1/2$$



$g(0,1) = 1/2$ $g(1,1) = -1/2$

$g(1,0) = 1/2$

Hence the matrix is;

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

$$= \frac{1}{2}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \#$$

Haar Transformation:

- It is transform which is based on the class of orthogonal, non-sinusodial, non-square fn, whose elements are the multiples of -1, 1, 0 by the power of $\sqrt{2}$.

- It is a computational efficient transfer function for 'N' point whose vectors require 2(N-1) additions & N multiplications.

- Steps:

① Determine the value of 'N' which represents the order of the system / the size of the kernel.

② Determine the total no. of bits required based upon the order of the function using the expression $n = \log_2 N$.

Step ③: Determine the values of the two specific terms PQ.

(i) $P \in [0, n-1]$ i.e. $0 \leq p \leq n-1$

(ii) If $P = 0$ then the value of q is either 0 or 1.

(iii) If $P \neq 0$, then $q \in [1, 2^P]$ i.e. $1 \leq q \leq 2^P$

Step 4: Determine value of K which tells us that total no. of rows in the Kernel by eqn

$$K = 2^P + q - 1$$

Step 5: Determine the value of Z where

$$Z \in \left[0, \frac{1}{N}, \frac{2}{N}, \ldots, \frac{N-1}{N}\right]$$

Step 6: If $K = 0$ then, (for first row of matrix, only value calculate करनें)

$$H_0(Z) = \frac{1}{\sqrt{N}} \quad \forall Z$$

Else, $K \neq 0$ (for other rows);

$$H_K(Z) = H_{pq}(Z) = \frac{1}{\sqrt{N}}$$

$$\begin{cases} +2^{P/2}; & \frac{q-1}{2^P} \leq 2\angle\theta \leq \frac{q-\frac{1}{2}}{2^P} \\ -2^{P/2}; & \frac{q-0.5}{2^P} \leq 2\angle\theta < \frac{q}{2^P} \\ 0; & \text{otherwise} \end{cases}$$

Q) Design kernel for Haar Transform of N=4.

Step 1: Here, N=4, which represents order of the system or size of the kernel.

Step 2:- Total no. of Bits required (n) = $\log_2 N$
i.e. $n = \log_2 4 = 2$.

Step 3: Determining values of P & q.

(i) P ranges from 0 to n-1 i.e. $0 \le p \le 2-1$
$\therefore p = 0, 1$

(ii) If p = 0 then q can be '0' or '1'

(iii) If P = 1, q will be, $1 \le q \le 2^p = 1 \le q \le 2$
$\therefore q = 1, 2$.

P	q
0	0
0	1
1	1
1	2

Step 4: Now, Determining total no. of rows in the kernel. i.e. $k = 2^p + q - 1$.

Now,

If $P=0, q=0 \Rightarrow K = 2^0 + 0 - 1 = 0$

If $P=0, q=1 \Rightarrow K = 2^0 + 1 - 1 = 1$

If $P=1, q=1 \Rightarrow K = 2^1 + 1 - 1 = 2$

If $P=1, q=2 \Rightarrow K = 2^1 + 2 - 1 = 3$

P	q	K
0	0	0
0	1	1
1	1	2
1	2	3

Step 5: Based on N, determining z where,

$$z \in \left[0, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}\right]$$

Step 6: If $K=0$, $H_0(z) = \frac{1}{\sqrt{N}} \quad \forall z$

i.e. $H_0(z) = \frac{1}{\sqrt{4}} = \frac{1}{2}$

→ If $K=1$, $P=0$ & $q=1$ Then,

$$H_1(z) = H_{01}(z) = \frac{1}{\sqrt{N}} \begin{bmatrix} +2^{P/2}, & \frac{q-1}{2^P} \le z < \frac{q-0.5}{2^P} \\ -2^{P/2}, & \frac{q-0.5}{2^P} \le z < \frac{q}{2^P} \\ 0, & \text{else} \end{bmatrix}$$

$$= \frac{1}{\sqrt{4}} \begin{bmatrix} +2^0, & \frac{1}{2^0} \le z < \frac{1-0.5}{2^0} \\ -2^0, & \frac{1-0.5}{2^0} \le z < \frac{1}{2^0} \\ 0 \end{bmatrix}$$

	z=0	z=1	z=2	z=3
K=1	1/2	1/2	-1/2	-1/2
K=2				
K=3				

$$= \frac{1}{\sqrt{4}} \begin{cases} +1 ; 0 \le z < 0.5 \\ -1 ; 0.5 \le z < 1 \\ 0 \end{cases}$$

$\therefore H_1(0) = \frac{1}{2}$

$H_1(1/4) = 1/2$

$H_1(2/4) = -1/2$

$H_1(3/4) = -1/2$

→ If K=2, P=1, q=1 then,

$$H_2(z) = H_{11}(z) - \frac{1}{\sqrt{N}} \begin{cases} +2^{P/2} ; \frac{q-1}{2^P} \le z < \frac{q-0.5}{2^P} \\ -2^{P/2} ; \frac{q-0.5}{2^P} \le z < \frac{q}{2^P} \\ 0 \end{cases}$$

$$= \frac{1}{\sqrt{4}} \begin{cases} +2^{1/2} ; \frac{1-1}{2^1} \le z < \frac{1-0.5}{2^1} \\ -2^{1/2} ; \frac{1-0.5}{2^1} \le z < \frac{1}{2^1} \\ 0 \end{cases}$$

$$= \frac{1}{\sqrt{4}} \begin{cases} +\sqrt{2} ; 0 \le z < 0.25 \\ -\sqrt{2} ; 0.25 \le z < 0.5 \\ 0 \end{cases}$$

$\therefore H_2(0) = \frac{1}{\sqrt{2}}$ $\quad \therefore H_2(2/4) = 0$

$\therefore H_2(1/4) = -\frac{1}{\sqrt{2}}$ $\quad \therefore H_3(3/4) = 0$

If $K=3$; $p=1$ & $q=2$ Then,

$$H_3(z) = H_{12}(z) = \frac{1}{\sqrt{N}} \begin{bmatrix} +2^{p/2}; \frac{q-1}{2^p} \le z < \frac{q-0.5}{2^p} \\ -2^{p/2}; \frac{q-0.5}{2^p} \le z < \frac{q}{2^p} \\ 0 \end{bmatrix}$$

$$= \frac{1}{\sqrt{4}} \begin{bmatrix} +2^{1/2}; \frac{2-1}{2^1} \le z < \frac{2-0.5}{2^1} \\ -2^{1/2}; \frac{2-0.5}{2^1} \le z < \frac{2}{2^1} \\ 0 \end{bmatrix}$$

$$= \frac{1}{2} \begin{bmatrix} +\sqrt{2}; 0.5 \le z < 0.75 \\ -\sqrt{2}; 0.75 \le z < 1 \\ 0 \end{bmatrix}$$

$\therefore H_3(0) = 0$ $\quad\quad$ $\therefore H_3(2/4) = 1/\sqrt{2}$

$\therefore H_3(1/4) = 0$ $\quad\quad$ $\therefore H_3(3/4) = -1/\sqrt{2}$

Hence, the matrix is;

$k=0$	$1/2$	$1/2$	$1/2$	$1/2$
$k=1$	$1/2$	$1/2$	$-1/2$	$-1/2$
$k=2$	$1/\sqrt{2}$	$-1/\sqrt{2}$	0	0
$k=3$	0	0	$1/\sqrt{2}$	$-1/\sqrt{2}$

$z=0 \quad z=1/4 \quad z=2/4 \quad z=3/4$

$$\therefore H = \frac{1}{\sqrt{4}} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ \sqrt{2} & -\sqrt{2} & 0 & 0 \\ 0 & 0 & \sqrt{2} & -\sqrt{2} \end{bmatrix}$$

2018 fall 8 a):

(i) Calculate Haar Transform T from given Image matrix F. And the reconstruct the original F by performing Inverse Haar Transform.

$$F = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

$$H = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ \sqrt{2} & -\sqrt{2} & 0 & 0 \\ 0 & 0 & \sqrt{2} & -\sqrt{2} \end{bmatrix}$$

⇒ Applying Haar Transform to obtain T.

$T = H \cdot F \cdot H^T$ where, H = Kernel for Haar Transform of order 4

H^T = Transpose of H

We know,

$$H = \frac{1}{\sqrt{4}}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ \sqrt{2} & -\sqrt{2} & 0 & 0 \\ 0 & 0 & \sqrt{2} & -\sqrt{2} \end{bmatrix}$$

$$H^T = \frac{1}{\sqrt{4}}\begin{bmatrix} 1 & 1 & \sqrt{2} & 0 \\ 1 & 1 & -\sqrt{2} & 0 \\ 1 & -1 & 0 & \sqrt{2} \\ 1 & -1 & 0 & \sqrt{2} \end{bmatrix}$$

1);

...ar Transform T from given ... F. And the reconstruct the original ...ng Inverse Haar Transform.

$$H = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ \sqrt{2} & -\sqrt{2} & 0 & 0 \\ 0 & 0 & \sqrt{2} & -\sqrt{2} \end{bmatrix}$$

$$\begin{bmatrix} \ldots 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

...aar Transform to obtain T.

where, H = Kernel for Haar Transform of order 4

H^T = Transpose of H

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ \sqrt{2} & -\sqrt{2} & 0 & 0 \\ 0 & 0 & \sqrt{2} & -\sqrt{2} \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & \sqrt{2} & 0 \\ 1 & 1 & -\sqrt{2} & 0 \\ 1 & -1 & 0 & \sqrt{2} \\ 1 & -1 & 0 & \sqrt{2} \end{bmatrix}$$

$$T = \frac{1}{\sqrt{4}}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ \sqrt{2} & -\sqrt{2} & 0 & 0 \\ 0 & 0 & \sqrt{2} & -\sqrt{2} \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{bmatrix} \cdot H^T$$

$$= \frac{1}{\sqrt{4}}\begin{bmatrix} 1+1+1+1 & 0+1+0+0 & 0+0+1+1 & 1+1+0+1 \\ 0 & 1 & -2 & 1 \\ 0 & \sqrt{2} & 0 & 0 \\ 0 & 0 & 0 & -\sqrt{2} \end{bmatrix} \cdot H^T$$

$$= \frac{1}{4}\begin{bmatrix} 10 & 0 & 3\sqrt{2} & -\sqrt{2} \\ 0 & 2 & -\sqrt{2} & -3\sqrt{2} \\ -\sqrt{2} & -\sqrt{2} & 2 & 0 \\ -\sqrt{2} & \sqrt{2} & 0 & 2 \end{bmatrix}$$

$$T = \begin{bmatrix} 2.5 & 0 & 3/2\sqrt{2} & -1/2\sqrt{2} \\ 0 & 1/2 & -1/2\sqrt{2} & -3/2\sqrt{2} \\ -1/2\sqrt{2} & -1/2\sqrt{2} & 1/2 & 0 \\ -1/2\sqrt{2} & 1/2\sqrt{2} & 0 & 1/2 \end{bmatrix}$$ #

2nd part

Applying Inverse Haar Transform,

$f = H^T \cdot T \cdot H$

$$= \frac{1}{\sqrt{4}} \begin{bmatrix} 1 & 1 & \sqrt{2} & 0 \\ 1 & 1 & -\sqrt{2} & 0 \\ 1 & -1 & 0 & \sqrt{2} \\ 1 & -1 & 0 & -\sqrt{2} \end{bmatrix} \begin{bmatrix} 5/2 & 0 & 3/2\sqrt{2} & -1/2\sqrt{2} \\ 0 & 1/2 & -1/2\sqrt{2} & -3/2\sqrt{2} \\ -1/2\sqrt{2} & -1/2\sqrt{2} & 1/2 & 0 \\ -1/2\sqrt{2} & 1/2\sqrt{2} & 0 & 1/2 \end{bmatrix}$$

• H

$$= \begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

fourier transform.

2024 spring 8a, 2023 spring 3a).

(b) Prove that multiplying the Image Intensity by $(-1)^{x+y}$, where (x,y) represents the coordinates of the pixels, centers the fourier transform.

t when do we need to center the fourier transform of an Image.

⇒ By default, Discrete fourier transform arranges the low-frequency components in the corners of the frequency spectrum,

while the high frequency components are in the center. This is inconvenient for analysis & visualization because we often want the low-frequency components (which represent most significant Image Information) to be at the center of the spectrum.

We need to center the fourier transform for better visualization, Image filtering & mathematical convenience.

2nd part:

2nd part

→ The 2D DFT of an image f(x,y) of size M x N is given by

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)\, e^{-j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)}$$

where F(u,v) represents frequency domain in filtering.

Now consider a modified image;

$$g(x,y) = f(x,y) \cdot (-1)^{x+y}$$

The DFT of g(x,y) is

$$G(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \left[f(x,y)(-1)^{x+y}\right] \cdot e^{-j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)}$$

Rewriting $(-1)^{x+y}$ as an exponential

$$(-1)^{x+y} = e^{j\pi(x+y)}$$

So, the DFT becomes

$$G(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)\, e^{j\pi(x+y)} \cdot e^{-j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)}$$

-T of an Image f(x,y) of
is given by

$$\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\, e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}$$

) represents frequency domain

: a modified Image;

$$\cdot(-1)^{x+y}$$

f g(x,y) is

$$= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} [f(x,y)(-1)^{x+y}]\cdot e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}$$

$(-1)^{x+y}$ as an exponential

$$e^{j\pi(x+y)}$$

DFT becomes

$$\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\, e^{j\pi(x+y)} \cdot e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}$$

$$= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\cdot e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}-\frac{x}{2}-\frac{y}{2}\right)}$$

$$= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\, e^{-j2\pi\left(\frac{(u-M/2)x}{M}+\frac{(v-N/2)y}{N}\right)}$$

$$\therefore G(u,v) = F\left(u-\frac{M}{2},\, v-\frac{N}{2}\right)$$

This means that spectrum F(u,v) has been shifted by M/2, N/2 in both direction effectively centering the fourier transform.

(1) Compute DFT of $f(x)=\{1,-1,-1,2\}$.

Soln

Here,

$f(x)=\{1,-1,-1,2\}$

$N=4$

Then discrete ~~inverse~~ fourier transform is given by

$$F(u)=\frac{1}{N}\sum_{n=0}^{N-1} F(x)\cdot e^{-j2\pi ux/N} \quad ;[u=0,1,..N$$

& the twiddle factor is

$$W_N^{ux} = e^{-j\frac{2\pi}{N}ux}$$

$$W_4^{ux} = e^{-j\frac{2\pi}{4}\frac{1}{2}ux}$$

$$= e^{-j\frac{\pi}{2}ux}$$

where,

$W_4^0=1 \quad W_4^2=-j \quad W_4^{2x}=-1$

$W_4^{3x}=j$

when $x=0$

$W_4^{0\cdot x} = e^{-j/2\pi u\cdot 0}$

$= e^0$

$=1$

$W_4^u = e^{-j\pi/2}$

$= -j$

$W_4^{2u} = e^{-j\pi}$

$=\cos\theta - j\sin$

$=-1-0$

$=-1$

when

$u=0$

$$F(0)=\frac{1}{4}\sum_{x=0}^{3} F(x)\cdot W_4^{ux}$$

$$=\frac{1}{4}\left[1\cdot W_4^0+(-1)\cdot W_4^0+(-1)W_4^0+(2)\cdot W_4^0\right.$$

$$=\frac{1}{4}(1\cdot 1+(-1)\cdot 1+(-1)\cdot 1+(2)\cdot 1)$$

$$=\frac{1}{4}(1-1-1+2)$$

$$=\frac{1}{4}$$

when $u=1$

$$F(u)=\frac{1}{4}\sum_{x=0}^{3} f(x)\cdot W_4^{1x}$$

$$=\frac{1}{4}\left[1\cdot W_4^0+(-1)W_4^1+(-1)W_4^2+(2)W_4^3\right)$$

$$=\frac{1}{4}\left[1\times 1+(-1)\times(-j)+(-1)\times(-1)+2\times j\right]$$

$$=\frac{1}{4}[2+3j]$$

When $u = 2$

$$f(2) = \frac{1}{4}\{(1)\cdot\omega_4^0 + (-1)\omega_4^2 + (-1)\omega_4^4 + (2)\cdot\omega_4^6\}$$

$$= -\frac{1}{2}$$

When $u = 3$

$$f(3) = \frac{-1+j}{4}$$

Ch-3

The one-dimensional fourier transform & its Inverse.

