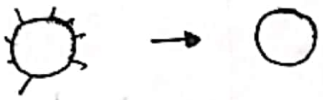
Lecture-11,12

Erosion: The process where all the pixels of the structuring element has to match those of the original image, in other words, have to fit.

Through erosion, the foreground size shrinks, and extra spiky edges are removed, therefore object boundaries are smoothened. Holes inside an object, however are not removed.

* Erosion can split apart joined object

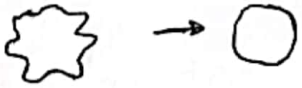


* Erosion can strip away extrusions.
* Erosion shrinks object.
* Removes irrelevant Details from binary image.

Equation: $(A \ominus B) = \{z \mid (B)_z \subseteq A\}$

Dilation: This is a process where any of the pixels of the structing element has to match those of the original image, in other words, have to hit. Through dilation, the foreground size enlarges. and extra outer layer is added, therefore object boundaries are smoothened. The holes inside the object vanish.

* Dilation can repair breaks
* Fills in holes.



* Dilation can repair intrusions.
* Dilation enlarges objects.

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

Boundary Extraction:

$$\beta(A) = A - (A \ominus B)$$

Edge Detection:

$$\beta(A) = (A \oplus B) - A$$

Duality of Erosion and Dilation:

- $(A \ominus B)^c = A^c \oplus \hat{B}$
- $(A \oplus B)^c = A^c \ominus \hat{B}$

~~[illegible]~~

Opening: The opening of image A by structuring element B, denoted by $A \circ B$ is simply erosion ~~followed~~ by a ~~dilation~~ dilation.

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

Effects: ① Smoothed the outline, by rounding off any sharp points
② Remove any parts that is smaller than the shape of structuring element used.
③ It will open / disconnect any thin bridges.
④ Doesn't remove holes inside image.
⑤ Doesn't make the basic core size of the image larger or smaller.

Closing: The closing of a image f by structuring element s, denoted by $A \bullet B$, is a simple ~~dial~~ dilation followed by an erosion.

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

Effect: ① Smoothed the outline, by filling holes.

② Connecting bridges will form.

③ Doesn't make the basic core size of shape larger or smaller.

Opening is the dual of closing

→ Opening the foreground pixels with a particular structuring element is equivalent to closing the background pixels with the same element.

Hit and Miss Transformation: commonly known as HMT, is a high level morphology method. It is specifically designed to find and locate specific patters or shapes in images by looking for a specific configuration of foreground and background pixels around the origin.

1 → foreground

0 → background

NaN → Don't Care

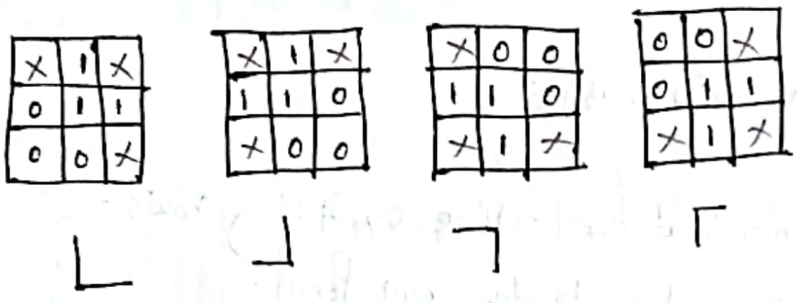
	1	
0	1	1
0	0	

* if pattern perfectly matches then it's a hit

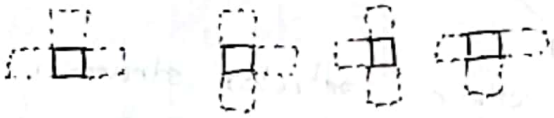
or

miss

Corner Detection:

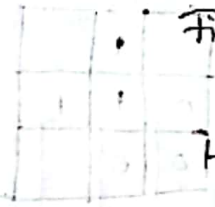


End Point:



Structuring element: A small image / template that helps to produce new image from the old one i.e small binary array. It is a shape mask used in basic morphological operations.

Can be of any shape and size, each has an origin.



Fit: if the SE perfectly matches / all pixels matches.

Hit: if any pixels a matches.

Origin-42

4(b)

i Reflection of B:

Assuming centre pixel is the origin of SE.

1
1
1

B

ii F dilated by B: if hit then 1 ; or 0

*considering padding of 0

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

iii F^c eroded by B; if fit then 1 ; or 0

F^c =

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

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

iv (F dilated by B) - F

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

v. Opening of F by B

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

F =

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

B =

1
1
1

Erosion:

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

* fit = 1
or = 0

Dilation:

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

vi. Closing of F by B

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

Dilation:

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

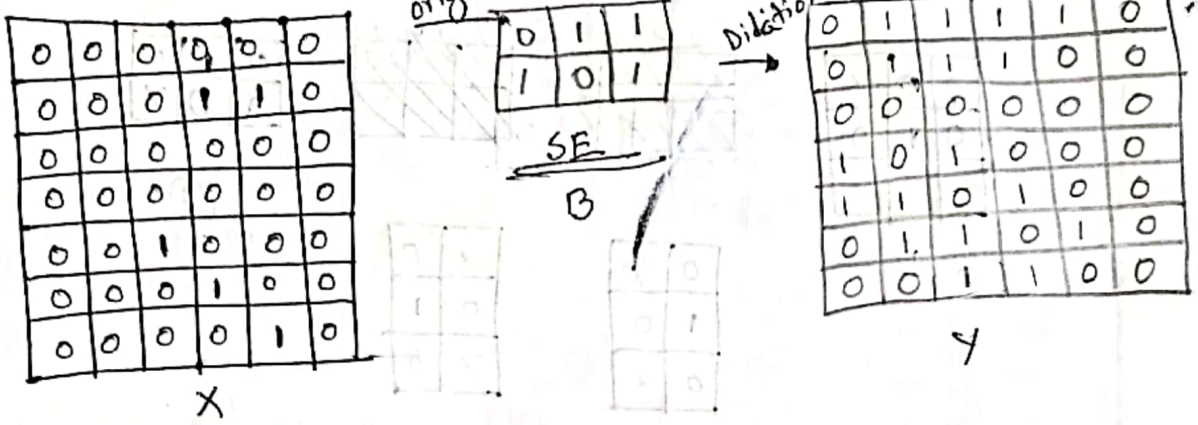
Erosion:

