Frequency Domain Transformation

Image Enhancement

- There are two broad categories of image enhancement techniques
  - i. **Spatial domain techniques**
    - ➢ Direct manipulation of image pixels (example: point operations)
  - ii. **Frequency domain techniques**
    - ➢ Image enhancement in the frequency domain is straightforward. We simply compute the **Fourier transform** of the image to be enhanced, multiply the result by a filter (rather than convolve in the spatial domain), and take the inverse transform to produce the enhanced image.
- Similar jobs can be done in the spatial and frequency domains
- Filtering in the spatial domain can be easier to understand
- Filtering in the frequency domain can be much faster – especially for large images

Fourier Transform

- Fourier transform transforms the one function (domain) to another function (domain) called frequency domain representation of the original function.
- The original function is often a function in time domain
- In image processing, the original function is the spatial domain.
- The term Fourier transform can refer either to the frequency domain representation of a function or the process that transforms one domain to another (example: Spatial domain to frequency domain)

1D Fourier Transform

- Let *f(x)* be a continuous function of a real variable *x*
- The Fourier transform of *f(x)*, denoted by $\Im\{f(x)\}$ is given by:

$$\Im\{f(x)\} = F(u) = \int_{-\infty}^{+\infty} f(x)\exp[-j2\pi ux]dx$$

- where $j = \sqrt{-1}$
- Given *F(u)*, *f(x)* can be obtained by using the *inverse Fourier transform:*

$$\Im^{-1}\{F(u)\} = f(x) = \int_{-\infty}^{+\infty} F(u)\exp[j2\pi ux]du.$$

- X represents time in second and u represents frequency (cycles/second)

Note:

- We are concerned with functions f(x) which are real, however the Fourier transform of a real function is, generally, complex. So,

$$F(u) = R(u) + jI(u)$$

The magnitude function |F(u)| is called the *Fourier spectrum* of f(x)

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

and φ(u) is the phase angle.

$$\varphi(u) = \tan^{-1}\left[\frac{I(u)}{R(u)}\right]$$

- The square of the spectrum,

$$P(u) = |F(u)|^2 = R^2(u) + I^2(u)$$

- is commonly called the *power spectrum* (or the *spectral density*) of f(x).

2D Fourier Transform

- The Fourier transform can be extended to 2 dimensions:

$$\Im\{f(x,y)\} = F(u,v) = \int\int_{-\infty}^{+\infty} f(x,y)\exp[-j2\pi(ux+vy)]dxdy.$$

- and the inverse transform

$$\Im^{-1}\{F(u,v)\} = f(x,y) = \int\int_{-\infty}^{+\infty} F(u,v)\exp[j2\pi(ux+vy)]dudv.$$

Application of Fourier Transform

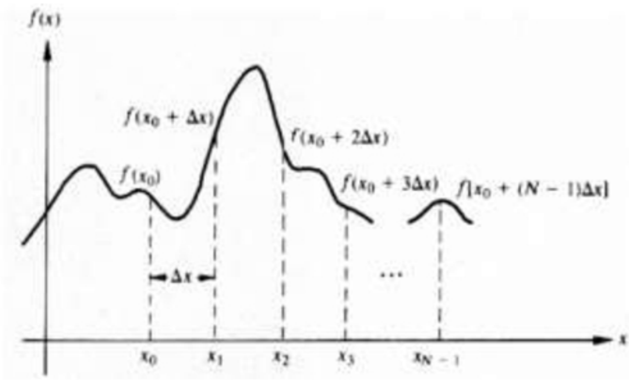
1. 1D Fourier transform is extensively used in signal processing.
2. 2D Fourier transform is used in image processing for image enhancement, image restoration, image encoding/decoding, image description and computer vision.

Discrete Fourier Transform: 1D

- Suppose a continuous function, f(x), is discretized into a sequence
  $\{f(x_0), f(x_0+\Delta x), f(x_0+2\Delta x), \ldots, f(x_0+[N-1]\Delta x)\}$
- by taking N samples Δx units apart
- Let *x* refer to either a continuous or discrete value by saying

$$f(x) = f(x_0 + x\Delta x)$$

- where *x* assumes the discrete values 0, 1, …, N-1 and
- {f(0),f(1),…,f(N-1)} denotes any N uniformly spaced samples from a corresponding continuous function



- The discrete Fourier transform is given by:

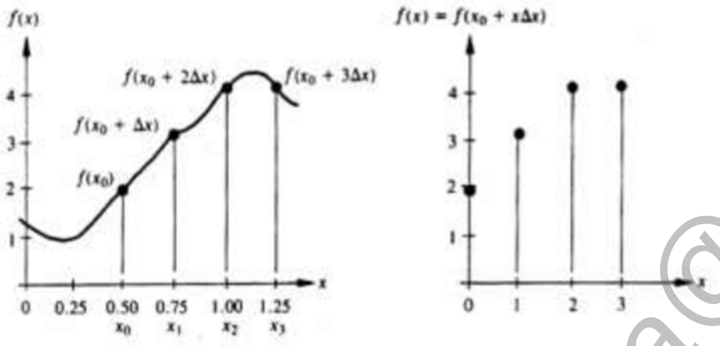
$$F(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)\exp[-j2\pi ux/N]$$

- for u=0, 1, … ,N-1
- The discrete inverse Fourier transform is given by:

$$f(x) = \sum_{u=0}^{N-1} F(u)\exp[j2\pi ux/N]$$

- for x=0, 1, … ,N-1
- The values of u=0, 1, … ,N-1in the discrete case correspond to samples of the continuous transform at 0, Δu, 2Δu, …, (N-1)Δu
  Δu and Δx are related by Δu=1/(N Δx)

Example:



- Consider sampling at x_0=.5, x_1=.75, x_2=1.0, and x_3=1.25
- Here Δx=.25 and x ranges from 0 → 3

Discrete Fourier Transform: 2D

- In the 2-D case:

$$F(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\exp[-j2\pi(ux/M + vy/N)]$$

- for *u=0 → M-1* and *v=0 → N-1*

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

- for *x=0 → M-1* and *y=0 → N-1*
- The discrete function f(x,y) represents samples of the continuous function at $f(x_0+x\Delta x,\ y_0+y\Delta y)$
  $\Delta u=1/(M\Delta x)$ and $\Delta v=1/(N\Delta y)$

Note:

EULER's FORMULA

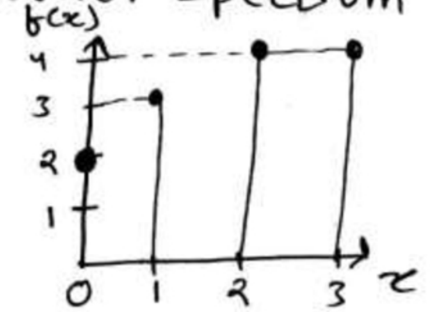
$e^{i\theta} = \cos\theta + i\sin\theta$

$e^{i\omega t} = \cos\omega t + i\sin\omega t$

where $i = \sqrt{-1}$

Example: $\exp[-j2\pi ux] = \cos(2\pi ux) - j\sin(2\pi ux)$

Find the fourier spectrum of the function below.



