Image Transforms

Image Transform is basically a mathematical tool, which allows us to convert image data from spatial domain to frequency domain.

Need for Image Transform

Most of image processing applications are real-time applications. Biometric applications such as finger-print based attendance, airport baggage screening system and radar image monitoring are image based real time applications. These applications require faster algorithms for speed and simplicity of operations. The frequency domain holds the key to both these requirements.

Image in Spatial Domain $\xleftrightarrow{\text{Transform}}$ Image in Frequency Domain

$f(x,y) = [\;]$ $\xleftrightarrow{\text{Transform}}$ $F(u,v) = [\;]$

$f(x,y)$ → Spatial Co-ordinates; [ ] → pixel

$F(u,v)$ → Frequency Co-ordinates; [ ] → Co-efficient

→ Each location in spatial domain is represented as pixel.

→ Each location in frequency domain is called Co-efficient.

Types of Image Transforms

Image transforms can be classified based on the nature of the basis function (Kernel) as

① Orthogonal Sinusoidal basis function

→ Discrete Fourier Transform (DFT)

→ Discrete Cosine Transform (DCT)

→ Discrete Sine Transform (DST)

② Non-Sinusoidal Orthogonal basis function

→ Haar Transform

→ Walsh Transform

→ Hadamard Transform

→ Slant Transform

③ Directional Transformation:

→ Hough Transform

→ Random Transform

→ Ridgelet Transform

→ Contourlet Transform

④ Statistical Transform

→ KL Transform

→ Singular Value Decomposition (SVD) Transform

Kernel : It is a function which transforms the co-ordinates from one domain to another domain.

Introduction to Fourier Transform

Let us assume a continuous function $f(x)$ and it's fourier transform is denoted by $F(u)$,

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

where 'u' represents the spatial frequency

The inverse Fourier Transform is expressed as,

$$f(x) = \int_{-\infty}^{\infty} F(u)\, e^{j2\pi ux}\, du$$

Hence, the Fourier Transform for the 2D image $f(x,y)$ is given by

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

and the inverse transform is given by

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

Discrete Fourier Transform

The DFT of a one-dimensional signal $f(x)$ is

$$F(u) = \sum_{x=0}^{M-1} f(x)\, e^{-j2\pi ux/M}$$

Similarly, the inverse DFT is given as

$$f(x) = \sum_{u=0}^{M-1} F(u)\, e^{j2\pi ux/M}$$

If $F(u) = R(u) + j\,I(u)$, where $R(u)$ is the real term and $I(u)$ is the imaginary term it's magnitude can be expressed as

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

phase angle $\phi(u)$ is defined as

$$\phi(u) = \tan^{-1} \frac{I(u)}{R(u)}$$ (∵ For recovering the Original Image)

The Power Spectrum is given by $P(u) = |F(u)|^2 = R^2(u) + I^2(u)$

If $f(x,y)$ is a two dimensional image of two dimensions M & N on a rectangular grid, then 2-D DFT is given by

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

for $u = 0, 1, \ldots M-1$ & $v = 0, 1, \ldots N-1$

and the inverse DFT (IDFT) is

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

When images are sampled in a square array, M = N

then

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

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

The magnitude & phase spectrum of 2D-images are

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

$$\& \quad \phi(u,v) = \tan^{-1} \frac{I(u,v)}{R(u,v)}$$

The Power Spectrum is given as $P(u,v) = R^2(u,v) + I^2(u,v)$

The DFT can be formulated for matrix operations of the form,

The forward transform is

$$F(u,v) = \text{Kernel} \times f(x,y) \times \text{Kernel}^T$$

The inverse 2D DFT is

$$f(x,y) = \frac{1}{N^2} \times \text{Kernel} \times F(u,v) \times \text{Kernel}^T$$

where Kernel (W) is formulated for N=4

$$W = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \qquad W = \begin{bmatrix} \omega^0 & \omega^0 & \omega^0 & \omega^0 \\ \omega^0 & \omega^1 & \omega^2 & \omega^3 \\ \omega^0 & \omega^2 & \omega^4 & \omega^6 \\ \omega^0 & \omega^3 & \omega^6 & \omega^9 \end{bmatrix}$$

$$\omega = e^{-\frac{j2\pi}{N}}$$

Properties of Fourier Transform

(i) **Separability** : This property allows a 2D transform to be computed in two steps by successive 1D operations on rows and columns of our image.