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

Decipher:

1(a)

i



ii

$Y = X \oplus B$

iii

$Z =$

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

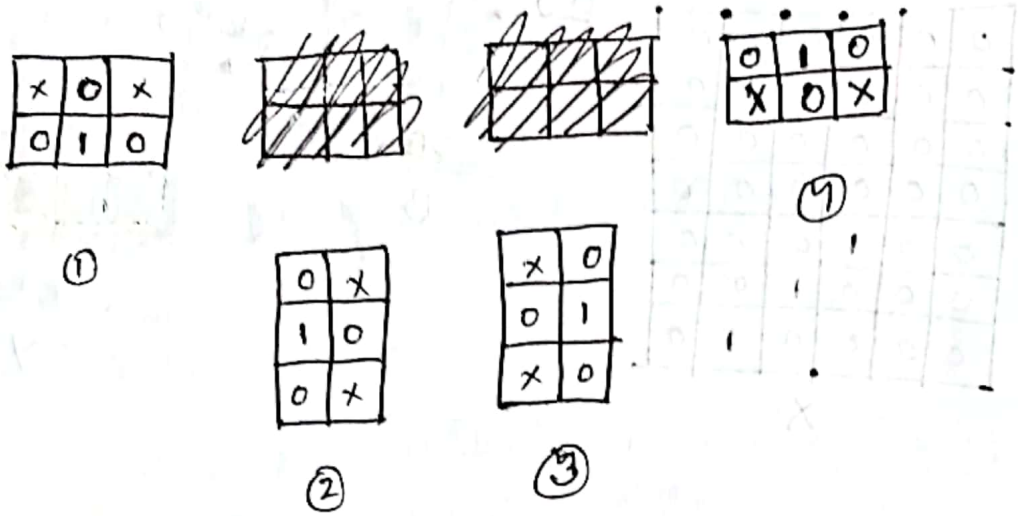
Erosion on Y using B.

iv $Z = Y \ominus B$

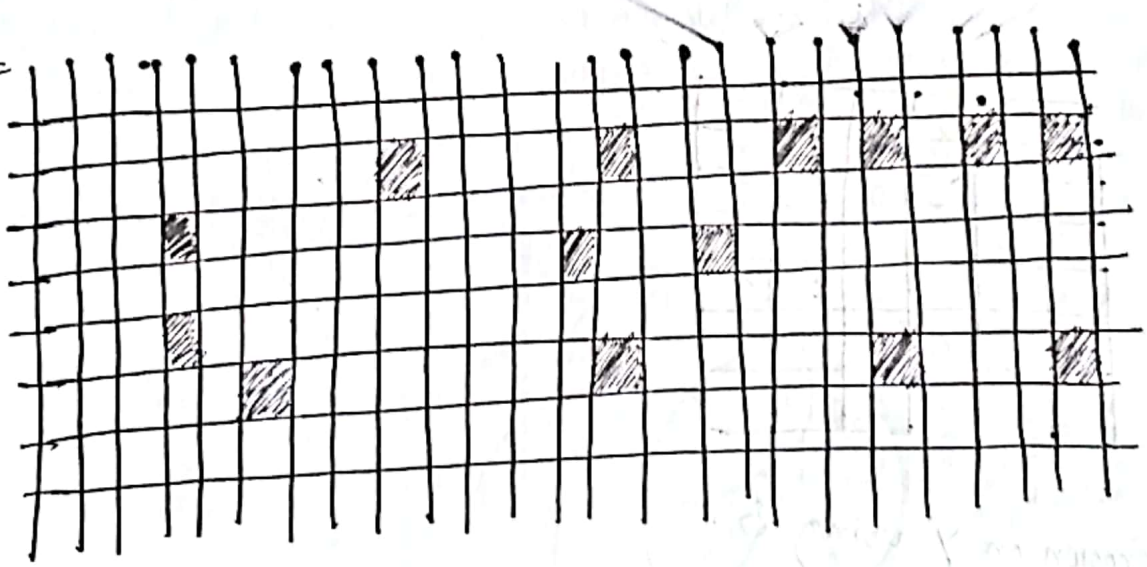
3(c)

i Hit and miss Transformation.

ii



iii



iv 14

Integer

2(c) → ✓

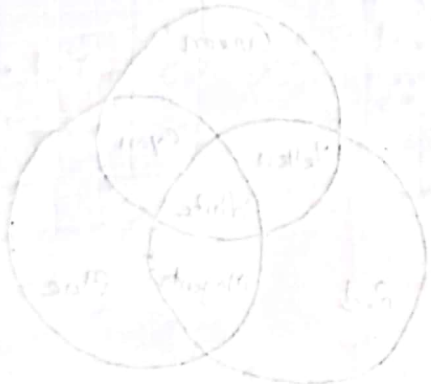
3(c)

i Explain Erosion

ii See previous page 3 (c) (ii)

iii Same logic as previous.

10 end points.



Lecture-15

Out of all cones of our eyes.

65% are Red light sensitive
33% " Green " "
2% " Blue " "

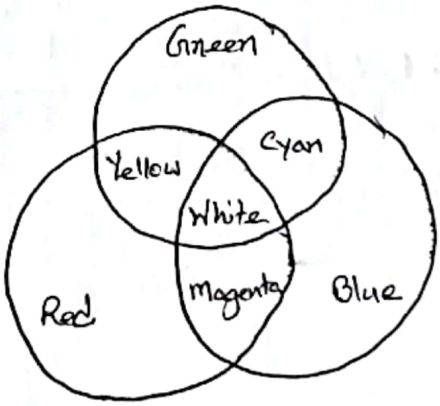
} out of these 3, blue are most sensitive.

