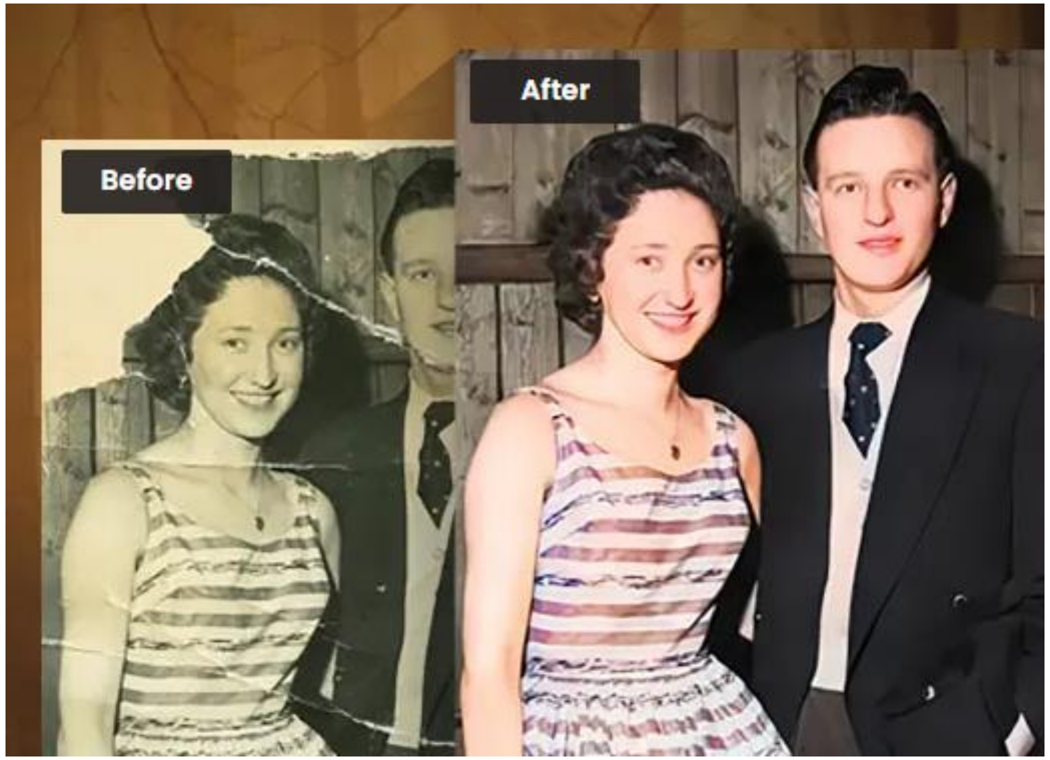
Unit 3

Image Restoration and Compression

Image Restoration

- **Image Restoration** is a family of inverse problems for obtaining a high quality image from a corrupted input image
- The purpose of image restoration is to "compensate for" or "undo" defects which degrade an image.
- Degradation comes in many
  - motion blur
  - noise, and
  - camera miss-focus.
- In cases like motion blur, it is possible to come up with an very good estimate of the actual blurring function and "undo" the blur to restore the original image.
- In cases where the image is corrupted by noise, the best we may hope to do is to compensate for the degradation it caused.

Cot..



Models for Image degradation and restoration process

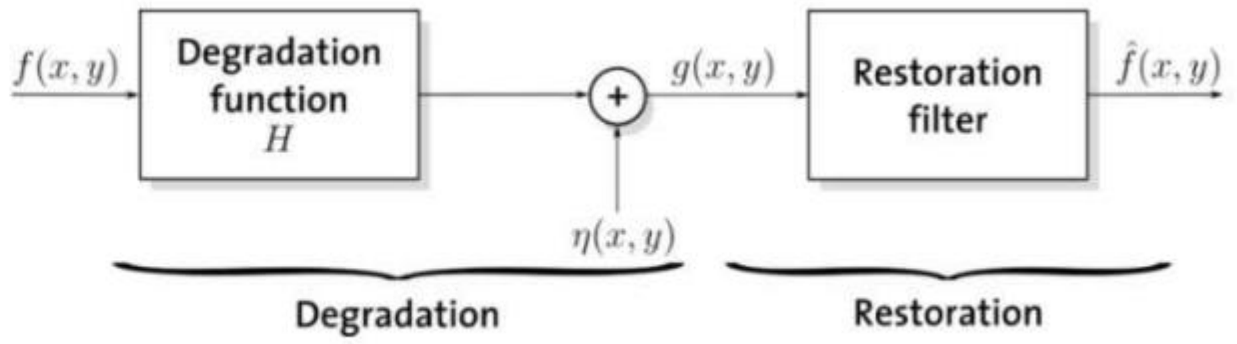


Fig: Image Degradation and Restoration Model

The input image f(x,y) is processed by degradation function H, that together with an additive noise term n(x,y) to produce degraded image g(x,y). the g(x,y) consist of some knowledge about noise term n(x,y) and some knowledge about degradation function H. The objective of image restoration is to obtain an estimate of original image f(x,y). Here, by some knowledge of H and n(x,y), we find the appropriate restoration filters, so that output image f'(x,y) is as close as original image f(x,y). Since, it is practically not possible (or very difficult) to completely restore the original image.

In Spatial Domain

Degraded image is represented by

g(x,y) = h(x,y) * f(x,y) + n(x,y)

Where,

- **h(x,y)** = Degradation function
- * = Indicates the convolution operation
- **f(x,y)** = Original Image
- **g(x,y)** = Degraded image
- **n(x,y)** = additive noise term

In Frequency Domain

Degraded image is represented by

$G(u,v) = H(u,v) * F(u,v) + N(u,v)$

If the restoration filter applied is R(u,v)

then,

$F'(u,v) = R(u,v) [G(u,v)]$

$F'(u,v) = R(u,v) H(u,v) F(u,v) + R(u,v) N(u,v)$

$F'(u,v) F(u,v)$

Where, F'(u,v) is the restored image

Here, restoration filter

R(u,v) is the reverse of degradation function H(u,v).

Noise Models

- Noise tells unwanted information in digital images.
- Noise produces undesirable effects such as artifacts, unrealistic edges, unseen lines, corners, blurred objects and disturbs background scenes.
- To reduce these undesirable effects, prior learning of noise models is essential for further processing.
- Digital noise may arise from various kinds of sources such as
- Charge Coupled Device (CCD) and Complementary Metal Oxide Semiconductor (CMOS) sensors
- In some sense, points spreading function (PSF) and modulation transfer function (MTF) have been used for timely, complete and quantitative analysis of noise models.
- Probability density function (PDF) or Histogram is also used to design and characterize the noise models.

Gaussian Noise

- It is also called as electronic noise because it arises in amplifiers or detectors. Gaussian noise caused by natural sources such as thermal vibration of atoms and discrete nature of radiation of warm objects.
- Because of its mathematical simplicity, the Gaussian noise model is often used in practice and even in situations where they are marginally applicable at best. Here, m is the mean and σ^2 is the variance.
- Gaussian noise generally disturbs the gray values in digital images. That is why Gaussian noise model essentially designed and characteristics by its PDF or normalizes histogram with respect to gray value. This is given as

$$p(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z-m)^2}{2\sigma^2}}$$

Rayleigh Noise

$$p(z) = \frac{2}{b}(z-a)e^{\frac{-(z-a)^2}{b}} for z \geq a, and p(z) = 0 otherwise.$$

Here mean m and variance σ^2 are the following:

$$m = a + \sqrt{\pi b/4}$$

$$\sigma^2 = \frac{b(4-\pi)}{4}$$

Rayleigh noise is usually used to characterize noise phenomena in range imaging.

Erlang (or gamma) Noise

$p(z) = \frac{a^b z^{b-1}}{(b-1)!} e^{-az} for z \geq 0 and p(z) = 0 otherwise.$

Here ! indicates factorial. The mean and variance are given below.

$m = b/a, 2 = b/a2$

Gamma noise density finds application in laser imaging.

Exponential Noise

$p(z) = ae^{-az} forz \geq 0andp(z) = 0otherwise.$

Here a > 0. The mean and variance of this noise pdf are:

$m = 1/a$
$2 = 1/\mathrm{a}2$

This density function is a special case of b = 1.

