Fourier transform properties:

1) Convolution property: Most powerful in DIP.

Convolution in spatial domain = Multiplication in frequency domain.

Convolution of 2 sequences $x(n)$ & $h(n)$

$$x(n) * h(n) = \sum_{k=-\infty}^{\infty} x(k)\, h(n-k)$$

For 2D array:

$$f(m,n) * g(m,n) = \sum_{a=0}^{N-1} \sum_{b=0}^{N-1} f(a,b)\, g(m-a, n-b)$$

Proof:

$$DFT\{f(m,n) * g(m,n)\} = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} \left\{ \sum_{a=0}^{N-1} \sum_{b=0}^{N-1} f(a,b)\, g(m-a, n-b) \right\} e^{-j\frac{2\pi}{N}mk}\, e^{-j\frac{2\pi}{N}nl}$$

$$= \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} \sum_{a=0}^{N-1} \sum_{b=0}^{N-1} f(a,b)\, g(m-a, n-b)$$

add and sub a, b →

$$e^{-j\frac{2\pi}{N}(m-a+a)k}$$

$$e^{-j\frac{2\pi}{N}(n-b+b)l}$$

$\therefore$

$$DFT\{f(m,n) * g(m,n)\} = F(k,l) \times G(k,l)$$

$$= \sum_{a=0}^{N-1} \sum_{b=0}^{N-1} f(a,b)$$

2) Separable property: Allows 2D transform to be computed in 2 operations on row and column of image

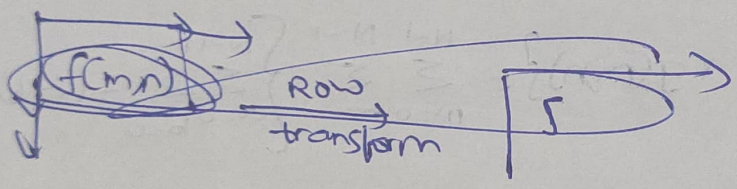
Proof:

$$F(k,l) = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} f(m,n)\, e^{-j\frac{2\pi}{N}mk}\, e^{-j\frac{2\pi}{N}nl}$$

$$= \sum_{m=0}^{N-1} \left[\sum_{n=0}^{N-1} f(m,n)\, e^{-j\frac{2\pi}{N}nl} \right] e^{-j\frac{2\pi}{N}mk}$$

$$= \sum_{m=0}^{N-1} f(m,l)\, e^{-j\frac{2\pi}{N}mk}$$

$$= F(k,l)$$



$f(m,n) \xrightarrow{\text{Row transform}} f(m,l) \xrightarrow{\text{Column transform}} f(k,l)$

3) Conjugate:

If DFT of $f(m,n) = F(k,l)$

then $DFT[f^*(m,n)] = F^*(-k,-l)$.

Proof: $F(k,l) = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} f(m,n)\, e^{-j\frac{2\pi}{N}mk}\, e^{-j\frac{2\pi}{N}nl}$

Apply conjugate :

$$F^*(k,l) = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} f(m,n) e^{j\frac{2\pi}{N}mk} e^{j\frac{2\pi}{N}nl}$$

Apply reverse

$$F^*(-k,-l) = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} f(m,n) e^{j\frac{2\pi}{N}m(-k)} e^{j\frac{2\pi}{N}n(-l)}$$

Q] Find 2D DFT of 4x4 image.

$$f(m,n) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Sol] 2D DFT = $F[k,l]$ = kernel * $f(m,n)$ * (kernel)^T

When N=4, kernel =

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

$$f[k,l] = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & -1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} * \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & -1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$= \begin{bmatrix} 16 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Q] Find inverse 2D DFT for above Question image

Sol]

$$f(m,n) = \frac{1}{N^2} \times \text{kernel} * F(k,l) * (\text{kernel})$$

~~N=4~~
$N \times N = 4 \times 4$

$$= \frac{1}{4^2} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \times \begin{bmatrix} 16 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} = \text{original image}$$

* (Check how to do matrix multiplication)

Huffman

1] RGB to HSI

RGB = (125, 200, 100)