Fourier transform:

$$F(u) = \int f(x)\, e^{-j2\pi ux}\, dx$$

Inverse F.T:

$$F(x) = \int F(u)\, e^{j2\pi ux}\, du$$

2-D F.T

$$F(u,v) = \int\int_{-\infty}^{\infty} f(x,y) \cdot e^{-j2\pi(ux+vy)}\, dx\, dy$$

The Inverse transforms:

$$F(x,y) = \int\int_{-\infty}^{\infty} F(u,v)\, e^{j2\pi(ux+vy)}\, dx\, dy$$

→ 2D - Fourier Spectrum is:

$$|F(u,v)| = \sqrt{R^2(u,v) + I^2(u,v)}$$

Phase angles is $\phi(u,v) = \tan^{-1}\dfrac{I(u,v)}{R(u,v)}$

Discrete Fourier Transform:

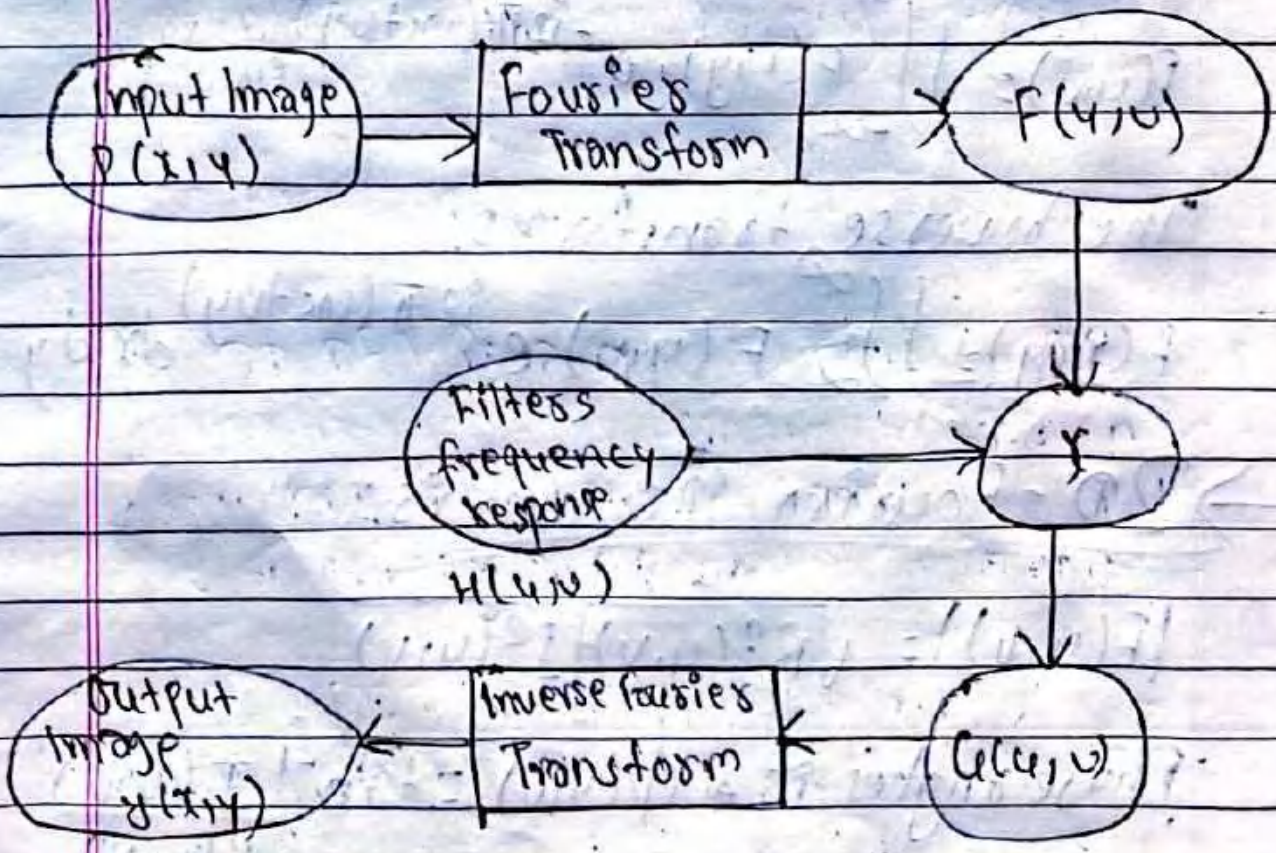
$$F(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x) \cdot e^{-j2\pi ux/N} \quad ; u = 0, 1, \ldots N-1$$

Inverse DFT:

$$f(x) = \sum_{x=0}^{N-1} f(u) \cdot e^{+j2\pi ux/N} \quad ; x = 0, 1, \ldots N-1$$

#

Steps Involved in Frequency domain Filtering:



ete fourier transform:

$$\frac{1}{N}\sum_{x=0}^{N-1} f(x)\cdot e^{-j2\pi ux/N} \quad ; u=0,1,\ldots N-1$$

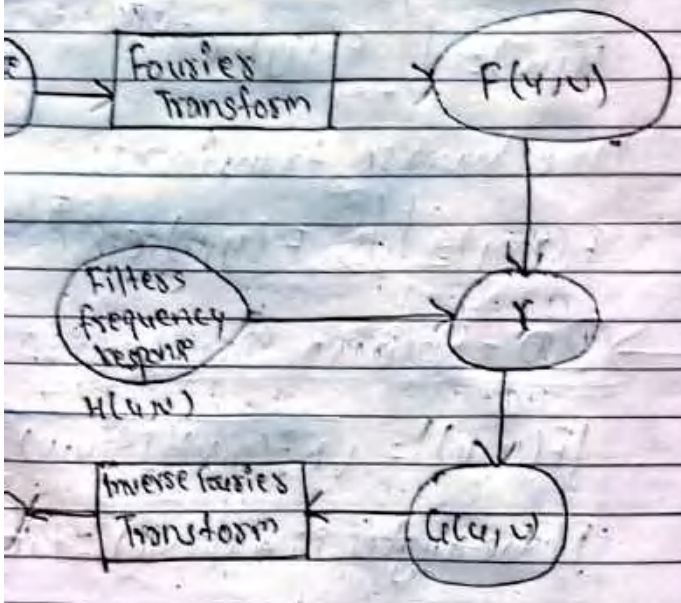
se DFT:

$$\sum_{x=0}^{N-1} f(x)\cdot e^{-j2\pi ux/N} \quad ; x=0,1,\ldots N-1$$

#

PS Involved in frequency domain

ing:



(I) Input Image f(x,y)

↳ The process starts with an Input Image in the spatial domain denoted as f(x,y)

↳ this is a grayscale or color Image that needs to be filtered to enhance certain features or remove noise.

(II) Fourier Transform F(u,v)

↳ The Image is transformed from the spatial domain to the frequency domain using the fourier transform (FT).

↳ The fourier transform decomposes the Image into its frequency components, represented as

$$F(u,v) = F\{f(x,y)\}$$

(III) Multiplication with filter's frequency Response H(u,v)

↳ A frequency domain filter H(u,v) is applied to modify the Image

$$G(u,v) = F(u,v) \times H(u,v)$$

Different frequency filters such as low-pass filter, high pass filter, band-pass filter.

(4) Inverse fourier Transform g(x,y).
↳ The filtered Image in the frequency domain G(u,v) is converted back to the spatial domain using Inverse fourier transform (IFT):

$$g(x,y) = F^{-1}\{G(u,v)\}.$$

↳ This reconstructs the processed Image with the applied filter effects.

(5) Output Image g(x,y)
The final output Image contains the effects of the frequency filtering.

Properties of 2D-DFT:

The 2D DFT is a crucial tool in IP for converting spatial domain Images in to the frequency domain. It provides Insights into the frequency components of an Image.

The 2D-DFT of an Image F(x,y) of size MxN given by

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \cdot e^{-2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)}$$

(1) Linearity:

The DFT of linear combination of two images is equal to the same linear combination of their individual DFTs.

If $F_1(x,y)$ & $F_2(x,y)$ have transforms $P_1(u,v)$ & $P_2(u,v)$ then

$$F\{aF_1(x,y) + bF_2(x,y)\} = aF_1(u,v) + bF_2(u,v)$$

(2) Periodicity:

The DFT is periodic in both frequency direction

$$F(u+M,v) = F(u,v+N) = F(u,v)$$

(3) Symmetry

If $f(x,y)$ is real-valued, then its DFT has conjugate symmetry

$$F(u,v) = F^*(-u,-v)$$

where F^* denotes the complex conjugate.

(4) Shift Property:
If an Image is shifted by (x_0, y_0) its DFT gets multiplied by a phase shift $e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}$
$F\{f(x-x_0, y-y_0)\} = F(u,v) \cdot e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}$ #

Image enhancement	Image Restoration
(i) Improves the visual appearance of Images.	(i) Reconstructs the original Image quality.
(ii) Focus on enhancing subjective quality.	(ii) Focus on recovering objective accuracy.
(iii) Enhances Image features like contrast, color, & sharpness.	(iii) Removes degradation like noise, blur & distortion.
(iv) Often used in photography to make Image appealing.	(iv) Used in fields like medical Imaging & forensic analysis.
(v) May introduce new artifacts.	(v) Aims to minimize & remove existing artifacts.
(vi) Techniques include filtering, histogram equalization & color adjustment.	(vi) Techniques include deblurring, denoising, & inpainting.
(vii) Adjustments based on user preference.	(vii) Adjustments based on mathematical models.
(viii) Generally Irreversible	(viii) Often aims to be reversib
(ix) Ex: Sharpening a photo to highlight details.	(ix) Restoring an old, damaged photograph by removing scratch

Short Notes of chapter 8 & 10

Ch-10:

(1) Steps of Pattern & Pattern Recognition System. Explain the issues with feature selection.

⇒ Pattern:

A pattern is a repeatable arrangement of elements or occurrences that exhibit consistency & regularity. i.e. It is an arrangement or structure of elements that can be recognized & identified.

Pattern Recognition:

Pattern recognition is the process of identifying & classifying patterns in data. In context of IPPR, it's about recognizing & interpreting patterns within images. This involves analyzing the pixel data of an image to identify objects, shapes or features.

Steps of Pattern Recognition: → 1st part.

↳ 5 or 8 marks

→

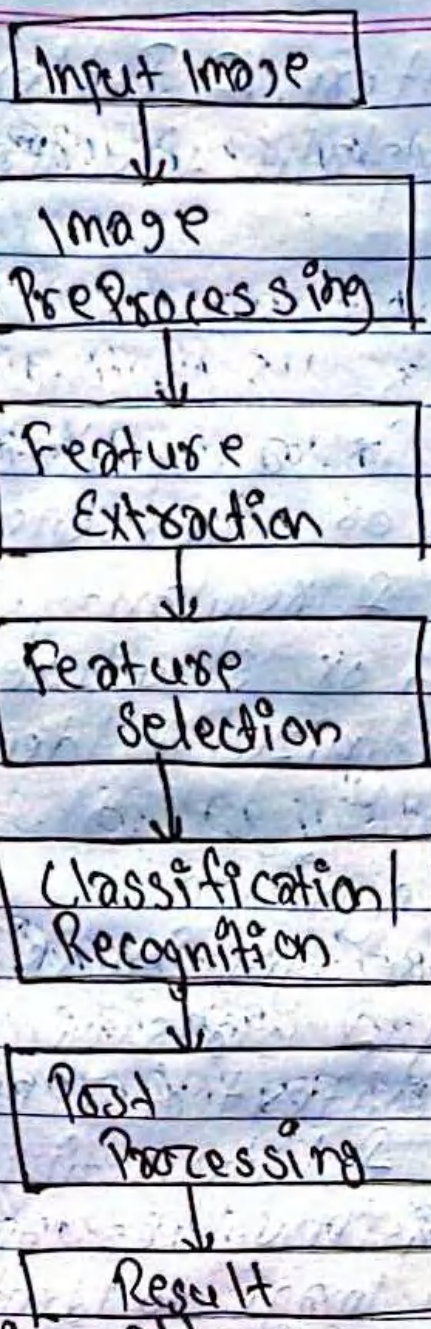
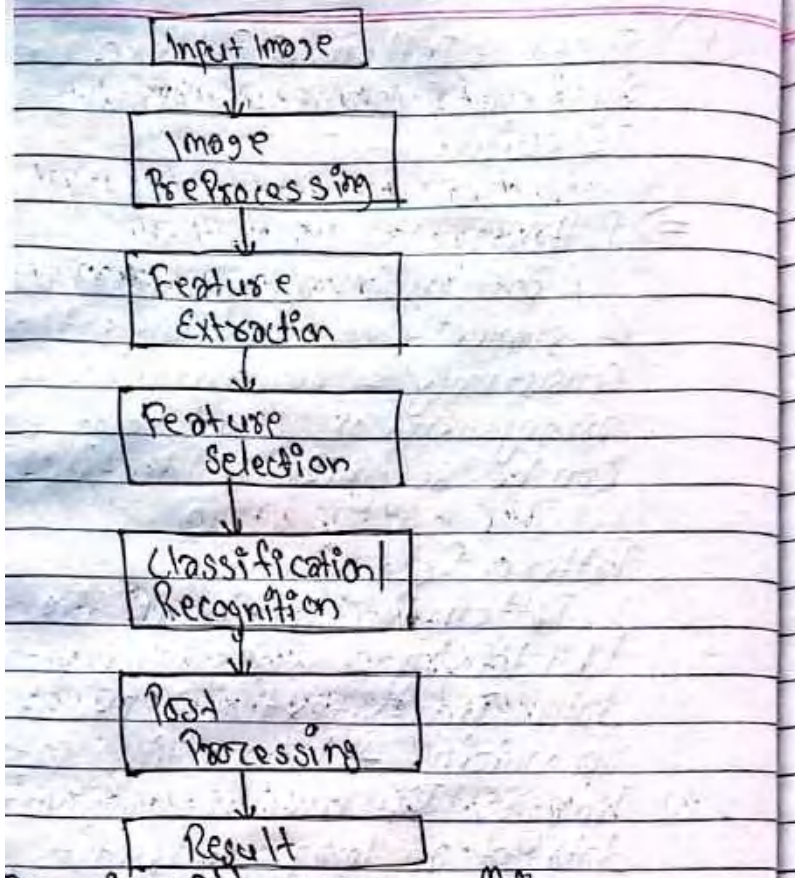


fig: Steps in pattern recognition.

① Pre-Processing:
- Remove noise & unwanted artifacts from image
- Enhance contrast & brightness for better feature visibility.
- Normalize Image size & orientation for consistent analysis.



teps in pattern recognition.

essing:
ve noise & unwanted artifacts from images
nce contrast & brightness for better
ure visibility.
malize image size & orientation
consistent analysis.

② Feature Extraction:
- Identify key characteristics like edges, corners & textures.
- Extract measurable properties that represent the pattern.
- Transform raw pixel data into meaningful numerical features.

③ Feature selection:
- Choose the most important features for classification.
- Reduce complexity by eliminating redundant information.
- Focus on features that best distinguish between pattern classes.

④ Classification / Recognition:
- Compare extracted features with known pattern models.
- Assign input patterns to predefined categories.
- Use algorithms like neural networks or decision trees.

→

⑤ Post Processing:
- Refine results using context & domain knowledge.
- Apply rules to correct misclassifications.
- combine multiple recognition results for improved accuracy.

2nd part (Assessment NEXT)

Explain the Issues with feature Selection:

⇒ ① Curse of Dimensionality:
- Too many features can lead to overfitting
- High-dimensional spaces require more samples.
- Computation becomes increasingly complex.

② Redundancy & correlation
- Many features contain overlapping Informatio
- Correlated features add burden without benifits.
- Difficult to Identify truly Independen features.

: using context & domain

o correct misclassifications.
ltiple recognition results for
curacy.

sues with feature selection:

ensionality:
atures can lead to overfitting
onal spaces require more

n becomes increasingly complex.

correlation
es contain overlapping information
features add burden
enifits.
to identify truly ~~the~~ Independent

③ Relevance Determination:
- Hard to distinguish important from unimportant features.
- Feature importance varies across pattern classes.
- Some features only useful in combination with others.

④ Selection method limitations:
- Filter methods miss feature interactions.
- Wrapper methods are computationally expensive.
- Embeeded methods may be algorithm-based.

⑤ Balancing Complexity & Performance:
- Too few features risk information loss.
- Too many features increase model complexity.
- Optimal features subset often requires experimentation.

Ch-9 + many short notes:

Pattern & Pattern Classes:

- A pattern is an arrangement of descriptors
- A pattern class is a family of patterns that share some common properties.
- Pattern classes are denoted $w_1, w_2, \ldots w_N$ where N is the number of classes.
- Pattern recognition by machine involves techniques for assigning patterns to their respective classes - automatically & with as little human intervention as possible.

The object or pattern recognition task consists of two steps:

(i) Feature selection (extraction)
(ii) Matching (classification.

There are 3 common pattern arrangements used in practice:

↳ Numeric vectors (for quantitative descriptions).

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

t many short notes:

ern & Pattern classes:

tern is an arrangement of descriptors.
attern class is a family of patterns
at share some common properties.
ttern classes are denoted $w_1, w_2, \ldots w_N$
here N is the number of classes.
tern recognition by machine involves
hniques for assigning patterns to
eir respective classes - automatically
with as little human intervention as
ssible.

bject or pattern recognition task
ts of two steps:
(i) Feature selection (extraction)
(ii) Matching (classification.

are 3 common pattern arrangements
in practice:
umeric vectors (for quantitative
escriptions).

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Strings & trees (for structural descriptions)

$x = abababa \ldots$

Example:
A pattern could be the arrangement of pixels that form the shape of a letter in an OCR system. Now, in OCR, pattern classes could include different categories for each letter of the alphabet (e.g. 'A', 'B', 'C') or for different handwriting types.

→ Decision-Theoritic Methods

Q) Explain Minimum Distance classifier fo pattern classification with necessary eqn & an example.

⇒ The minimum distance classifier is used to classify unknown Image data to classes which minimize the distance betn the Image data & the class in multi-feature space. The distance is defined as an Ind of similarity so that the minimum distanc is identical to the maximum similarity

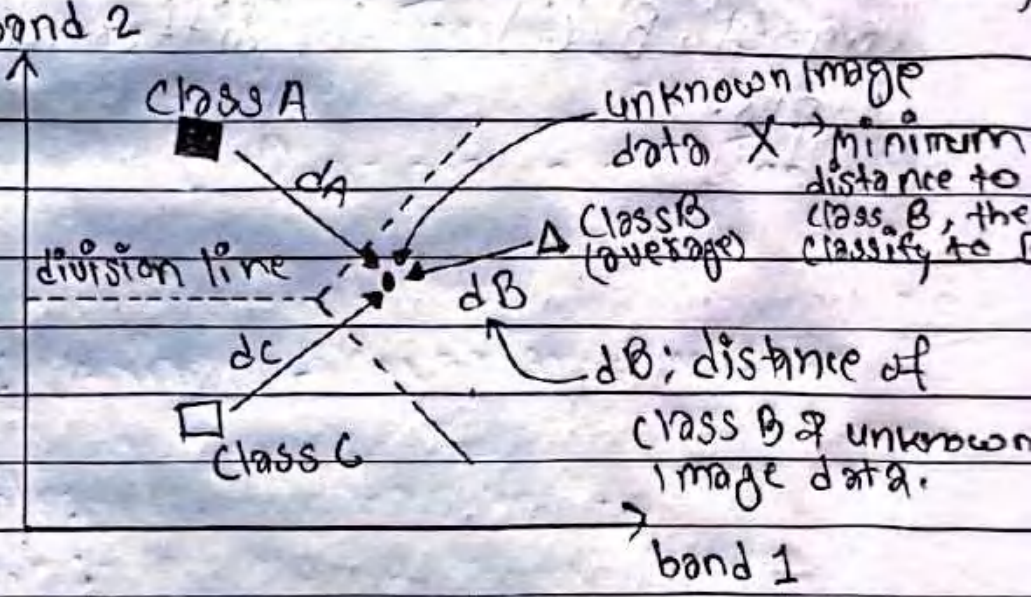


fig: Concept of Minimum Distance Classifier.