Primary colours of light are additive (RGB)

=> Red, Green, Blue.

Combining them we get white.

RGB is used primarily in digital displays, such as computer monitors, TV screens and projectors. It is based on the principle of adding light to create colours. In RGB, each colour channel (RGB) starts at zero intensity and as we increase the ~~ins~~ intensity of each channel, we add more light to the mix, When all channel are set to maximum intensity, we get white light, when all at zero we get black. As we are getting colours by emitting and adding different spectrum of light, it is called additive.



R + G = Yellow
G + B = Cyan
B + R = Magenta

Primary colours of pigment are subtraction (cmy)

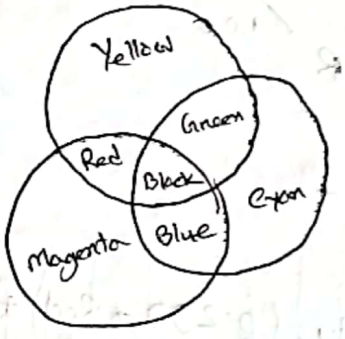
⇒ Cyan, Magenta and Yellow

⇒ Combining them we get black

cmy is primiraly used in colour printing, such as in magazines, brochures and posters, it is based on the principle of subtracting colour to create colour. In cmy, each colour channel (Cyan, Magenta, Yellow) starts at its maximum intensity and as we decrease the intensity of each channel, we subtract colour from the mix. When all chanel intensity is at maximum, we get black, when zero we get white. As we are decreasing the intensity it is called subtractive.

CMYK

Here! K = key / Black



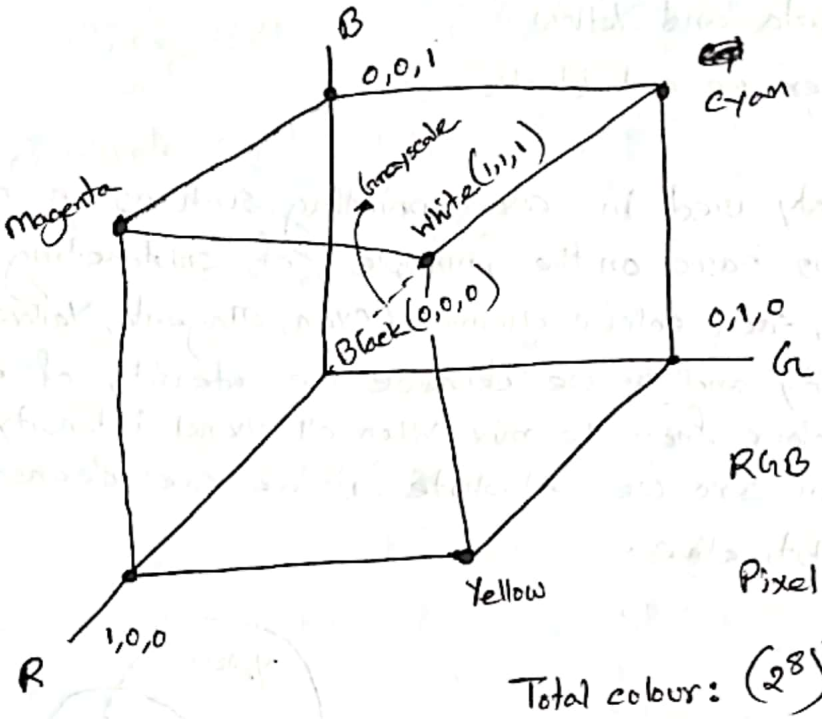
Cyan = White - Red

Magenta = White - Green

Yellow = White - Blue

# Additive	Subtractive
① We Get Colours by adding / mixing different colours.	① We get colours by subtracting different Colours.
② emits different spectrum of light	② Absorbs different spectrum of light.
③ Additive colour system use coloured light passing through the image elements to reconstruct the full colours on the screen.	③ Subtractive systems use multi layered coloured dyes to reconstruct the spectrum on the film and then white light projects through the image on to the screen.

RGB colour Model



RGB of 24-bit colour cube.

Pixel depth = 8 bit per plane

Total colour: $(2^8)^3 = 16,777,216$

Light intensity: $(0.299 * \text{Red}) + (0.587 * \text{Green}) + (0.144 * \text{Blue})$ for human eye.

Green appear the brighest.

Lecture-8

Image Segmentation: In computer vision segmentation is the process of partitioning a digital image into multiple segments. (set of pixels, also known as image objects)

* Divide the image into different regions.
* Separate objects from background and given them individual ID.

Segmentation algorithm are two types based on one of the two basis properties of intensity values:

i. Similarity: Partitioning an image into regions that are similar according to a set of predefines criteria.
   * Thresholding: based on pixel intensities
   * Region Based: Grouping Similar pixels.
   * Match based: Comparison to a given template.

ii. Discontinuity: Partitioning an image based on sharpe changes in intensity.
   * Edge based: detection of edges that separate regions from each other.
   * Watershed: find regions corresponding to local minima in intensity.

Canny Edge Detector

Three main Criteria:

1. Good Detection: The ability to locate and mark all real edges.
2. Good Localisation: Minimal distance between the detected edge and real edge.
3. clear Response: Only one response per edge.