Exponential noise is also commonly present in cases of laser imaging.

Uniform Noise

$p(z) = \frac{1}{b-a} if a \leq z \leq b, and p(z) = 0 otherwise.$

The mean and variance are given below.

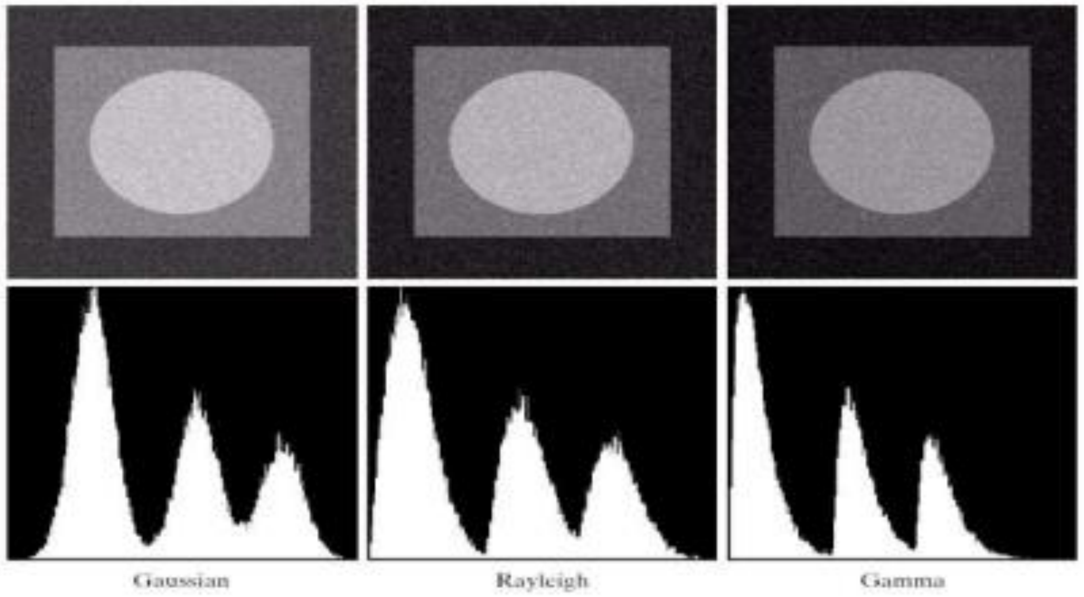
$m = \frac{a+b}{2}$
$\sigma^2 = \frac{(b-a)^2}{12}$

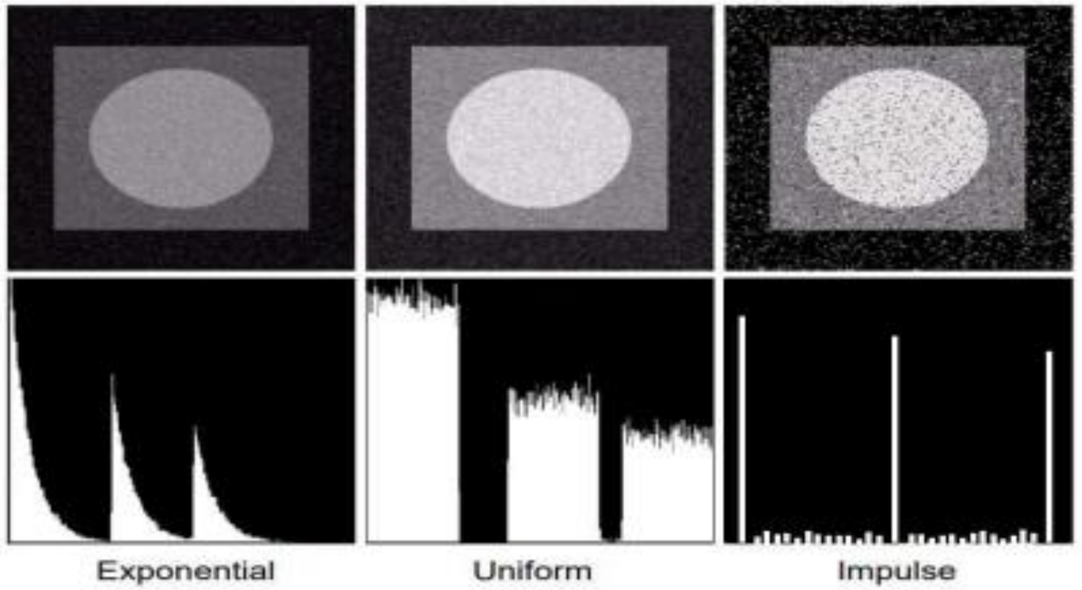
Uniform noise is not practically present but is often used in numerical simulations to analyze systems.

Impulse Noise

$p(z) = Pa for z = a, p(z) = Pb for z = b, p(z) = 0 otherwise.$

If b > a, intensity b will appear as a light dot in the image. Conversely, level a will appear like a black dot in the image. Hence, this presence of white and black dots in the image resembles to salt-and-pepper granules, hence also called salt-and-pepper noise. When either P_a or P_b is zero, it is called unipolar noise. The origin of impulse noise is quick transients such as faulty switching in cameras or other such cases.


Gaussian
Rayleigh
Gamma


Exponential
Uniform
Impulse

Restoration Filters

Mean Filters:

Arithmetic mean filter:

This is the simplest of the mean filters, S_{xy} represent the set of coordinates in a rectangular sub-image window (*neighborhood*) of size $m*n$, centered at point (x,y).

The arithmetic mean filter computes the average value of the corrupted image $g(x,y)$ in the area defined by S_{xy}. The value of the restored image f' at point (x,y) is simply the arithmetic mean computed using the pixels in the region defined by S_{xy}. A mean filter smooth local variations in an image, and noise is reduced as a result of blurring.

Arithmetic mean filter can define by an expression

$$\hat{f}(x,y) = \frac{1}{mn} \sum_{(s,t)\in S_{xy}} g(s,t)$$

Where, m and n are the height and width of S_{xy}

Mask of arithmetic mean filter do convolution to image with this mask to filter the image

$$\frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

Then, Result will replace the central pixel of the given image.

Geometric mean filter:

A geometric mean filter achieves smoothing comparable to the arithmetic mean filter, but it tends to loss less image detail in the process.

Geometric mean filter can define by an expression

$$\hat{f}(x,y) = \left[\prod_{(s,t)\in S_{xy}} g(s,t) \right]^{\frac{1}{mn}}$$

Here, each restored pixel is given by the product of the pixels in the sub-image window, raised to the power ***1/mn***.

Then, Result will replace the central pixel of the given image.

Example: Apply geometric mean filter to given image

1	2	3
4	5	6
7	8	9

Solution:

$$(\mathbf{1} \times \mathbf{2} \times \mathbf{3} \times \mathbf{4} \times \mathbf{5} \times \mathbf{6} \times \mathbf{7} \times \mathbf{8} \times \mathbf{9})^{\frac{1}{9}}$$

$(362880)^{0.11}$

4.09

Result will replace the central pixel of the given image.

Restored Image is:

1	2	3
4	4.09	6
7	8	9

Harmonic mean filter:

It works well for salt noise, but fails for pepper noise. It also works well for Gaussian noise.

Harmonic mean filter can define by an expression

$$\hat{f}(x,y) = \frac{mn}{\sum_{(s,t)\in S_{xy}} \frac{1}{g(s,t)}}$$

Then, Result will replace the central pixel of the given image.

Example: apply harmonic mean filter to given image

1	2	3
4	5	6
7	8	9

Solution:

$$= \frac{9}{\frac{1}{1}+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\frac{1}{5}+\frac{1}{6}+\frac{1}{7}+\frac{1}{8}+\frac{1}{9}}$$

$$= \frac{9}{1+0.5+0.33+0.25+0.2+0.17+0.14+0.13+0.11}$$

$$= \frac{9}{2.83}$$

= 3.18

Result will replace the central pixel of the given image.

Restored Image is:

1	2	3
4	3.83	6
7	8	9

Order Statistics Filters:

Spatial filters whose response is based on ordering (*sorting*) the values of the pixels contained in the image area encompassed by the filter is called order statics filters.