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

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

$$= \sum_{y=0}^{N-1} F(u,y)\, e^{-j2\pi vy/N}$$

$$= F(u,v) \quad //$$

② Translation

If $f(x,y) \xleftrightarrow{DFT} F(u,v)$ then

$$~~f(x-x_0,~~ f(x-x_0, y-y_0) \xleftrightarrow{DFT} F(u,v)\, e^{-j2\pi\left(\frac{ux_0}{M}+\frac{vy_0}{N}\right)}$$

and $$f(x,y)\, e^{j2\pi\left(\frac{ux_0}{M}+\frac{vy_0}{N}\right)} \xleftrightarrow{DFT} F(u-u_0, v-v_0)$$

Proof:

We know that

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

then

$$DFT\left[f(x-x_0, y-y_0)\right] = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x-x_0, y-y_0)\, e^{\frac{-j2\pi ux}{M}} \cdot e^{\frac{-j2\pi vy}{N}}$$

$$= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x-x_0, y-y_0)\, e^{\frac{-j2\pi u(x-x_0+x_0)}{M}} \cdot e^{-\frac{j2\pi v(y-y_0+y_0)}{N}}$$

$$= \left[\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x-x_0, y-y_0)\, e^{\frac{-j2\pi u(x-x_0)}{M}} \cdot e^{\frac{-j2\pi v(y-y_0)}{N}}\right] e^{\frac{-j2\pi ux_0}{M}} \cdot e^{\frac{-j2\pi vy_0}{N}}$$

$$= F(u,v)\, e^{-j2\pi\left(\frac{ux_0}{M}+\frac{vy_0}{N}\right)} \quad //$$

③ Periodicity

The Fourier transform and it's inverse are periodic. If N is the period,

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

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

Scanned by CamScanner

④ Distribution

The Fourier transform is distributive over addition.

It is represented as,

$$DFT\left[f_1(x,y) + f_2(x,y)\right] = DFT\left[f_1(x,y)\right] + DFT\left[f_2(x,y)\right]$$

⑤ Scaling

Any scaling in the spatial domain is reflected in frequency domain, i.e.

If $f(x,y) \xleftrightarrow{DFT} F(u,v)$ then

$$f(ax, by) \xleftrightarrow{DFT} \frac{1}{ab} F\left(\frac{u}{a}, \frac{v}{b}\right)$$

⑥ Average Value

The Average value of Discrete Fourier Transform is defined as

$$F(0,0) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)$$

$\therefore$ $F(0,0)$ is called DC Component (or) Average Value

⑦ Convolution

If $f(x,y) \xleftrightarrow{DFT} F(u,v)$ then

$$f(x,y) * g(x,y) \xleftrightarrow{DFT} F(u,v) \cdot G(u,v)$$

⑧ Parseval's Theorem

The sum of the squares of the Pixels of an image gives the energy of the image. This can be calculated by taking the square of the magnitude of the Fourier coefficients. This is known as Parseval's Theorem.

$$P(u,v) = \sum |f(x,y)|^2 = \sum |F(u,v)|^2$$

Ex: what is the dc component of the following image?

$$f(x,y) = \begin{bmatrix} 1 & 3 & 4 \\ 5 & 6 & 7 \\ 8 & 9 & 11 \end{bmatrix}$$

Sol: The DC Component $F(0,0)$ is the average of the pixel values of the given image.

$$F(0,0) = \frac{1}{3\times 3}(1+3+4+5+6+7+8+9+11)$$

$$= \frac{1}{9}(54) = 6$$

Ex: Apply DFT to the following image and prove that DFT works or not?

$$f(x,y) = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Sol: The forward transform,

$$F(u,v) = \text{Kernel} \times f(x,y) \times \text{Kernel}^T$$

$$F(u,v) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \times \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & j \end{bmatrix}$$

$$F(u,v) = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$F(u,v) = \begin{bmatrix} 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \end{bmatrix}$$

The inverse 2D DFT is

$$f(x,y) = \frac{1}{N^2} \text{Kernel} \times F(u,v) \times \text{Kernel}^T$$

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

$$= \frac{1}{16} \begin{bmatrix} 8 & 8 & 8 & 8 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$f(x,y) = \frac{1}{16} \begin{bmatrix} 32 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Hence the original image is recovered without any loss of information.

Discrete Cosine Transform (DCT)

Discrete Cosine Transform is widely used for image compression because of it's high energy packing capabilities.

DCT of a 1D sequence of length N can be given as

$$C(u) = W(u) \times \sum_{x=0}^{N-1} f(x) \cos\left(\frac{\pi(2x+1)u}{2N}\right)$$

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

where

$$W(u) = \begin{cases} \frac{1}{\sqrt{N}} & \text{for } u = 0 \\ \sqrt{\frac{2}{N}} & \text{for } u \neq 0 \end{cases}$$

When $u = 0$

$$\Rightarrow C(u) = C(0) = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} f(x)$$, which is called as DC Component

For other values of u, the components obtained are called AC Components (co-efficients).

The inverse DCT is given as

$$f(x) = \sum_{u=0}^{N-1} W(u) \times C(u) \cos\left(\frac{\pi(2x+1)u}{2N}\right)$$

The two-dimensional DCT is defined as,

$$C(u,v) = W(u)\, W(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \cdot \cos\left(\frac{\pi(2x+1)u}{2N}\right) \cdot \cos\left(\frac{\pi(2y+1)v}{2N}\right)$$

for $u, v = 0, 1, 2, \ldots N-1$. $W(u)$ and $W(v)$ can be calculated by using the one-dimensional.