The algorithm runs in 5 steps:

Step-1: Smoothing (Blurring of the image to remove noise)

Let, $f(x,y)$ denote input image and $G(x,y)$ denote Gaussian function:

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

A smoothed image, $f_s(x,y)$ will be form,

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

Step-2: Finding gradients (The edges should be marked where the gradients of the image has large magnitudes)

- Prewitt edge operator

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

Horizontal (g_y)

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

vertical (g_x)

Gradient magnitude,

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

and

$$\alpha(x,y) = \arctan(g_y/g_x)$$

- Sobel edge operator

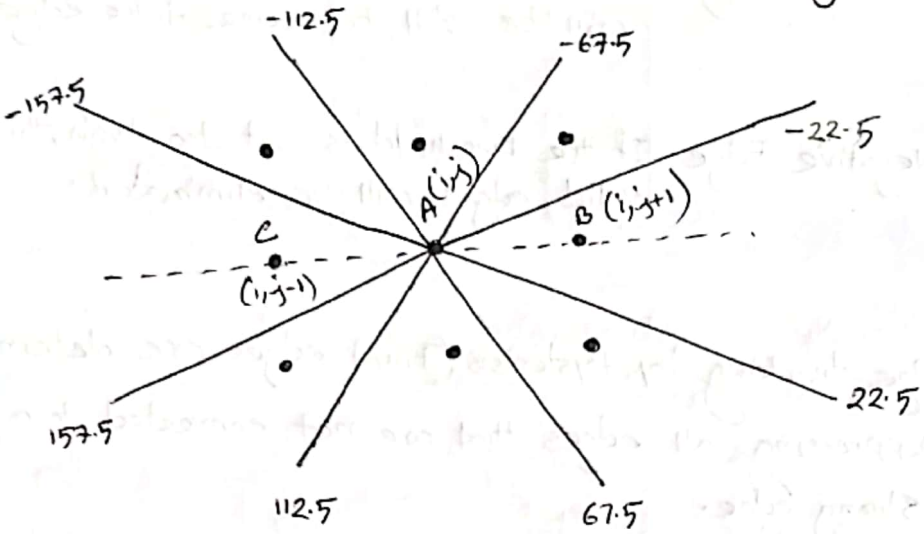
-1	-2	-1
0	0	0
1	2	1

Horizontal (g_y)

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

vertical (g_x)

Step-3: Only local maxima should be marked as edges.



For each pixels, the neighbouring pixels are located in horizontal, vertical and diagonal directions (0°, 45°, 90°, 135°)

$$M(x,y) = \begin{cases} |\nabla S|(i,j) & \text{if } |\nabla S|(i,j) > |\nabla S| \text{ (i,j+1) and} \\ & |\nabla S|(i,j) > |\nabla S|(i,j-1). \\ 0 & \text{otherwise} \end{cases}$$

$g_N(x,y)$ will be the output

Step-4: Double thresholding (Potential edges are determined by thresholding)

To solve the problem of "which edges are really edges and which are not" Canny use the Hysteresis Thresholding:

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

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

and $g_{NL}(x,y) = g_{NL}(x,y) - g_{NH}(x,y)$

The ratio of the high and low threshold should be two or three to one.

#False Positive Edge: If the threshold is set too low, there will be still be some false edge.

#False Negative Edge: If the threshold is set too high, then actual valid edges will be eliminated.

Step-5 : Edge tracking by Hysteresis: Final edges are determined by suppressing all edges that are not connected to a very strong edge.

For the edge pixels, of $g_{NH}(x,y)$ and $g_{NL}(x,y)$

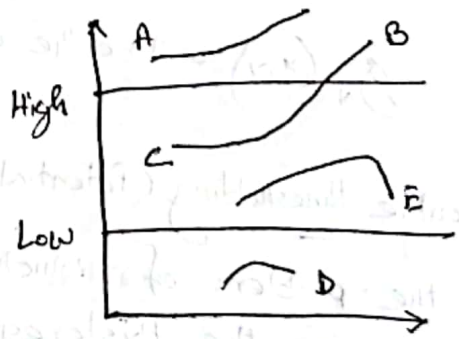
* Any edges with intensity greater than 'High' are the sure edge.
* Any edges with intensity lower than 'low' are sure to be non-edge.
* The edges between high and low thresholds are classified as edges only if they are connected to a sure edge otherwise discarded.