Commonly used order statistics filters include

- Median filter
- Max and min filter
- Midpoint filter
- Alpha trimmed mean filter

Median filter:

The best known order statistic filter is the median filter, which replaces the value of the central pixel by the median of the intensity levels in the neighborhood pixel.

To find the median value we have to sort the pixels of given image in ascending order first.

$$\hat{f}(x,y) = \underset{(s,t)\in S_{xy}}{median}\{g(s,t)\}$$

Then, Result will replace the central pixel of the given image.

The value of the pixel at (x,y) is included in the computation of the median.

For certain types of random noise it provides excellent noise reduction capabilities with considerably less blurring than linear smoothing filter of the same size. It is particularly good when salt and pepper noise is present.

Max filter and Min filter:

Max filter is good for pepper noise and Min filter is good for salt noise.

Max filter equation is

$$\hat{f}(x,y) = \max_{(s,t)\in S_{xy}} \{g(s,t)\}$$

Min filter equation is

$$\hat{f}(x,y) = \min_{(s,t)\in S_{xy}} \{g(s,t)\}$$

Then, Result will replace the central pixel of the given image.

Midpoint filter:

It simply computes the midpoint between the maximum and minimum values in the area encompassed by the filter. It is good for Gaussian noise and uniform noise.

$$\hat{f}(x,y) = \frac{1}{2}\left[\max_{(s,t)\in S_{xy}} \{g(s,t)\} + \min_{(s,t)\in S_{xy}} \{g(s,t)\}\right]$$

Then, Result will replace the central pixel of the given image.

Alpha-Trimmed Mean Filter:

It is useful in situation involving multiple types of noise, such as a combination of salt-and-pepper noise and Gaussian noise.

Suppose, we delete the ***d/2*** lowest and ***d/2*** highest intensity values of ***g(s,t)*** in the neighborhood of S_{xy}. Let $g_r(s,t)$ represents the remaining ***(mn – d)*** pixels, the filter formed by averaging these remaining pixels is called an Alpha-Trimmed mean filer and represented by expression

$$\hat{f}(x,y) = \frac{1}{mn-d} \sum_{(s,t)\in S_{xy}} g_r(s,t)$$

Where, the value of ***d*** can range from ***0*** to ***mn-1***, when ***d = 0***, the alpha-trimmed filter reduces to the arithmetic mean filter and if ***d = mn-1***, filter becomes median filter.

Then, Result will replace the central pixel of the given image.

Band rejected Filters

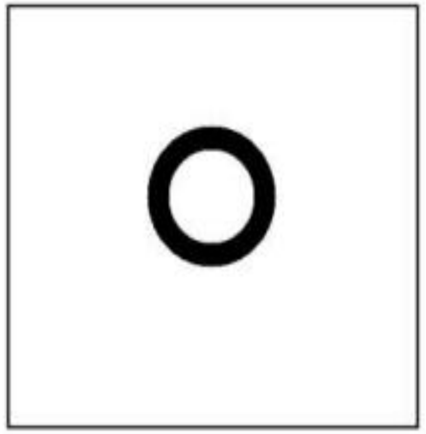
- A band reject filter is useful when the general location of the noise in the frequency domain is known. A band reject filter blocks frequencies within the chosen range and lets frequencies outside of the range pass through.

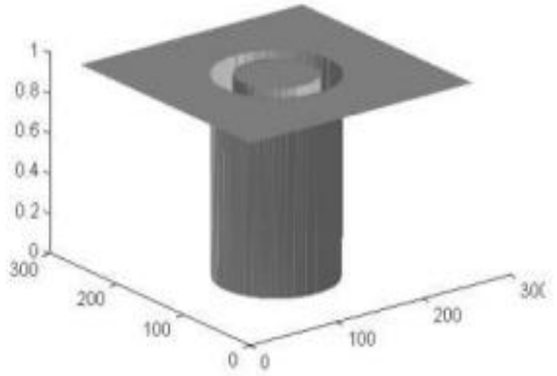
Ideal Band Reject Filter

$$H(u,v) = \begin{cases} 0 & \text{if } D_0 - \frac{W}{2} \le D(u,v) \le D_0 + \frac{W}{2} \\ 1 & \text{otherwise} \end{cases}$$

Where,

- ***D(u,v)*** *is the distance of pixel* ***(u,v)*** *to the frequency center* ***(0,0)*** *and can be calculated by formula:* $D(u,v) = [u^2 + v^2]^{1/2}$
- ***W*** *is the width of the band (range)*
- D_0 *is the radial center*



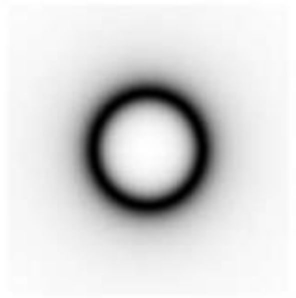


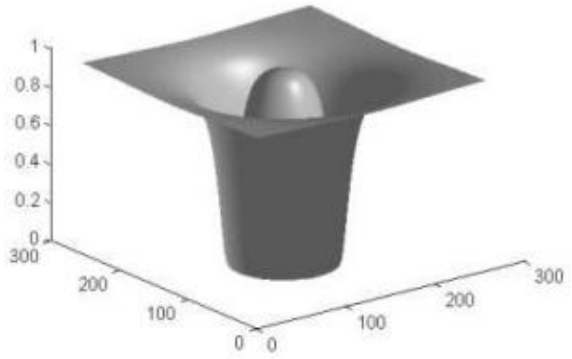
Butterworth Band Reject Filter:

$$H(u,v) = \frac{1}{1 + \left[\frac{WD(u,v)}{D^2(u,v) - D_0^2} \right]^{2n}}$$

Where,

- ***D(u,v)*** *is the distance of pixel* ***(u,v)*** *to the frequency center* ***(0,0)*** *and can be calculated by formula:* $D(u,v) = [u^2 + v^2]^{1/2}$
- ***W*** *is the width of the band (range)*
- D_0 *is the radial center*
- ***n*** *is the order of the filter*



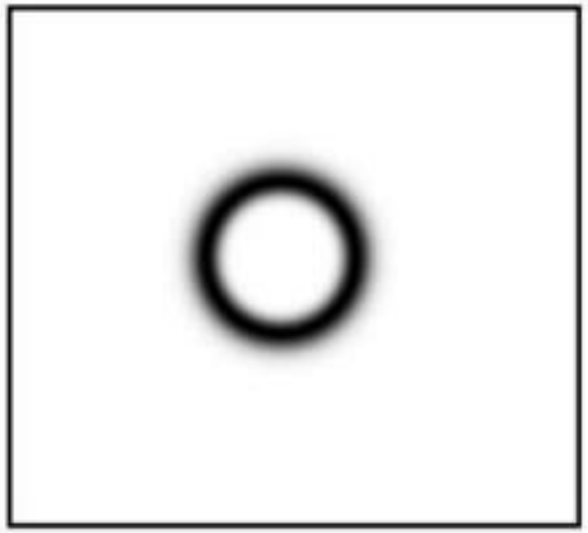


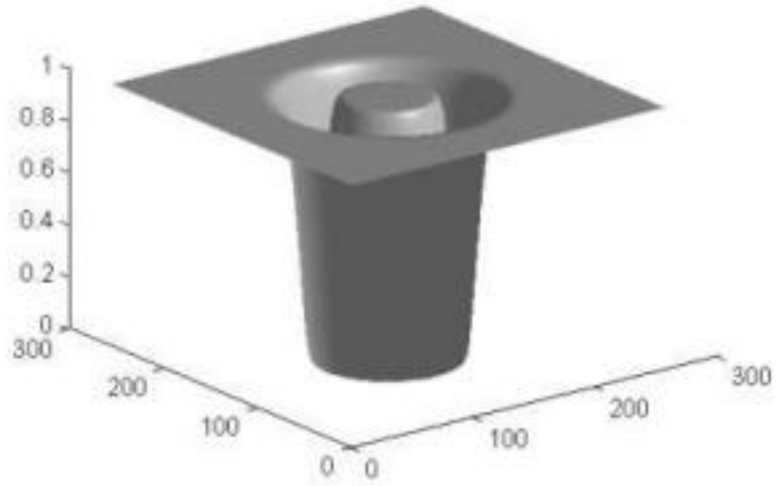
Gaussian Band Reject Filter:

$$H(u,v) = 1 - e^{-\left[\frac{D^2(u,v) - D_0^2}{WD(u,v)}\right]^2}$$

Where,

- ***D(u,v)*** *is the distance of pixel* ***(u,v)*** *to the frequency center* ***(0,0)*** *and can be calculated by formula:* $\mathbf{D(u,v) = [u^2 + v^2]^{1/2}}$
- ***W*** *is the width of the band (range)*
- $\mathbf{D_0}$ *is the radial center*