$f(0) = 2$
$f(1) = 3$
$f(2) = 4$
$f(3) = 4$

Soln:

We know

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

Here $N = 4$ (i.e 0, 1, 2 and 3)

$$\therefore F(u) = \frac{1}{4} \sum_{x=0}^{3} f(x) \cdot e^{\frac{-j \cdot 2\pi ux}{4}}$$

$$= \frac{1}{4} \sum_{x=0}^{3} f(x) \cdot e^{\frac{-j\pi ux}{2}}$$

- Now, At $u = 0$, i.e $f(0) = ?$

$$F(0) = \frac{1}{4} \sum_{x=0}^{3} f(x) \cdot e^{0} = \frac{1}{4} \sum_{x=0}^{3} f(x)$$

$$= \frac{1}{4} [f(0) + f(1) + f(2) + f(3)]$$

$$= \frac{1}{4} [2 + 3 + 4 + 4]$$

$$= \frac{13}{4} \quad \text{i.e} \left(\underbrace{\frac{13}{4}}_{R} + \underbrace{0}_{I} \cdot J \right)$$

$\therefore$ Fourier Spectrum: $|F(0)| = \sqrt{R^2 + I^2}$

$$= \sqrt{\left(\frac{13}{4}\right)^2 + 0^2}$$

$$= \frac{13}{4}$$

- At $u = 1$, i.e $F(1) = ?$

$$F(1) = \frac{1}{4} \sum_{x=0}^{3} f(x) \cdot e^{-\frac{j\pi x}{2}}$$

$$= \frac{1}{4}\left[f(0)\cdot e^{0} + f(1)\cdot e^{-j\frac{\pi}{2}} + f(2)\cdot e^{-j\pi} + f(3)\, e^{-\frac{3j\pi}{2}}\right]$$

$$= \frac{1}{4}\left[2\cdot 1 + 3\cdot\left(\cos\frac{\pi}{2} - j\cdot\sin\frac{\pi}{2}\right) + 4\cdot\left(\cos\pi - j\cdot\sin\pi\right) + 4\cdot\left(\cos\frac{3\pi}{2} - j\cdot\sin\frac{3\pi}{2}\right)\right]$$

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

$$= \frac{1}{4}\left[2 - 3j - 4 + 4j\right]$$

$$= \frac{1}{4}\left[-2 + j\right]$$

$$= -\frac{1}{2} + \frac{1}{4}j$$

$\therefore$ Fourier Spectrum $|F(1)| = \sqrt{R^2 + I^2}$

$$= \sqrt{\left(-\frac{1}{2}\right)^2 + \left(\frac{1}{4}\right)^2}$$

$$= \frac{\sqrt{5}}{4}$$

Similarly,

$F(2) = -\frac{1}{4}$ and $|F(2)| = \frac{1}{4}$

$F(3) = -\frac{1}{4}(2+j)$ and $|F(3)| = \frac{\sqrt{5}}{4}$

Perform inverse fourier transformation for previous Fourier transform terms.

Soln:

We know,

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

Here $N=4$,

$$\therefore f(x) = \sum_{u=0}^{3} F(u) \cdot e^{\frac{j\pi ux}{2}}$$

Given

$F(0) = 3.25$

$F(1) = \frac{1}{4}(-2+j)$

$F(2) = -\frac{1}{4}$

$F(3) = -\frac{1}{4}(2+j)$

- Now at $x=0$, i.e. $f(0) = ?$

$$f(0) = \sum_{u=0}^{3} F(u) \cdot e^{0}$$

$$= \sum_{u=0}^{3} F(u)$$

$$= F(0) + F(1) + F(2) + F(3)$$

$$= 3.25 + \frac{1}{4}(-2+j) - \frac{1}{4} - \frac{1}{4}(2+j)$$

$$= 3.25 - \frac{1}{2} + \cancel{\frac{j}{4}} - \frac{1}{4} - \frac{1}{2} - \cancel{\frac{j}{4}}$$

$$= 3.25 - 1.25$$

$$= 2$$

- At $x=1$, i.e. $f(1) = ?$

$$f(1) = \sum_{u=0}^{3} F(u) \cdot e^{\frac{j\pi u}{2}}$$

$$= F(0) \cdot e^{0} + F(1) \cdot e^{\frac{j\pi}{2}} + F(2) \cdot e^{j\pi} + F(3) \cdot e^{\frac{3j\pi}{2}}$$

$$= 3.25 \cdot 1 + \frac{1}{4}(-2+j) \cdot \left(\cos\frac{\pi}{2} + j \cdot \sin\frac{\pi}{2}\right) - \frac{1}{4} \cdot (\cos\pi + j \cdot \sin\pi) - \frac{1}{4}(2+j)\left(\cos\frac{3\pi}{2} + j\sin\frac{3\pi}{2}\right)$$

$$= 3.25 + \frac{1}{4}(-2+i)\cdot(0+i) - \frac{1}{4}(-1+0) - \frac{1}{4}(2+i)(0-i)]$$

$$= 3.25 + \frac{i}{4}(-2+i) + \frac{1}{4} + \frac{i}{4}(2+i)$$

$$= 3.25 - \frac{i}{2} + \frac{i^2}{4} + \frac{1}{4} + \frac{i}{2} + \frac{i^2}{4}$$

$$= 3.25 - \cancel{\frac{i}{2}} - \cancel{\frac{1}{4}} + \cancel{\frac{1}{4}} + \cancel{\frac{i}{2}} - \frac{1}{4}$$