The inverse DCT is defined as,

$$f(x,y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} W(u)\,W(v)\,C(u,v) \cos\left(\frac{\pi(2x+1)u}{2N}\right) \cdot \cos\left(\frac{\pi(2y+1)v}{2N}\right)$$

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

Properties of DCT

(1) The Cosine Transform is real and orthogonal.

i.e $C = C^*$ & $C \cdot C^T = I$

(2) The DCT is helpful in removing the ~~data~~ redundant data from an image.

(3) The energy compaction efficiency of the DCT is excellent for highly correlated data.

(4) The DCT is seperable since the transform for rows and columns are identical.

(5) The DCT can be called as a symmetric transformation.

Prob: Find the co-efficients of given image using DCT?

$$f(x,y) = \begin{bmatrix} 2 & 4 \\ 3 & 1 \end{bmatrix}$$

Sol: We know that, $N = 2$

$$C(u,v) = W(u) \cdot W(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \cdot \cos\left(\frac{\pi(2x+1)u}{2N}\right) \cdot \cos\left(\frac{\pi(2y+1)v}{2N}\right)$$

$$C(0,0) = W(0)\,W(0) \sum_{x=0}^{1} \sum_{y=0}^{1} f(x,y) \cdot 1$$

$$= \frac{1}{\sqrt{2}} \cdot \frac{1}{\sqrt{2}} \left[f(0,0) + f(0,1) + f(1,0) + f(1,1) \right]$$

$$= \frac{1}{2} \left[2+4+3+1 \right] \Rightarrow C(0,0) = 5$$

$$C(0,1) = W(0)\,W(1) \sum_{x=0}^{1} \sum_{y=0}^{1} f(x,y) \cdot 1 \cdot \cos\left(\frac{(2y+1)\pi}{4} \right)$$

$$= \frac{1}{\sqrt{2}} \sqrt{\frac{2}{2}} \sum_{x=0}^{1} \sum_{y=0}^{1} f(x,y) \cos \cdot \frac{(2y+1)\pi}{4}$$

$$= \frac{1}{\sqrt{2}} \left[f(0,0) \cos\frac{\pi}{4} + f(0,1) \cos\frac{3\pi}{4} + f(1,0) \cos\frac{\pi}{4} + f(1,1) \cos\frac{3\pi}{4} \right]$$

$$= \frac{1}{\sqrt{2}} \left[2\left(\frac{1}{\sqrt{2}}\right) + 4\left(\frac{-1}{\sqrt{2}}\right) + 3\left(\frac{1}{\sqrt{2}}\right) + 1\left(\frac{-1}{\sqrt{2}}\right) \right] = 0$$

$$C(1,0) = W(1)\,W(0) \sum_{x=0}^{1} \sum_{y=0}^{1} f(x,y) \cdot 1 \cdot \cos\left(\frac{(2x+1)\pi}{4} \right)$$

$$= \frac{1}{\sqrt{2}} \left[f(0,0) \cos\frac{\pi}{4} + f(0,1) \cos\frac{\pi}{4} + f(1,0) \cos\frac{3\pi}{4} + f(1,1) \cos\frac{3\pi}{4} \right]$$

$$= \frac{1}{\sqrt{2}} \left[2 \times \frac{1}{\sqrt{2}} + 4 \times \frac{1}{\sqrt{2}} + 3 \times \left(\frac{-1}{\sqrt{2}}\right) + 1\left(\frac{-1}{\sqrt{2}}\right) \right] = 1$$

$$C(1,1) = W(1)\,W(1) \sum_{x=0}^{1} \sum_{y=0}^{1} f(x,y) \cdot \cos\left(\frac{(2x+1)\pi}{4} \right) \cdot \cos\left(\frac{(2y+1)\pi}{4} \right)$$

$$= \sqrt{\frac{2}{2}} \sqrt{\frac{2}{2}} \left[f(0,0) \cos\frac{\pi}{4} \cos\frac{\pi}{4} + f(0,1) \cos\frac{\pi}{4} \cos\frac{3\pi}{4} + f(1,0) \cos\frac{3\pi}{4} \cos\frac{\pi}{4} + f(1,1) \cos\frac{3\pi}{4} \cos\frac{3\pi}{4} \right]$$

$$= 2 \times \frac{1}{2} + 4 \times \frac{-1}{2} + 3 \times \frac{-1}{2} + 1 \times \frac{-1}{2} = -2$$

$$\therefore \; C(u,v) = \begin{bmatrix} 5 & 0 \\ 1 & -2 \end{bmatrix}$$

DISCRETE SINE TRANSFORM

The sine transform pair of one-dimensional image $f(x)$ is defined as,

$$S(u) = \sqrt{\frac{2}{M+1}} \sum_{x=0}^{M-1} f(x) \sin\left(\frac{(x+1)(u+1)\pi}{M+1}\right)$$

& The inverse DST is

$$f(x) = \sqrt{\frac{2}{M+1}} \sum_{u=0}^{M-1} S(u) \sin\left(\frac{(x+1)(u+1)\pi}{M+1}\right)$$

Similarly for two-dimensional image $f(x,y)$, the DST is defined as

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

and The inverse DST is defined as,

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

Fast Fourier Transform

The 2D-DFT can be calculated by applying the 1D DFT for each column of $f(x,y)$ and then performing a 1D DFT for each row on the resulting complex data. However the DFT algorithm require summations and additions of order $(MN)^2$. Even super Computers failed to implement these equations and their related calculations. To solve this problem a faster version of Fourier Transform was introduced.