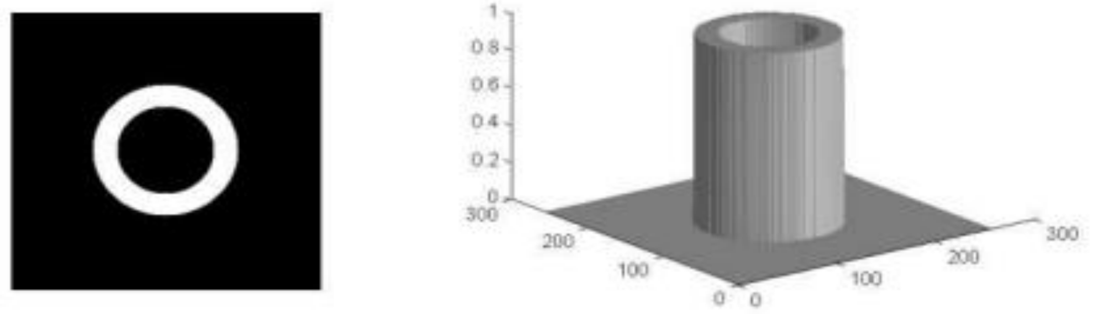
Band pass Filters

- Band pass filter performs the opposite operation of a band reject filter.
- It let only a portion of the frequency pass. Performing straight band pass filtering on an image is not a common procedure because it generally removes too much image details.

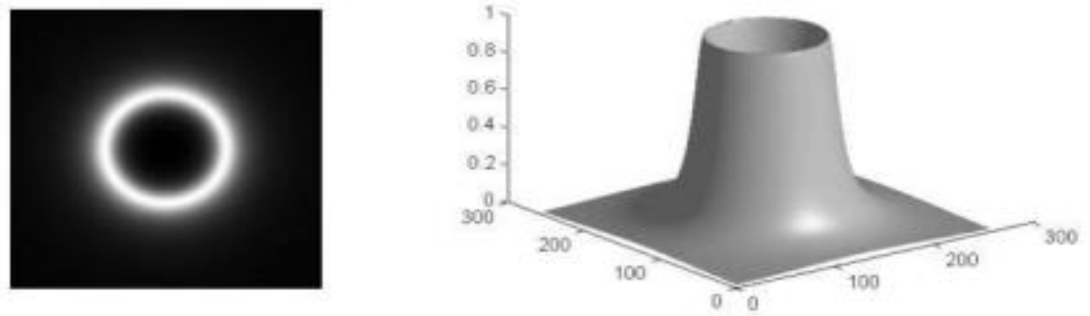
- This filter is defined by the expression

- $H_{BP}(u,v) = 1 - \ H_{BP}(u,v)$

Ideal Band Pass Filter:



Butterworth Band Pass Filter:



Butterworth band pass filter at order of 1

Gaussian Band Pass Filter:

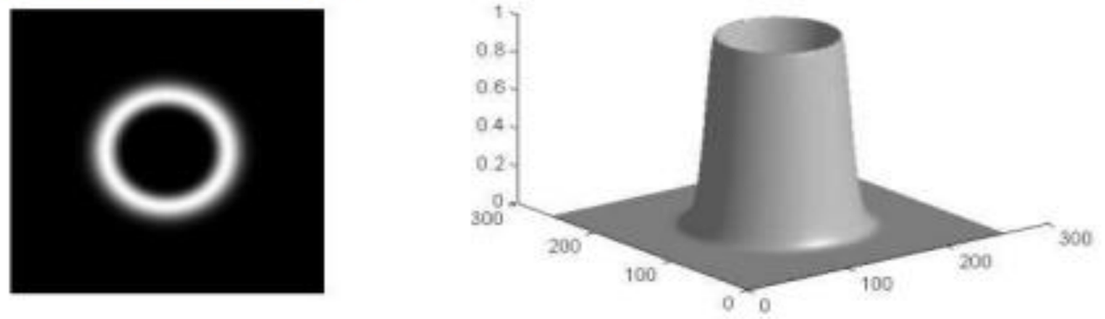


Image Compression

- Image compression involves reducing the file size of an image while maintaining as much of its quality as possible.
- This is useful for saving storage space and improving the performance of websites and applications.
- There are two main types of image compression
  - **Lossy Compression**
  - **Lossless Compression:**

Why image compression important?

- Compressed images load faster than uncompressed images. This matters because the speed at which webpages and applications load has a huge impact on SEO, conversion rates, the user's digital experience, and other crucial metrics. Improving web performance is one of the major ways that developers optimize websites.
- Image compression is typically used alongside other methods for improving web performance as well. For instance, a CDN caches content to deliver it more quickly to end users. Load balancing helps prevent web servers from becoming overloaded. The use of lazy loading can allow the most important content of a webpage to load even faster. Overall, however, image compression is often one of the quickest ways of fixing slow page performance.

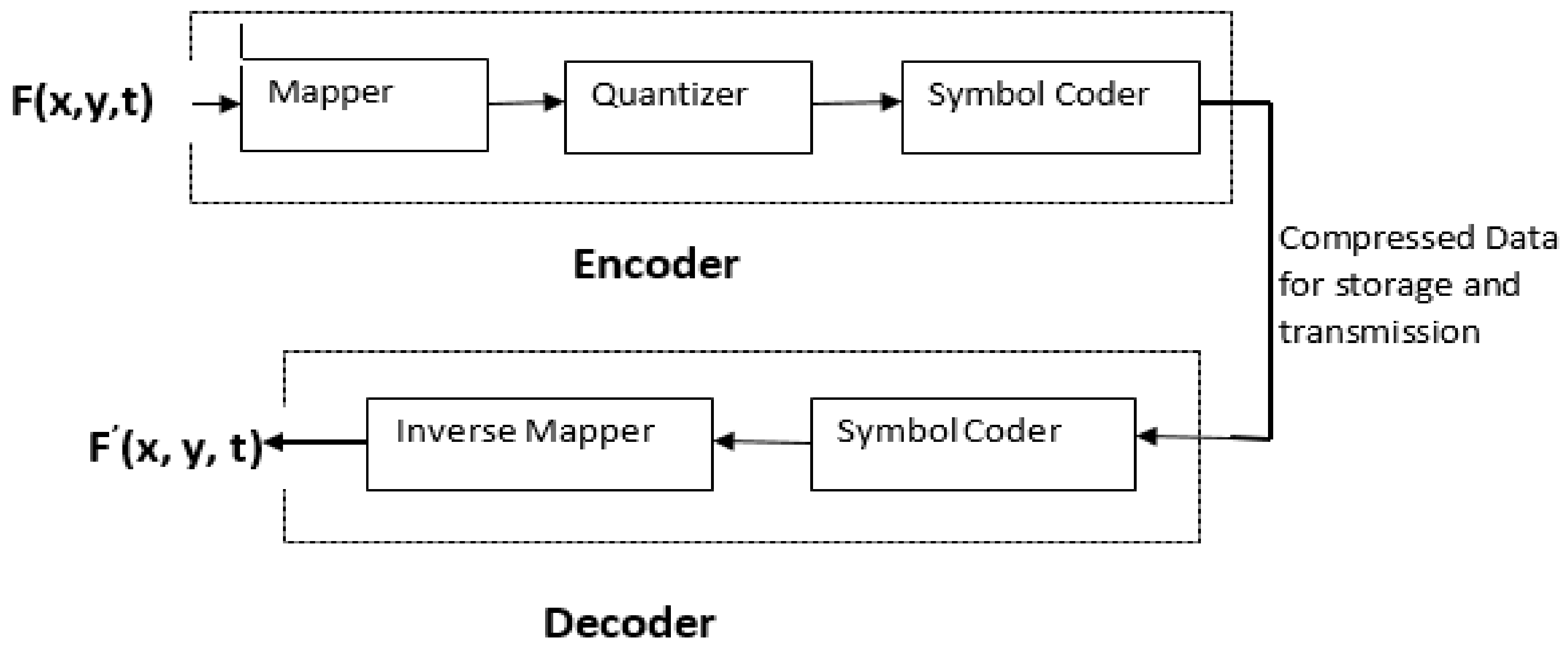
Lossy Compression and Lossless Compression

S.NO	Lossy Compression	Lossless Compression
1.	Lossy compression is the method which eliminate the data which is not noticeable.	While Lossless Compression does not eliminate the data which is not noticeable.
2.	In Lossy compression, A file does not restore or rebuilt in its original form.	While in Lossless Compression, A file can be restored in its original form.
3.	In Lossy compression, Data's quality is compromised.	But Lossless Compression does not compromise the data's quality.
4.	Lossy compression reduces the size of data.	But Lossless Compression does not reduce the size of data.