Above figure shows the concept of a minimum distance classifier. The follow distances are often used in this procedure

(1) Euclidian Distance:

$$d_k^2 = (x - \mu_k)^t (x - \mu_k)$$

Is used in cases where the variances of the population classes are different to each other. The Euclidian distance is theoretically identical to similarity index.

(2) Normalized Euclidian distance:

The Normalized Euclidian distance is proportional to the similarity index, as shown in figure, in the case of difference variance.

$$d_k^2 = (x - \mu_2)^t \sigma_k^{-1} (x - \mu_2)$$

(3) Mahalanobis distance:

In cases where there is correlation between the axes in feature space, the Mahalanobis distance with variance covariance matrix, should be used as shown in figure.

$$d_k^2 = (x - \mu_k)^t \Sigma_k^{-1} (x - \mu_k)$$

where X: vector of Image data (n bands)

$X = [x_1, x_2, \ldots, x_n]$

μ_k: mean of the k^{th} class.

$\mu_k = [m_1, m_2, \ldots m_n]$

σ_k = variance matrix

$$\sigma_k = \begin{bmatrix} \sigma_{11} & 0 & \cdots & 0 \\ 0 & \sigma_{22} & & 0 \\ \vdots & & & \\ 0 & \cdots & & \sigma_{nn} \end{bmatrix}$$

Σ_k: variance-covariance matrix

Example:

Scenario:

You have a system designed to classify fruits based on their color.

fruit classes:

- Apple: Mean color value = 1 (red)
- Banana: Mean color value = 2 (yellow)
- Orange: Mean color value = 3 (orange)

- Unknown fruit:
  A unknown fruit has a color value of 1.5.

- Euclidean Distance Calculation:
  - Distance to Apple $|1.5-1| = 0.5$
  - " " Banana $|1.5-2| = 0.5$
  - " " Orange $|1.5-3| = 1.5$

- Classification Decision:
  The system classifies the unknown fruit as either an apple or banana since both have the smallest distance (0.5).
  ↳ Additional features or rules may be needed to finalize the decision betn apple & banana.

Neural Network In Image Processing

→ Neural network is a machine learning program, or model, that makes decisions in a manner similar to the human brain, by using processes that mimic the way biological neurons work together to identify phenomena.

→ In Image processing, neural networks are used to analyze, interpret, & manipulate images by learning patterns & features directly from pixel data.

→ How they work in Image Processing:

(i) Input:
Images are fed into the network as numerical data.

(ii) Layers:
The network has an Input layer, hidden layers, & an output layer. Hidden layers transform raw pixel data into meaningful features (e.g. edges, shapes) through weighted connections & activation functions.

(iii) Learning:
During Training, the network adjusts weights using backpropagation &

optimization to minimize errors, learning from labeled examples.

(iv) Output:
The network produces results like classifications segmentations or enhanced images.

Use of Various Neural Network in IPRR:

(i) Multilayer Perceptron (MLP):
- Fully connected feedforward network with input, hidden & output layers.
- Suitable for simple image classification with low-dimensional data.
- Processes flattened pixel values to output class labels.
- Example: MNIST digit classification.

(ii) Convolutional Neural Network (CNN):-
- Uses convolutional, pooling, & fully connected layers.
- Ideal for tasks like object recognition, segmentation etc.
- Reduces parameters by exploiting spatial relationships.
- Ex:- Cats Vs. dogs.

(iii) Recurrent Neural Network (RNN):-
- Processes sequential data with loops & memory.
- Applied to tasks involving sequences or image captions.
- Ex:- Generating captions like "A dog runs in a park."
- less suited for static images.

(iv) Generative Adversarial Networks (GANs):
- Two networks:
  Generator (creates images) & discriminator (evaluates them).
- Ex:- Super-resolution or generating synthetic faces.
- Produces high-quality, realistic outputs.

Application of NN in ~~Image Processi~~ Pattern Recognition:

→ (i) Image Processing
↳ facial Recognition, medical diagnosis.
(ii) Speech Recognition:
↳ Virtual Assistants like Siri or Alexa.
(iii) NLP
↳ Sentiment Analysis, handwriting recognition
(iv) Time-series Analysis:
↳ Stock market prediction, ECG signal classification.
↳ fraud detection in financial transactions.
(v) Anamoly Detection:
↳ fraud detection in financial transactions.

Application of NN in IPPR:

(i) NN for Image compression
(ii) " " Pattern Recognition
(iii) " " for Perception (Perception).

Perceptron Network:

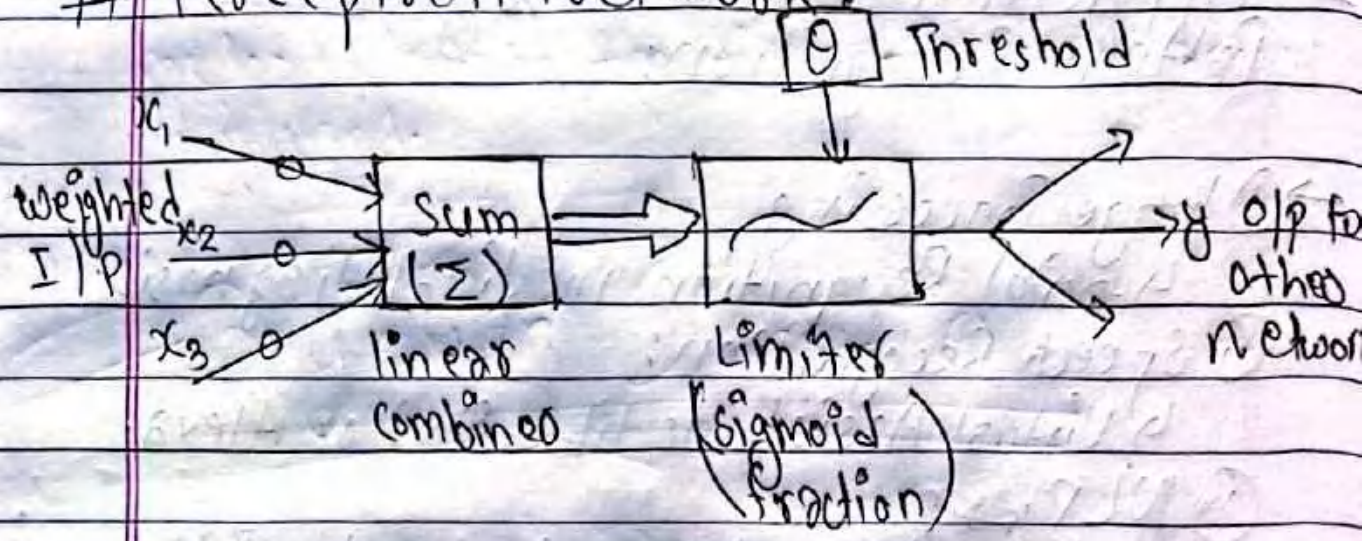
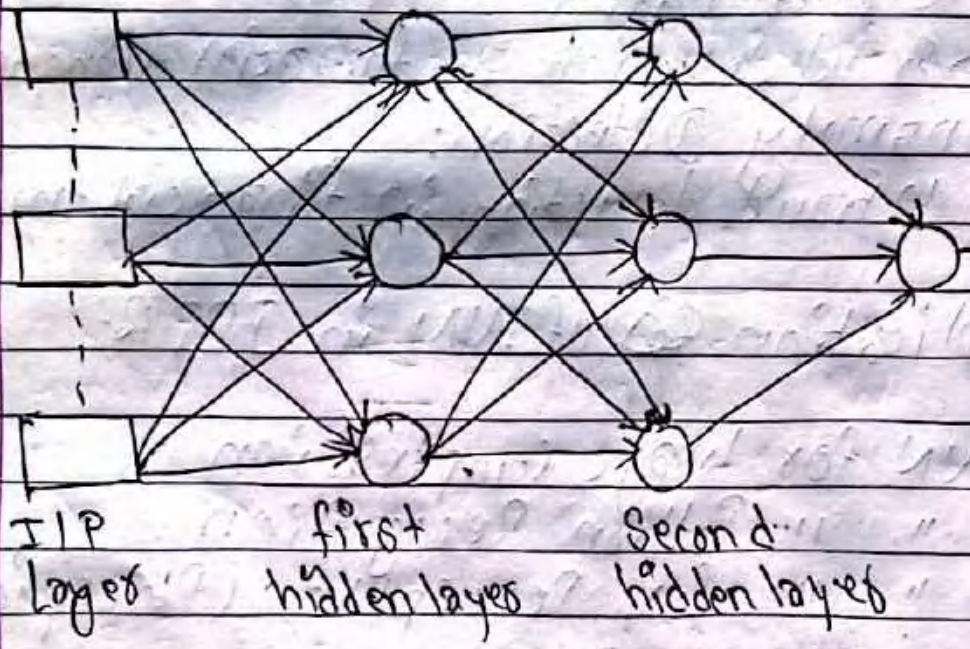


fig: Block diagram of perceptron.



(fig: Formation of o/p by perceptron).

It is one of the earliest ANN model.

Here, i/p's from one or more previous neurons are individually weighted & then summed.

The result is non-linearly scaled betⁿ 0 & 1. And then, the result is passed to the neurons in the next layers for output result.

Several perceptions can be combined to form multi-layer perception (MLP). So MLP is a development from the simple perception in which extra hidden layers are added.

Here, more than one layer can be used.

Generally, connections are allowed from input layer, hidden layer & so on. At first i/p are feeded into the i/p layer & get multiplied by inter-connection weights as they are passed from the i/p layer to the first hidden layer.

After first layer then go to the second layer & finally data is multiplied by inter-connection weight & so on.

Back Propagation never take the perception methods but at the time of sensing accuracy it is used.

Bayesian Classifier:

A Bayesian classifier is simple probabilistics classifier based on the Bayesian Theorem.

The Bayesian classifier, classified the features to contribute the probability of hypothetical evidence.

$$P(H_i/E) = \frac{P(E/H_i) \cdot P(H_i)}{\sum_{n=1}^{K} P(E/H_n) \cdot P(H_n)}$$

Where,

$P(E/H_i)$ = The probability that we will observe evidence 'E' given that Hypothesis H_i is true.

$P(H_i)$ = A prior probability H_i is true.

$P(H_i/E)$ = The probability that hypothesis H_i given "E" is true.

K = The no. of possible

Hopfield Network:

- A Hopfield network is a recurr network used for assoc where the network is de & recall patterns.
- It is typically used for b storage & recall
- It operates on the principal minimization.
- It is a fully connected, sym where each neuron is conne other neuron.

$P(H_i)$: A prior probability that the Hypothesis H_i is true.

$P(H_i/E)$: The probability that we will observe hypothesis H_i given that evidence "E" is true.

K : The no. of possible Hypotheois.

Hopfield Network :

- A Hopfield network is a recurrent neural network used for associative memory, where the network is designed to store & recall patterns.
- It is typically used for binary pattern storage & recall
- It operates on the principal of energy minimization.
- It is a fully connected, symmetric network where each neuron is connected to every other neuron.

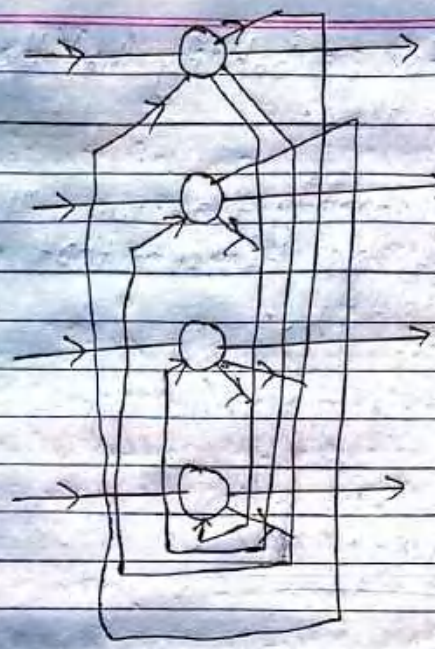


fig: Structure of Hopfield network

Hopfield nets can either have units that takes on values of 1 or -1, or units that take on values of 1 or 0.

So, the two possible definitions for unit i's activation, a_i are:

(1) $a_i \leftarrow \begin{cases} 1 & \text{if } \sum_j \\ -1 & \text{oth} \end{cases}$

(2) $a_i \leftarrow \begin{cases} 1 & \text{if } \sum_j \\ 0 & \text{oth} \end{cases}$

Where:

- w_{ij} is the strength of weight from unit j to u of the connection).
- s_j is the state of uni
- θ_i is the threshold

Hopfield nets have a scalar with each state of the ne to as the "energy", E, where:

$$E = -\frac{1}{2} \sum_{i,j} w_{ij} s_i s_j -$$

(1) $a_i \leftarrow \begin{cases} 1 & \text{if } \sum_j w_{ij} s_j > \theta_i, \\ -1 & \text{otherwise.} \end{cases}$

(2) $a_i \leftarrow \begin{cases} 1 & \text{if } \sum_j w_{ij} s_j > \theta_i, \\ 0 & \text{otherwise} \end{cases}$

Where:

- w_{ij} is the strength of the connection weight from unit j to unit i (the weight of the connection).

- s_j is the state of unit j.

- θ_i is the threshold of unit i.

Hopfield nets have a scalar value associated with each state of the network referred to as the "energy", E, of the network, where:

$$E = -\frac{1}{2} \sum_{i,j} w_{ij} s_i s_j - \sum_i \theta_i s_i$$

Spatial low pass filter:

It is used in image processing to remove high-frequency components, which typically correspond to noise or fine details.

The filter allows low-freq. components (smooth areas) of the image to pass through while attenuating the high-frequency components (sharp edges & noise).

Its purpose is to smooth an image, reduce noise, & blur fine details.

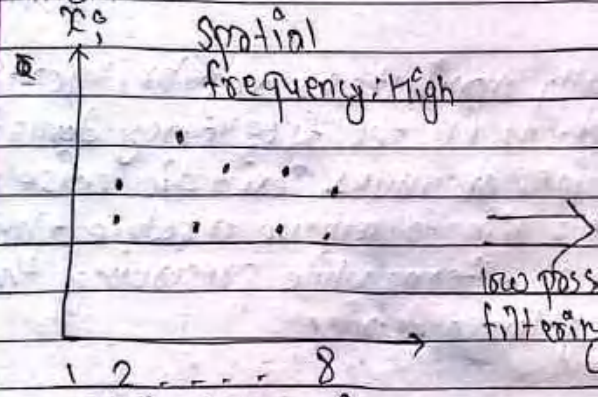
It reduces high-frequency variations & keeps low-frequency information, which is often the dominant feature of an image.

The output of a spatial low-pass filter is obtained by convolving the image $I(x,y)$ with the filter kernel $H(x,y)$.

$$I_{filtered}(x,y) = \sum_{i=-m}^{m} \sum_{j=-n}^{n} I(x+i, y+j) H(i,j)$$

Where,

- $I(x,y)$: The input image at
- $I_{filtered}(x,y)$: The filtered
- $H(i,j)$: The filter kernel (
- The kernel $H(i,j)$ typically
represent a smoothing function,
function.



Page

ow pass filter:

Image processing to remove
y components, which typically
oise or fine details.

lows low-freq. components
s) of the image to pass through
uating the high-frequency
sharp edges & noise).

to smooth an image, reduce
ur fine details.

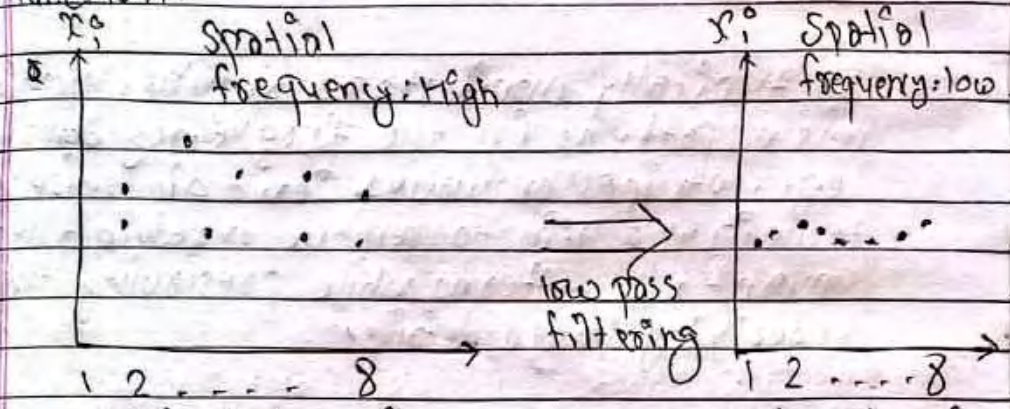
gh-frequency variations & keeps
information, which is often the
ature of an image.

of a spatial low-pass filter
by convolving the image $I(x,y)$
ter kernel $H(x,y)$.

$$(x,y) = \sum_{i=-m}^{m} \sum_{j=-n}^{n} I(x+i, y+j) H(i,j)$$

Date
Page

- $I(x,y)$: The input image at position (x,y)
- $I_{filtered}(x,y)$: The filtered image at position (x,y)
- $H(i,j)$: The filter kernel (low-pass filter)
- The kernel $H(i,j)$ typically has values that represent a smoothing function, such as a Gaussian function.



Periodic Noise Reduction:

Periodic noise in an image refers to repetitive patterns or interference, often caused by sources like electric equipment or environmental factors.

It typically manifests as regular, repeating noise patterns in the frequency domain (eg., stripes or waves. Periodic noise reduction techniques aim to remove or reduce these unwanted patterns while preserving the underlying image details.

Steps:
- Perform a 2D Fourier Transform to convert the image into the frequency domain.
- Identify the frequency components corresponding to periodic noise.
- Apply a filter to attenuate these noisy frequencies.
- Apply the Inverse Fourier Transform to return the image to the spatial domain.

Image Averaging:

It is a technique in IP used to noise by combining multiple images of same scene or object.

By averaging the pixel values across images, random noise, which is unpredictable, can be reduced, while underlying image features are

By averaging multiple images taken of same scene, random variations (noise) cancel out, while consistent features (edges etc.) remain.