200	0	0
0	0	101
0	0	0

g_{NH}

0	0	57
0	45	0
50	0	0

g_{NL}



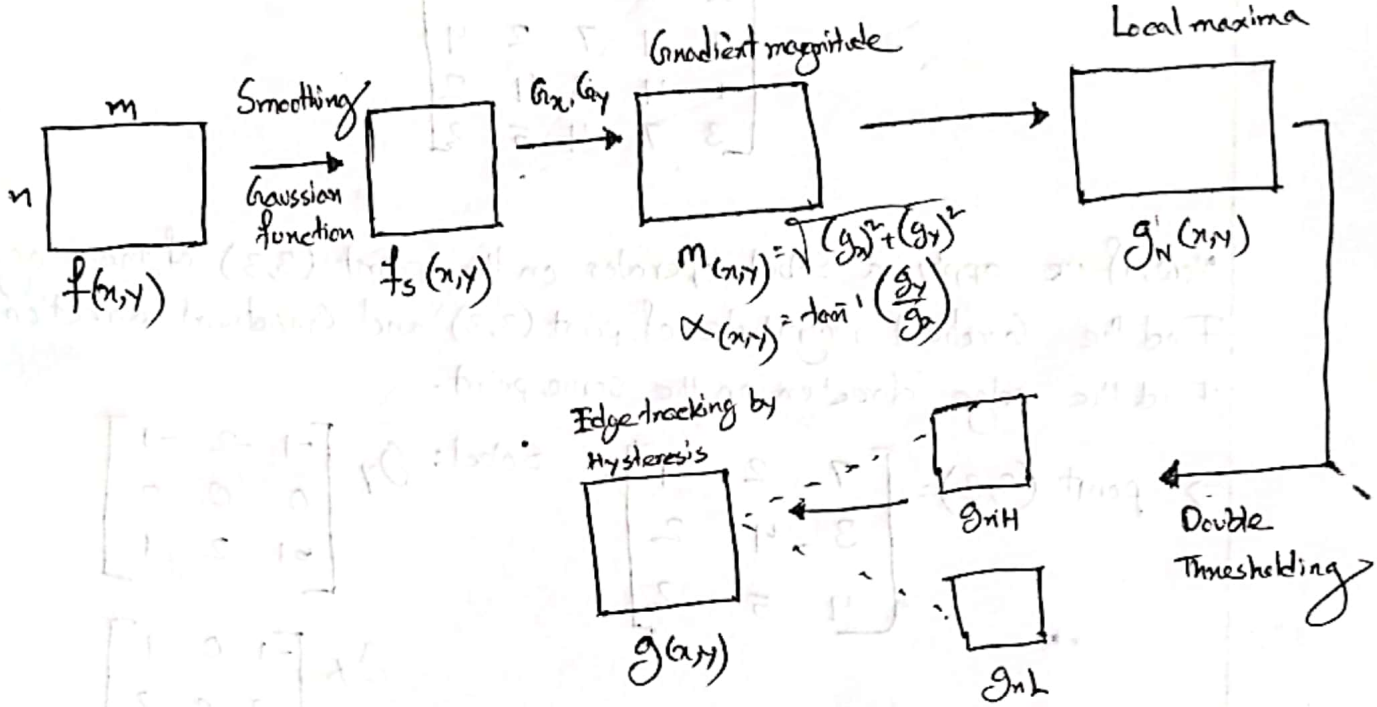
$g(x,y)$

200	0	57
0	0	101
0	0	0

Decipher

3(e)

i



ii

iii Line Detection:

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

R_1 = Horizontal

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

R_2 = +45°

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

R_3 = Vertical

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

R_4 = -45°

What is an Gradient Operator? Suppose we have a 5×5 image with values are

$$\begin{bmatrix} 3 & 7 & 6 & 2 & 0 \\ 2 & 5 & 8 & 5 & 1 \\ 3 & 4 & 7 & 2 & 4 \\ 1 & 4 & 3 & 4 & 2 \\ 3 & 7 & 4 & 5 & 2 \end{bmatrix}$$

Now, if we apply a Sobel operator on the point (3,3) of the image. Find the Gradient magnitude of point (3,3) and Gradient direction? Find the edge direction on the same point.

$\Rightarrow$ point (3,3): $\begin{bmatrix} 7 & 2 & 4 \\ 3 & 4 & 2 \\ 4 & 5 & 2 \end{bmatrix}$ Sobel: $g_y \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$

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

$g_y = (7\times-1)+(2\times-2)+(4\times-1)+(3\times0)+(4\times0)+(2\times0)$
$+(4\times1)+(5\times2)+(2\times1)$

$= -7-4-4+4+10+2 = 1$

$g_x = (7\times-1)+(2\times0)+(4\times1)+(3\times-2)+(4\times0)+(2\times2)+(4\times-1)+(5\times0)+(2\times1)$

$= -7+4-6+4-4+2 = -7$

Gradient magnitude: $\sqrt{(-7)^2+(1)^2} = 5\sqrt{2}$

Gradient direction: $\tan^{-1}\left(\frac{1}{-7}\right) = -8.13^\circ$

We know Gradient direction and Edge direction are perpendicular to each other.

Edge Direction: $-8.13 \mp 90^\circ$

$= -98.13^\circ$

Lecture-13,14

Origin-42

1-(b)

i) Given image,

3	4	7
3	4	7
3	4	7

8-bit image

For 8-bit image, codeblock will be 9-bit

Dictionary Location	Entry
0	0
1	1
⋮	⋮
255	255
256	⋮
⋮	
511	

Current Recognized sequence	Pixel Being Processed	Encoded output	Dictionary Location (Codeword)	Dictionary Entry
	3			
3	4	3	256	3-4
4	7	4	257	4-7
7	3	7	258	7-3
3	4			
3-4	7	256	259	3-4-7
7	3			
7-3	4	258	260	7-3-4
4	7			
4-7		257		

New Codebook:

Dictionary Location	Entry
0	0
⋮	⋮
255	255
256	3-4
257	4-7
258	7-3
259	3-4-7
260	7-3-4

The output sequence: 3-4-7-256-258-257

Extra:

Decoding:

3 4 7

3 4 7

3 4 7

ii

In LZW coding process, it reduces spatial data redundancy. Intensities of each pixel may correlate to it's neighbours. Many informations are unnecessary replicated in the spatial redundancy. LZW try to minimize the unnecessary data. It assigns fixed length code words to variable length sequence of source symbols.

iii

Before compression,

Total pixels = 9

Each pixels contains 8 bit

∴ original image size = 9×8 = 72 bits.

After compression,

Total output = 6

Each pixels contains 9 bit

∴ compressed image size = 6×9 = 54 bits.

So, Compressed ratio = $\frac{72}{54}$ = 1·33 : 1

Origin-42

5(a)

i) Image Compression is needed for
   * Data storing at low storage
   * Data transmission at minimum cost

ii) Three types of Data Redundancy
   * Coding Redundancy: Most 2D intensity arrays contain more bits than are needed to represent the intensities.
     It tries to minimize codeword length.
   * Spatia and Temporal redundancy: Pixels of most 2D-intensity array are correlated spatially and video sequence are temporally correlated.
   * Irrelavent information: Most 2D intensity arrays contain ~~intensity~~ information that is ignored by the visual system of human.

iii) 5.3 bits/pixel entropy means minimum 5.3 bits required to represent a pixel without losing any data.

$$\text{Compression Ratio} = \frac{512 \times 512 \times 8}{512 \times 512 \times 5.3} = 1.509 : 1$$

Origin-42

5 (b)