The FFT algorithm is developed by using a successive doubling method which reduces calculations to the order $MN \log_2 MN$. The FFT algorithm is designed as follows,

The Fourier Transform can be represented as,

$$F(u) = \sum_{x=0}^{M-1} f(x) e^{-\frac{j2\pi ux}{M}}$$

where $u = 0, 1, \ldots M-1$

The above eq'n can be written as

$$F(u) = \sum_{x=0}^{M-1} f(x) W_M^{ux}$$

where $W_M = e^{-\frac{j2\pi}{M}}$

Let us consider the number of points M is equal to a power of 2. So M can be expressed as $M = 2K$ where K is a positive integer.

By substituting the value of M in above eq'n

$$F(u) = \sum_{x=0}^{2K-1} f(x) W_{2K}^{ux}$$

We can rewrite above equation as

$$F(u) = \sum_{x=0}^{K-1} f(2x) W_{2K}^{u(2x)} + \sum_{x=0}^{K-1} f(2x+1) W_{2K}^{u(2x+1)}$$

$$F(u) = \sum_{x=0}^{K-1} f(2x) W_{2K}^{u(2x)} + \sum_{x=0}^{K-1} f(2x+1) W_{2K}^{2ux} \cdot W_{2K}^{u}$$

But $W_{2K}^{2ux} = W_K^{ux}$

$$\Rightarrow F(u) = \sum_{x=0}^{K-1} f(2x) W_K^{ux} + \sum_{x=0}^{K-1} f(2x+1) W_K^{ux} \cdot W_{2K}^{u}$$

$$\Rightarrow F(u) = F_e(u) + F_o(u) W_{2K}^{u}$$

where the functions $F_e(u)$ and $F_o(u)$ are even and odd functions.

Due to the periodicity properties,

$$W_K^{u+K} = W_K^{u} \; \& \; W_{2K}^{u+K} = -W_{2K}^{u}$$

Then,

$$F(u+K) = F_e(u) - F_o(u) W_{2K}^{u}$$

The FFT algorithm is based on the principle called " Divide & Conquer Method". As per this principle, if the problem is big, it is divide into sub-problems and then solved, the individual solutions are combined to yield the final solution.

—

WALSH Transform

The Walsh Transform for 1-D Dimensional image $f(x)$ is defined as,

$$w(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x) \prod_{i=0}^{n-1} (-1)^{[b_i(x)\, b_{n-1-i}(u)]}$$

where

$$g(x,u) = \frac{1}{N} \prod_{i=0}^{n-1} (-1)^{[b_i(x)\, b_{n-1-i}(u)]}$$ is called Kernel

of the Walsh Transform.

where $b_i(x)$ represents the i^{th} bit of the binary representation of 'x'.

The inverse 1-D Walsh Transform is

$$f(x) = \frac{1}{N} \sum_{u=0}^{N-1} w(u) \prod_{i=0}^{n-1} (-1)^{[b_i(x)\, b_{n-1-i}(u)]}$$

Similarly for, 2-D Dimensional images,

$$w(u,v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \prod_{i=0}^{n-1} (-1)^{[b_i(x)\, b_{n-1-i}(u) + b_i(y)\, b_{n-1-i}(v)]}$$

and the kernel is defined as,

$$g(x,y,u,v) = \frac{1}{N} \prod_{i=0}^{n-1} (-1)^{[b_i(x)\, b_{n-1-i}(u) + b_i(y)\, b_{n-1-i}(v)]}$$

which is seperable and Symmetric.

The inverse 2D-Walsh Transform

$$f(x,y) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} w(u,v) (-1)^{[b_i(x)\, b_{n-1-i}(u) + b_i(y)\, b_{n-1-i}(v)]}$$

Ex:

Walsh Transform for N = 2

$N = 2 = 2^n \Rightarrow n = 1$

From the definition of kernel

$$g(x,u) = \frac{1}{N} \prod_{i=0}^{n-1} (-1)^{b_i(x)\, b_{n-1-i}(u)}$$

$$= \frac{1}{4} \prod_{i=0}^{1} (-1)^{b_i(x)\, b_{1-1-i}(u)}$$

$$g(x,u) = \frac{1}{2} (-1)^{b_0(x)\, b_0(u)}.$$

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

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

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

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

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

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

If the order of N is high, it is difficult to calculate. So we go for the short cut method,

① Write the binary representation of x

② Write the binary representation of u and consider in reverse order

③ Check for the number of overlaps of '1' between u and x. If it is 0 (or) even, the sign is positive, Else negative