Let's say we have N images $I_1, I_2, \ldots$ of the same scene. The averaged image I at each pixel location (x, y) is computed as:

$$I(x, y) = \frac{1}{N} \sum_{i=1}^{N} I_i(x, y)$$

where,

$I_i(x, y)$: The pixel value at position (x, y) in the ith image.

$I(x, y)$: The averaged pixel value at (x, y).

N: The number of images being averaged.

Image Averaging:

It is a technique in IP used to reduce noise by combining multiple images of the same scene or object.

By averaging the pixel values across a set of images, random noise, which is typically unpredictable, can be reduced, while the underlying image features are preserved.

By averaging multiple images taken from the same scene, random variations (noise) tend to cancel out, while consistent features (objects, edges etc.) remain.

Let's say we have N images $I_1, I_2, \ldots, I_N$ of the same scene. The averaged image $I(x,y)$ at each pixel location (x,y) is computed as:

$$I(x,y) = \frac{1}{N} \sum_{i=1}^{N} I_i(x,y)$$

where,

$I_i(x,y)$: The pixel value at position (x,y) in the i^{th} image.

$I(x,y)$: The averaged pixel value at position (x,y).

N: The number of images being averaged.

descriptors are also called ... that are useful when working with region boundaries. Types: FD & Shape Number.

Ch-8

Fourier Descriptor (short note):

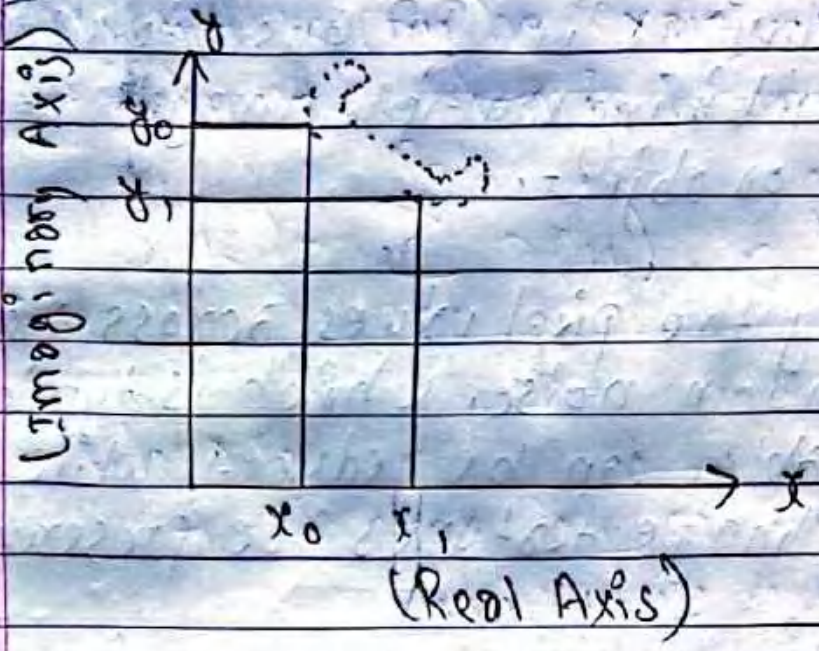


fig: A digital boundary & its representation as a complex sequence. Point (x_0, y_0) is selected arbitary or starting point. Point (x_1, y_1) is the next counter clockwise point in a sequence.

- Fourier Descriptor can be explained by using the following steps:

(1) Define the arbitary point (x_0, y_0) & the coordinate point or pairs (x_0, y_0), (x_1, y_1), (x_2, y_2) ... (x_{K-1}, y_{K-1}) are encountered in traversing the boundary in counter-clockwise direction.

(ii) These coordinates can be expressed in the form $x(k) = x_k$ & $y(k) = y_k$.
The boundary can be represented as the sequence of coordinates as:

$$s(k) = [x(k), y(k)] \text{ for } k = 0, 1, 2, \ldots K-1.$$

The each coordinate pair can be treated as a complex number so that $s(k) = x(k) + j\,y(k)$

(iii) The discrete fourier transform of 1D sequence, $s(k)$ can be written as:

$$a(u) = \frac{1}{K} \sum_{k=0}^{K-1} s(k) \cdot e^{-j 2\pi uk/K}$$

where

$$u = 0, 1, 2, \ldots K-1$$

(iv) The complex coefficient $a(u)$ are called the "Fourier Descriptors" of the boundary. The Inverse fourier transform of these coefficient is:

$$s(k) = \sum_{u=0}^{K-1} a(u) \cdot e^{j 2\pi uk/K}$$

(v) Therefore,
for boundary descriptors, the Fourier Descriptors can be explained as Fourier coefficient which is defined as:

$$\hat{s}(k) = \sum_{u=0}^{P-1} s(u) \cdot e^{j2\pi uk/P}$$

where,
P = Preceeding eqn for s(u)
where P = 0, 1, ... P-1

$\hat{s}(k)$ = Approximation to s(k).

Shape Number:

The shape number of a boundary generally based on four directions chain codes, is defined as the first difference of smallest magnitude.

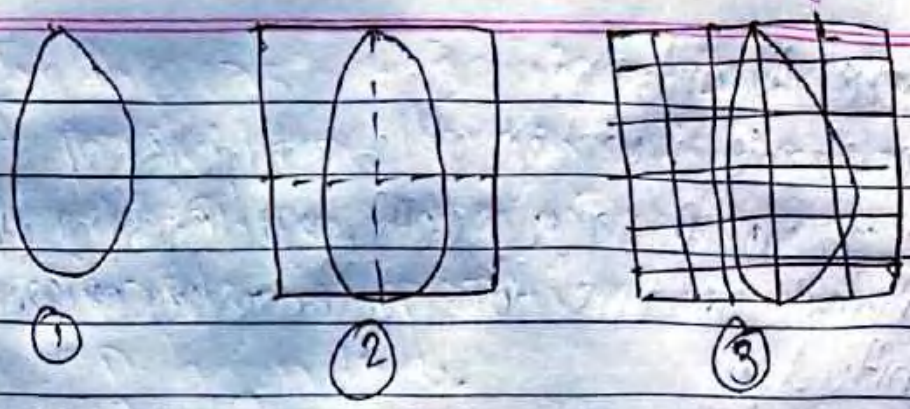
The order n of a shape number is defined as the number of digits in its representation.

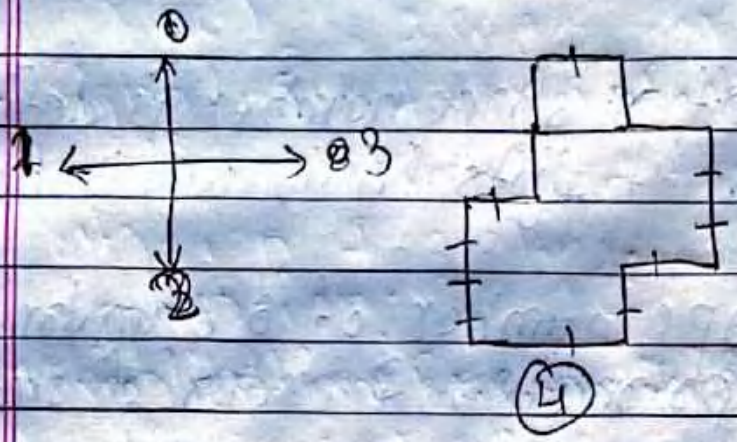
Thus the shape number of a boundary is given by the parameter & order of shape number length.

The four directional chain codes can be made insensitive to the starting point by using the integer of minimum magnitude & made rotation that are multiples of 90° by using the first difference of the code.

Thus, shape numbers are insensitive to the starting point & rotation that are multiple of 90°.

Shape number is illustrated step by step as:.





∴ chain code = 000 03003223 2221211

Difference = 3 0003131131 10031 0310

Shape no. = 0 0031 0330 13 0031303

(fig: Steps in the generation of shape number).

Application:

(i) Character Recognition

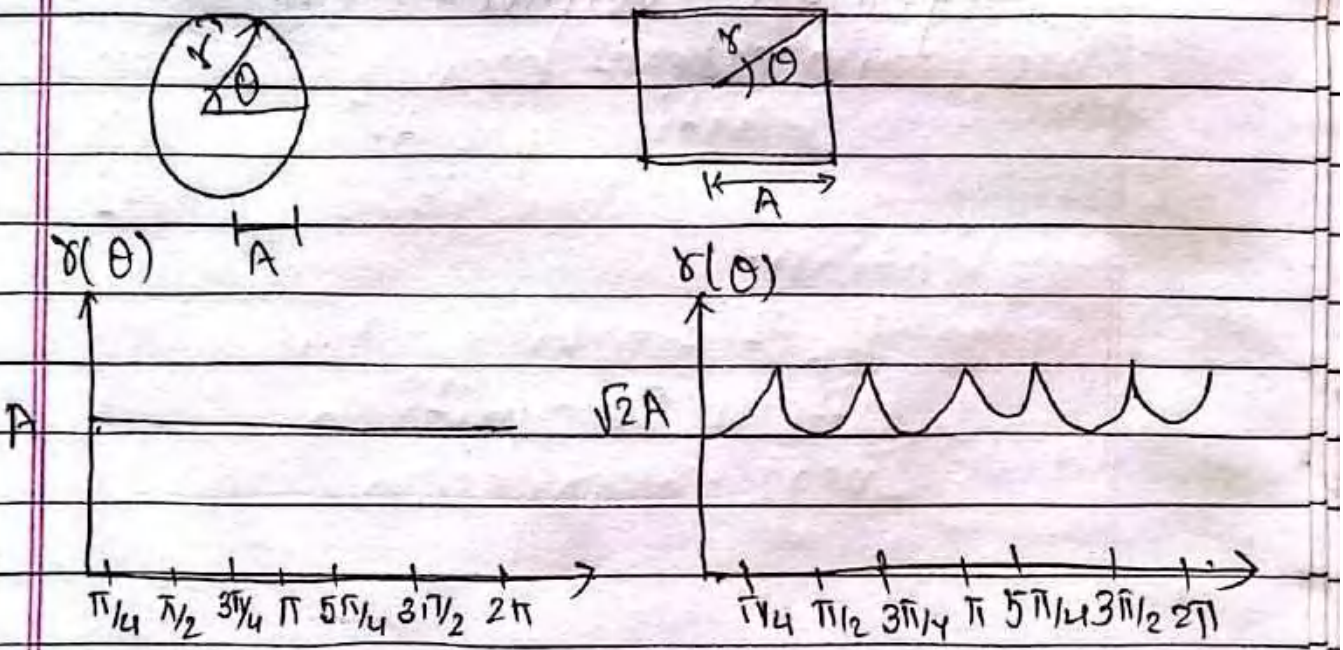
(ii) Symbol classification

Signature :

A signature is a 1-D functional representation of a boundary & may be generated in various ways.

One of the simplest way is to plot the distance from the centroid to the boundary as a fn of an angle.

This approach is invarient to translation but is dependent on rotation & scaling.



Here, In case of circle, the radius is constant throughout the angle & is shown in figure.

→ In the case of square & angle 45°, It posses the longer distance as compose at 90°. Thus, the graph/signature consists of repetation of pattern.

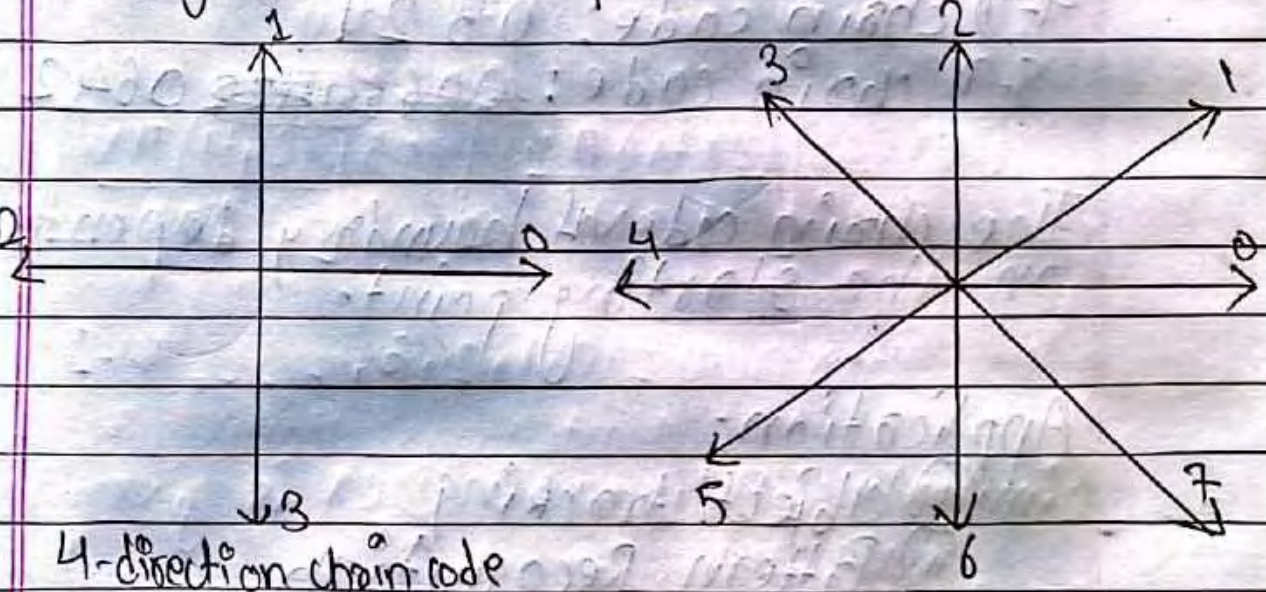
$$r(\theta) = \begin{cases} A\sec\theta & \text{for } 0 \leq \theta \leq \pi/4 \\ A\cos\theta & \text{for } \pi/4 \leq \theta \leq \pi/2 \end{cases}$$

Application:

(i) Gesture Recognition
(ii) logo Matching
(iii) Signature verification.

Chain Code:

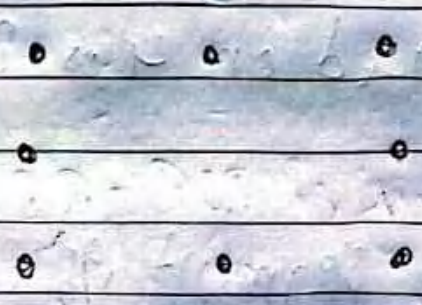
- It is used to represent a boundary by a connected sequence of straight-line segments of specified length & direction.
- Representation is based on 4 or 8 connectivity of the segments.
- Direction of each segment is coded by using a number sequence.



- Chain code of a boundary depends on the starting point.
- Boundary code is formed as a sequence of such directional numbers which often called

as freeman chain code.

Example:
Consider a square:



4-D chain code: 0321
8-D chain code: [redacted]0642

The chain code of boundary depends on the starting point.

Application:
(i) Object tracking
(ii) Pattern Recognition
(iii) Image compression

Discrete Cosine Transformation (DCT):-

- It is calculated using Fast Fourier Transform algorithm, so it is also called as Fast Transform.

- It is used for JPEG Image Compression. It does not have any windowing problem like in 2D-FT.

- The 1-D DCT is defined by eqn:-

$$C(w) = \alpha(w) \sum_{x=0}^{N-1} f(x) \cos\left[\frac{(2x+1)\pi w}{2N}\right]$$

for $w = 0, 1, 2, \ldots, N-1$

- Similarly, the inverse DCT is defined by

$$f(x) = \sum_{w=0}^{N-1} \alpha(w) C(w) \cos\left[\frac{(2x+1)\pi w}{2N}\right] *$$

for $x = 0, 1, 2, \ldots, N-1$

- where $\alpha(\omega)$ is defined as,

$$\alpha(\omega) = \begin{cases} \sqrt{1/N} & \text{for } \omega = 0 \\ \sqrt{2/N} & \text{for } \omega = 0, 1, 2, 3, \ldots, N-1 \end{cases}$$

→ The 2D-DCT is given by:-

$$C(\omega, v) = \alpha(\omega)\,\alpha(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y)$$

$$\cos\left[\frac{(2x+1)\pi\omega}{2N}\right] \cos\left[\frac{(2y+1)\pi v}{2N}\right]$$

for $\omega, v = 0, 1, 2, 3, \ldots, N-1$

Similarly 2D Inverse DCT is given by:-

$$f(x, y) = \sum_{\omega=0}^{N-1} \sum_{v=0}^{N-1} \alpha(\omega) \cdot \alpha(v) \cdot C(\omega, v)$$

$$\cos\left[\frac{(2x+1)\pi\omega}{2N}\right] \cos\left[\frac{(2y+1)\pi v}{2N}\right]$$

for $x, y = 0, 1, 2, 3, \ldots, N-1$.

- DCT is symmetric & separate transform.

Ch-6 Morphological Image Processing

→ Morphology is concerned with image analysis methods whose outputs describe image content (i.e. extract "meaning" from an image).

→ Morphological image processing is the study of shape & structure in digital images using mathematical operations.

* Morphological Operator:

↳ It is a fundamental image processing technique that modifies the structure of an objects in a binary or grayscale image.

↳ These operator works based on the shape & structure of object using a structuring element (kernel) to process the image.

→ A binary image is an image whose pixel values are 0 (representing black) or 1 (representing white i.e. 255).

Structuring element:

↳ It is represented by a matrix of 0s & 1s; for simplicity, the zero entries are often omitted.

↳ The origin of structuring element must be clearly identified.

Fitting:

↳ The SE is said to fit an image if, for each of its pixels that is set to 1, the corresponding image pixels is also 1

Hitting:

The SE is said to hit, or Intersect an image if, for any of its pixels that is set to 1, the corresponding image pixels is also 1.

Dilation:

It is a morphological operator used to grow or thicken objects in binary images. The dilation of a binary images A by a structuring element B is defined as:

$$A \oplus B = \{ z : (\hat{B})_z \cap A \neq \phi \}$$

This eqn is based on obtaining the reflection of B about its origin & translating (shifting) this reflection by z.

Then the dilation of A by B is the set of all structuring element origin locations

where the reflected & translated B overlaps with A by atleast one element.