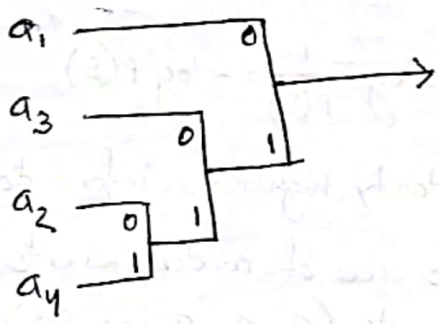
i) Given Image,

18	16	16
11	11	14
11	14	16

PDF =

Pixel Value	Probability
$a_1 = 11$	$3/9 = 0.33$
$a_2 = 14$	$2/9 = 0.22$
$a_3 = 16$	$3/9 = 0.33$
$a_4 = 18$	$1/9 = 0.11$

Using Huffman Coding:



Symbol	Codeword
a_1	0
a_3	10
a_2	110
a_4	111

ii) Entropy of the image : $\sum_{k=1}^{L-1} -P_r(r_k) \log_2 P_r(r_k)$

original image entropy, H :- $(0.33 \times \log_2(0.33) + 0.22 \times \log_2(0.22) + 0.33 \log_2(0.33) + 0.11 \times \log_2(0.11))$

$= 1.88$ bits/pixels.

iii) Huffman coding reduces coding Redundancy.

$L_{avg} = (0.33 \times 1) + (0.33 \times 2) + (0.22 \times 2) + (0.11 \times 4) = 1.98$ bits/pixel.

$H < L_{Avg}$ ∴ Loseless.

iv) Compression ratio $= \frac{3\times3\times8}{3\times3\times1.98} = 4.04:1$

(ans)

Enigma-41

2(a)

i) A random event E with probability P(E) carries I(E) units of information;

$$I(E) = \log \frac{1}{P(E)} = -\log P(E)$$

- Higher the uncertainty, higher the information content

If we have a source of random events from a discrete set of events $(a_1, a_2, a_3 \cdots, a_j)$ with probabilities, $P(a_1), P(a_2), P(a_3) \cdots P(a_j)$ then average information per event or the entropy of the source,

$$H = \sum_{j=1}^{j} -P(a_j) \log P(a_j)$$

Considering Pixel intensities as random events

$$H = \sum_{k=1}^{L-1} -P_r(r_k) \log P_r(r_k)$$

L = Gray levels / # of intensities
r_k = input image intensity
$P_r(r_k)$ = is normalized histogram of input image

* Entropy is the measurement of the average information in an image.

ii) $H = -\Big((0.04\log_2(0.04)) + (0.16\log_2(0.16)) + (0.2\log_2(0.2)) + (0.28\log_2(0.28)) + (0.2\log_2(0.2)) + (0.12\log_2(0.12))\Big)$

$= 2.41$ bits/pixel

Pixel values	Probabilities
a_6 18	1/25 = 0.04
a_4 16	4/25 = 0.16
a_3 14	5/25 = 0.2
a_2 12	7/25 = 0.28
a_1 11	5/25 = 0.2
a_5 17	3/25 = 0.12

iii)

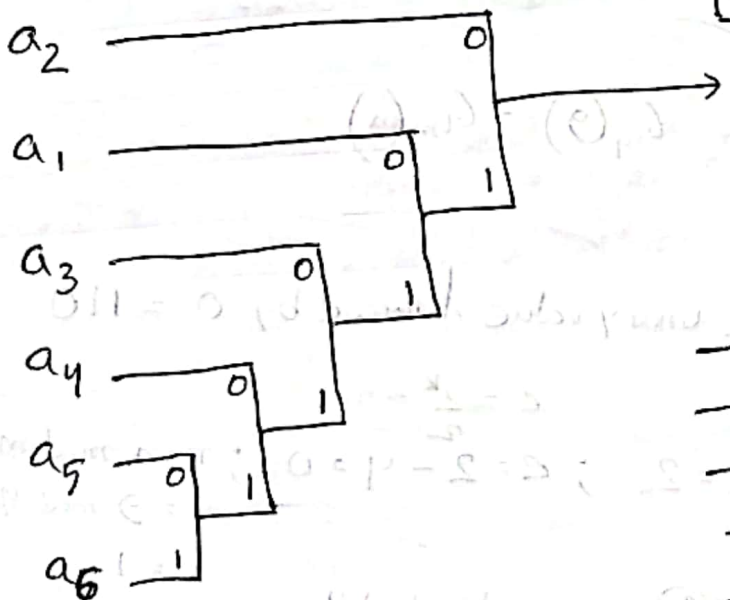
~~$a_1 = 0.2$~~
~~$a_2 = 0.28$~~
~~$a_3 = 0.$~~

$a_2 = 0.28$
$a_1 = 0.2$
$a_3 = 0.2$
$a_4 = 0.16$
$a_5 = 0.12$
$a_6 = 0.04$

iv) Compression ratio = $C_R = \frac{8}{2.72}$

$= 2.94 : 1$

v) $R_D = 1 - \frac{1}{C_R} = 0.65$



Symbol	Codeword
a_2	0
a_1	10
a_3	110
a_4	1110
a_5	11110
a_6	11111

$L_{avg} = (0.04\times5) + (0.16\times4) + (0.2\times3) + (0.28\times1) + (0.2\times2) + (0.12\times5)$

$= 2.72$ bits/pixel

Enigma-41

2(b)

i) $f(x,y)$

1	4	3
2	3	4
2	4	5

3-bit

$\xrightarrow{+2}$

3	5	5
4	5	6
4	6	7

$M=3$; $N=3$

Root mean Square Error:

$$e_{rms} = \left[\frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left(f'(x,y)-f(x,y)\right)^2\right]^{1/2}$$

$$= \left[\frac{1}{3\times3}\left((3-1)^2+(6-4)^2+(5-3)^2+(4-2)^2+(5-3)^2 + (6-4)^2+(4-2)^2+(6-4)^2+(7-5)^2\right)\right]^{1/2}$$

$$= \sqrt{4} = 2$$

ii) Golomb code for $G_4(9) = G_m(n)$

Step: 1

$\left\lfloor \frac{9}{4} \right\rfloor = 2$ ∴ unary value followed by 0 = 110

$c = 2^k - m$

Step: 2 $k = \lceil \log_2 4 \rceil = 2$; $c = 2^2 - 4 = 0$; $r = n \bmod m$
$= 9 \bmod 4$
$= 1$