$$= 3.25 - \frac{1}{4}$$

$$= 3$$

Similarly calculate

$b(2) = 4$

and

$b(3) = 4$ #

Filtering in Frequency Domain

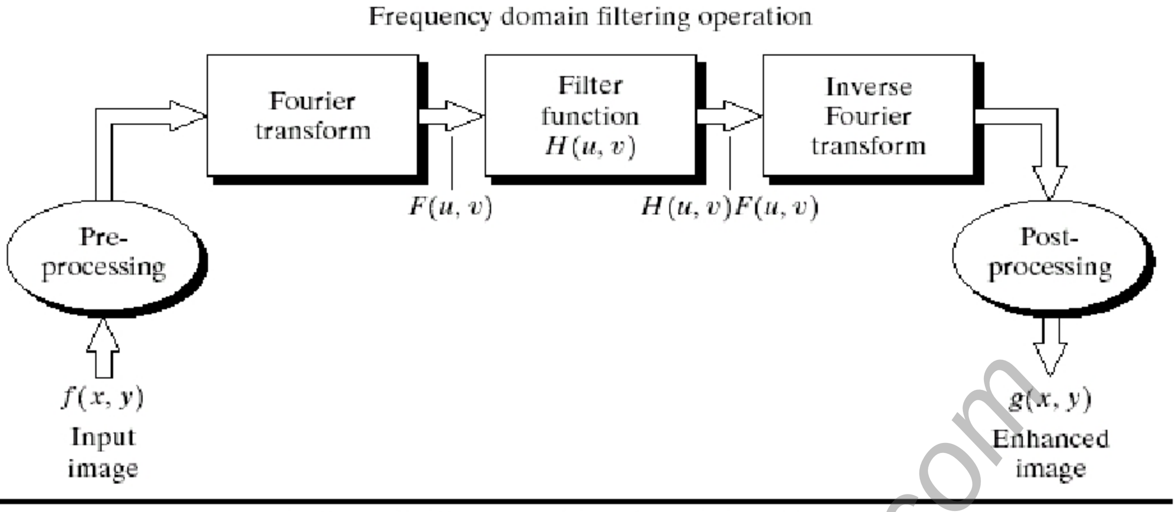
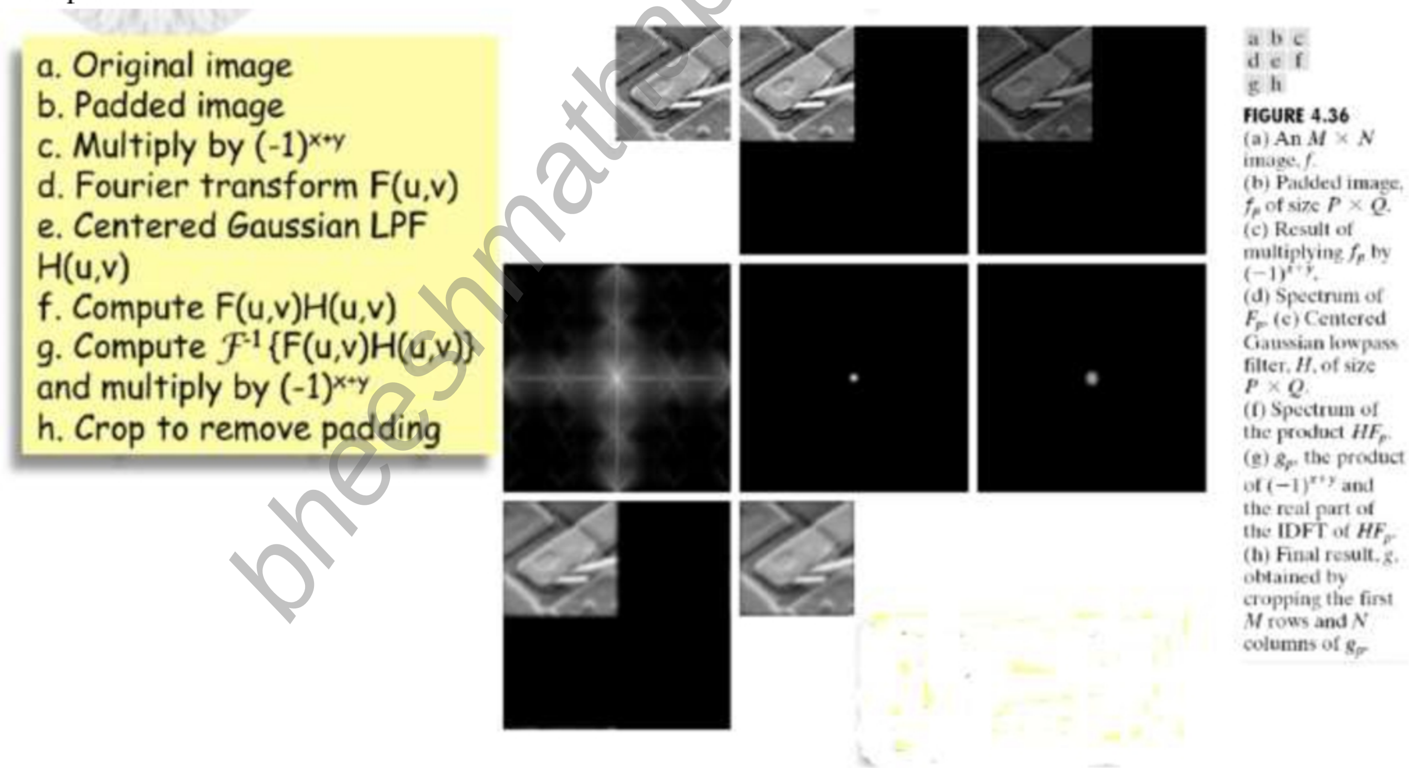


Fig: Steps for Frequency Domain Filtering Operation

- The filtering operation in frequency domain is quite straight forward. We carry out the following steps.
  - The first step is pre-processing. In this step, different operation like zero padding, multiplying the input image by $(-1)^{X+Y}$ so as to centre the transform etc. are performed.
  - Next, DFT is computed G(u,v) of the pre-processed image.
  - Next, the obtained 2D G (u, v) is simply multiplied(convolved) by a filter function H(u,v) which may be a low pass filter, high pass filter, Butterworth filter etc.
  - Next, the inverse DFT of the obtained result is computed.
  - Next, the real part of previous obtained result is extracted.
  - Finally, the result obtained is post processed with different operation like multiplying by $(-1)^{X+Y}$, Cropping etc.

Example



Some Basic Filter in Frequency Domain

- A filter supress some frequencies in the transform while leaving other unchanged.
- There are two types of filter

1. Low pass filter (Smoothing filter)
   - Low frequency in Fourier transform are responsible for general grey level appearance of an image over smooth area.
   - A filter which attenuates high frequency while passing only low frequency is known as low pass filter.
   - Such filter always appears smoother.
2. High pass filter (Sharpening filter)
   - High frequency in Fourier transform are responsible for details such as edge and noise.
   - A filter which attenuates low frequencies while passing high frequency is known as high pass filter.
   - Such filter always appears sharper.

Smoothing Frequency Domain Filters

- Edge and other sharp transition such as noise in the gray level of an image contribute significantly to high frequency content of its Fourier transform.
- Thus, smoothing may be achieved in the frequency domain by attenuating a specified range of high frequency components in the transform of a given image.
- Let us reproduce the expression representing the basic model for filtering as

$$\mathbf{F(u,v) = G(u,v).H(u,v)}$$

Where, G(u,v) is the Fourier Transform of the image to be smoothed and H(u,v) is transfer function which yields F(u,v) by attenuating the high frequency components of G(u,v).

- There are three types of low pass filter

i. Ideal LPF
ii. Butterworth Filter
iii. Gaussian LPF

Ideal Low pass filter

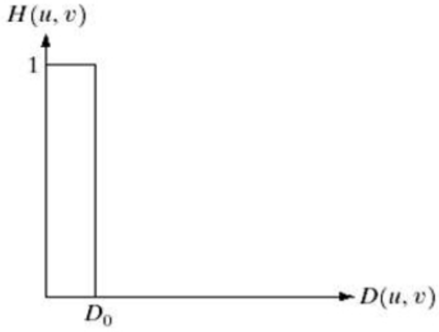
- The simplest low pass filter may be a filter which cuts off all high frequency components of the Fourier transform that are at a distance greater than a specified distance D_0 from the origin of the transform. Such type of filter is known as 2D ideal low pass filter.
- The centre coefficient F (M/2, N/2) correlates to lower frequency of the original image. As we move away from it i.e. centre, in any direction the Fourier coefficients correlate to higher and higher frequencies.
- *Low pass filters* – only pass the low frequencies, drop the high ones
- It has the following transfer function.

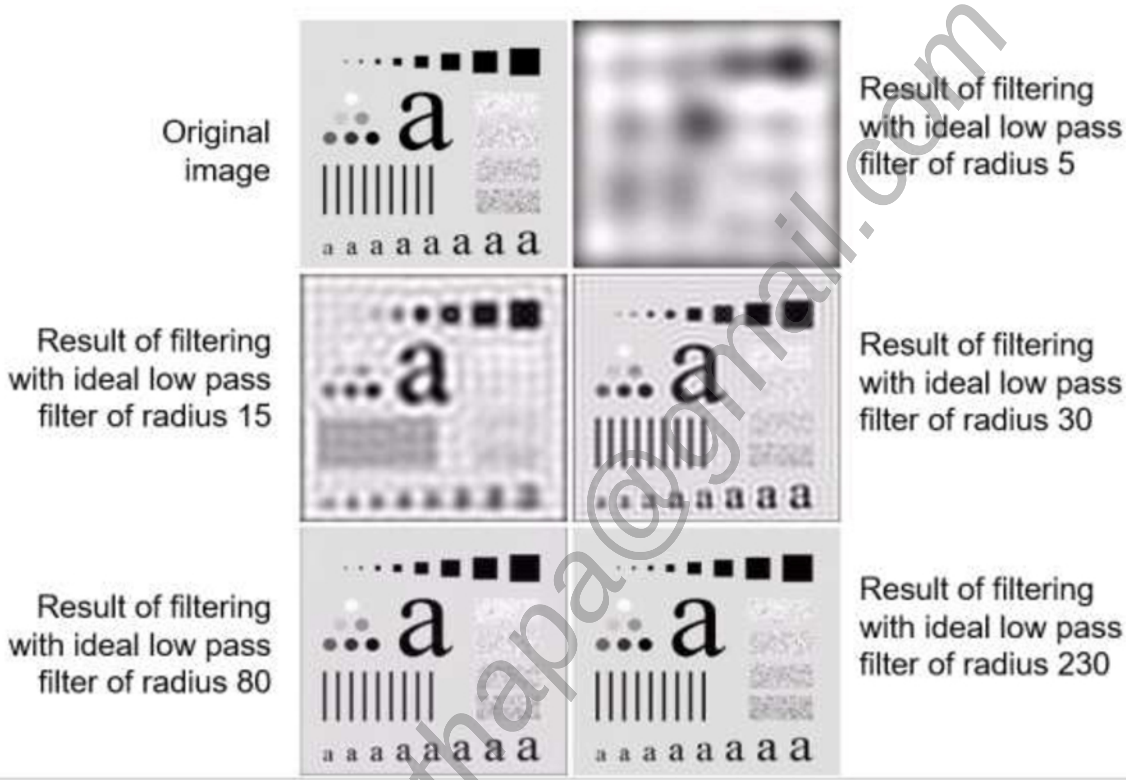
$$H(u,v) = \begin{cases} 1 \text{ if } D(u,v) \le D_0 \\ 0 \text{ if } D(u,v) > D_0 \end{cases}$$