Erosion

It is a morphological operator used to shrink or thin objects in binary images. The erosion of binary image A by a structuring element B is defined as

$$A \ominus B = \{z : (B)_z \cap A^c \neq \phi\}$$

The erosion of A by B is set of all structuring elements origin location where the translated B does not overlap with the background of A.

Opening Operation:

The opening operation erodes an images & then dilutes the eroded images using the same structuring elements for both operation i.e.

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

where A is the original images & B is the structuring elements.

The opening operation is used to remove region of an object that cannot contain structuring elements, smooth object contours, & break thin connections.

Ex:

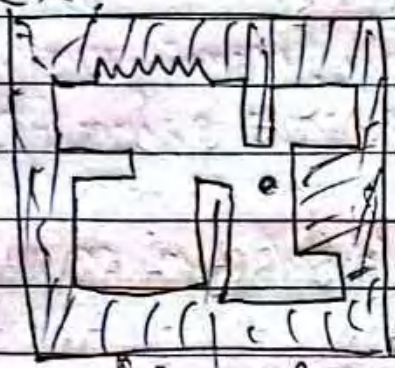


fig: original binary image

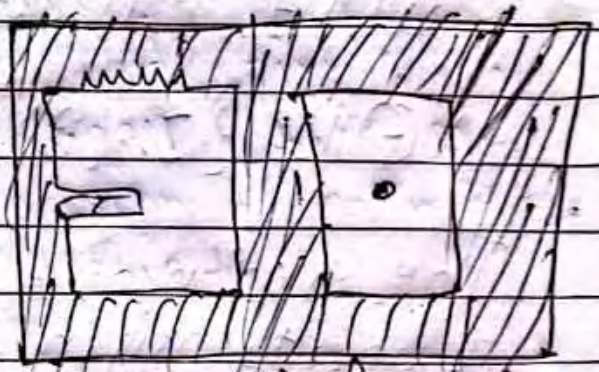


fig: Result of opening with square structuring element of size 20 pixels

Closing Operation:

The closing operation dilates an images & then erodes the dilated image using the same structuring element for both operation.

i.e.

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

where A is the original image & B is the structuring elements.

The closing operation fill holes that are smaller than the structuring elements, joint narrow breaks, fill gaps in contours & smoothes objects contours.

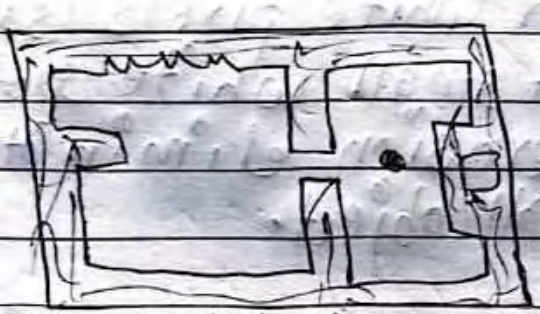


fig: Original Image

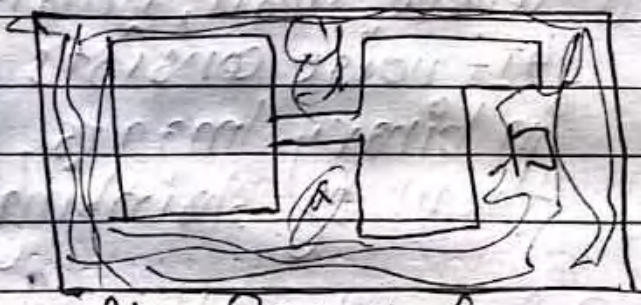


fig: Result of Closing with square structuring elements of size 20 pixels.

Define the morphological erosion & dilation operators. Which of these can be considered as a noise-reduction technique? why?

⇒ Erosion operation can be considered as noise reduction technique.

Erosion is a morphological operation that removes pixels from the boundaries of objects. It works by applying a structure elements to the image. A pixel is retained only if all pixels under the structuring element are part of the object otherwise, it is removed.

→ How it reduces noise?

① Remove small, isolated white pixels (salt noise)
↳ If noise consists of small white spots in binary images, erosion eliminates them by shrinking all object.

② Separates objects that are connected by thin lines.
↳ Erosion helps break narrow connections bet^n objects, which is useful for text recognition.

③ Reduces false edges:
↳ In grayscale images, erosion reduces minor variations by smoothing the edges.

→ Example:
Consider an image with small white noise (salt noise).

After applying erosion:
- small white dots (noise disappear).
- only larger object remains, but they becomes thinner.

Advs:
↳ Remove small noises particles effectively.
↳ Preserve overall shape while eliminating noise.
↳ useful for preprocessing in OCR & fingerprint analysis.

But, dilation is the opposite of erosion. It add pixels to object boundaries expanding shape of objects. It fills small black holes (pepper noise). It smoothens broken part of objects i.e. if an object has small gaps or breaks dilation connects them.

Chapter 5

lossy Compression & lossless Compression:

(1) lossy Compression:
- Reduces file size by permanently removing some data
- Commonly used for multimedia files like Images, audio & video.
- Achieves higher compression ratios, making files much smaller.
- often used in applications where minor loss in quality is acceptable, such as streaming services.
- once compressed, original data cannot be fully restored.
- Ex include JPEG, MP3, MP4 etc.
- offers faster upload & download times due to smaller file sizes.

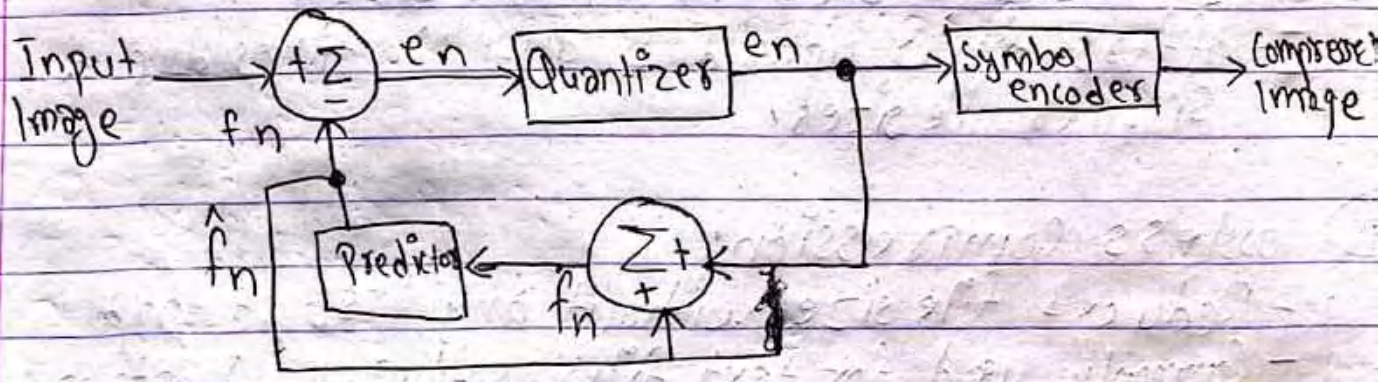
(11) lossless Compression:
- Reduces file size without any loss of data
- Commonly used for text files, software, & databases where data Integrity is crucial.
- Achieves lower compression ratios, resulting in larger file sizes compared to lossy methods.
- Preferred for applications requiring high quality, such as medial Imaging & archival storage.
- Original data can be fully restored after decompression
- Ex include PNG for Images, FLAC for audio, ZIP for files
- often used in scenarios where data accuracy & Integrity are more Important than file size reduction

Coding Redundancy:

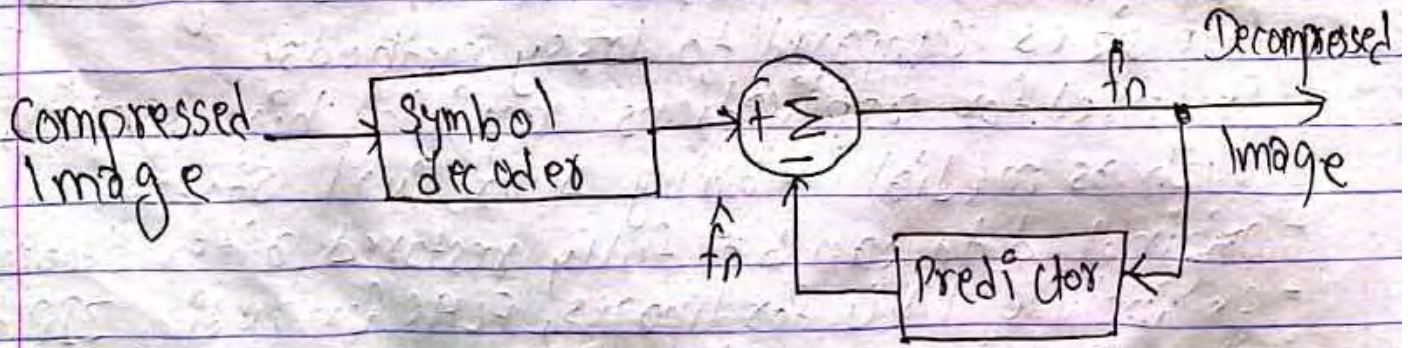
↳ Coding Redundancy relates to how Information is expressed through codes representing data, such as the gray levels within an Image.

↳ When these codes use excessive symbols to represent each gray level, more than what's required, the resultant Image is described as having coding redundancy.

Working Principle of lossy Predictive Coding:



(a) encoder



(b) Decoder

lossy predictive coding is a type of data compression technique commonly used in audio & video compression.

The key idea is to predict the value of each data point based on its context & then encode only the difference (or error) betⁿ the actual value & the predicted value.

Working:

(i) For each data point, a prediction is made based on previous data points.

(ii) The difference betⁿ the actual data point & the predicted value is calculated. This difference is called the prediction error or residual.

(iv) The prediction error is then quantized, which means it is approximated to reduce the number of bits required to represent it. This step introduces the lossy aspect of the compression as it can lead to some loss of information.

(v) The quantized prediction error is then encoded using an entropy coding scheme like Huffman coding.

(vi) To construct the original data, the decoder uses the encoded prediction error & the same prediction method to estimate the original data points.

The quantizer absorbs the nearest value of a prediction error & maps it into a limited range of output denoted by e_n at encoding & f_n at decoding, which is defined as:

e_n is given by:

$$e_n = f_n - \hat{f}_n$$

& , f_n is given by:

$$f_n = e_n + \hat{f}_n$$ ##

Why is Image Compression necessary?

(i) Storage efficiency
(ii) Transmission speed
(iii) Cost Reduction
(iv) Performance
(v) Memory usage.

Run length coding:

It is a very simple form of the data compression which represents each subsequence of identical symbols by a pair like (L, a) where L, is the subsequence & a is the recurring symbol.

Run-length coding describes each row of the image by a sequence of length that describes successive runs.

for ex:, aaabbbaaa is coded as 3a3b3a.

The most common method are as follows:
(i) To specify the value of the first run of each row.
(ii) To assume that each row begins with a white run.

for ex :- 
black(1) white(2) black(3)

from above figure, the run length cod can be written as: 1 black 2 white 3 black.

In this technique, the numeric run length coding can also be defined as:

00000 222111 = 0 5 2 4 1 3

Algorithm:

(i) Initialize an empty result string.

(ii) Set a counter count to 1.

(iii) Traverse the input string from the second element to the end.

↳ If the current element is equal to the previous element:

↳ Increment the counter count.

↳ Else:

- Append the previous element & counter count to the result string.
- Reset the counter count to 1.

(iv) Append the last character & it's count to 1.

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

Example:

Run-length Pair

= (5,0), (3,0), (2,1)
(5,1), (5,1), (5,1)

No. of pairs = 6

Max length = 5

↓

3 bits in binary

No. of bits per Pixel = 1 (0 or 1)

Total no. of pixels = 6 * (3+1)

= 24

No. of pixels for original Image = 5 * 5 = 25

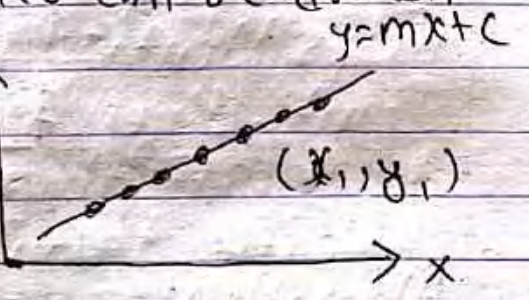
Compression ratio = $\frac{25}{24}$ = 1.042 : 1 #

Hough Transform :-

- It is a feature extraction method for detecting simple edges shapes such as circle, lines etc in an image.
- Takes images created by edge detection operators but most of the time, edge map is disconnected.
- Therefore Hough Transform is used to connect the disjoint edge points.
- The eqn of line

$$y = mx + c.$$

- Problem is that, Infinite lines can be drawn connecting these points.



∴ an edge point in x-y plane is transformed to the c-m plane.

let point (x_i, y_i), then

$$y_i = mx_i + c$$

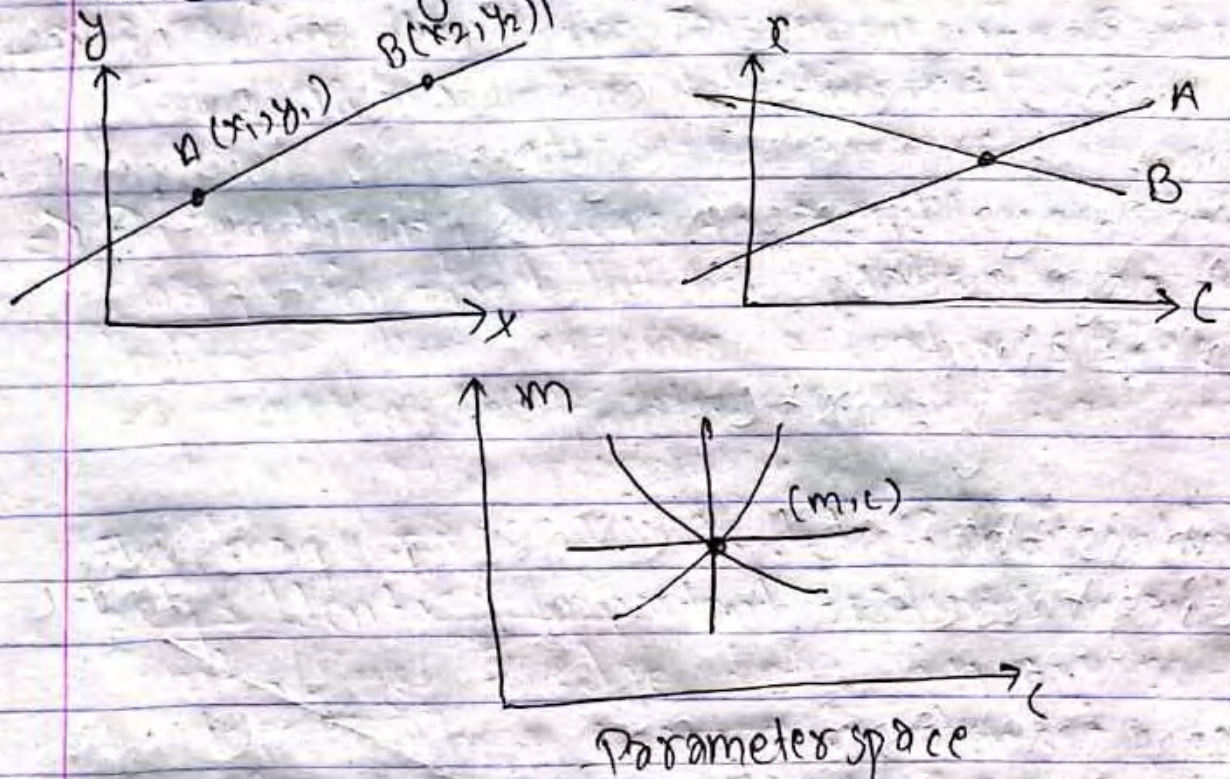
Or, $\boxed{c = -x_i m + y_i}$

Now any edge point in x-y plane can be written in c-m plane.

→ A common intersection point indicates that the edge point are the part of the same line.

→ Parameter space is also called Hough space.

→ If A & B are two point connected by a line in spatial domain, they will be intersecting line in Hough space.



Algorithm:

1. load the Image
2. Determine the Image edge using any edge detector.
3. quantize the parameter space P.
4. Repeat the process for all pixels
   If the pixel is an edge pixel then,
   $c = -x(m) + y$
   $P(c,m) = P(c,m) + 1$
5. Show the hough space.
6. Find the local maxima in parameter space.
7. Draw the line using local maxima.

Ch-7

→ Image Segmentation:

Image segmentation is the process by which a digital image is partitioned into various subgroups (of pixels) the complexity of the image, & thus analyzing the images becomes simpler.

→ level of subdivision depends on the problem being solved.
→ Segmentation stops when objects of Interests is an application have been isolated.

Two approaches based on the image properties:
(i) Similarity Detection.
(ii) Discontinuity Detection

(i) Similarity Detection:
↳ relies on detecting similar pixels in an images based on a threshold, region growing, region Spreading & region merging.

(ii) Discontinuity Detection:
↳ opposite of Similarity Detection approach where the algorithm rather search for discontinuity.

Edge detection, Points detection, line detection follows this approach.

Region Based Segmentation:

(Q) Define the term Discontinuity & Explain Point, line & edge detection methods.

=> Discontinuity refers to a sudden or significant change in pixel Intensity values in an Image, Indicating the presence of features like edges, lines or points.

In Image processing, detecting discontinuities helps Identify Important features in an Image. Point, line & edge detection are methods to locate these changes in Intensity using Specific techniques, often Involving masks or filters.

(1) Point Detection:
- Point detection finds Isolated pixels with a sharp Intensity difference from their neighbors, such as single bright, or dark spot. Used to detect small, isolated features like noise.
- Working:
A mask (e.g. Laplacian filter) is applied to the Image. The mask checks the difference between a pixels Intensity & the average of its neighbors.