Sol] $R = \frac{125}{255} = 0.49$ $G = \frac{200}{255} = 0.78$

$B = \frac{100}{255} = 0.39$

$$H = \cos^{-1}\left[\frac{\frac{1}{2}\left[(R-G)+(R-B)\right]}{\left[(R-G)^2+(R-B)(G-B)\right]^{1/2}}\right]$$

$$S = 1 - \left[\frac{3}{R+G+B}\min(R,G,B)\right]$$

$$I = \frac{1}{3}(R+G+B)$$

HSI to RGB

(1) $0 \le H < 120°$

$$B = I(1-S)$$

$$R = I\left[1 + \frac{S\cos H}{\cos(60-H)}\right]$$

$$G = 3I - (R+B)$$

(2) $120 < H < 240$: $H = H - 120$

$R = B$ in (1)

$G = R$ in (1)

$B = G$ in (1)

(3) $240 < 360 \Rightarrow H$ $\quad H = H - 240$

$G = B$ in (1)

$B = R$ in (1)

$R = G$ in (1)

Gaussian filtering: It is a low pass filter

It uses average smoothing

It uses Gaussian kernel

standard deviation & kernel size is used to smoothen image.

Reduces noise, blur

used in CV, Graphics, ML

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\left(x^2/2\sigma^2\right)}$$

$$= \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}$$

Gaussian noise: known as additive white Gaussian noise.

It is identically distributed.

It is common in sensor, communication channel.

It adds randomness to signal

$$P(z) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\left(\frac{z-\mu}{\sqrt{2}\sigma}\right)^2}$$

Quiz 3:

1) Adjusts pixels based on it's position.
Modifies only neighborhood.
Reduces noise
Used in photography, medical imaging

2) 2D DFT: Operation converts spatial to frequency domain

Convolution:

$$x(n) * h(n) = \sum_{k=-\infty}^{\infty} x(k)\, h(n-k)$$

3) 256×256 = no. of pixels

Total bits = no of pixel × bit per pixel
$= 256 \times 256 \times 8$

4) $P(0) = \frac{5}{16}$ $P(1) = \frac{4}{16}$ $P(2) = \frac{5}{16}$ $P(3) = \frac{2}{16}$

Entropy $= -P_i \log_2 P_i$

$$= \frac{-5}{16}\log_2\frac{5}{16} - \frac{4}{16}\log_2\frac{4}{16} - \frac{5}{16}\log_2\frac{5}{16} - \frac{2}{16}\log_2\frac{2}{16}$$

5)

Image Restoration	Enhancement
1) Receive original image from degraded	Make image better
2) Remove blur	Enhance visual quality
3) Medical imaging	photography

6) Color Image smoothy reduces noice & improves quality of color image by averagy neighborhood.

Neighborhood averagy:

1) Replace each pixel by avg value of surrounding

$$F(x,y) = \frac{1}{m \times n} \sum_{k=a}^{a} \sum_{l=-b}^{b} F(i+k, j+l)$$

$$F = \{R, G, B\}$$

7) Orthogonality:

$$\frac{1}{N^2} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} a_{k,l}(m,n) * a_{k',l'}(m,n)$$

$$= \delta(k-k', l-l')$$

Sec Committee Huffman Coding

source. It is important to note that Huffman coding is optimal at the block level and it is not optimal at the symbol-level encoding.

6. *For the image shown below, compute the degree of compression that can be achieved using (a) Huffman coding of pixel values, (b) run-length coding, assuming 2 bits to represent the pixel value and 2 bits to represent the run length.*

$$\begin{bmatrix} 3 & 3 & 3 & 2 \\ 2 & 3 & 3 & 3 \\ 3 & 2 & 2 & 2 \\ 2 & 1 & 1 & 0 \end{bmatrix}$$

Solution From the given image, it is clear that the maximum value is 3. To encode this maximum value, a minimum of two bits is required. Totally, 16 pixels are present in the given matrix. This implies that a minimum of $16 \times 2 = 32$ bits are required to code this image.

(a) Huffman coding of pixels

Step 1 *Probability of occurence of pixel values*

The first step in Huffman coding is to determine the probability of occurrence of each pixel value. From the given image, we get the following:

Probability of occurence of pixel value 0 is $p(0) = \frac{1}{16}$

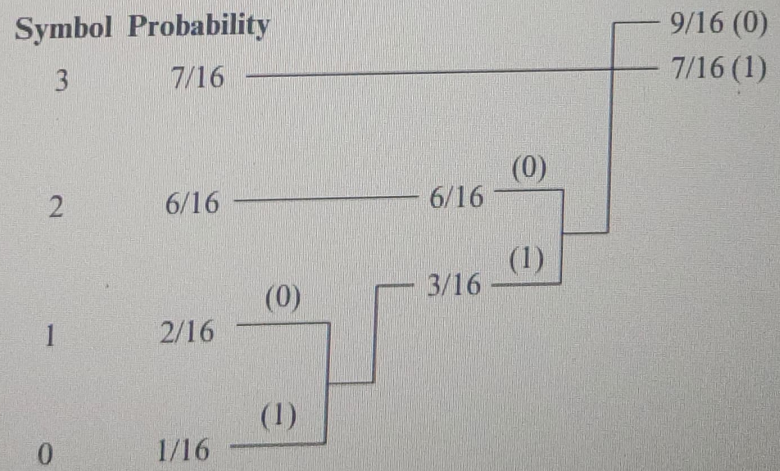
Probability of occurence of pixel value 1 is $p(1) = \frac{2}{16}$