where $D(u,v)$ is given as:

$$D(u,v) = [(u - M/2)^2 + (v - N/2)^2]^{1/2}$$

i.e. D(u, v) is the distance from the point (u, v) to the origin of frequency. D_0 is specified non negative quantity i.e. Cut-off distance.





Butterworth LPF

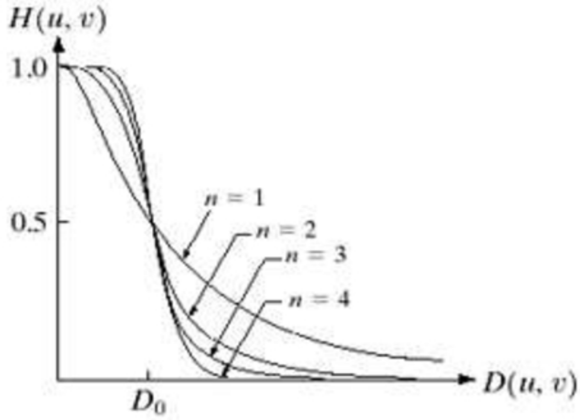
- The transfer function of a Butterworth lowpass filter of order *n* with cut-off frequency at distance D_0 from the origin is defined as:

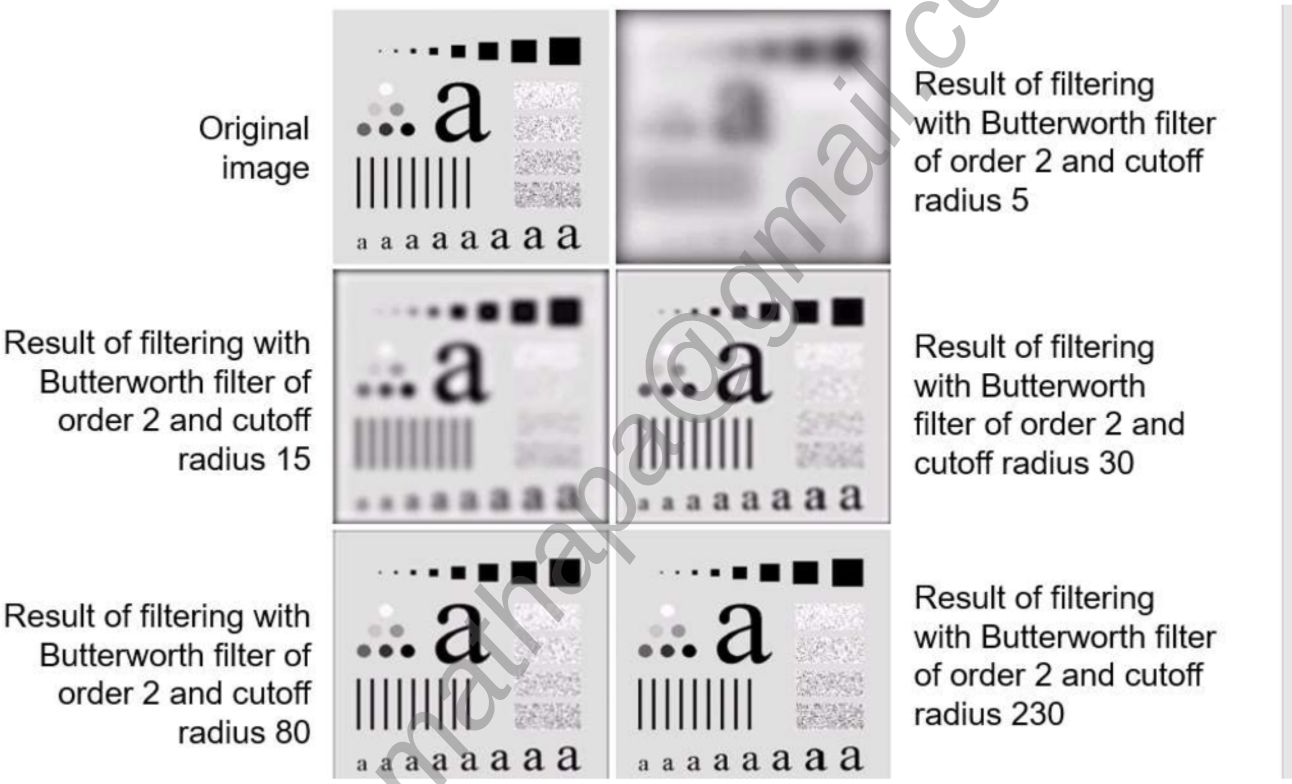
$$H(u,v) = \frac{1}{1+[D(u,v)/D_0]^{2n}}$$

where $D(u,v)$ is given as:

$$D(u,v) = [(u-M/2)^2 + (v-N/2)^2]^{1/2}$$

i.e. D(u, v) is the distance from the point (u, v) to the origin of frequency. D_0 is specified non negative quantity.





Gaussian LPF

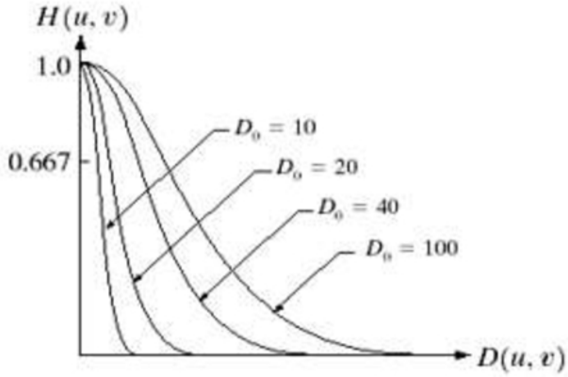
- The transfer function of a Gaussian lowpass filter is defined as:

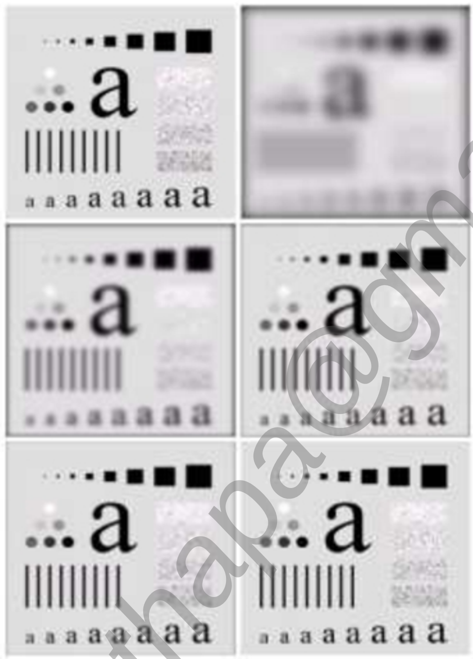
$$H(u,v) = e^{-D^2(u,v)/2D_0^2}$$

where $D(u,v)$ is given as:

$$D(u,v) = [(u - M/2)^2 + (v - N/2)^2]^{1/2}$$

i.e. D(u, v) is the distance from the point (u, v) to the origin of frequency. D_0 is specified non negative quantity.