Process:
- convolve the image with a mask like:

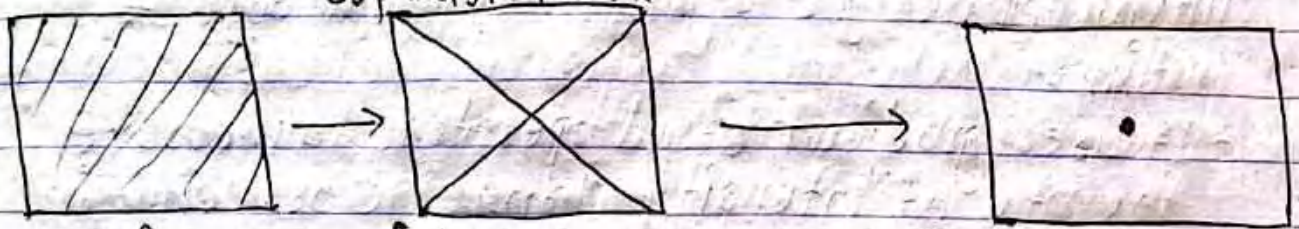


fig: Point detection.

- If the result exceeds a threshold (eg; |R|>T), the pixel is marked as point.

② Line Detection:

- It identifies thin, linear structures in an image, such as horizontal, vertical or diagonal lines.
- Specific masks are designed to detect lines in different orientations by emphasizing intensity changes along a line while supressing others.
- Useful in road detection or structural analysis.

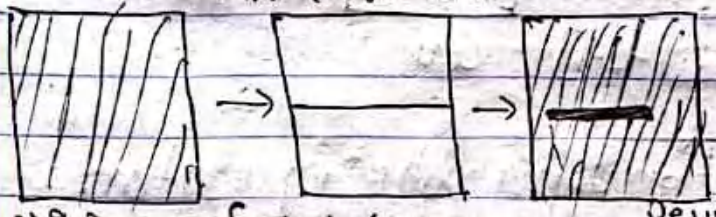


fig: line detection for horizontal lines.

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

fig: Vertical line detection mask

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

fig: -45° line detection mask.

③ Edge detection:

- It locates boundaries between regions where intensity changes significantly, like object outlines.
- It uses gradient based operators to measure the rate of intensity change across neighboring pixels.
- It is used to identify object boundaries in complex scenes.
- Common methods include Sobel, prewitt, or Roberts operators.

For Sobel:

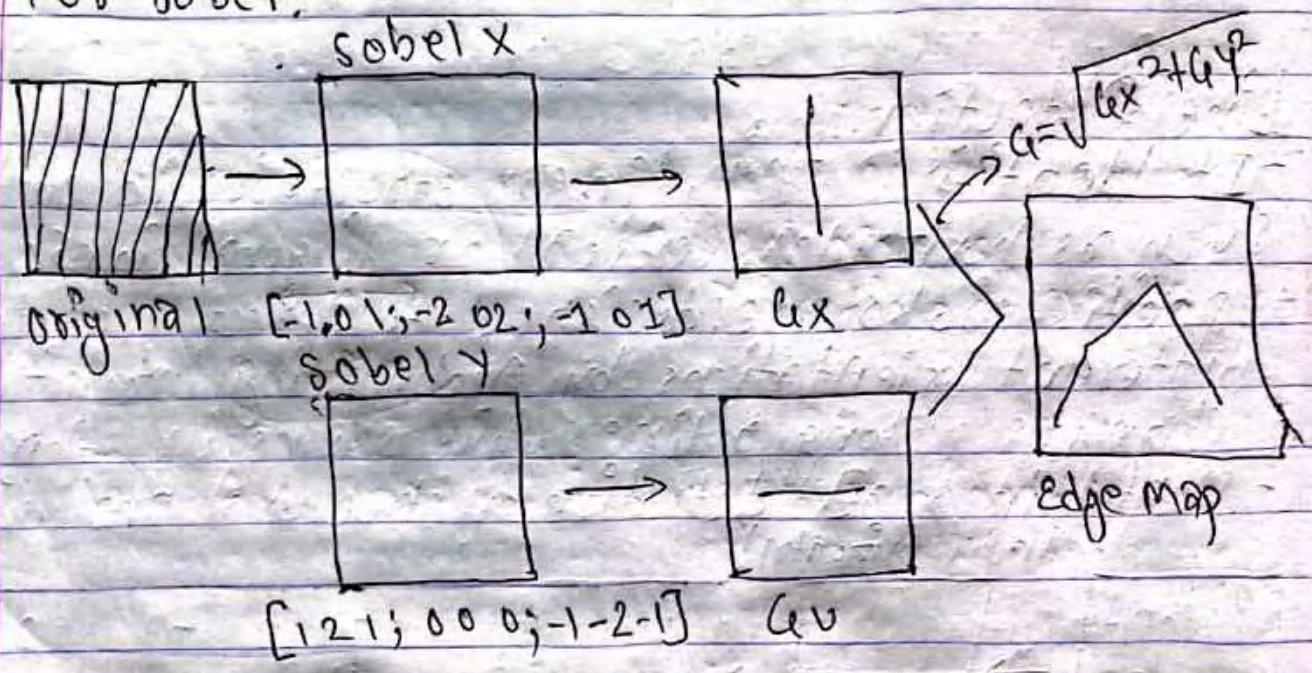
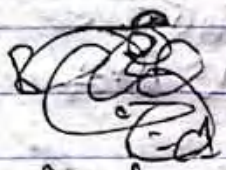


fig: Edge detection using Sobel operator.

Region Based Segmentation:

- Based on sets
- Each Image is a set of region R,
  - every pixels belongs to one region.
  - One pixel can only belong to a single region.



$R = \bigcup_{i=1}^{S} R_i$, R_i Intersect $R_i = \phi$

Basic formulation:

(i) Every pixel must be in a region
(ii) Points in a region must be connected.
(iii) Region must be disjoint
(iv) Different region have different properties.

→ Why region based segmentation:
↳ Edges detection & thresholding not always effective
↳ Region-based segmentation effective in every Image.

Edge linking & Boundary detection:

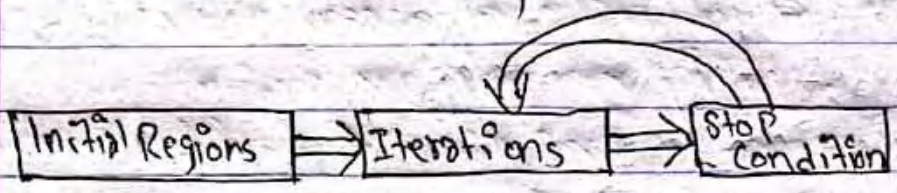
Edge linking & boundary detection are techniques used to detect & connect edges in an Image, forming meaningful object boundaries.

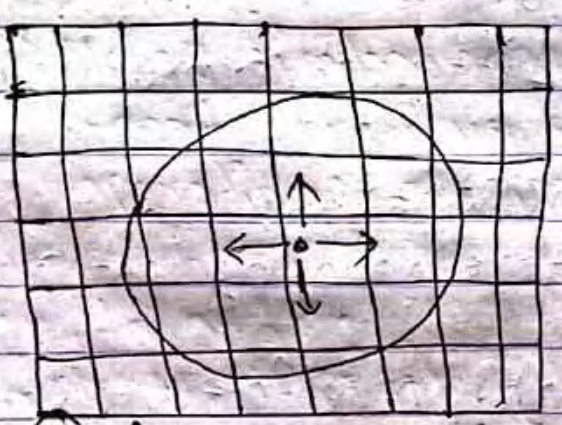
P.T.O
→

Region Growing:

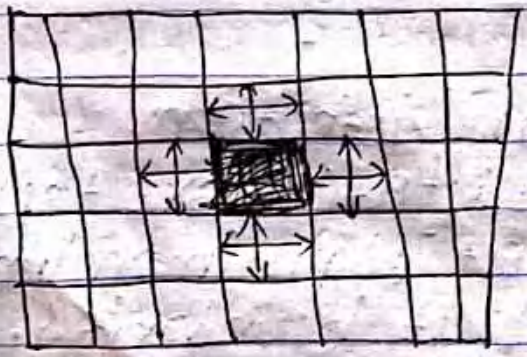
It is a pixel based segmentation technique that groups pixels into larger region based on predefined similarity criteria.

It starts with seed points & expand the region by adding neighbouring pixels that meets a similarity condition.





(a) fig: start of growing a region.



(b) Growing process after a few Iteration.

Algorithm:

(1) Choose seed points
   - manually selected by user
   - chosen based on Intensity, colour of texture.
(2) Define a similarity criteria (e.g. Intensity difference < threshold)
(3) Add neighbouring pixels that meet the criteria.
(4) Continue until no more pixels satisfy the condition.

Pseudocode:

(1) Region Growing (Image, seed, threshold).
(i) Initialize an empty list 'region' to store region pixels.
(ii) Initialize a queue & add the 'seed' pixel to it.
(iii) Mark 'seed' as visited
(iv) While 'Q' is not empty
   (a) Remove a pixel 'p' from 'Q'
   (b) Add 'p' to region.
   (c) for each neighbour 'N' of 'p'
      i) If 'N' is not visited ||Intensity(N) - Intensity(seed)|| <= threshold:
         → Add 'N' to Q
         → Mark 'N' as visited

(v) Return the 'region' containing the segmented area.

Region Merging:

It is a top down Image segmentation technique where an Image is Initially divided into many small region (usually one per pixel) & then similar adjacent regions are merged Iteratively based on similarity criteria.

Algorithm:

(i) Divide Image into an Initial set of regions.
   - One region per pixels.
(ii) Define a similarity criteria for merging regions.
(iii) Merge similar regions.
(iv) Repeat previous step until no more merge operations are possible.

Pseudo code:

Region Merging (Image, Threshold):

(i) Treat each pixel as Individual region
(ii) Define a similarity criteria (e.g. Intensity differences Thresholding).
(iii) Set 'merged = True' (to track if merging occurs).
(iv) while 'merged' is True:
   (a) set 'merged = False'
   (b) For each pair of adjacent region (R_1, R_2).
      i) If similarity (R_1, R_2) $\leq$ threshold:
         - merge R_1 & R_2 into a single region.

- set 'merged = True'

(v) Return the final segmented image.

Applications:

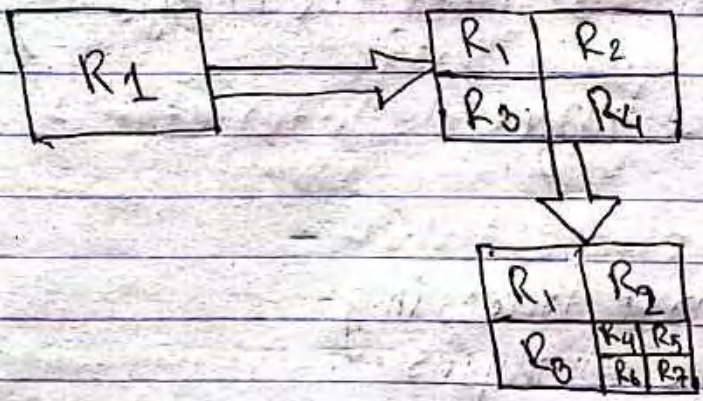
(i) Medical Imaging

(ii) Satellite Image Processing

(iii) Object Recognition.

Region Splitting:

Region splitting is a top down image segmentation technique where an image is recursively divided into smaller regions until each region meets a defined homogeneity criteria.



Algorithm:

(1) One initial set that includes the whole image.

(2) Define homogeneity criterion.

- Intensity variance.

- color uniformity
- Texture variation.

③ Iteratively Split regions into sub-regions.
④ Stop when no more splittings are possible.

→ Pseudocode:
Region-splitting (Image, Threshold):

① Treat the entire Image area single region.
② Define a homogeneity criterion (e.g. Intensity difference & threshold).
③ If a region does not satisfy the criterion (e.g. Intensity difference & threshold)
   (a) Split the region into four sub region
   (b) Recursively check each sub region.

④ Stop when all regions meet the homogeneity condition
⑤ Return segmented Image.

Application:
① Document Image Processing
② Medical Image Analysis.

Thresholding:

Thresholding is usually the first step in any segmentation approach. It works by setting a threshold value & classifying pixels as either foreground or background based on their intensity values.

↳ The threshold value is chosen following some criteria or feature of an image.
↳ Then every pixel of an image is compared to threshold values.
↳ On comparison all the pixels having intensity value greater than the thresholding value belong to one region & the rest pixels belongs to other region.

→ Grayscale Image:

A grayscale image is an image where each pixel has a single intensity value, typically ranging from 0 to 255. Here 0 represents black, 255 represents white & values in between represent shades of gray. Segmentation means dividing the image into different regions based on these intensity values.

→ Intensity thresholding:

Intensity thresholding is a method to segment a grayscale image by comparing the intensity of each pixel to one or more threshold values. It seperates the image into distinct regions, like foreground & background, based on these comparisons.

Single Thresholding (Global Thresholding):

In single thresholding a single intensity value T is ~~either~~ chosen & all pixels in the images are classified into two groups:

Foreground → pixels with higher intensity greater than T.
Background → Pixels with intensity less than or equal to T.

Mathematical Representation:

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

where
$g(x,y)$ is segmented image
$f(x,y)$ is the grayscale intensity at pixel (x,y)
T is the threshold value

Example:
Consider a grayscale image where objects have high intensity (e.g. white) & the background is dark. If we set T=128 (assuming pixel intensity range is 0-155).
Pixels with intensity above 128 → white
" " " below 128 → black

Multiple Thresholding (Multi-level Thresholding):
when an Image contains multiple objects with different Intensity levels, a single Threshold may not be sufficient. Multiple thresholding uses two or more threshold values $T_1, T_2, T_3 \ldots T_n$ to divide the Image into multiple regions.

Mathematical representation:
For two thresholds T_1 & T_2:

$$g(x,y) = \begin{cases} 0 & \text{if } f(x,y) \leq T_1 \text{ (Background)} \\ 1 & \text{if } T_1 < f(x,y) \leq T_2 \text{ (object 1)} \\ 2 & \text{if } f(x,y) > T_2 \text{ (object 2)} \end{cases}$$

Examples:
If a medical Image contains bones, muscles & fat tissues:
$T_1 = 80$ (Fat)
$T_2 = 150$ (muscles)
$T_3 = 200$ (bones)

⇒ Methods for selecting multiple thresholds.
↳ Otsu's methods
↳ K-means clustering.

Basic Global Thresholding

↳ Based on the histogram of an Image
↳ Portition the Image histogram using a single global threshold.
↳ The success of this technique very strongly depends on how will the histogram can be portitioned.

→ Algorithm:
The basic global threshold, T is calculated as follows:
(i) Select an initial estimate for T (typically the average gray level in the image).
(ii) Segment the Image using T to produce two groups of pixels i.e. G1 pixels with grey levels $\geq T$ & G2 pixels with grey levels $< T$
(iii) compute the average gray levels of pixels in G1 to give μ_1 & G2 to give μ_2
(iv) Compute a new threshold value:

$$T = \frac{\mu_1 + \mu_2}{2}$$

(v) Repeat step 2-4 until the difference in T in successive Iterations is less than a predefined limit T.

Methods of Selecting Threshold value.

(i) Manual Thresholding
(ii) Histogram-based methods
(iii) Otsu's method (optimal thresholding)
(iv) Clustering-Based Methods (K-Means).
(v) Entropy Based Methods
(vi) Adaptive thresholding

(1) Manual Thresholding:

A predefined threshold value is selected based on prior knowledges of the Image.

How it works:

- The user manually sets a threshold T
- All pixels with intensity greater than T are classified as foreground & others as background.

Example:

If an Image has done background (0-100) Intensity & bright objects (150-255 Intensity) a threshold like T=128 can separate them.

Disadvantages:

- Require manual tuning for different Images
- Not suitable for Images with variable lighting.

(10) Histogram based Methods:
- Uses the intensity distribution (histogram) to find an optimal threshold.
- If the histogram has two peaks (bimodal distribution), the threshold is set at the lowest point (valley) between them.
- The threshold is set at the average or median intensity between these two peaks.

Ex:
If the histogram has peaks at 50 & 200 the threshold can be

$$T = \frac{50 + 200}{2} = 125$$

Disadvantages:
- fails if the histogram does not have clear peaks.
- struggles with multimodal images.

(11) Otsu's Method:
- Otsu's method automatically selects an optimal threshold by minimizing intra-class variance (variance within object & background)

How it works:
- Compute the intra class variance for different thresholds.
- Selects the threshold where variance is

minimum, ensuring maximum separation.

Examples:
for a grayscale image, otsu's method finds threshold automatically based on intensity distribution.

Disadvantages:
- Fails in images where objects background intensities overlap.
- Computationally expensive.

Basic Adaptive Thresholding :

Adaptive thresholding is a way to process grayscale images when lighting or contrast isn't the same everywhere.

Adaptive thresholding picks a different value for each pixel based on its nearby area. This makes it great for images with shadows or uneven light.

Algorithm.

Consider a grayscale image I of size M×N.

(i) Select a neighbourhood for each pixel (x,y)

(ii) Use a square window of size w×w centered at (x,y).

(iii) Calculate the local threshold by computing the average intensity of all pixels within the window. This value is the local threshold T(x,y).

(iv) Classify the pixel:

  (i) If I(x,y) > T(x,y), set B(x,y) = 255 (foreground)

  (ii) Otherwise, set B(x,y) = 0 (background).

(v) Repeat the process for every pixel in the Image.

(vi) Handle border pixels by skipping them, using a smaller window or padding the image with extrapolated values. #

→ Edge detection methods for Image segmentation.

→ Edge detection methods are aimed at identifying the boundaries of objects within an Image.
→ These boundaries or edges help separate different objects or areas for segmentation.

Methods:
(i) Gradient based Methods / Filters → Seperate long an asked Give different first order derivative based gradient filters
(ii) Canny Edge detector. → IMP (Most asked).

(i) Gradient based Methods / Filters: *****
- Detects edges by identifying areas where the Image Intensity changes rapidly.
- These filters measure the gradient, or the first order derivative of the Image Intensity.

→ Working (detecting edges with gradient based filters):-
(i) Apply horizontal & vertical gradient filters (e.g. sobel).
(ii) Calculate the Gradient magnitude using $G = \sqrt{G_x^2 + G_y^2}$.
(iii) Threshold the gradient magnitude to identify edges.

Gradient Based filters Include:
(i) Sobel Operator / Filters:
- Uses first order derivatives to calculate the gradient of Image Intensity.
- Highlights edges by detecting high gradients in horizontal & vertical directions.
- Applies convolution masks to the Image to compute the gradient in horizontal & vertical directions.