.	Algorithms used in Lossy compression are: **Transform coding, Discrete Cosine Transform, Discrete Wavelet Transform, fractal compression** etc.	Algorithms used in Lossless compression are: **Run Length Encoding, Lempel-Ziv-Welch, Huffman Coding,Arithmetic encoding** etc.
6.	Lossy compression is used in Images, audio, video.	Lossless Compression is used in Text, images, sound.
7.	Lossy compression has more data-holding capacity.	Lossless Compression has less data-holding capacity than Lossy compression technique.
8.	Lossy compression is also termed as irreversible compression.	Lossless Compression is also termed as reversible compression.

Image compression models

- Compression has two types i.e. Lossy and Lossless technique.
- A typical image compression system comprises of two main blocks An Encoder (Compressor) which is a software application used to compress an image and Decoder (Decompressor) which is a software application used to de-compress an image.
- The image f(x,y) is fed to the encoder which encodes the image to make it suitable for transmission.
- The decoder receives this transmitted signal and reconstructs the output image f’(x,y).
- If the system is an error free the image f’(x,y) will be a replica of f(x,y).
- Input and output function f(x,y,t) and f’(x,y,t) respectively are used for video application where t specified the time.


F(x,y,t)
Mapper
Quantizer
Symbol Coder
Encoder
Compressed Data for storage and transmission
F´(x, y, t)
Inverse Mapper
Symbol Coder
Decoder

- **Encoder (Compressor):**
- The three basic types of the redundancies in an image are interpixel, coding redundancies and psychovisual redundancies. Run length coding is used to eliminate or reduce inter-pixel redundancies, Huffman encoding is used to eliminate or reduce coding redundancies and psychovisual is used to eliminate interpixel redundancies. The input image is passed through a **Mapper**. The mapper reduces the interpixel redundancies. The mapping stage is a lossless technique and hence is a reversible operation. Run-length coding is used to compress the image in this stage. The output of a mapper is passed to a **Quantizer** block. The quantizer block reduces the psychovisual redundancies. It compresses the data by eliminating some redundant information and hence is an irreversible operation. The quantizer block uses JPEG compression which means a lossy compression. Hence in case of lossless compression, the quantizer block is omitted.
- The final block of the encoder is **Symbol Coder**. This block creates a variable length code to represent the output of the quantizer. The shortest code length is used for most frequent quantizer output to reduce coding redundancy. This operation is reversible. The Huffman code is a typical example of the symbol coder.
- **Decoder (De-compressor):** The **Symbol Decoder** and **Inverse Mapper** of decoder block perform exactly the reverse operation of the symbol coder and the mapper blocks respectively. It is important to note that the decoder has only two blocks. Since quantization is irreversible hence, an inverse quantizer block does not exist.

standards and coding Techniques

- Image compression is a process that makes image files smaller. Image compression most often works either by removing bytes of information from the image, or by using an image compression algorithm to rewrite the image file in a way that takes up less storage space.
- Compressing an image is an effective way to ensure that the image loads quickly when a user interacts with a website or application. It is an important part of image optimization.

Inter-pixel Redundancy (Run Length Coding)

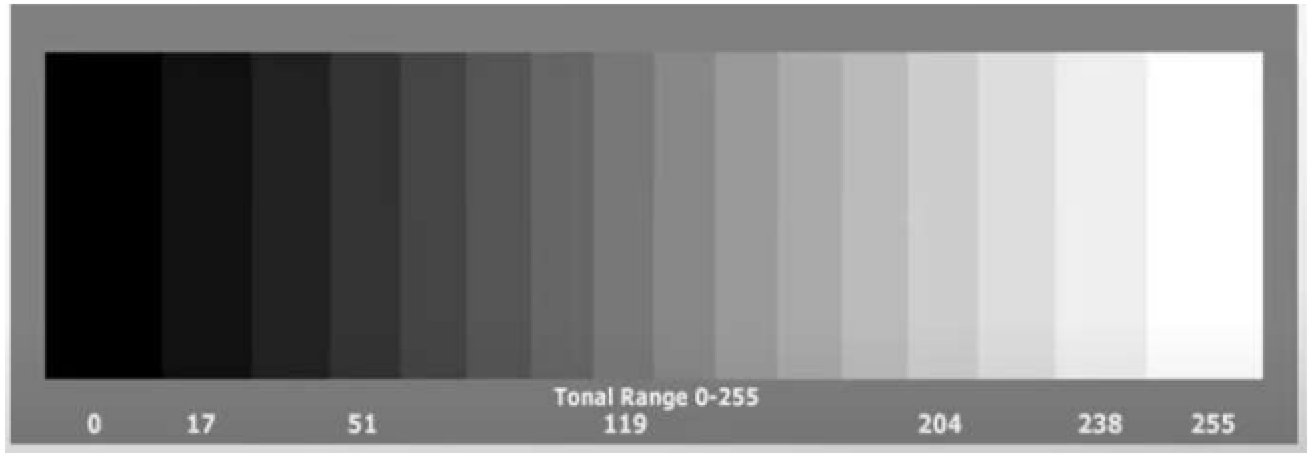
- It is a simplest data compression technique. It is a form of lossless data compression in which runs of data (sequences in which the same data value occurs in many consecutive data elements) are stored as a single data value and count, rather than as the original run. This is most useful on data that contains many such runs.

- The general idea behind this method is to replace consecutive repeating occurrences of a symbol by one occurrence of the symbol followed by the number of occurrences.

- Example 1: Suppose string is AAAAAAA then Run-length encoding is A7 (A is character and 7 is number of times appear that string)
- Example 2: If input string is “WWWWAAADEXXXXXX” then the Run-length encoding is W4A3D1E1X6.

Psychovisual Redundancy (4-bit Improved Gray Scale Coding: IGS Coding Scheme)

- The tendency of certain kinds of information to be relatively unimportant to the human visual system, The Psychovisual redundancies exist because human perception does not involve quantitative analysis of every pixel or luminance value in the image. During human eye visualization certain information can be eliminated without significantly degrading image quality. For example:



Here, in the above image, there are different levels of shades, human eyes can't differentiate all the shades like gray scale 253, 254, and 255 are like same so, 253 and 254 can illuminate and replace by 255.