Original image

Result of filtering with Gaussian filter with cutoff radius 5

Result of filtering with Gaussian filter with cutoff radius 15

Result of filtering with Gaussian filter with cutoff radius 30

Result of filtering with Gaussian filter with cutoff radius 85

Result of filtering with Gaussian filter with cutoff radius 230

Sharpening Frequency Domain Filter

- An image can be blurred by attenuating high frequency components of its Fourier transform.
- Since, edge and other abrupt changes in grey level are associated with high frequency components so image sharpening may be achieved by applying high pass filter which attenuates the low frequency components without disturbing high frequency information in the Fourier transform.
- The Fourier function of high pass filter may be obtained using the following relationship.

$$H_{hp}(u, v) = 1 - H_{lp}(u, v)$$

High pass filter are the reverse of low pass filter.

- Types of High Pass Filter(HPF)
  1. Ideal HPF
  2. Butterworth HPF
  3. Gaussian HPF

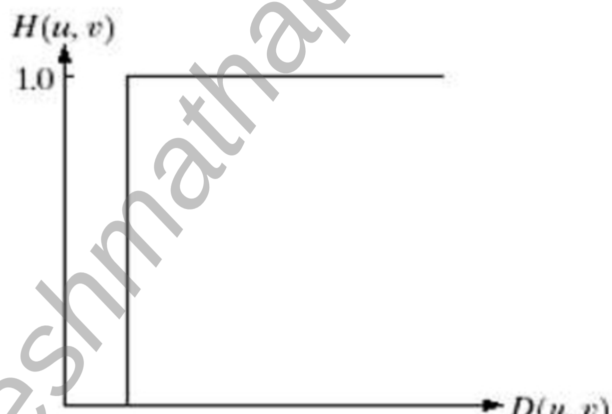
Ideal HPF

- A 2D ideal HPF may be described with the help of following equation

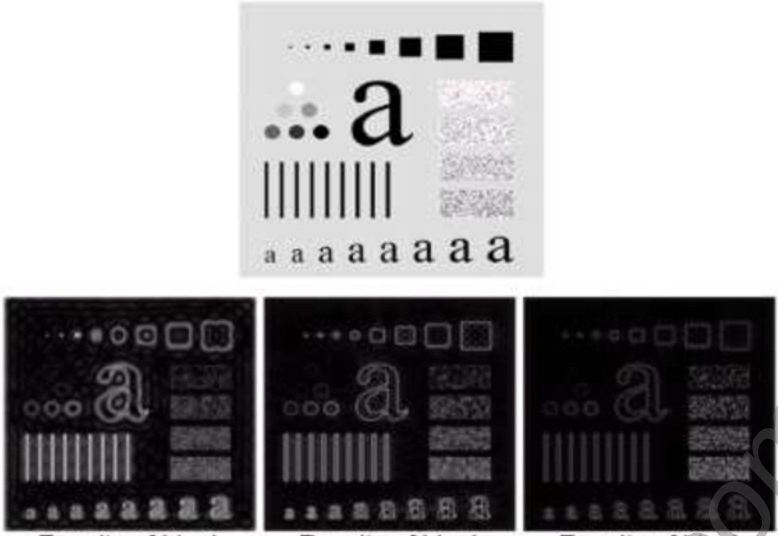
$$H(u,v) = \begin{cases} 0 \text{ if } D(u,v) \leq D_0 \\ 1 \text{ if } D(u,v) > D_0 \end{cases}$$

i.e. D_0 is specified non negative quantity. D(u, v) is the distance from the point (u, v) to the origin of frequency given as

$$D(u,v) = [(u - M/2)^2 + (v - N/2)^2]^{1/2}$$



Example:



Results of ideal high pass filtering with $D_0 = 15$

Results of ideal high pass filtering with $D_0 = 30$

Results of ideal high pass filtering with $D_0 = 80$

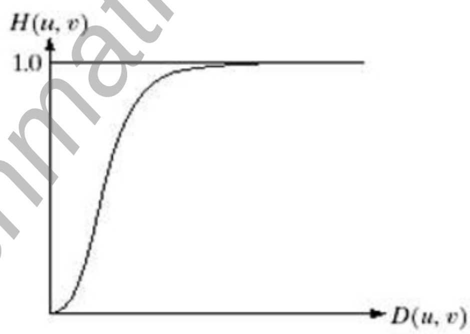
Butterworth HPF

- The transfer function of the butterworth HPF of order 'n' with cut off frequency at a distance D_0 from the origin is described with the help of following equation

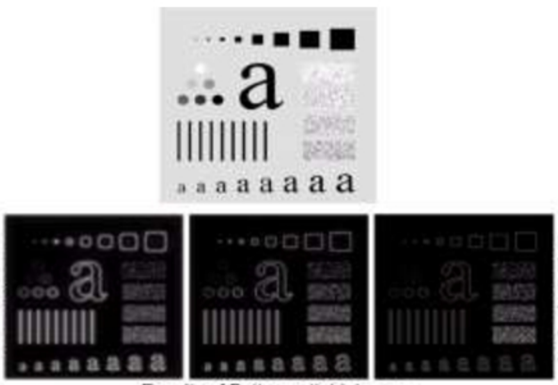
$$H(u,v) = \frac{1}{1+[D_0 / D(u,v)]^{2n}}$$

i.e. D_0 is specified non negative quantity. D (u, v) is the distance from the point (u, v) to the origin of frequency given as

$$D(u,v) = [(u - M/2)^2 + (v - N/2)^2]^{1/2}$$



Example:



Results of Butterworth high pass filtering of order 2 with $D_0 = 15$

Results of Butterworth high pass filtering of order 2 with $D_0 = 30$

Results of Butterworth high pass filtering of order 2 with $D_0 = 80$

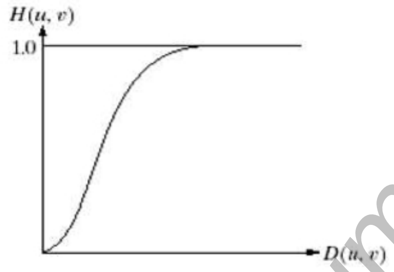
Gaussian HPF

- The transfer function of Gaussian HPF with cutoff frequency at distance D_0 from origin is described with the help of following equation.