Ex : for N=4&8 the Kernel for Walsh Transform

$$W_{4\times4} = \frac{1}{4}\begin{bmatrix} + & + & + & + \\ + & + & - & - \\ + & - & + & - \\ + & - & - & + \end{bmatrix}$$

$$W_{8\times8} = \frac{1}{8}\begin{bmatrix} + & + & + & + & + & + & + & + \\ + & + & + & + & - & - & - & - \\ + & + & - & - & + & + & - & - \\ + & + & - & - & - & - & + & + \\ + & - & + & - & + & - & + & - \\ + & - & + & - & - & + & - & + \\ + & - & - & + & + & - & - & + \\ + & - & - & + & - & + & + & - \end{bmatrix}$$

The Advantage of walsh Transform is fourier transform is based on trigonometric terms where as walsh transform consists of a series of expansions of basis function whose values are only +1 or -1. These functions can be implemented more efficiently in a digital environment than exponential basis function of the fourier transform.

Hadamard Transform

The basic function of the Hadamard transform is also +1 and −1, like walsh transform. It is defined for 1D image is

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

where, $g(x,u) = \frac{1}{N} (-1)^{\sum_{i=0}^{n-1} b_i(x) b_i(u)}$ is kernel

The inverse Hadamard Transform, f(x) is given as,

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

The two-dimensional Hadamard Transform is defined as,

$$H(u,v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) (-1)^{\sum_{i=0}^{n-1} [b_i(x) b_i(u) + b_i(y) b_i(v)]}$$

where Kernel is

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

The inverse Hadamard transform is

$$f(x,y) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} H(u,v) (-1)^{\sum_{i=0}^{n-1} [b_i(x) . b_i(u) + b_i(y) . b_i(v)]}$$

Ex: For N = 2 Hadamard Kernel for 1D image is

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

For $N = 2 = 2^n \Rightarrow n = 1$

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

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

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

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

$$g(1,1) = \frac{1}{2}(-1)^1 = -1/2$$

$$H_2 = \frac{1}{2}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Similarly, Hadmard Transform

for N = 4

$$H_4 = \frac{1}{4}\begin{bmatrix} H_2 & H_2 \\ H_2 & -H_2 \end{bmatrix}$$

for N = 8

$$H_8 = \frac{1}{8}\begin{bmatrix} H_4 & H_4 \\ H_4 & -H_4 \end{bmatrix}$$

//

Ex: Prove that Hadamard Transform works for the following image?

$$f = \begin{bmatrix} 2 & 2 \\ 2 & 1 \end{bmatrix}$$

Sol: The Forward Transform $= \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} 2 & 2 \\ 2 & 1 \end{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$

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

$$= \frac{1}{2}\begin{bmatrix} 7 & 1 \\ 1 & -1 \end{bmatrix}$$

The Inverse transformation is given as

$$= \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\frac{1}{2}\begin{bmatrix} 7 & 1 \\ 1 & -1 \end{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$= \frac{1}{4}\begin{bmatrix} 8 & 0 \\ 6 & 2 \end{bmatrix}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$= \frac{1}{4}\begin{bmatrix} 8 & 8 \\ 8 & 4 \end{bmatrix} = \begin{bmatrix} 2 & 2 \\ 2 & 1 \end{bmatrix} //$$

HAAR Transform

The Haar Transform consists of elements +1, −1 or 0. The Kernel of Haar Transform is generated by using following Procedure.

1. Find the order N. Let $n = \log_2 N$
2. Determine P and q
   - (a) P ranges from 0 to $n-1$
   - (b) If $P=0$ then $q = 0$ (or) 1, Else $1 \le q \le 2^P$

(3) Determine the value of k

$$k = 2^p + q - 1$$

(4) Determine $z \Rightarrow \left[\frac{0}{N}, \frac{1}{N}, \frac{2}{N}, \cdots \frac{N-1}{N}\right]$

(5) If $k=0$ then $h_k(z) = h_0(z) = \frac{1}{\sqrt{N}}$ for $z \in (0,1)$

Else,

$$h_k(z) = h_{pq}(z) = \frac{1}{\sqrt{N}} \cdot \begin{cases} 2^{p/2} & \text{if } \frac{q-1}{2^p} \leq z \leq \frac{q-1/2}{2^p} \\ -2^{p/2} & \text{if } \frac{q-1/2}{2^p} \leq z \leq \frac{q}{2^p} \\ 0 & \text{otherwise.} \end{cases}$$

Ex: Generate Haar basis for $N=2$

(1) $N = 2 \Rightarrow n = \log_2 2 = 1$

(2) Then value of $p = 0$

So $q = 0$ (or) 1

(3) Determine $k = 2^p + q - 1$

p	q	k
0	0	0
0	1	1

(4) $z = \left[\frac{0}{2}, \frac{1}{2}\right]$

(5) $p = 0$

$q=0$ $\quad$ $q=1$

Case ① If $k=0$

$n=0$ & $n=1$.