Coding Redundancy (Huffman Coding):

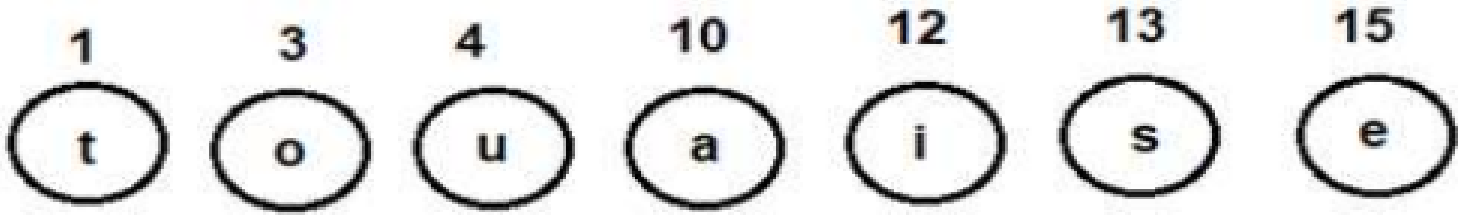
- Huffman coding is one of the basic lossless compression methods that have proven useful in image and video compression standards. When applying Huffman encoding technique on an Image, the source symbols can be either pixel intensities of the Image, or the output of an intensity mapping function.
- The first step of Huffman coding technique is to reduce the input image to a ordered histogram, where the probability of occurrence of a certain pixel intensity value is as
- **Probability of Pixel =** Number of Pixel / Total Number of Pixel
- Where, Number of Pixel is the number of occurrence of a pixel with a certain intensity value and Total Number of Pixel is the total number of pixels in the input Image.
- **Build a Huffman Tree:**
    1. Combine the two lowest probability leaf nodes into a new node.
    2. Replace the two leaf nodes by the new node and sort the nodes according to the new probability values.
    3. Continue the steps (a) and (b) until we get a single node with probability value 1.0. We will call this node as root

    Calculate Huffman Code: Backtrack from the root, assigning „0" to left child and „1" to the right child in each intermediate node, till we reach the leaf nodes. Then assemble the code 0 and 1 from root to pixel position

Example: Create Huffman Tree for the following characters along with their frequencies using the Huffman algorithm. Also find the Huffman code for each character.

Characters	**Frequency**	**Probability**
A	10	0.17
E	15	0.26
I	12	0.21
O	3	0.05
U	4	0.07
S	13	0.22
T	1	0.02

Create leaf nodes for all the characters and add them to the min heap.



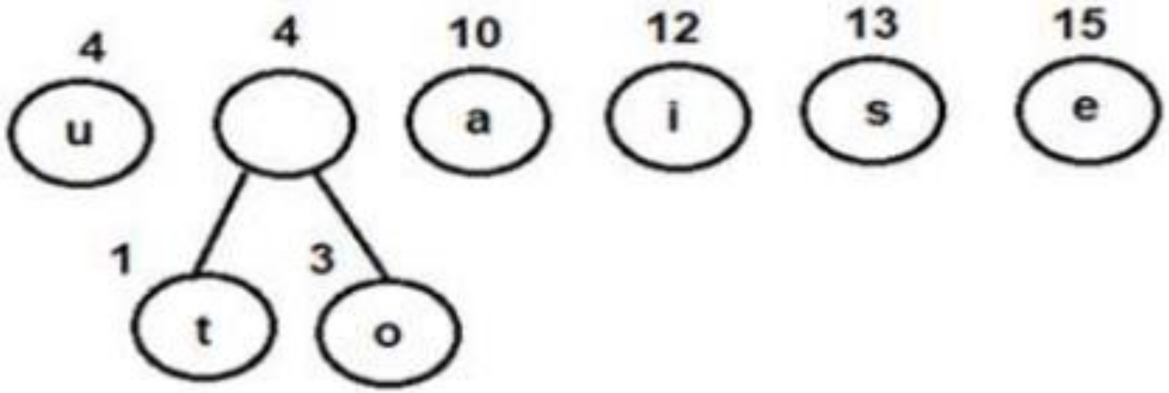
Repeat the following steps till heap has more than one node

Step 1: Extract two nodes, say **x** and **y,** with minimum frequency from the heap

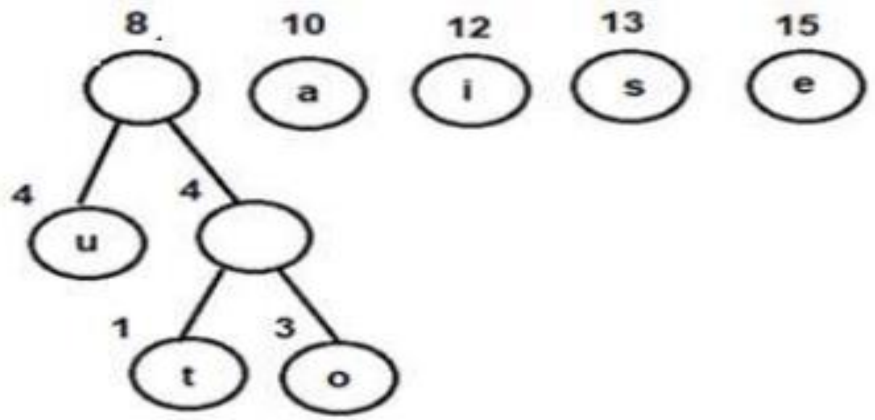
Step 2: Create a new internal node **z** with **x** as its ***left child*** and **y** as its ***right child***. Also frequency (z) = frequency (x) + frequency (y)

Step 3: Add z to min heap

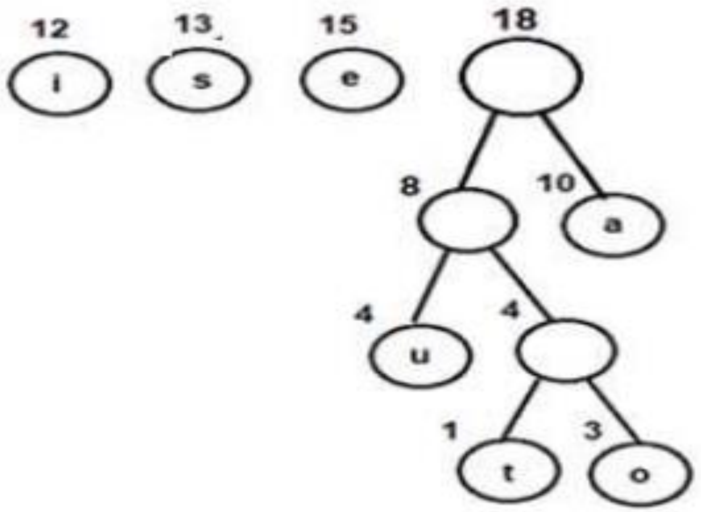
Here in above min heap, node **t** and node **o** have minimum frequency, combine those two and place to min heap.



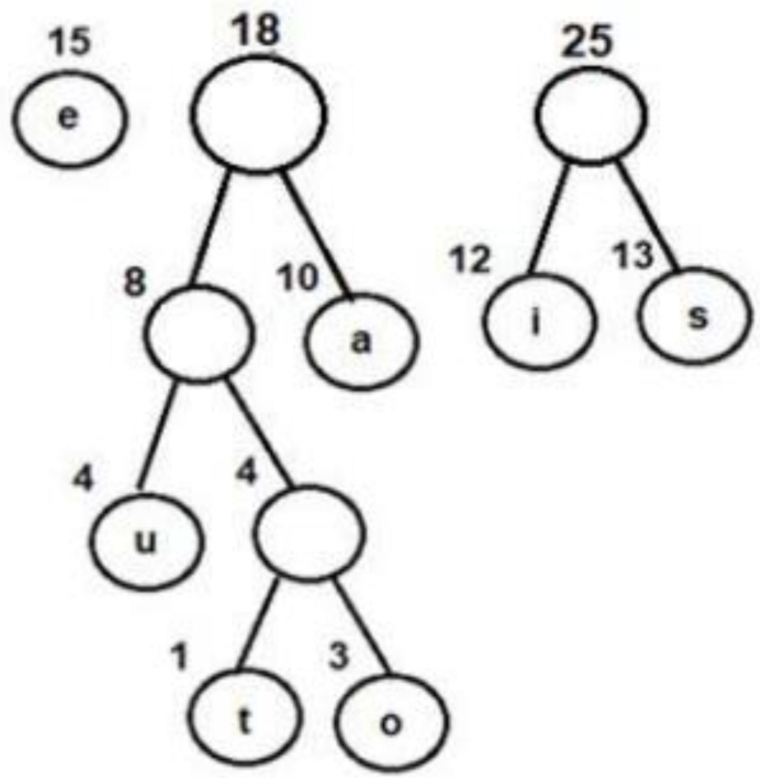
Now, combine node **u** and new tree and add to heap.