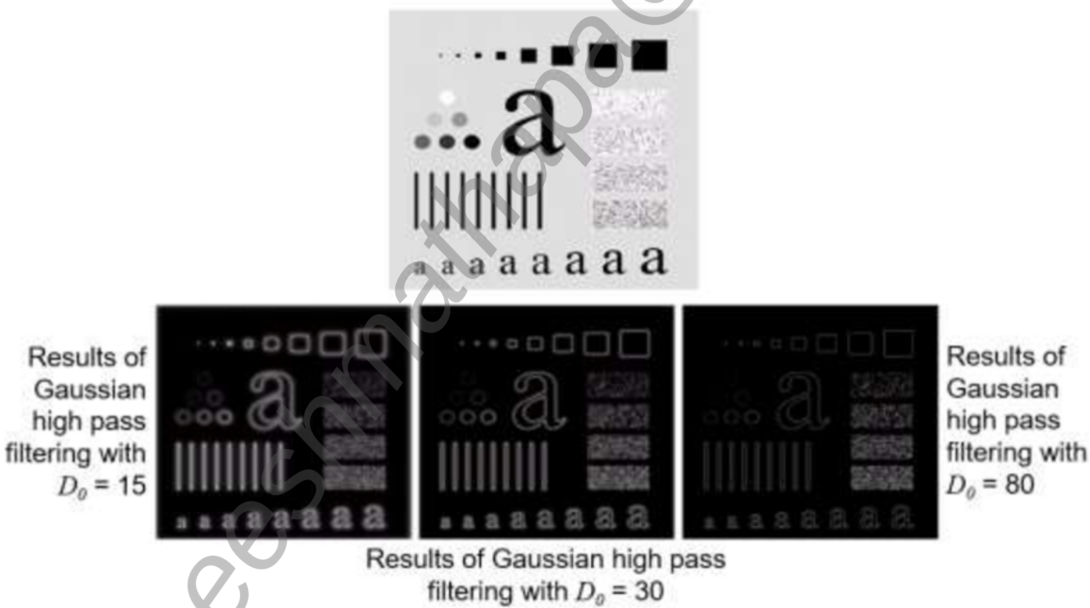
$$H(u,v) = 1 - e^{-D^2(u,v)/2D_0^{\,2}}$$

i.e. D_0 is specified non negative quantity. D (u, v) is the distance from the point (u, v) to the origin of frequency given as

$$D(u,v) = [(u - M/2)^2 + (v - N/2)^2]^{1/2}$$

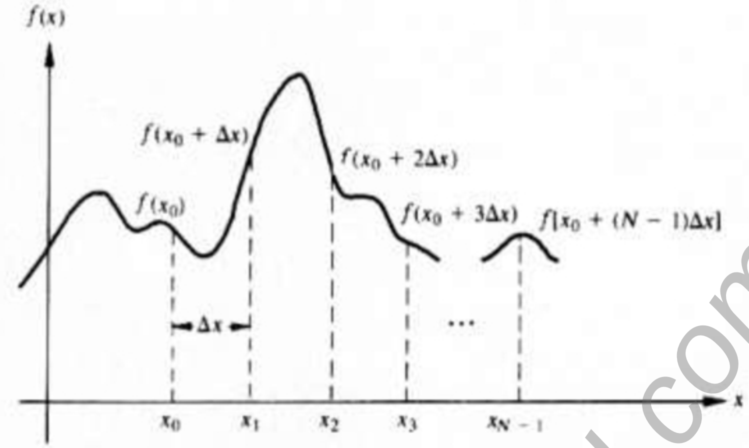


Example:



Fast Fourier Transform

- FFT is an algorithm to compute discrete Fourier transform and its inverse.
- DFT is obtained by decomposing a sequence of values into a components of frequencies F(u).



- In above figure f(x) is a continuous function to be changed by taking N-Sample.
  - Suppose a continuous function, f(x), is discretized into a sequence
    $\{f(x_0), f(x_0+\Delta x), f(x_0+2\Delta x), \ldots, f(x_0+[N-1]\Delta x)\}$
  - by taking N samples Δx units apart
  - Let *x* refer to either a continuous or discrete value by saying

$$f(x) = f(x_0 + x\Delta x)$$

  - where *x* assumes the discrete values 0, 1, …, N-1 and
  - {f(0),f(1),…,f(N-1)} denotes any N uniformly spaced samples from a corresponding continuous function

  - The discrete Fourier transform is given by:

$$F(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)\exp[-j2\pi ux/N]$$

  - for u=0, 1, … ,N-1
  - The discrete inverse Fourier transform is given by:

$$f(x) = \sum_{u=0}^{N-1} F(u)\exp[j2\pi ux/N]$$

  - for x=0, 1, … ,N-1
  - The values of u=0, 1, … ,N-1in the discrete case correspond to samples of the continuous transform at 0, Δu, 2Δu, …, (N-1)Δu
    Δu and Δx are related by Δu=1/(N Δx)

- In the 2-D case:

$$F(u,v)=\frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}f(x,y)\exp[-j2\pi(ux/M+vy/N)]$$

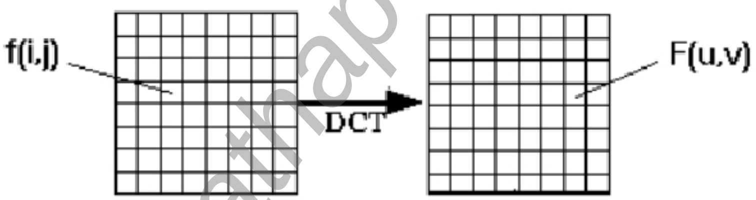
- for *u=0* $\rightarrow$ *M-1* and *v=0* $\rightarrow$ *N-1*

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

- for *x=0* $\rightarrow$ *M-1* and *y=0* $\rightarrow$ *N-1*
- The discrete function f(x,y) represents samples of the continuous function at $f(x_0+x\Delta x,\ y_0+y\Delta y)$
  $\Delta u=1/(M\Delta x)$ and $\Delta v=1/(N\Delta y)$

Discrete Cosine Transform

- DFT is very popular due to its good computational efficiency.
- However, DFT has some strong disadvantages for some application
  - i. It is complex i.e. DFT possesses real as well as complex coefficients
  - ii. It has comparable poor energy compaction capacity.
    (The ability to pack the energy of the spatial sequence into a few frequency components as possible is called energy compaction)
- DCT is similar to DFT but it is only concerned with only the real parts



- Since, DCT provides high energy compaction capacity so this method widely used in **image compression (e.g. JPEG compression). If compaction is high, we can only transmit or store a few coefficients.**
- The general equation for a 1D, DCT is defined by the following equation:

$$DCT = F(u) = \alpha(u)\cdot\sum_{x=0}^{N-1} f(x)\cdot\cos\left[\frac{\pi(2x+1)u}{2N}\right]$$

where $u=0$ to $N-1$
and
$$\alpha(u)=\frac{1}{\sqrt{N}} \text{ for } u=0$$
$$=\sqrt{\frac{2}{N}} \text{ for } u=1 \text{ to } N-1$$

- The general equation for a 1D, IDCT is defined by the following equation:

$$IDCT = f(x) = \sum_{u=0}^{N-1}\alpha(u)\cdot F(u)\cdot\cos\left[\frac{\pi(2x+1)u}{2N}\right]$$

where $x=0$ to $N-1$

- The general equation for a 2D, DCT is defined by the following equation:

$$DCT = F(u,v) = \alpha(u)\cdot\alpha(v)\cdot\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cdot\cos\left[\frac{\pi(2x+1)u}{2N}\right]\cdot\cos\left[\frac{\pi(2y+1)v}{2N}\right]$$

where $\alpha(u) = \frac{1}{\sqrt{N}}$ for $u = 0$

$= \sqrt{\frac{2}{N}}$ for $u = 1$ to $N-1$

and [redacted] $u,v = 0$ to $N-1$

Here, the input image is of size N X N. f (x, y) is the intensity of the pixel in row x and column y, F (u, v) is the DCT coefficient in row u and column v of the DCT matrix.