$r' = r + c = (1) \rightarrow$ represent in k-bits

∴ $r' = 01$

Step-3 Golomb code = 'unary code value followed by 0' + r'

= 11001

Recorsive-40

5(a)

iii. Loseless compression technique that reduces the size of a digital image without losing any information or quality. It's also known as reversible as original file can be restored after decompression. Example- Arithmatic coding, Huffman coding, Golomb coding etc

iv) Arithmatic Coding is a lossless technique that encodes data by creating a code string which represents a fractional value on the number line between 0 to 1.

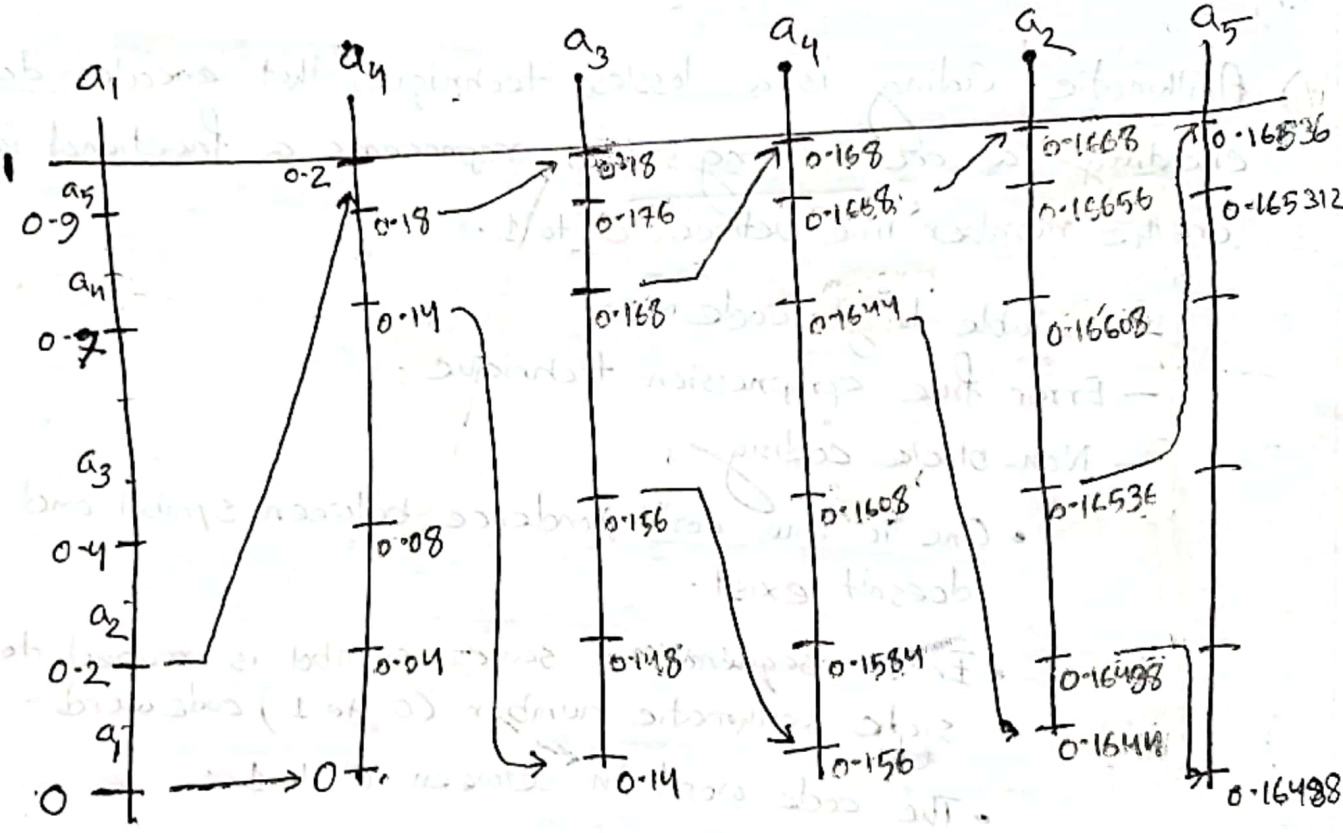
- Variable length code
- Error free compression technique.
- Non-block coding
  - One to one correspondence between symbol and code doesn't exist.
  - Entire sequence of source symbol is maped to a sigle arithmatic number (0 to 1) codeword.
  - The code word in between 0 to 1.

- Gives higher compression ratio than Huffman coding.

Given Sequence: $a_1\ a_4\ a_3\ a_4\ a_2\ a_5$

v)

Source Symbol	Probability	Initial Subinterval
a_1	0.2	[0.0, 0.2)
a_2	0.2	[0.2, 0.4)
a_3	0.3	[0.4, 0.7)
a_4	0.2	[0.7, 0.9)
a_5	0.1	[0.9, 1.0)



0.0012

0.00024

∴ Arithmatic code for sequence $a_1\ a_4\ a_3\ a_4\ a_2\ a_5$

is 0.165350

Integer-43

2(a)

Solved

3(a)

3	10	3	24	35
3	10	10	24	3
3	10	10	24	35
3	10	10	24	35

6-bit image.

