

12. Type of image

The image types we will consider are:

- 1) Binary
- 2) grayscale
- 3) color
- 4) multispectral.

Example: find image size with 640*480 pixels

Total no. of bits = $640 \times 480 \times 8 \text{ bit} = 2457600 \text{ bit}$ Or $640 \times 480 \times 1 \text{ Byte} = 307200 \text{ Bytes}$

Convert to Byte: $2457600 / 8 = 307200 \text{ Byte}$

Convert to KByte: $307200 \text{ Byte} / 1024 = 300 \text{ KB}$

Examples: Color Image uses 32 bits with high = 200 and width = 200 pixels?
Size= $200 \times 200 \times 4$ Bytes = 160000B / 1024 = 156.25KB.

13. Image file formats

Some popular file formats are listed in the table below with their respective extensions:

Format Name	Description	Recognized Extensions
GIF	Graphics Interchange Format	.gif
JPEG	Joint Photographic Experts Group	.jpg, .jpeg
TIFF	Tagged Image File Format	.tif, .tiff
BMP	Windows Bitmap	.bmp
PNG	Portable Network Graphics	.png

5- PNG

PNG meant to supersede GIF standard

Features of PNG:

- ✚ Support up to 48 bits per pixel - more accurate colors
- ✚ Support description of gamma-correction and
- ✚ alpha-channel for controls such as transparency
- ✚ Support progress display through 8×8 blocks.

✚ Compress

✚ Can only be used online,

✚ Not for all application

3- TIFF

- ✚ Tagged Image File Format stores many different types of images (e.g., bit-map, greyscale, 8-bit & 24-bit RGB, etc.). **large file size**
- ✚ Developed by the Aldus Corp. in the 1980's and later supported by the Microsoft
- ✚ TIFF is a typically lossless format.
- ✚ It does not provide any major advantages over JPEG and is not as user-controllable
it appears to be declining in popularity
- ✚ **No color loss, raw data, no compress.**

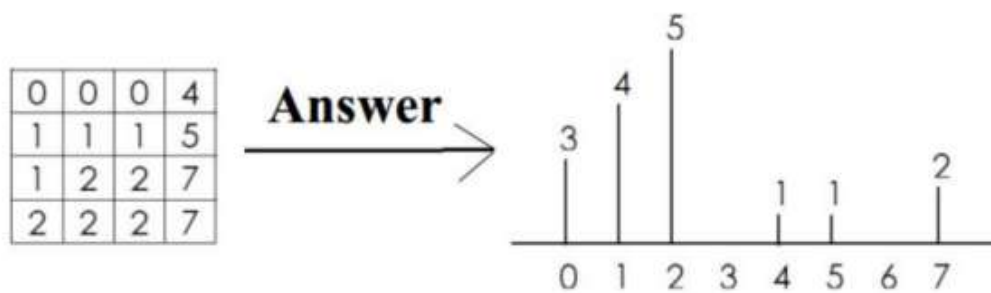
2- JPEG Standard

- ✚ A standard for photographic image compression created by the Joint Photographic Experts Group
- ✚ Takes advantage of limitations in the human vision system to achieve high rates of compression.
- ✚ Lossy compression which allows user to set the desired level of quality/compression.
- ✚ Minor color and detail loss through compression.
- ✚ Not good for sharp edges or text
- ✚ Millions of color

1- GIF

- Graphics Interchange Format (GIF) devised by the UNISYS Corp. and CompuServe, initially for transmitting graphical images over phone lines via modems.
- One of the simplest is the 8-bit GIF format, and we study it because it is easily understood, and also because of its historical connection to the WWW and HTML markup language as the first image
- type recognized by net browsers. **Perfect for online.**
- Limited to only 8-bit (256) colour images, suitable for images with few distinctive colours (e.g., graphics, drawing)
- GIF89a: supports simple animation, transparency index etc.
- **Great for sharp edges or text.**

Example: Plot the Histogram of the following example with 4x4 matrix of a 3-bit image.



Example: Apply histogram stretching for the following sub image :

7	12	8
[20	9	6]
10	15	1

Where: **Max =255** ;

Min =0

Solution:

$$St(r, c) = \left[\frac{I(r, c) - I(r, c)_{min}}{I(r, c)_{max} - I(r, c)_{min}} \right] (Max - Min) + Min$$

$I(r, c)_{min} = 1$;

$I(r, c)_{max} = 20$;

Max =255 ;

Min =0

$$I_{(0,0)} = [7-1 / 20-1] * [255 - 0] + 0 = 80.5$$

$$I_{(0,1)} = [12-1 / 20-1] * [255 - 0] + 0 = 147.6$$

$$I_{(0,2)} = [8-1 / 20-1] * [255 - 0] + 0 = 93.9$$

$$I_{(1,0)} = [20-1 / 20-1] * [255 - 0] + 0 = 255$$

$$I_{(1,1)} = [9-1 / 20-1] * [255 - 0] = 107.3$$

$$I_{(1,2)} = [6-1 / 20-1] * [255 - 0] + 0 = 67.1$$

$$I_{(2,0)} = [10-1 / 20-1] * [255 - 0] + 0 = 120.7$$

$$I_{(2,1)} = [15-1 / 20-1] * [255 - 0] + 0 = 187.8$$

$$I_{(2,2)} = [1-1 / 20-1] * [255 - 0] + 0 = 0$$

80	147	93
[255	107	67]
120	187	0

Example: Apply histogram shrink for the following sub image :