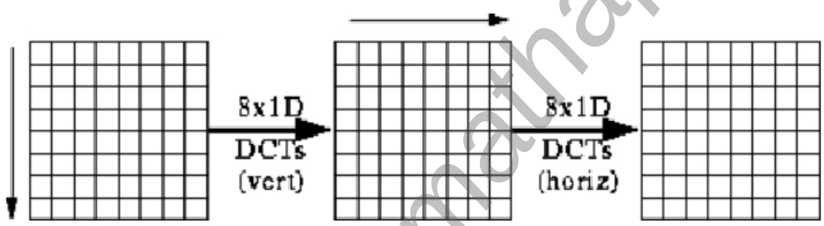
- The general equation for a 2D, IDCT is defined by the following equation:

$$IDCT = f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} \alpha(u)\alpha(v)\cdot F(u,v)\cdot\cos\left[\frac{\pi(2x+1)u}{2N}\right]\cdot\cos\left[\frac{\pi(2y+1)v}{2N}\right]$$

where $x,y = 0,1 - - N-1$

- Signal energy lies at low frequency in image, t appears in the upper left corner of the DCT. Compression can be achieved since the lower right values represent higher frequencies, and generally small enough to be neglected with little visible distortion.
- Also we can compute 2D DCT using 1D DCT as below
  - apply 1D DCT (Vertically) to Columns
  - apply 1D DCT (Horizontally) to resultant Vertical DCT above
  - or alternatively Horizontal to Vertical.

The equations are given by:



Example

Image Compression using DCT: JPEG Compression

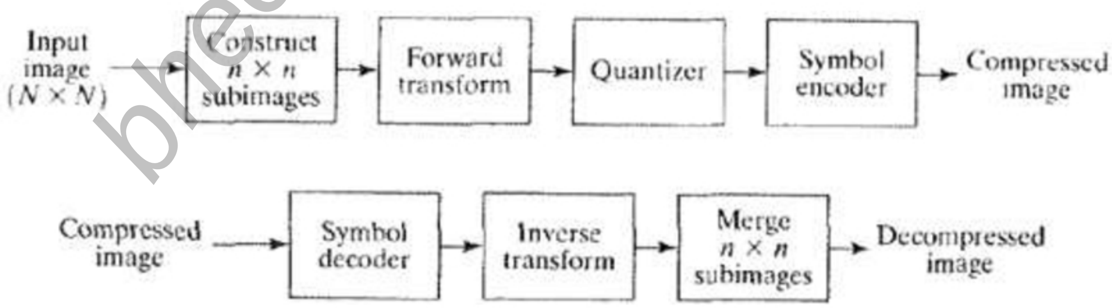
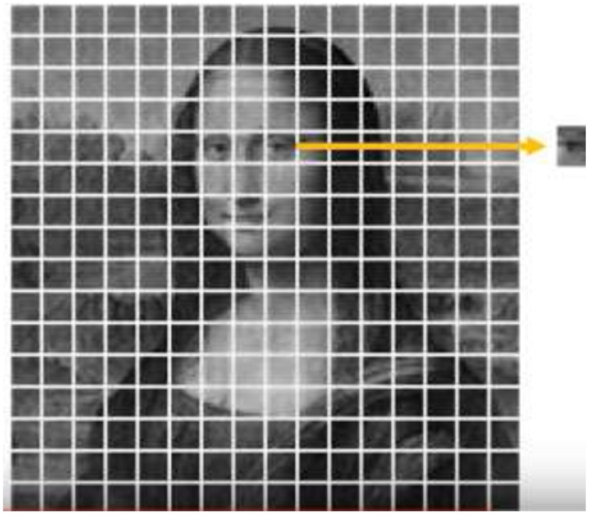


Fig: Steps In JPEG compression

- To perform the JPEG coding, an image is first subdivided into blocks of 8x8 pixels.



98	92	94	93	93	107	109	107
102	121	138	118	89	75	59	67
142	127	116	111	105	95	71	52
117	89	39	39	51	72	76	62
189	145	61	68	85	74	89	80
180	167	147	130	130	136	125	126
156	157	158	150	142	116	106	124
153	134	133	136	132	125	149	160

Fig:8 * 8 Block

- The Discrete Cosine Transform (DCT) is then performed on each block. For each block, the top-left coefficients F (0,0) correlates to lower frequency of the original image block, which is called DC coefficient. As we move away from the F(0,0) in all directions the DCT coefficients correlate to higher and higher frequencies, where F(7,7) corresponds to the highest frequency. DCT block for the above input block is shown below.

886	108	48	28	11	-2	-5	-2
-147	4	-27	-1	-18	3	6	-5
89	-55	-49	-58	-1	6	16	6
43	-4	18	5	-2	-7	-1	-2
-73	-57	63	36	8	8	-17	-1
-21	-27	-9	7	-4	-1	5	-1
53	-18	2	18	-5	7	4	1
17	31	-2	13	15	-2	-10	3

Fig : F(u,v)

- This generates 64 coefficients which are then quantized as below to reduce their magnitude.

$$F^Q(u,v) = \left[\frac{F(u,v)}{Q(u,v)}\right]$$

59	54	48	60	24	3	44	40
36	100	26	2	4	44	55	44
89	58	50	60	7	55	66	55
100	4	70	88	6	6	77	88
37	57	85	90	77	66	88	100
21	27	10	106	7	77	99	150
100	100	101	156	108	188	100	112
70	106	107	101	100	125	123	124

Fig: Q(u, v)

Since, lower frequency components constitute more image information, so quantizer is chosen such that upper left corner values are smaller than bottom right values. So the quantized output is obtained as

15	2	1	0	0	-1	0	0
-4	0	-1	-1	-1	0	0	0
1	-2	-1	-1	-1	0	0	0
0	-1	0	0	-1	-1	0	0
-2	-1	0	0	0	0	0	0
-1	-1	-1	0	-1	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Fig: Quantized 8*8 Block

- The compression is achieved in two stages; the first is during quantization and the second during the entropy coding (RLE and Huffman Encoding) process.
- JPEG decoding is the reverse process of coding.

Hadamard Transform

- The elements of a basic vectors of a Hadamard transform takes only the binary values i.e. +1 and -1 in its kernel matrix.
- Hadamard transform can be implemented in order of O[Nlog$_2$N] addition and subtraction.
- Hence this is very simple and easy to implement but unlike DCT it doesn't have good energy packing capacity.

⇒ The 1-D, forward Hadamard kernel is given by the relation

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

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

where summation in the exponent is performed in modulo 2 arithmetic and $b_k(z)$ is the kth bit in the binary representation of z.

⇒ The 1-D Hadamard transform is given by

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

where $N=2^n$, and u has values in the range 0, 1, 2, ..., N−1

⇒ The Hadamard kernel forms a matrix having orthogonal rows and columns

⇒ The inverse Hadamard kernel is given by

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

The 2-D Kernels are given by

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

and

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

⇒ The 2-D Hadamard transform pair

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

and

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

- The Hadamard transform kernel H_n is a N * N Matrix where N= 2^n, which can be recursively defined as

$$\mathbf{H_n} = \frac{1}{\sqrt{2}}\begin{pmatrix} \mathbf{H}_{n-1} & \mathbf{H}_{n-1} \\ \mathbf{H}_{n-1} & -\mathbf{H}_{n-1} \end{pmatrix}$$

- The Hadamard Matrix for the lowest order N=2 is

We have

$$H_n = \frac{1}{\sqrt{N}}\begin{bmatrix} H_{n-1} & H_{n-1} \\ H_{n-1} & -H_{n-1} \end{bmatrix}$$

then

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

$$H_1 = \frac{1}{\sqrt{2}}\begin{bmatrix} H_0 & H_0 \\ H_0 & -H_0 \end{bmatrix} \quad (\text{let } H_0 = 1)$$

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

Similarly,

For $N=4$, $N=2^n \Rightarrow n=2$

$$H_2 = \frac{1}{\sqrt{2}}\begin{bmatrix} H_1 & H_1 \\ H_1 & -H_1 \end{bmatrix}$$

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

- **Sequential ordering** is based on the concept of number of sign changes along the rows.
  H_3 can be computed as below

$$H_3 = \frac{1}{\sqrt{2}} \begin{bmatrix} H_2 & H_2 \\ H_2 & -H_2 \end{bmatrix}$$

$$= \frac{1}{2\sqrt{2}} \begin{array}{c} \\ \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix} \end{array} \begin{array}{c} \text{No. of sign change} \\ 0 \\ 7 \\ 3 \\ 4 \\ 1 \\ 6 \\ 2 \\ 5 \end{array}$$

- Sorting the position of rows on the basis of number of sign changes, we get

$$\frac{1}{2\sqrt{2}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix} \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array}$$

- So this will transform the data in increasing order of frequency.