$$h_k(z) = \frac{1}{\sqrt{2}}$$

Case ② If $k=1$; $p=0$ & $q=1$

Condition ① $0 \leq z \leq 1/2$

② $1/2 \leq z < 1$

③ otherwise

For $z=0$ condition ① Satisfied

$$h(z) = \frac{1}{\sqrt{2}} 2^{0/2} = \frac{1}{\sqrt{2}}$$

$z = 1/2$ Condition ② satisfied

$$h(z) = \frac{1}{\sqrt{2}} (-2^{0/2}) = -1/\sqrt{2}$$

Hence the Haar basis for $N=2$

k \ n	0	1
0	$1/\sqrt{2}$	$1/\sqrt{2}$
1	$1/\sqrt{2}$	$-1/\sqrt{2}$

$$A_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

for $N=4$

$$A_4 = \frac{1}{\sqrt{4}} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ -\sqrt{2} & -\sqrt{2} & 0 & 0 \\ 0 & 0 & -\sqrt{2} & -\sqrt{2} \end{bmatrix}$$

Similarly, the Haar basis for $N=8$

$$A_8 = \begin{bmatrix} \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} \\ \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & -\frac{1}{2\sqrt{2}} & -\frac{1}{2\sqrt{2}} & -\frac{1}{2\sqrt{2}} & -\frac{1}{2\sqrt{2}} \\ \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \\ \frac{1}{\sqrt{2}} & -\frac{1}{2\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & 0 & 0 & -0 & -\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}$$

Slant Transform

The slant transform is an orthogonal transform containing saw tooth waveform. It is widely used in image compression. The slant transform of the order 2x2 is defined as,

$$S_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

In generally, an NxN matrix can be

$$S_N = \left[\begin{array}{cc|c|cc|c} 1 & 0 & 0 & 1 & 0 & 0 \\ a_N & b_N & & -a_N & b_N & \\ \hline \multicolumn{2}{c|}{0} & I_{N/2-2} & \multicolumn{2}{c|}{0} & I_{N/2-2} \\ \hline 0 & 1 & 0 & 0 & -1 & 0 \\ -b_N & a_N & & b_N & a_N & \\ \hline \multicolumn{2}{c|}{0} & I_{N/2-2} & \multicolumn{2}{c|}{0} & I_{N/2-2} \end{array}\right] \times \begin{bmatrix} S_{N/2} & 0 \\ 0 & S_{N/2} \end{bmatrix}$$

where,

$$a_N = \left[\frac{3N^2}{4(N^2-1)}\right]^{1/2} \quad \& \quad b_N = \left[\frac{N^2-4}{4(N^2-1)}\right]^{1/2}$$

for $N=4$

$$a_4 = \left[\frac{\cancel{3}\times 16}{\cancel{4}\times \cancel{18}}\right]^{1/2} = \frac{2}{\sqrt{5}} \qquad I_{N/2-2} = I_0$$

$$b_N = \left[\frac{12}{4\times 15}\right]^{1/2} = \frac{1}{\sqrt{5}}\,.$$

$$\therefore \quad S_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ a_4 & b_4 & -a_4 & b_4 \\ 0 & 1 & 0 & -1 \\ -b_4 & a_4 & b_4 & a_4 \end{bmatrix} \times \begin{bmatrix} S_2 & 0 \\ 0 & S_2 \end{bmatrix}$$

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

$$S_{4\times4} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 3/\sqrt{5} & 1/\sqrt{5} & -1/\sqrt{5} & -3/\sqrt{5} \\ 0 & -1 & -0 & 1 \\ 1/\sqrt{5} & -3/\sqrt{5} & 3/\sqrt{5} & -1/\sqrt{5} \end{bmatrix}$$

—

Similarly for $N=8$

$$a_8 = \left(\frac{3\times 64}{4\times 63}\right)^{1/2} = \left(\frac{16}{21}\right)^{1/2} = \frac{4}{\sqrt{21}}$$

$$I_{N/2-2} = I_2$$

$$b_8 = \left(\frac{60}{4\times 63}\right)^{1/2} = \frac{\sqrt{5}}{\sqrt{21}}$$

$$S_8 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ a_8 & b_8 & 0 & 0 & -a_8 & b_8 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ -b_8 & a_8 & 0 & 0 & b_8 & a_8 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} S_4 & 0 \\ 0 & S_4 \end{bmatrix}$$

$$S_8 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 4/\sqrt{21} & \sqrt{5}/\sqrt{21} & 0 & 0 & -4/\sqrt{21} & \sqrt{5}/\sqrt{21} & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ -\sqrt{5}/\sqrt{21} & 4/\sqrt{21} & 0 & 0 & \sqrt{5}/\sqrt{21} & 4/\sqrt{21} & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} S_4 & 0 \\ 0 & S_4 \end{bmatrix}.$$