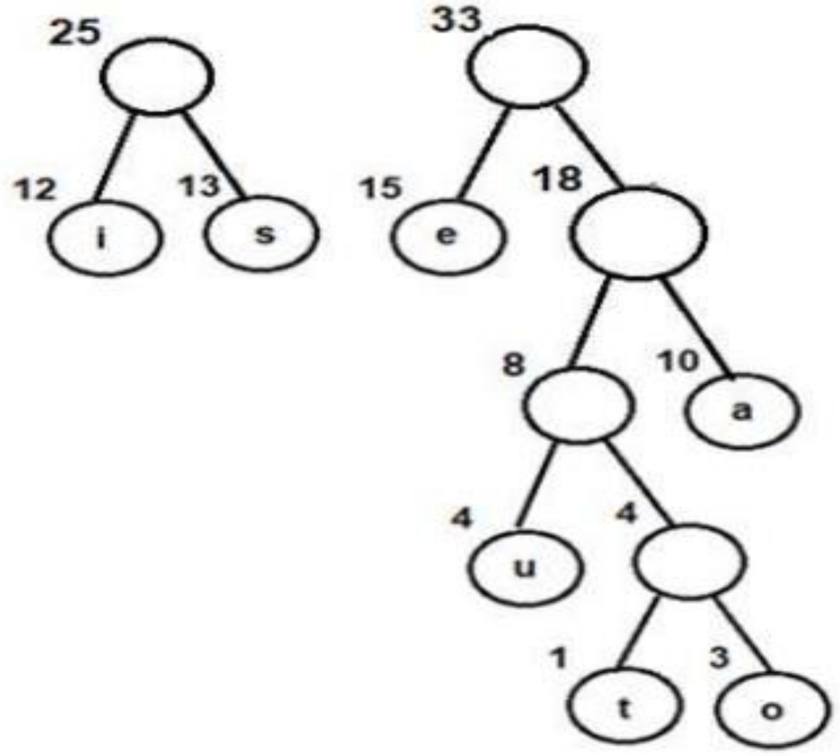
Now, combine node **a** and new tree and add to min heap.



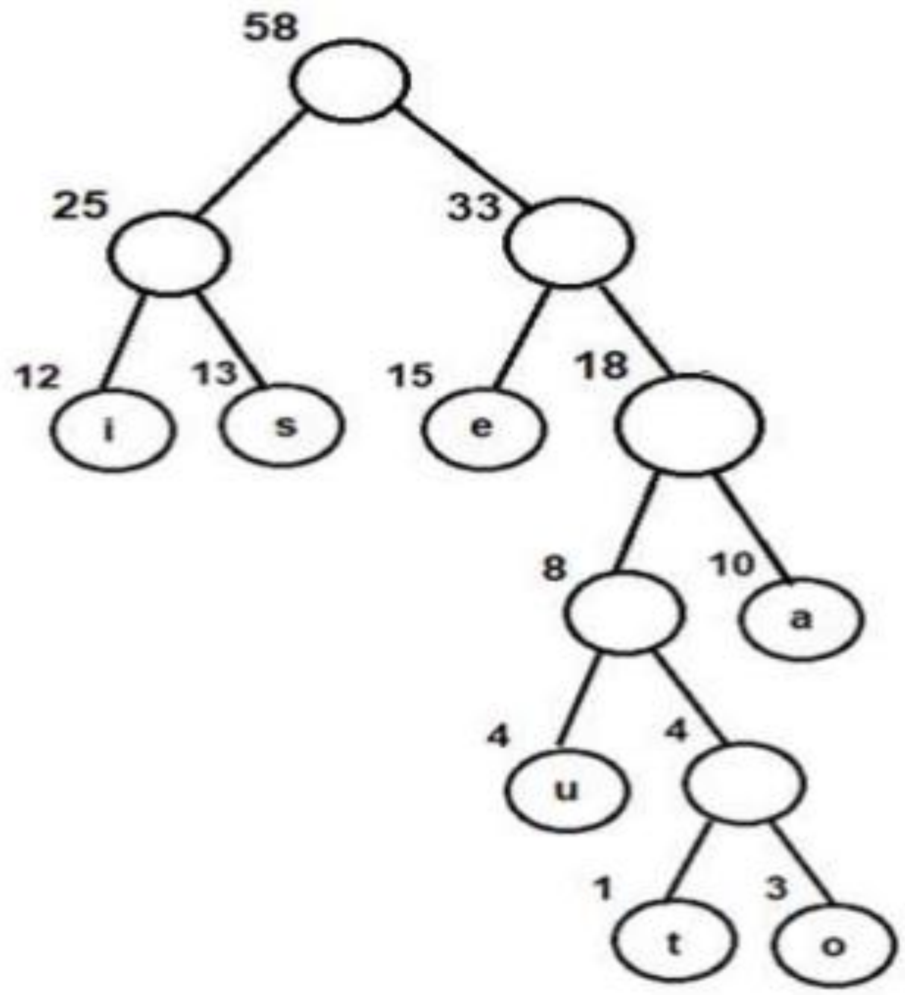
Now, combine node **i** and **s** which are minimum two nodes, and add to min heap



Now, combine **e** with tree having root node 18 and add to min heap



Now, combine remaining two trees

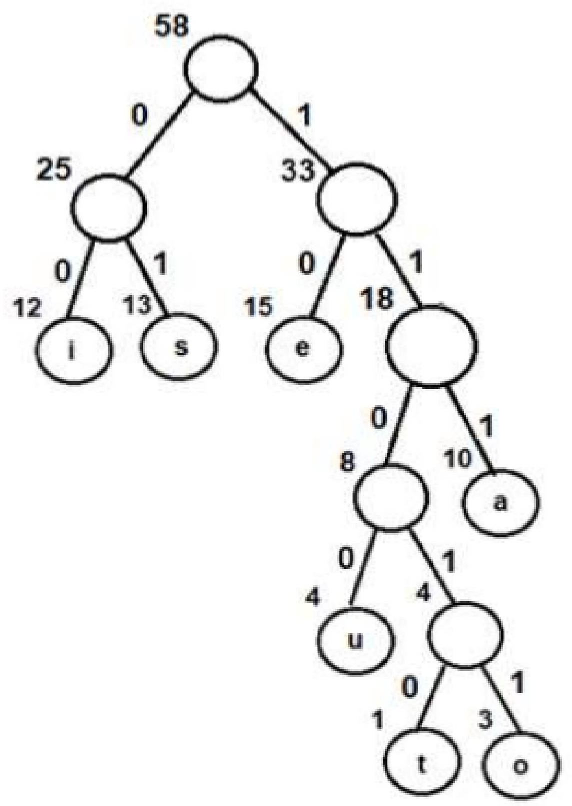


This is the final Huffman tree.

Calculate Huffman Code:

Traverse Huffman tree to get Huffman code for each characters.

Place 0 for left child and 1 for right child in the Huffman tree that we previously calculated.