↳ Horizontal mask (G_x):

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

↳ Vertical mask (G_y):

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

↳ Gradient magnitude = $\sqrt{G_x^2 + G_y^2}$

(2) Prewitt Filter:

- Similar to sobel operators.
- Uses simpler convolution masks to detect edges in horizontal & vertical directions.
- Easy to use.

↳ Horizontal mask (G_x):

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

↳ Vertical mask (G_y):

$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

↳ Gradient magnitude = $\sqrt{G_x^2 + G_y^2}$.

(3) Roberts Cross Filter:
- Another first order derivative method.
- Detects edges by calculating the difference between diagonally adjacent pixels.
- Applies 2×2 convolution mask for edge detection.

Kernel 1: $\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

Kernel 2: $\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$

↳ Gradient Magnitude: $\sqrt{G_1^2 + G_2^2}$ //

******

(2) Canny Edge Detector (2nd method for edge detection):
- A multistep Algorithm that combines several techniques for robust edge detection.

Steps:
- Smooth the Image using a Gaussian filter to reduce noise.
- Calculate the gradient magnitude & direction.
- Apply non-maximum suppression to thin out the edges.
- Use double thresholding to Identify strong & weak edges.
- Perform edge tracking by hysteresis to finalize the edge detection.

Gaussian Kernel: $G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$ //

(1) What are the problems associated with the first & 2nd order derivative filters for edge detection during segmentation? Explain with suitable example.

⇒ First order derivative filters:
First order derivative filters measure the rate of intensity change (gradient) in an image. They detect edges by highlighting area where intensity changes significantly.

Problems:
(1) Sensitivity to Noise:
- These filters amplify small intensity variations including noise, which can lead to false edges.

Ex: In a noisy grayscale image of a face, random pixel variations (noise) might be detected as edges, making the outline of the face uneven with extra unwanted lines.

(2) Thick edges:
- They produce edges that are wider than necessary because they respond to gradual intensity changes, not just sharp ones.

Example: For an image of a square object, the edge might appear as a thick band instead of a crisp single pixel line, complicating precise segmentation.

(2) Direction Dependency:
Filters like sobel calculate gradients in specific directions (eg. horizontal or vertical), so they may miss edges in other orientations unless combined.
Ex:- A diagonal edge in an image might be weak or undetected if only horizontal Sobel is applied.

2) Second Order derivative filters:
Second order derivative filters eg. Laplacian measure the rate of change of the gradient, detecting edges where the gradient crosses zero (sharp transitions).

Problems:
(1) High Sensitivity to Noise:
- These filters are even more sensitive to noise than first order filters because the amplify small fluctuations twice (second derivative).
Ex:- In an image of a smooth road with slight noise, the laplacian might highlight noise speckles as edges instead of just the road boundaries.

(11) No edge detection information:
They detect where intensity changes occur but don't indicate the direction or strength of the edges making it harder to interpret.
Ex:- for a circular object, the laplacian marks the edge but doesn't tell if the intensity increases inward or outward, confusing segmentation.

(iii) Double Edges:
- They produce positive & negative responses on either side of an edge (zero-crossing), which can create double lines instead of single edges.
Ex:- In an image of a wall against a sky, the edge might appear as two parallel lines (one positive, one negative) instead of one clean boundary.

→ Ex:-
Consider an image of a white circle on a gray background with some noise:

Sobel (first order): Detects the circle's edges but adds noise specks & makes the edge thick.

Laplacian (second order): Marks the edge sharply but shows double lines around the circle & highlights noise as false edges. //

Noise PDF used in IP
or
Noise Models in an Image with their Probability Density function.

(I) Gaussian Noise:
It arises in an Image due to factors such as electronic circuit noise & sensor noise due to poor Illumination or high temperature.
It causes smooth variations in pixel intensity.

PDF:
$$P(z) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(z-\mu)^2}{2\sigma^2}}$$
σ = standard deviation.
z = Pixel value
μ = Mean (often 0)

(II) Salt-&-Pepper noise:
- This noise shows up as random black & white dots, like salt & pepper due to sudden signal disruptions or faulty sensors.

PDF:
$$P(z) = \begin{cases} P_a & \text{if } z = a \text{ (e.g. 0)}, \\ P_b & \text{if } z = b \text{ (e.g. 255)}, \\ 0 & \text{otherwise} \end{cases}$$
where, P_a, P_b : Probabilities of low/high values.

(III) Uniform noise:
- Uniform noise spreads evenly across range, less common in real Images but seen in quantization errors.

PDF:
$$P(z) = \begin{cases} \frac{1}{b-a} & \text{if } a \le z \le b \\ 0 & \text{otherwise} \end{cases}$$
where;
- a, b = Range limits.

(4) Exponential Noise:
Exponential Noise, attentied up to photon counting, has skewed shape & affects Imaging systems like X-rays.

PDF: $P(z) = \begin{cases} \lambda e^{-\lambda z} & \text{if } z \geq 0, \\ 0 & \text{if } z < 0 \end{cases}$

where; λ = Rate Parameter

(5) Rayleigh Noise:
Found in range Imaging (e.g. radar), Rayleigh noise comes from the magnitude of signals with gaussian components.

PDF: $P(z) = \begin{cases} \frac{z}{\sigma^2} e^{-\frac{z^2}{2\sigma^2}} & \text{if } z \geq 0, \\ 0 & \text{if } z < 0. \end{cases}$

where, σ : scale parameter.

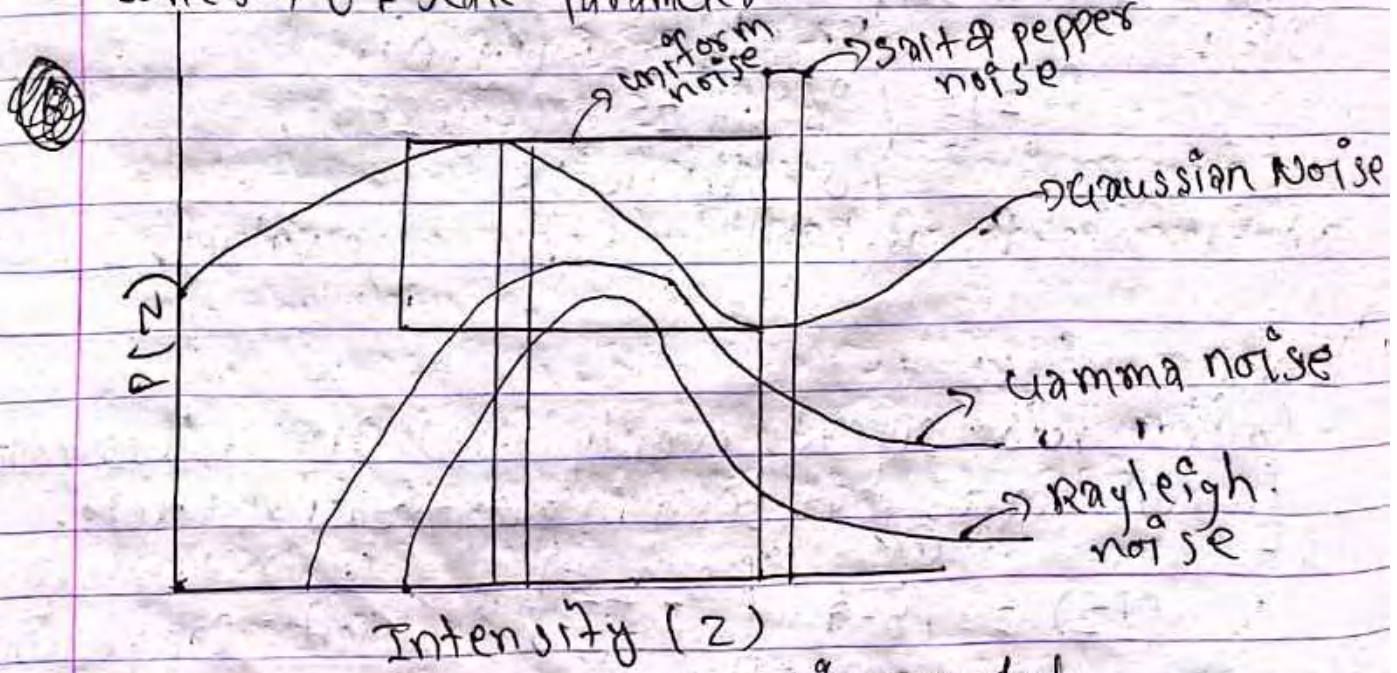


fig: PDF for Image Noise models.

Salt & Pepper Noise:

Salt & pepper noise is a type of digital image noise that appears as randomly occurring white & black pixels scattered throughout an image.

i) Salt noise: The white pixels that appear brighter than surrounding pixels.
ii) Pepper noise: The black pixels that appear darker than the surrounding pixels.

Q) Which spatial filter is used to reduce salt-and-pepper noise? Why?

=) The Median filter is a non-linear filter used to remove salt & pepper noise by replacing each pixel value with the median value from its surrounding neighbourhood.

working:
- Consider a window W (commonly a 3x3 neighbourhood) around a pixel at (x, y).
- The filter sorts all pixel values in the window & selects the median value.
- mathematically the filtered image $I'(x,y)$ is given by:

$$I'(x,y) = \text{median}\{I(s,t) \mid (s,t) \in W_{xy}\}$$

where $I(s,t)$ are the pixel values in the window W_{xy}.

→ Why?

↳ The median value is not affected by extreme values (ie. the salt (max value) & pepper (min value) noise) which are outliers.

↳ Unlike linear filters that average values (which can blur edges), the median filter preserves sharp transitions because the median operation does not create new pixel values.

↳ This makes the median filter effective in removing impulse noise while maintaining the important details of the image.

Example:

consider a 3X3 window in an image where the pixel intensity values are;

100	102	103
101	255	104
100	102	101

Here, the pixel with intensity 255 represents salt noise (an outlier). To apply the median filter

(1) list the values:

100, 102, 103, 101, 255, 104, 100, 102, 101

(2) Sort the values:

100, 100, 101, 101, 102, 102, 103, 104, 255

(3) Find the median

The median is 102.

Thus, the central pixel (which was 255) is replaced by 102. This process effectively removes the noise while preserving the details of the image.

Adaptive Mean Filter:

The adaptive mean filter is a noise-reduction technique that adapts its smoothing based on local image statistics.

Instead of applying a fixed average (mean) over a set neighbourhood, it computes the local mean & variance within a window around each pixel & then uses these values to decide how much smoothing is appropriate at that location.

Filtering Process:
- At each pixel, compute the local mean $\mu(i,j)$ & local variance $\sigma^2(i,j)$ in a predefined window.

- The filter then adjusts the output pixel value $f(i,j)$ using a formula that typically resembles

$$f(i,j) = \mu(i,j) + \frac{\sigma^2(i,j) - \sigma^2_n}{\sigma^2(i,j)} [g(i,j) - \mu(i,j)]$$

Here,
- $g(i,j)$ is the original pixel value.
- σ^2_n is the estimated noise variance.

- The filter reduces the amount of smoothing to preserve important details.
If the local variance is zero (indicating a completely uniform area), the output is set to the local mean to avoid division by zero.

Q) What are the main sources of noises in digital image? How can you use harmonic & contraharmonic mean filters to remove noise from an image?

=> Image noise can come from various sources including:

(i) Sensor noise:
Issues with the camera or sensor of scanner can introduce noise.

(ii) Environmental Conditions:
Poor lighting or electronic interference can add random variations

(iii) Quantization Noise:
Occurs during the conversion of analog signals to digital form.

(iv) Transmission Errors:
Errors during the transmission of images can introduce noise.

To remove noise from images, harmonic & contra harmonic mean filters can be used.

(1) Harmonic mean filter:
The harmonic mean filter computes the output pixel as the harmonic mean of the intensities in a neighborhood.

$$H = \frac{n}{\sum_{i=1}^{n} \frac{1}{g_i}}$$

where n is the number of pixels in the window & g_i is the intensity of each pixel.

Excellent for Gaussian noise, as it smooths random fluctuations effectively.

(11) Contra-Harmonic Mean Filter:

The contra Harmonic mean filter is the generalization of the harmonic mean filter.

It uses a power-based formula with an order parameter Q.

$$CH = \frac{\sum_{i=1}^{n} g_i^{Q+1}}{\sum_{i=1}^{n} g_i^{Q}}$$

where Q controls the filter's behavior.

Q>0: emphasizes larger values

Q<0: emphasizes smaller values

Q=0: Reduces to the arithmetic mean.

Q=1; Becomes the harmonic mean (sort of a special case).

It can handle salt & pepper noise by tuning Q based on the dominant noise type.

Q) Differentiate betⁿ smoothening & sharpening filters. Explain some basic gray level transformation techniques used for image enhancement in spatial domain.

⇒

Smoothening filters	Sharpening filters
(i) Reduce noise & small intensity variations	(i) Enhance edges & fine details.
(ii) Average pixel values in a neighbourhood.	(ii) Amplify differences betⁿ pixels.
(iii) Tend to blur edges & details.	(iii) Can introduce noise if over applied.
(iv) Useful for removing Caussain or salt-and-pepper noise.	(iv) Useful for highlighting boundaries.
(v) Ex:- mean filter	(v) Ex:- laplacian filter.

Basic gray level transformation techniques:

(1) Linear Transformation:
- Changes pixel brightness using formula.
  $S = L - 1 - r$
- Enhances white or grey details in dark regions, like x-Rays.
- used to stretch contrast by spreading out pixel values.
- Ex:- Converting dark, low-contrast x-Ray images to clearer ones.

(ii) Logarithmic Transformation:
- Formula: $S = c \log(1+r)$
- makes dark areas brighter while keeping bright areas from becoming too bright
- compresses the range of very bright pixel values.
- Useful when Image has very bright & very dark areas together.
- Ex: Making details visible in both shadows & highlights of a photo.

(iii) Power-law (Gamma) Transformation:
- Formula: $S = c \times r^{\gamma}$
- $\gamma < 1$ brightens darker regions (commonly used)
- $\gamma > 1$ darkens mid-tones & enhances bright regions.
- fixes Images that look too dark or too light on screen.
- Ex: correcting Images taken in poor lighting conditions.

Others are:
(i) Piecewise-linear Transformations
(iv) Histogram Equalizations.
(ii) Bit-plane slicing
(iv) Image Negatives.

Q) What are the uses of derivative based filters?

⇒ - Detect edges wher pixel Intensity changes rapidly.
- Highlight boundaries betn different objects.
- Enhance fine details & texture in images.
- Support corner detection & texture extraction
- Identify organ boundaries in medical Imagin

Laplacian filter

⇒ IMP

Q) Derive mask for laplacian second order derivative based.

⇒ Here,
The derivative operator laplacian for an Image is defined as:

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

for X-direction,

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

for y-direction,

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

By substituting eq^n's of x & y direction in derivative operator we get;

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

The equation represented in terms of mask: (4-neighbor)

0	1	0
1	-4	1
0	1	0

//

When the diagonals also considered then the eqn becomes,

$$\Delta^2 f(x,y) = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) + f(x+1,y+1) + f(x-1,y-1) + f(x+1,y-1) + f(x-1,y+1) - 8(f(x,y))$$

The mask representation of the above equation, (for sharper edge detection

1	1	1
1	-8	1
1	1	1

//

Method of Image sharpening using the laplacian:

The laplacian enhances edges & details by amplifying regions of rapid intensity change in an image.

Use the second order derivative mask:

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

center -4, neighbors 1, highlights edges (positive/negative values).

Steps:

convolve the mask with the original image $f(x,y)$ to get $\nabla^2 f(x,y)$.

Add the laplacian (scaled) back to the original image:

$$g(x,y) = f(x,y) - k \cdot \nabla^2 f(x,y)$$

where k controls sharpening strength.

Subtracting $\nabla^2 f$ boosts edges (since laplacian is negative at edge peaks, subtraction increases intensity there). Hence using these edge become more pronounced, making the image $g(x,y)$ appear sharper.

Ch 1 & 2

Digital Image:
A digital image is a representation of a 2D image as a finite set of digital values called picture elements or pixels.
Pixel values typically represent gray levels, colors, heights, opacity etc.

Digital Image Processing:
It is the processing of digital image by means of digital.

The field of DIP is related to the extracting of the attribute of the image, processing, the attributes and classification & recognization of object/image.

Image Enhancement:
It is the process of enhancing the appearance of an image or a subset of the image, to improve contrast or visualization of image features of interest or to facilitate more accurate subsequent image analysis.

Image Acquisition:
It is an action of retrieving image from an external source for further processing. Image disturbance & noise are removed during the pre-processing step.

Components of an Image Processing System:

Digital Image Processing uses different computer algorithms to perform image processing on the digital images.

Components:

(i) Image sensors:
It senses the intensity, amplitude, co-ordinates & other features of the images & passes the result to the image processing hardware. It includes the problem domain.

(ii) Image Processing hardware:
It is the dedicated hardware that is used to process the instructions obtained from the image sensors. It passes the result to general purpose computer.

(iii) Computer: Used in the IPS.

(iv) Image Processing software:
Software that includes all the mechanisms & algorithms that are used in IPS.

(v) Mass storage: stores the pixels of the images during the processing.

(vi) Hard Copy Devices: once the image is processed then it is stored in the hard copy device. It can be pendrive.

(vii) Image display: It includes the monitor or display screen that displays the processed image.

(ix) Network: It is the connection of all the above elements of the Image processing system.

→ fields thay use DIP:
(i) Image sharpening & restoration.
(ii) medical field
(iii) office automation
(iv) Geographic Information system
(v) Artistic effect.
(vi) pattern recognition.