$$S_8 = \frac{1}{2\sqrt{2}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \frac{7}{\sqrt{21}} & \frac{5}{\sqrt{21}} & \frac{3}{\sqrt{21}} & \frac{1}{\sqrt{21}} & \frac{-1}{\sqrt{21}} & \frac{-3}{\sqrt{21}} & \frac{-5}{\sqrt{21}} & \frac{-7}{\sqrt{21}} \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ \frac{1}{\sqrt{5}} & \frac{-3}{\sqrt{5}} & \frac{3}{\sqrt{5}} & \frac{-1}{\sqrt{5}} & \frac{1}{\sqrt{5}} & \frac{-3}{\sqrt{5}} & \frac{3}{\sqrt{5}} & \frac{-1}{\sqrt{5}} \\ \frac{3}{\sqrt{5}} & \frac{1}{\sqrt{5}} & \frac{-1}{\sqrt{5}} & \frac{-3}{\sqrt{5}} & \frac{-3}{\sqrt{5}} & \frac{-1}{\sqrt{5}} & \frac{1}{\sqrt{5}} & \frac{3}{\sqrt{5}} \\ \frac{7}{\sqrt{105}} & \frac{-1}{\sqrt{105}} & \frac{-9}{\sqrt{105}} & \frac{-17}{\sqrt{105}} & \frac{17}{\sqrt{105}} & \frac{9}{\sqrt{105}} & \frac{1}{\sqrt{105}} & \frac{-7}{\sqrt{105}} \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ \frac{1}{\sqrt{5}} & \frac{-3}{\sqrt{5}} & \frac{3}{\sqrt{5}} & \frac{-1}{\sqrt{5}} & \frac{-1}{\sqrt{5}} & \frac{3}{\sqrt{5}} & \frac{-3}{\sqrt{5}} & \frac{1}{\sqrt{5}} \end{bmatrix}$$

KL Transform (Karhunen-Loeve Transform or Hotelling Transform)

Generally KL transform is applied to n-dimensional vector population to get another set of n-dimensional vector population.

Consider a population of random vectors of the form,

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

and the mean vector of the population is defined as the expected value, $m_x = E\{x\}$

The expected value of a vector is obtained by taking the expected value of each element. The co-variance matrix can be calculated as

$$C_x = E\left\{(x-m_x)(x-m_x)^T\right\}$$

The covariance matrix is real and symmetric. For M random vectors, when M is large enough, the mean vector and co-variance matrix can be approximately calculated as,

$$m_x = \frac{1}{M}\sum_{k=1}^{M} x_k$$

$$\& \quad C_x = \frac{1}{M}\sum_{k=1}^{M} x_k x_k^T - m_x m_x^T$$

So, It is possibile to find a set of n - orthonormal eigen vectors and eigen values of the matrix C_x satisfies the equation,

$$C_x e_i = \lambda_i e_i \quad \text{for } i=1,2,\ldots n-1$$

where e_i and λ_i are eigen vectors and corresponding eigen values of the co-variance matrix C_x, For covariance eigen vectors e_i is arrange the eigen values in descending order so that $\lambda_i \geq \lambda_{i+1}$ for $i=1,2,\ldots,n-1$. let A be the transform kernel matrix and it is formed by its first row is the largest eigen value and the rest of row ~~being~~ arranged

Scanned by CamScanner

in descending order with the last row being the smallest eigen value. Then the transformation x vectors into y vectors of the form

$$y = A(x - m_x)$$

This transform is called Karhunen-Loeve (or) Hoteling Transform. The mean of the y-vector will be zero and its covariance matrix is

$$C_y = A C_x A^T$$

where C_y is a diagonal matrix whose main diagonal elements are the eigen values of C_x

$$\text{i.e } C_y = \begin{bmatrix} \lambda_1 & 0 & 0 \cdots & 0 \\ 0 & \lambda_2 & 0 \cdots & 0 \\ 0 & 0 & \lambda_3 \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 \cdots & \lambda_n \end{bmatrix}$$

The original vector x can be reconstructed as,

$$x = A^T y + m_x$$

Applications

(i) Hoteling Transform is used in the object recognition proc to remove the effects of rotation caused due to image analysis process.

② It is used for compressing the data required to represents the image.

Singular-Value Decomposition (SVD) Transform

The SVD transform of an image is written as

$$g = SVD(A)$$

The SVD method transforms the given matrix A into the product $U \times S \times V^{-1}$. Let us assume that the image ~~and~~ has m rows and n columns. Then the image A can be written as,

$$A = U \times S \times V^{-1} = U \times S \times V^{T}$$

where, The matrix U is an $(m \times m)$ orthogonal matrix. The column vectors form an orthonormal set.

i.e $u_i^T \times u_j = 1$ for $i = j$ & 0 for $i \neq j$

The matrix V is an $n \times n$ orthogonal matrix and its column form an orthonormal set. S is a matrix whose dimensions are $m \times n$ with singular values as

$$S = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 & 0 \\ 0 & \sigma_2 & \cdots & 0 & 0 \\ \vdots & & & & \\ 0 & 0 & \cdots & \sigma_{n-1} & 0 \\ 0 & 0 & \cdots & 0 & \sigma_n \\ 0 & 0 & & 0 & 0 \end{bmatrix}$$

where $\sigma_1, \sigma_2 \ldots \sigma_n$ are called singular values which are the square roots of the eigen values and form the diagonal of S. Hence the image can be expressed as,

$$A = U \times S \times V^T$$

$$= \sum_{i=1}^{r} \sigma_i u_i v_i^T$$

This is equal to the outer product expansion of the singular vectors u and v. This is expressed as,

$$A = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_r u_r v_r^T$$

Here r is called the rank of the matrix. The rank of matrix is the no. of non-zero singular values of the image.