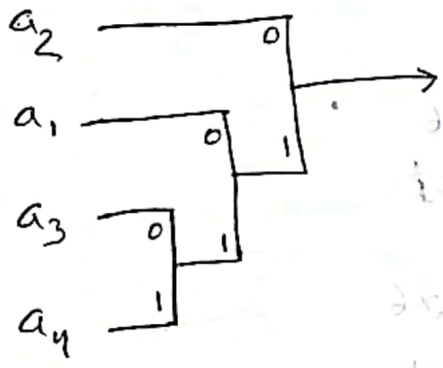
Pixel values	Probabilities
a_1 3	6/20 = 0.3
a_2 10	7/20 = 0.35
a_3 24	4/20 = 0.2
a_4 35	3/20 = 0.15

a_2 — 035

a_1 — 0.30

a_3 — 0.2

a_4 — 015



Symbol	Codeword
a_2	0
a_1	10
a_3	110
a_4	111

$$L_{Avg} = (0.3 \times 2) + (0.35 \times 1) + (0.2 \times 3) + (0.15 \times 3)$$

$$= 2 \text{ bits/pixel.}$$

ii)

First row: (3,1) (10,1) (3,1) (24,1) (35,1)

2nd row: (3,1) (10,2) (24,1) (3,1)

3nd row: (3,1) (10,2) (24,1) (35,1)

4th row: (3,1) (10,2) (24,1) (35,1)

RLC compressed file:

3,1,10,1,3,1,24,1,35,1,3,1,10,2,24,1,3,1,3,1,10,2,24,1,35,1,3,1,
10,2,24,1,35,1.

iii) Compressed ratio, C_R (Huff) $= \frac{6}{2} = 3$

~~Compressed ratio, C_R~~

For RLC;

original size = 4×5×6

= 120 bits

compressed size = 17×2×6

= 204 bits.

compressed ratio, $C_R = \frac{120}{204} = 0.58$

Decipher-44

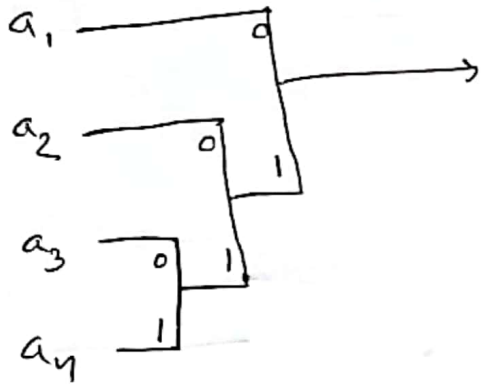
3(a)

i)

3	10	3	21	34
3	10	3	21	34
3	10	3	21	34
3	10	3	21	34

6-bit image

Pixel	Prob
$a_1 = 3$	$8/20 = 0.4$
$a_2 = 10$	$4/20 = 0.2$
$a_3 = 21$	$4/20 = 0.2$
$a_4 = 34$	$4/20 = 0.2$



Symbol	Codeword
a_1	0
a_2	10
a_3	110
a_4	111

$$L_{avg} = (0.4 \times 1) + (0.2 \times 2) + (0.2 \times 3) + (0.2 \times 3)$$

$$= 2 \text{ bits/pixel.}$$

$$\therefore \text{compression ratio} = \frac{6}{2} = 3$$

ii) $R_D = 1 - \frac{1}{3} = 0.667 = 66.7\%$ data were redundant.

$$H = -\left(0.4 \log_2(0.4) + 0.2 \log_2(0.2) + 0.2 \log_2(0.2) + 0.2 \log_2(0.2)\right)$$

$$= 1.92 \text{ bits/pixel}$$

here, $H < L_{avg}$ So, no. data loss occured

iii) $$\text{compression ratio} = \frac{8}{5.3} = 1.50$$

(Ans)

Lecture-4,5,6,7

Decipher-44

3(f)

i

2	5	8
5	1	3
4	7	2

→ Zero padding

0	0	0	0	0
0	2	5	8	0
0	5	1	3	0
0	4	7	2	0
0	0	0	0	0

Average filter (3×3) =

1/3	1/3	1/3
1/3	1/3	1/3
1/3	1/3	1/3

(1,1) Pixel = $\frac{1}{3}(0+0+0+0+2+5+0+5+1)$

$= 4.33 \approx 4$

(1,2) Pixel = $\frac{1}{3}(2+5+8+5+1+3)$

$= 8$

$(1,3) = \frac{1}{3}(5+8+1+3) = 5.67 \approx 6$

$(2,1) = \frac{1}{3}(2+5+5+1+4+7) = 8$

$(2,2) = \frac{1}{3}(2+5+8+5+1+3+4+7+2) = 12.33 \approx 12$

$(2,3) = \frac{1}{3}(5+8+1+3+7+2) \approx 8.67 = 9$

$(3,1) = \frac{1}{3}(5+1+4+7) = 5.67 \approx 6$

$(3,2) = \frac{1}{3}(5+1+3+4+7+2) = 7.33 \approx 7$

$(3,3) = \frac{1}{3}(1+3+7+2) = 3.25 \approx 3$

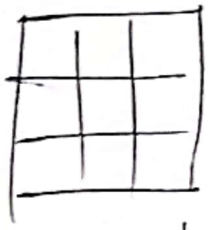
Filtered output image:

4	8	6
8	12	9
6	7	3

(Ans)

ii)

0	2	1	0
5	2	3	8
0	1	3	4
3	0	7	2



3×3 median filter

~~(2,2) → 0, 0, 1, 1, 2, 2, 3, 3, 5~~

~~→ 2~~

(2,2) → 0, 0, 1, 1, (2), 2, 3, 3, 5

→ 2

(2,3) → 0, 1, 1, 2, (2), 3, 3, 4, 8

→ 2

(3,2) → 0, 0, 1, 2, (3), 3, 3, 5, 7

→ 3

(3,3) → 0, 1, 2, 2, (3), 3, 4, 7, 8

→ 3

output image:

0	0	0	0
0	2	2	0
0	3	3	0
0	0	0	0

iii) no need.