70	120	80
[200	90	60]
100	150	10

Where: **Shrink max =100 ; shrink min =20**

Solution:

$$\text{Shrink}(I(r, c)) = \left[\frac{\text{Shrink}_{\text{MAX}} - \text{Shrink}_{\text{MIN}}}{I(r, c)_{\text{MAX}} - I(r, c)_{\text{MIN}}} \right] [I(r, c) - I(r, c)_{\text{MIN}}] + \text{Shrink}_{\text{MIN}}$$

$I(r, c)_{\text{min}} = 10 ; \quad I(r, c)_{\text{max}} = 200 ; \quad \text{Shrink max} = 100 ; \quad \text{shrink min} = 20$

$$I_{(0,0)} = [100 - 20 / 200 - 10] * [70 - 10] + 20 = 45.2$$

$$I_{(0,1)} = [100 - 20 / 200 - 10] * [120 - 10] + 20 = 66.3$$

$$I_{(0,2)} = [100 - 20 / 200 - 10] * [80 - 10] + 20 = 49.4$$

$$I_{(1,0)} = [100 - 20 / 200 - 10] * [200 - 10] + 20 = 100$$

$$I_{(1,1)} = [100 - 20 / 200 - 10] * [90 - 10] + 20 = 53.68$$

$$I_{(1,2)} = [100 - 20 / 200 - 10] * [60 - 10] + 20 = 41.05$$

$$I_{(2,0)} = [100 - 20 / 200 - 10] * [100 - 10] + 20 = 57.89$$

$$I_{(2,1)} = [100 - 20 / 200 - 10] * [150 - 10] + 20 = 78.94$$

$$I_{(2,2)} = [100 - 20 / 200 - 10] * [10 - 10] + 20 = 20$$

45	66	49
[100	53	41]
57	78	20

Example: The source of information A generates the symbols {A0, A1, A2, A3 and A4} with the corresponding probabilities {0.4, 0.3, 0.15, 0.1 and 0.05}. Encoding the source symbols using binary encoder and Shannon-Fano encoder gives:

Source Symbol	P_i	Binary Code	Shannon-Fano
A0	0.4	000	0
A1	0.3	001	10
A2	0.15	010	110
A3	0.1	011	1110
A4	0.05	100	1111
L_{avg}	$H = 2.0087$	3	2.05

The Entropy of the source is:

$$H = - \sum_{i=0}^4 P_i \log_2 P_i = 2.0087 \text{ bit/symbol}$$

Since we have 5 symbols ($5 < 8 = 2^3$), we need 3 bits at least to represent each symbol in binary (fixed-length code). Hence the average length of the binary code is

$$L_{avg} = \sum_{i=0}^4 P_i l_i = 3 (0.4 + 0.3 + 0.15 + 0.1 + 0.05) = 3 \text{ bit/symbol}$$

Thus the efficiency of the binary code is

$$\eta = \frac{H}{L_{avg}} = \frac{2.0087}{3} = 67\%$$

Since we have 5 symbols ($5 < 8 = 2^3$), we need 3 bits at least to represent each symbol in binary (fixed-length code). Hence the average length of the binary code is

$$L_{avg} = \sum_{i=0}^4 P_i l_i = 3 (0.4 + 0.3 + 0.15 + 0.1 + 0.05) = 3 \text{ bit/symbol}$$

Thus the efficiency of the binary code is

$$\eta = \frac{H}{L_{avg}} = \frac{2.0087}{3} = 67\%$$

Example: Design Huffman codes for $A = \{a_1, a_2, \dots, a_5\}$, having the probabilities $\{0.2, 0.4, 0.2, 0.1, 0.1\}$.

Symbol	Step 1	Step 2	Step 3	Step 4	Codeword
a_2	0.4	0.4	0.4	0.6 0	1
a_1	0.2	0.2	0.4 } 0	0.4 1	01
a_3	0.2	0.2 } 0	0.2 } 1		000
a_4	0.1 } 0	0.2 } 1			0010
a_5	0.1 } 1				0011

Letter	Probability	Codeword
a_2	0.4	1
a_1	0.2	01
a_3	0.2	000
a_4	0.1	0010
a_5	0.1	0011

Example:

Find the size in Kbyte for a 10 minutes Full HD (1920x1080) video sequence (RGB type) with a frame rate = 30 fps.

Solutions:

RGB type = 24 bits=3Bytes

Time in seconds = 10 x 60 = 600

Video size = 1920 x 1080 x 3 x 30 x 600

Video size = 11547360000 Bytes

To convert to KBytes

Video size = 11547360000/1024

= 11276718.75 Kbytes

Video Rate (bits) = width x height x bit per pixel x frame rate

Example:

What is the bit rate for high-definition TV (HDTV)?

Solution: $1920 \times 1080 \times 30 \times 24 = 1492992000 = 1.39 \text{ Gbps}$

HDTV uses digital signals to broadcast high quality video signals. The HDTV screen is normally a ratio of 16: 9.

There are 1920 by 1080 pixels per screen, and the screen is renewed 30 times per second. Twenty-four bits represents one color pixel.