The SVD transform is useful in image compression, image restoration and object recognition.

Problems

① Assume that a 15m high structure is observed from a distance of 75m. What is the size of the retinal image?

Dec-2013

Sol:

Object height = 15m

Distance b/w object & observer = 75m

Let size of the retinal image (h) = ?

We know that

$$\frac{\text{height of object}}{\text{Distance b/w object \& observer}} = \frac{h}{17\,mm}$$

$$\Rightarrow \frac{15}{75} = \frac{h}{17mm} \Rightarrow \boxed{h = 3.4mm}$$

② Assume that a 10m high structure is observed from a distance of 50m. What is the size of retinal image?

Sol:

$$\frac{10}{50} = \frac{h}{17mm} \Rightarrow \boxed{h = 3.4mm}$$

③ A Common measure of transmission for digital data is baud rate, defined as the no. of bits transmitted per second?. Generally, transmission is acomplished in packets Consists of starting bit, a byte of information and a stop bit. Using this approach, answer the following?

ⓐ How many minutes would it take to transmit a 512×512 image with 128 gray levels at 300 baud?

(b) what would be the time be at 9600 baud?

(c) Repeat (a) & (b) for 1024 x 1024 image with 128 gray levels?

Sol: Given that

(a) Size of image = 512 x 512 (M x N)

$\Rightarrow M = N = 512$

No. of gray levels $L = 126$

$\Rightarrow 2^k = 126 \Rightarrow \boxed{k = 7}$

Baud Rate $B = 300$ baud

Total no. of bits required to store $= N^2 k = 512 \times 512 \times 7$

$= 1835008$ bits

The no. of seconds to transmit the given image is

$$= \frac{1835008}{300} = 6116.7 \text{ Sec (or) } 101.945 \text{ mins}$$

(b) If Baud Rate is 9600

The no. of Seconds required to transmit the given image is $= \frac{1835008}{9600} = 191.15$ Sec (or) 3.19 mins

(c) For a 1024 x 1024 image

$M = N = 1024$

No. of gray levels $L = 128 \Rightarrow 2^k = 128$

$\boxed{k = 7}$

The total no. of bits required to store

$$= N^2 k = 1024 \times 1024 \times 7$$

$$= 7340032 \text{ bits}$$

At a baud rate of 300, the (no. of seconds) time required to transmit image $= \dfrac{7340032}{300} = 24466.78$ Sec (or) 407.78 mins

At a baud rate of 9600, the time required to transmit image $= \dfrac{7340032}{9600} = 764.59$ Sec (or) 12.74 min

④ Let $V = \{0, 1\}$, Compute the D_e, D_4 and D_8 distances b/w two pixels p and q. Let the pixel co-ordinates of p and q be (3,0) and (2,3) respectively, for the image

0	1	1	1
1	0	0	1
1	1	1	1 q
1 (p)	1	1	1

sol:
The Euclidean distance is

$$D_e = \sqrt{(x-s)^2 + (y-t)^2}$$

$$= \sqrt{(3-2)^2 + (0-3)^2} = \sqrt{10}$$

The D_4 distance is

$$D_4 = |x-s| + |y-t| = 1 + 3 = 4$$

The D_8 Distance is

$$D_8 = \max(|x-s|, |y-t|) = \max(1,3) = 3$$

(5) An object is 15 Cm wide and is imaged with a sensor of size 8.8 × 6.6 mm from a distance 0.7 m. What should be the required focal length?

Sol: The focal length is expressed as,

$$f = \frac{uM}{M+1}$$

where u = distance b/w object and sensor

M = Magnification factor

$$\Rightarrow M = \frac{\text{Size of the image}}{\text{Size of object}} = \frac{8.8}{150} = 0.0587$$

$$\therefore \quad f = \frac{700 \times 0.0587}{1.0587} = 38.81 \text{ mm}$$

(6) A Medical image has a size of 8×8 inches. The Sampling resolution is 5 Cycles/mm. How many pixels are required? Will an image of size 256×256 be enough?

Sol: The Sampling Resolution is 5 Cycles/mm

For better quality, 2 pixels per cycle are required

∴ For given resolution 10 pixels per mm required

E: 1 inch = 25.4 mm

The size of given image = 8×8 inches

$= 8 \times 25.4 \times 8 \times 25.4 = 203.2 \times 203.2$ mm

The minimum no. of pixels required $= 203.2 \times 10 \times 203.2 \times 10 = 2032 \times 2032$