Key stages in I P

steps in DIP:

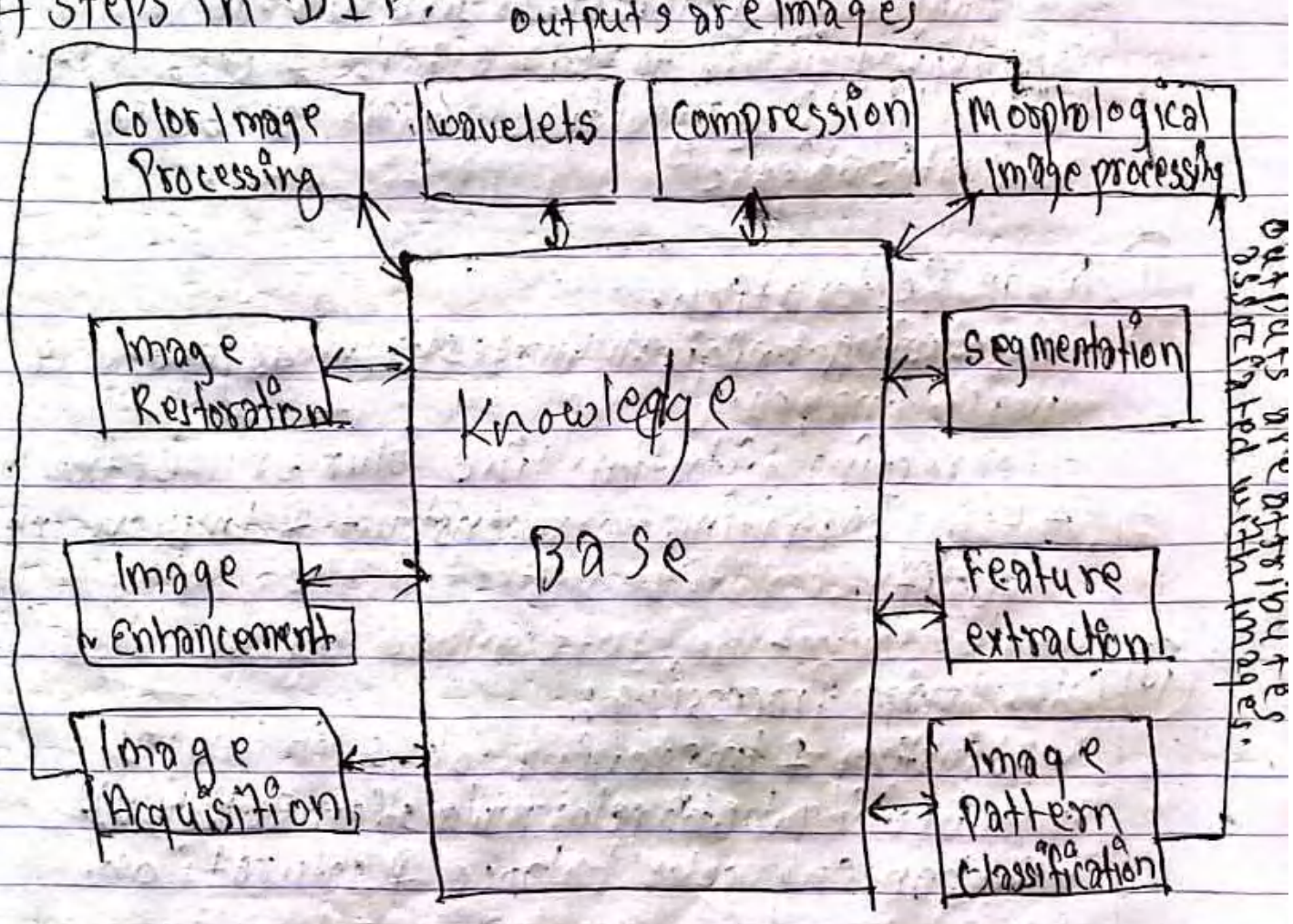


fig: fundamental steps in DIP..

(1) Image Acquisition:
- capturing an image using device.
- converting the captured image into a digital format.
- storing the digital image for further processin

(11) Image Enhancement:
- improving image quality by adjusting brightness, contrast, or sharpness.
- Applying filters to reduce noise & enhance detail
- Highlighting specific features for better visualization.

(111) Image Restoration.
- Recovering original images from degraded or noisy images.
- correcting distortions like blur or noise from Imag
- Using algorithms to reconstruct & Improve Image quality.

(IV) Color Image Processing:
- Adjusting & Improving colors in Images.
- converting bet^n color models (e.g RGB, HSV).
- Enhancing color balance & saturation.

(V) Wavelets:
- breaking Image into frequency components.
- Efficient for compression & analysis
- Enhances resolution in focused areas.

(vi) Image Compression:
- Reducing Image size for storage.
- Using lossless or lossy techniques.
- Balancing quality & size.

(vii) Morphological Image Processing:
- Processing shapes & structures.
- Applying dilation, erosion etc.
- Extracting structural information.

(viii) Image Segmentation:
- Dividing Image into regions.
- Grouping similar pixels.
- Isolating areas of interest.

(ix) Feature Extraction:
- Identifying key features (edges, textures)
- simplifying raw data.
- used for recognition & classification.

(x) Pattern Recognition:
- Identifying patterns from features.
- classifying objects based on patterns.
- Using algorithms for accurate recognition.

Difference bet^n Image sampling & Image Quantization:

Image Sampling	Image Quantization
(i) Converts continuous to discrete.	(i) Reduces color values.
(ii) Determines Image resolution.	(ii) Determines color depth.
(iii) Higher Sampling = more detail.	(iii) Compression to reduce file size.
(iv) Similar to choosing file count to mosaic	(iv) Similar to choosing specific colors.

Q) Explain spatial resolution & Intensity level resolution with examples.

⇒ Spatial Resolution:

Spatial Resolution states that the clarity of an Image cannot be determined by the pixel resolution.

The no. in an Image does not matter.

Spatial resolution can be defined as the smallest discernible detail in an Image.

we cannot compare two different types of Images to see that which one is clear or which

one is not.

If we have to compare the two images; to see which one is more clear or which has more spatial resolution, we have to compare two image of the same size.

Example:
Imagine two photos of the same park.

(1) High spatial resolution:
The photo shows a lot of detail, like individual blades of grass & small flowers.

(ii) low spatial Resolution:
Shows less detail, like only large objects such as trees & benches.

(2) Intensity level Resolution:

It refers to the number of Intensity levels used to represent the Image.

The more Intensity level used, the finer the level of detail discernible in an Image

Intensity level resolution is usually given in terms of the number of bits used to store each Intensity.

Ex: Consider a grayscale image of a scene. If the intensity level resolution is 8-bit, it means there are $2^8 = 256$ different shades of gray that can be represented, from black to white.

If you reduce the intensity level resolution to 4-bit, only $2^4 = 16$ different shades of gray can be represented. The higher the intensity level resolution, the smoother the transitions between shades, making the image appear more detailed & realistic.

Q) Explain basic relationships like neighbours, connectivity & distance measures b/t pixels in any digital image with suitable example

⇒ Neighbors of a pixel:

Any pixel p(x,y) has two vertical & two horizontal neighbors, given by;

$(x+1, y), (x-1, y), (x, y+1), (x, y-1)$

This set of pixels are called the 4 neighbors of P & is denoted by $N_4(P)$.

Each of them are at a unit distance from P.

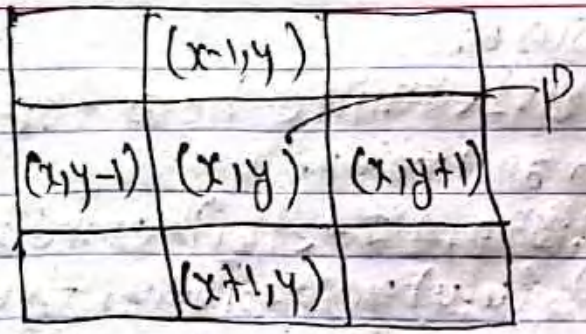


Fig: Neighbors of pixels (4-neighbors).

(2) Connectivity:

- To determine whether the pixels are adjacent in some sense.
- let v be the set of gray level values used to define connectivity; then Two pixels P, q that have values from the set V are:
  - 4 connected, if q is in the set $N_4(P)$
  - 8 connected, if q is in the set $N_8(P)$
  - m connected, iff
    - q is in $N_4(P)$ or
    - q is in $N_8(P)$ & set $[N_4(P) \cap N_4(q)]$ is empty

Ex: Consider the following sample Image pixel & vector V = {1, 2}

0	1	2
0	1	0
0	0	2

(a) Input

0	1	2
0	1	0
0	0	2

(b) 4-way Connected

0	1	2
0	1	0
0	0	2

(c) 8-way connected

0	1	2
0	1	0
0	0	2

(d) m-way Connected

- It defines how pixels are grouped based on their relationships with neighbours.

(3) Distance measure:

- It helps to determine how far apart two pixels are in an image.
- Given pixels p, q & z with coordinates (x,y), (s,t), (u,v) respectively, the distance function D has following properties:

(a) $D(p,q) \geq 0$ [$D(p,q)=0$, iff $p=q$]
(b) $D(p,q) = D(q,p)$
(c) $D(p,z) \leq [D(p,q) + D(q,z)]$

Different Distance measures:

(i) Euclidean Distance:

$$D_E(p,q) = [(x-s)^2 + (y-t)^2]$$

(ii) City Block Distance:

$$D_4(p,q) = |x-s| + |y-t|$$

(iii) Chess-Board Distance

$$D_8(p,q) = \max(|x-s|, |y-t|)$$

		2		
	2	1	2	
2	1	0	1	2
	2	1	2	
		2		

(a) City Block / D4 Distance

2	2	2	2	2
2	1	1	1	2
2	1	0	1	2
2	1	1	1	2
2	2	2	2	2

(b) Chess Board / D8 Distance

Brightness Adaptation:

- It is the ability of the human visual system to adjust to different levels of illumination.
- It enables us to see in both bright & dim light by adapting the sensitivity of our eyes.

Brightness discrimination:

- Ability to distinguish different levels of brightness.
- It's about detecting small differences in light intensity.

Neighbouring of Pixel

It refers to the relationship between pixels in an image based on their spatial adjacency.

Types:

(1) 4-connectivity (4-neighbours):

Any pixel p(x, y) has two vertical & two horizontal neighbours give by;

$$(x+1, y), (x-1, y), (x, y+1), (x, y-1)$$

This set of pixels are called the 4 Neighbors of P & is denoted by $N_4(P)$.

Each of them are at unit distance from P.

Ex:-
N_1
N_2 P N_3
N_4

Here N_1, N_2, N_3 & N_4 4-neighbors of Pixel

(2) 8-connectivity (8-Neighbors):

A pixel P at coordinates (x, y) has 8 neighbors, which include the 4-neighbors plus the 4 diagonal neighbors.
Given by;

$$N_8(P) = \{(x, y+1), (x, y-1), (x+1, y), (x-1, y), (x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1)\}$$

Ex:-

$$\begin{bmatrix} N_1 & N_2 & N_3 \\ N_4 & P & N_5 \\ N_6 & N_7 & N_8 \end{bmatrix}$$

Here, N_1-N_8 are the 8-Neighbors of pixel P.

(3) Diagonal Connectivity:
A pixel P at coordinates (x, y) has 4 diagonal neighbors, which are the pixels located diagonally around it.

Ex:- The pixel P has diagonal neighbors

$$N_d(P) = \{(x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1)\}$$

$$\begin{bmatrix} D_1 & & D_2 \\ & P & \\ D_3 & & D_4 \end{bmatrix}$$ Here D_1 to D_4 are the diagonal neighbors of P. #

Average Filtering:

Also known as mean filtering is a simple, commonly used technique to smooth images by reducing noise.

It works by replacing each pixel value in an image with the average value of its neighbour.

Algorithm:

(i) Input: Original image of size MxN filter size fxf.
(ii) output: smoothed image of size MxN.
(iv) steps:
  (i) Initialize an empty output image of the same size as the orfginal image.
  (ii) for each pixel in the original image:
    (A) compute the average value of the pixels neighbors within the filter window.
    (B) Assign the completed average valve to the corresponding pixel in the output image.
  (iii) Return the smoothed output image.

→

Pseudocode:

```
function average_filter(image, filter_size):
    M, N = image.height, image.width
    offset = filter_size
    output_image = create_empty_image(M, N)

    for i from offset to M-offset-1:
        for j from offset to N-offset-1:
            sum = 0
            for k from -offset to offset:
                for l from -offset to offset:
                    sum += image[i+k][j+l]
            output_image[i][j] = sum / (filter_size
                                        * filter_size)

    return output_image
```

Ex:

For a 3x3 filter, the filter window for a pixel P at (i, j) would look like this:

N1 N2 N3
N4 P N5
N6 N7 N8

The average value would be calculated as:

average_value = (N1+N2+N3+N4+P+N5+N6+N7+N8)/9

Contrast stretching & Thresholding:

(i) Contrast stretching:
- Enhances image contrast by stretching the intensity range of pixel values.
- It is used to increase the difference between the darkest & brightest regions, making the image more visually clear.
- It maps the minimum & maximum pixel values in the image to a new range (usually 0 to 255 for 8-bit images).
- It increases the contrast bet^n light & dark areas.

Formula:

$$S(x,y) = \frac{(I(x,y) - I_{min})}{(I_{max} - I_{min})} \times (S_{max} - S_{min}) + S_{min}$$

where Imin & Imax are the original min & max values, & Smin & Smax are the desired output range.

(ii) Thresholding:
- converts a grayscale image into binary image based on threshold intensity.
- Pixels are classified as either foreground (white) or background (black).
- Each pixel is compared to a threshold value T.
- If the pixel intensity $\geq T$, set it to white (255), otherwise set it to black (0).
- formula:

$$s(x,y) = \begin{cases} 255 & \text{if } I(x,y) \geq T \\ 0 & \text{if } I(x,y) < T \end{cases}$$

#

Histogram of an image represents the frequency of occurrence of gray level in th image. It provides the global description of image.

Intensity level slicing:

- It can be used to segment certain gray level regions from the rest of the image.

- It is an image enhancement technique used to highlight specific intensity ranges within an image while supressing others.

Q) How can you use histogram equalization technique for contrast enhancement?

⇒ It is a process for increasing the contrast in an image by spreading the histogram out to be approximately uniformly distributed.

- The gray level of an image that has been subjected to histogram equalization are spread out & always reach white

- The increase of dynamic range produces an increase in contrast.

- for image with low contrast, histogram equalization has adverse effect of the increasing visual graininess.

- It requires construction of transformation function S_k:

$$S_{1k} = T(r_k) = \sum_{j=0}^{k} \frac{n_j}{m*N}$$

$$S_k = T(r_k) = \frac{L-1}{M*N} \sum_{j=0}^{k} n_j$$

where T_k is k^{th} gray level
n_k is no. of pixels with that gray level
$M \times N$ is no. of pixel in the Image &
K is 0, 1, ... L-1

Here $T(r_k)$ must satisfy two conditions:
(1) It must be single valued & monotonically increasing in the range $0 \leq r \leq 1$.
(2) $T(r_k)$ must be within the range
$0 \leq t(r_k) \leq 1$ or
$0 \leq s \leq 1$.

#Algorithm for Histogram Equalization:
(1) Grayscale Image with pixel values ranging from 0 to L-1.
(2) Calculate the histogram of the Image, which is a frequency distribution of pixel Intensities.
(3) Normalize the histogram to obtain the probability Distribution function (PDF) of pixel Intensities.
(4) Calculate the cumulative sum of the PDF to get the CDF
(5) Normalize the CDF to map the minimum valued

& the max^n value to L-1.

⑥ Map Intensities: Use the normalized CDF to map the original pixel values to equalized values.

Pseudo code:

```
def hist_equalization(image):
    hist = compute_histogram(image)
    pdf = hist / float(image.size)
    cdf = pdf.cumsum()
    cdf_min = cdf[cdf > 0].min()
    cdf_max = cdf.max()
    cdf_normalized = (cdf - cdf_min) *
                     (255 / (cdf_max - cdf_min))
    equalized_image = cdf_normalized[image]
    return equalized_image

def compute_histogram(image):
    hist = np.zeros(256, dtype=int)
    for pixel_value in image.flatten():
        hist[pixel_value] += 1
    return hist
```

//

Zooming:

It simply means enlarging a picture in a sense that the detail in the image became more visible & clear.

① Zooming by replication:
- It is also known as nearest neighbour interpolation.
- In this method, we just replicate the neighbouring pixels.
- Zooming is nothing but increase amount of sample or pixels.

Algorithm:

Eg:- a 4×4 image

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

→ column

① Repeat each pixel as shown in the example below:

1	1	2	2	3	3	4	
5	5	6	6	7	7	8	
9	9	10	10	11	11	12	
13	13	14	14	15	15	16	

⑪ Repeat each rows as shown in the example below:

1	1	2	2	3	3	4	4
1	1	2	2	3	3	4	4
5	5	6	6	7	7	8	8
5	5	6	6	7	7	8	8
9	9	10	10	11	11	12	12
9	9	10	10	11	11	12	12
13	13	14	14	15	15	16	16
13	13	14	14	15	15	16	16

Here 4x4 Image is zoomed to 8x8 Image
- This method can be repeated to get bigger size Image.
- Zooming by replication can be implemented on the computer by using replication mask (window).

(2) Zooming by Interpolation:
- Instead of replicating each pixels, average of two adjacent pixels along rows is taken & placed between pixels.
- The involves estimating the values of the new pixels based on the values of the existing pixels.
- The same operation is then performed along the column.
- Linear interpolation is performed on zero interlaced Image.

Ex:

(i)

1	1.5	2	2.5	3	3.5	4	2
0	0	0	0	0	0	0	0
5	5.5	6	6.5	7	7.5	8	4
0	0	0	0	0	0	0	0
9	9.5	10	10.5	11	11.5	12	6
0	0	0	0	0	0	0	0
13	13.5	14	14.5	15	15.5	16	8

(ii) Find the average value along row & place between pixels.

1	1.5	2	2.5	3	3.5	4	2
3	3.5	4	4.5	5	5.5	6	3
5	5.5	6	6.5	7	7.5	8	4
7	7.5	8	8.5	9	9.5	10	5
9	9.5	10	10.5	11	11.5	12	6
11	11.5	12	12.5	13	13.5	14	7
13	13.5	14	14.5	15	15.5	16	8

(iii)

1	2	2	3	3	4	4	2
3	4	4	5	5	6	6	3
5	6	6	7	7	8	8	4
7	8	8	9	9	10	10	5
9	10	10	11	11	12	12	6
11	12	12	13	13	14	14	7
13	14	14	15	15	16	16	8

→ Zooming in Interpolation is more efficient than zooming by replication.