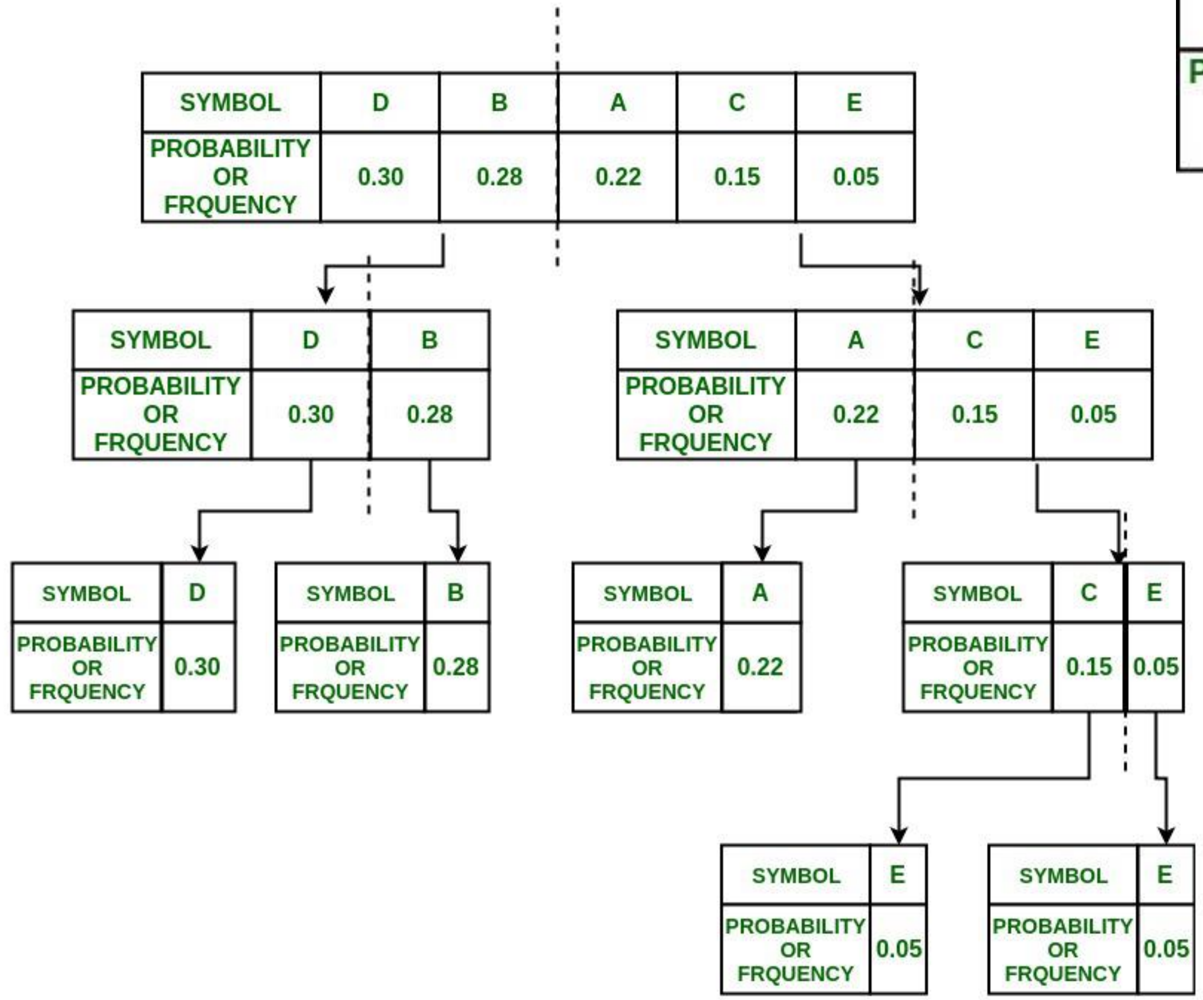
Characters	Probability	Huffman Codes	Code Length	Average Code Length Sum(Pi x CLi)
A	0.17	111	3	0.51
E	0.26	10	2	0.52
I	0.21	00	2	0.42
O	0.05	11011	5	0.25
U	0.07	1100	4	0.28
S	0.22	01	2	0.44
T	0.02	11010	5	0.1
Average Code Length (ACL)				**2.52**

Shannon-Fano Algorithm for Data Compression

- Create a list of probabilities or frequency counts for the given set of symbols so that the relative frequency of occurrence of each symbol is known.
- Sort the list of symbols in decreasing order of probability, the most probable ones to the left and the least probable ones to the right.
- Split the list into two parts, with the total probability of both parts being as close to each other as possible.
- Assign the value 0 to the left part and 1 to the right part.
- Repeat steps 3 and 4 for each part until all the symbols are split into individual subgroups.

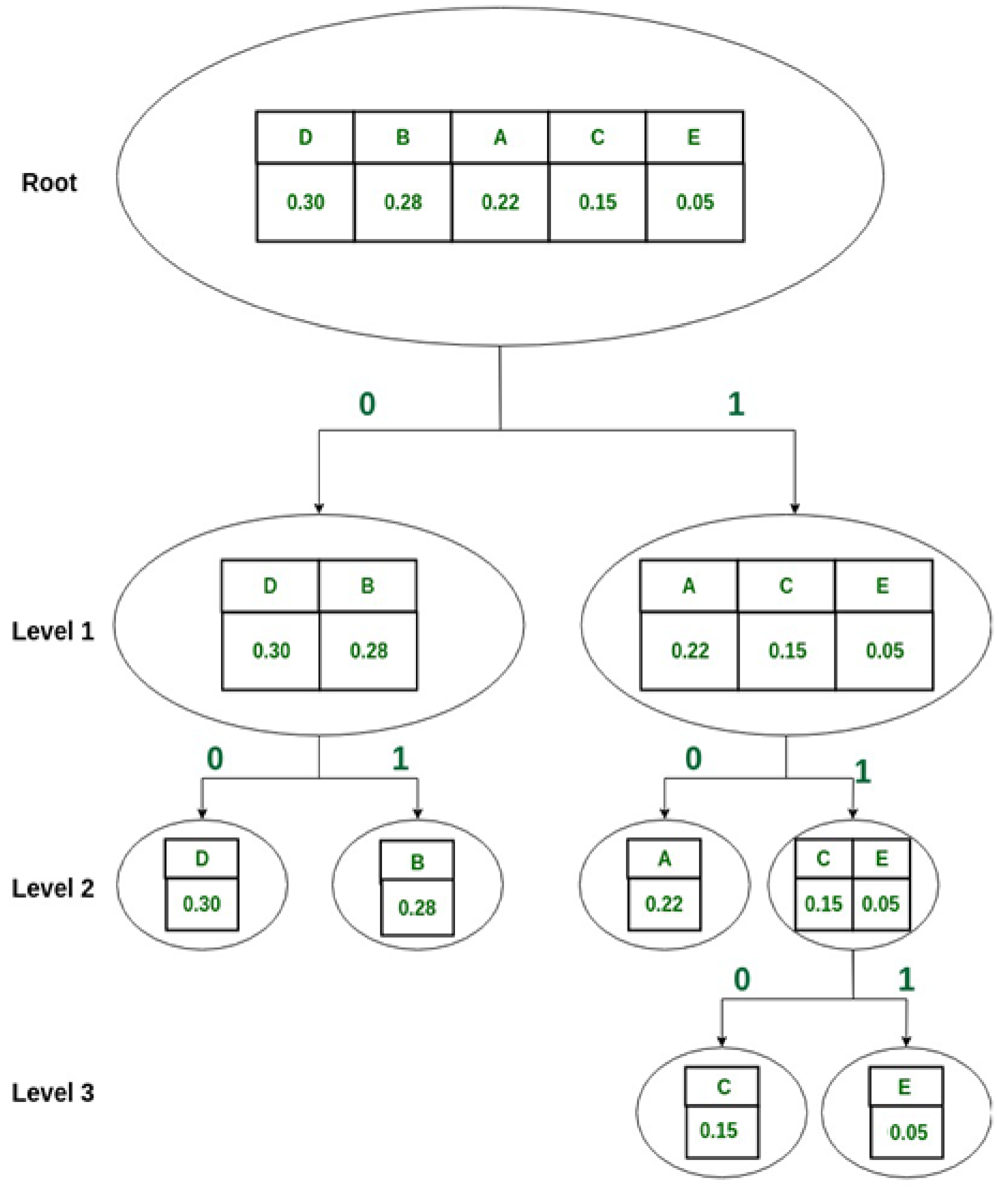
SYMBOL	A	B	C	D	E
PROBABILITY OR FRQUENCY	0.22	0.28	0.15	0.30	0.05

THE SYMBOLS AND THEIR PROBABILITY / FREQUENCY
ARE TAKEN AS INPUTS.
(In case of Frequency, the values can be any number)



SYMBOL D B A C E
PROBABILITY OR FRQUENCY 0.30 0.28 0.22 0.15 0.05
SYMBOL D B
PROBABILITY OR FRQUENCY 0.30 0.28
SYMBOL A C E
PROBABILITY OR FRQUENCY 0.22 0.15 0.05
SYMBOL D
PROBABILITY OR FRQUENCY 0.30
SYMBOL B
PROBABILITY OR FRQUENCY 0.28
SYMBOL A
PROBABILITY OR FRQUENCY 0.22
SYMBOL C E
PROBABILITY OR FRQUENCY 0.15 0.05
SYMBOL E
PROBABILITY OR FRQUENCY 0.05
SYMBOL E
PROBABILITY OR FRQUENCY 0.05

THE SYMBOLS ARE CONTINUED TO BE DIVIDED INTO TWO
TILL EACH SYMBOL BECOME SEPARATED



TREE AFTER STEP 4

SYMBOL	A	B	C	D	E
PROBABILI TY	0.22	0.28	0.15	.30	.05
SHANNON CODE:	10	01	110	00	111