Probability of occurence of pixel value 2 is $p(2) = \frac{6}{16}$

Probability of occurence of pixel value 3 is $p(3) = \frac{7}{16}$

Step 2 *Formation of Huffman tree*

In Huffman tree formation, the pixels are arranged in descending order.



From the Huffman tree, the code for the symbols are given below:

Code for the symbol 3 is 1.

Code for the symbol 2 is 00

Code for the symbol 1 is 010

Code for the symbol 0 is 011

The mean bits per pixel using Huffman coding is given by $\sum_k p_k l_k$

For this problem, the mean bits per pixel is $\frac{7}{16}\times 1+\frac{6}{16}\times 2+\frac{2}{16}\times 3+\frac{1}{16}\times 3=\frac{28}{16}=1.75$

Originally, we are in need of two bits to represent the pixel, but through Huffman coding we are in need of only 1.75 bits to represent the pixel. Hence the compression ratio achieved through Huffman coding is given by

$$\text{compression ratio} = \frac{2}{1.75}=1.14$$

$$\text{compression ratio} = \frac{2}{1.75} = 1.14$$

(b) Run-length coding of pixels

By scanning the image from left to right and from top to bottom, we can write the run length as (3, 3) (2, 2) (3, 4) (2, 4) (1, 2) (0, 1). The first element in the bracket represents the pixel value and the second element represents the number of occurence of the pixel value. We use two bits to represent the pixel value and two bits to represent the number of occurences of the pixel value.

This implies the following:

To code (3, 3) we need $2 + 2 = 4$ bits

To code (2, 2) we need $2 + 2 = 4$ bits

To code (3, 4) we need $2 + 2 = 4$ bits

To code (2, 4) we need $2 + 2 = 4$ bits

To code (1, 2) we need $2 + 2 = 4$ bits

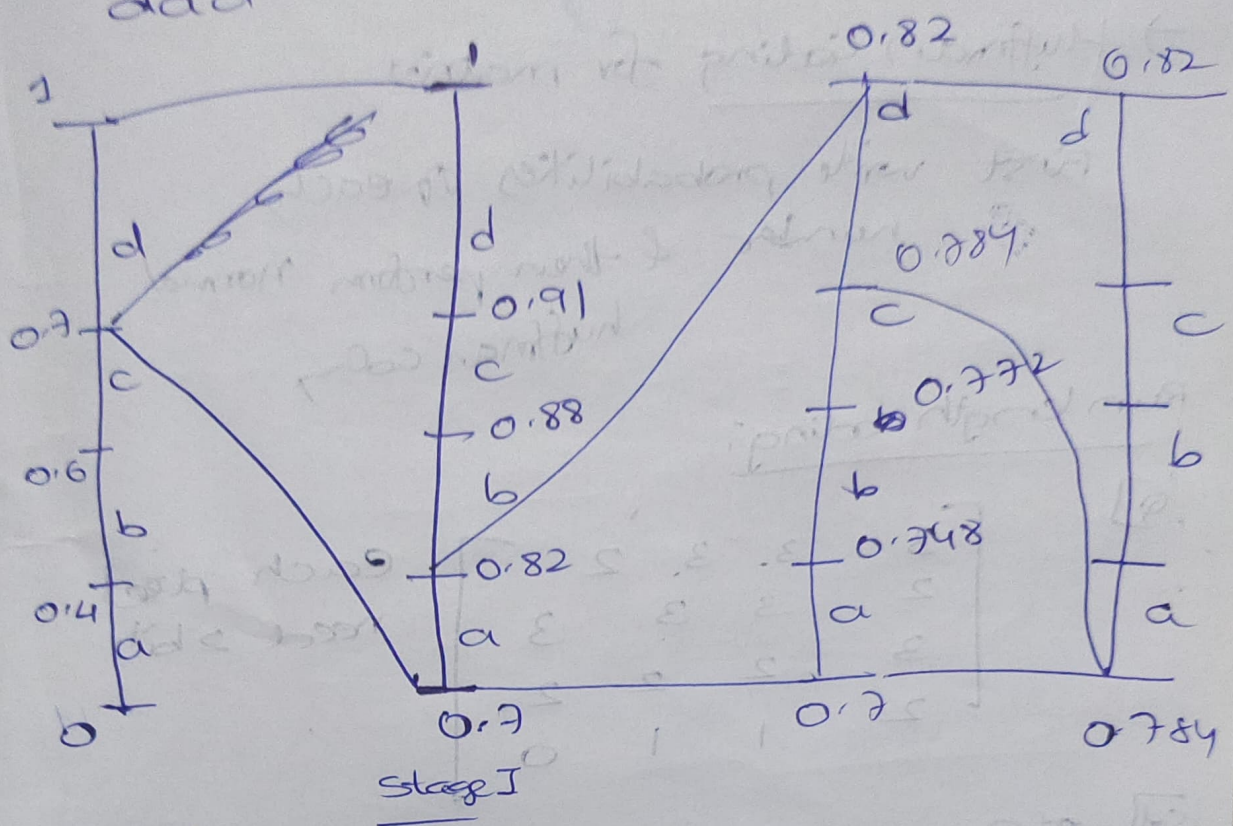
To code (0, 1) we need $2 + 2 = 4$ bits

Totally, we are in need of 24 bits. Originally, we were in need of 32 bits, and hence the compression that can be achieved through run-length coding is 32/24=1.33

9] symbol:

a	b	c	d
0.4	0.2	0.1	0.3

dad



Stage I:

Range of symbol

= lower limit + diff (probability of symbol)

diff upper - lower limit = 1 - 0.7 = 0.3

For a: 0.7 + 0.3 (0.4) = 0.82

For b: 0.82 + 0.3 (0.2) = 0.88

For c: 0.88 + 0.3 (0.1) = 0.91

Stage 2: diff 0.82 - 0.7 = 0.12

For a: 0.7 + 0.12 (0.4) = 0.748

For b: 0.748 + 0.12 (0.2) = 0.772

For c: 0.772 + 0.12 (0.1) = 0.784

$$\text{tag} = \frac{0.784 + 0.82}{2} = 0.802$$

Image restoration is a crucial task in image processing aimed at enhancing the quality of an image by removing or reducing degradation caused by various factors such as noise, blur, and other distortions. The purpose of image restoration is to recover the original, undistorted image as closely as possible.

The model of image degradation and restoration process typically involves the following steps:

1. **Image Degradation**: This step represents the process by which an image gets degraded due to various factors such as motion blur, defocus blur, sensor noise, etc. Image degradation can be modeled as a combination of degradation functions such as blurring, noise addition, and other distortions.

2. **Degradation Model**: The degradation model describes the relationship between the original image and the degraded image. It typically consists of two main components:

   a. **Blur Model**: This component models the blurring effect on the image caused by factors like motion or defocus. It can be represented mathematically using convolution with a point spread function (PSF) or impulse response.

   b. **Noise Model**: This component models the noise added to the image during the acquisition or transmission process. Common types of noise include Gaussian noise, salt-and-pepper noise, etc.

etc.

3. **Restoration Process:** The restoration process involves recovering the original image from the degraded image. It typically consists of two main steps:

   a. **Estimation:** In this step, an estimate of the original image is computed using various techniques such as inverse filtering, deconvolution, or model-based approaches.

   b. **Post-processing:** After obtaining the initial estimate, post-processing techniques are applied to further enhance the quality of the restored image. This may involve techniques such as regularization, noise filtering, or sharpening.

A Model of Image Degradation/Restoration Process

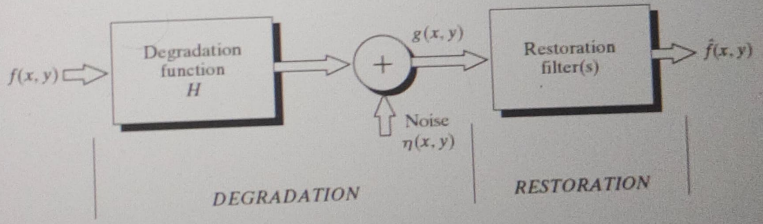


FIGURE 5.1 A model of the image degradation/restoration process.

$$g(x, y) = h(x, y) \star f(x, y) + \eta(x, y)$$

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