Haar Transform

- Haar transform is effective in applications such as signal and image compression in electrical and computer engineering as it provides a simple and computationally efficient approach for analysing the local aspects of a signal.
- The Haar function $h_k(x)$ are defined on a continuous interval $x \epsilon [0,1]$ and for K=0, 1…..N-1, where $N=2^n$. The integer K can be uniquely decomposed as

$$K = 2^P + q - 1$$

Where $0 \leq P \leq n-1$; q=0,1 for p=0 and $1 \leq q \leq 2^p$ for $p \neq 0$

For example, when N=4, $N=2^n$ i.e. n=2 amd K=0,1,2,3

We have

K	0	1	2	3
P	0	0	1	1
q	0	1	1	2

Representing k by (p,q) the Haar function are defined as

$$h_0(x) = h_{0,0}(x) = \frac{1}{\sqrt{N}}$$

$$h_k(x) = h_{p,q}(x) = \frac{1}{\sqrt{N}} \begin{cases} 2^{p/2} & \frac{q-1}{2^p} \le x < \frac{q-1/2}{2^p} \\ -2^{p/2} & \frac{q-1/2}{2^p} \le x < \frac{q}{2^p} \\ 0 & \text{otherwise} \end{cases}$$

$x \in [0,1]$

Example:

Construct Haar matrix for N=4.

Construct Haar matrix for $N=4$ case.

Soln: We know

$H_r = H_k(x)$ where $k = 0, 1 - - - N-1$

such that

$k = 2^p + q - 1$

and $0 \le p \le n-1$

$1 \le q \le 2^p$ for $p \ne 0$

$q = 0, 1$ for $p = 0$

Here given $N=4$

$\therefore k = 0, 1, 2$ and 3 and $p = 0, 1$

k	0	1	2	3
p	0	0	1	1
q	0	1	1	2

$N = 2^n$

$4 = 2^n$

$2^2 = 2^n$

$\therefore n = 2$

and

$0 \le p \le 1$

Representing k by (p,q) the Haar function are defined as

$$h_0(x) = h_{00}(x) = \frac{1}{\sqrt{N}}$$

$$h_k(x) = h_{pq}(x) = \frac{1}{\sqrt{N}} \begin{cases} 2^{p/2} & \frac{q-1}{2^p} \le x < \frac{q-1/2}{2^p} \\ -2^{p/2} & \frac{q-1/2}{2^p} \le x < q/2^p \\ 0 & \text{otherwise} \end{cases}$$

$x \in [0,1]$

L - - - I

where $k = 0, 1, 2$ and 3 represents row
$x = 0, 1/4, 2/4$ and $3/4$ represent columns.

- Now For $k=0$: $p=0$ and $q=0$

$$h_0(x) = h_{0,0}(0) = \frac{1}{\sqrt{N}} = \frac{1}{\sqrt{4}} = 1/2$$

$$h_{0,0}(1/4) = \frac{1}{\sqrt{N}} = 1/2$$

$$h_{0,0}(2/4) = 1/\sqrt{N} = 1/2$$

$$h_{0,0}(3/4) = 1/\sqrt{N} = 1/2$$

- For $k=1$: $p=0$ and $q=1$
we have from equation I

$$h_1(x) = h_{0,1}(x) = \frac{1}{\sqrt{N}} \begin{cases} 2^{p/2} & 0 \le x < 1/2 \\ -2^{p/2} & 1/2 \le x < 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\therefore h_1(0) = h_{0,1}(0) = \frac{1}{\sqrt{4}} * 2^{0/2} = 1/2 * 1 = 1/2$$

$$h_1(1/4) = 1/2$$
$$h_1(2/4) = -1/2$$
$$h_1(3/4) = -1/2$$

- For $k=2$: $p=1$ and $q=1$
we have

$$h_2(x) = h_{1,1}(x) = \frac{1}{\sqrt{N}} \begin{cases} 2^{p/2} & 0 \le x \le 1/4 \\ -2^{p/2} & 1/4 \le x \le 1/2 \\ 0 & \text{otherwise} \end{cases}$$

$\therefore h_2(0) = h_{1,1}(0) = \frac{1}{\sqrt{4}} \times 2^{1/2} = \frac{\sqrt{2}}{2}$

$h_2(1/4) = h_{1,1}(1/4) = \frac{1}{\sqrt{4}} \times -2^{1/2} = -\frac{\sqrt{2}}{2}$

$h_2(2/4) = h_{1,1}(2/4) = \frac{1}{\sqrt{4}} \times 0 = 0$

$h_2(3/4) = h_{1,1}(3/4) = \frac{1}{\sqrt{4}} \times 0 = 0$

- For $k=3$: $p=1$ and $q=2$

We have

$$h_3(x) = h_{1,2}(x) = \frac{1}{\sqrt{N}} \begin{cases} 2^{p/2} & 1/2 \le x < 0.75 \\ -2^{p/2} & 0.75 \le x \le 1 \\ 0 & \text{otherwise.} \end{cases}$$

$\therefore h_3(0) = h_{1,2}(0) = \frac{1}{\sqrt{4}} \times 0 = 0$

$h_3(1/4) = h_{1,2}(1/4) = \frac{1}{\sqrt{4}} \times 0 = 0$

$h_3(2/4) = h_{1,2}(2/4) = \frac{1}{\sqrt{4}} \times 2^{1/2} = \frac{\sqrt{2}}{2}$

$h_3(3/4) = h_{1,2}(3/4) = \frac{1}{\sqrt{4}} \times -2^{1/2} = -\frac{\sqrt{2}}{2}$

$\therefore$ The Haar matrix for $N=4$ is as below

$$\begin{bmatrix} 1/2 & 1/2 & 1/2 & 1/2 \\ 1/2 & 1/2 & -1/2 & -1/2 \\ \sqrt{2}/2 & -\sqrt{2}/2 & 0 & 0 \\ 0 & 0 & \sqrt{2}/2 & -\sqrt{2}/2 \end{bmatrix} \longrightarrow \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ \sqrt{2} & -\sqrt{2} & 0 & 0 \\ 0 & 0 & \sqrt{2} & -\sqrt{2} \end{bmatrix}$$

Assignment Solution:

1. Write the Properties of Fourier Transform.

Properties of Fourier transform are:-

① Addition theorem. $F[f(x,y)+g(x,y)] = F(u,v)+G(u,v)$

② Shift theorem. $F[f(x-a,y-b)] = F(u,v)\,e^{-j2\pi(ua+vb)}$

③ Similarity theorem. $F[f(ax,by)] = \frac{1}{|ab|}F(u/a, v/b)$

④ Convolution theorem. $F[f(x,y)*g(x,y)] = F(u,v)\,G(u,v)$

⑤ Rayleigh's theorem. $\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} |f(x,y)|^2\,dx\,dy = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} |F(u,v)|^2\,du\,dv$

⑥ Evenness and oddness $f_e(x) \Rightarrow f(x) = f(-x)$ and $f_o(x) \Rightarrow f(-x) = -f(x)$

⑦ Power theorem. $\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y) * g^*(x,y)\,dx\,dy = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)\,G^*(u,v)\,du\,dv$

2. Write the properties of DCT.

① The cosine transform is real and orthogonal.

i.e. $C = C^* \Rightarrow C^{-1} = C^T$

where C^* = conjugate

C^T = transform

② The cosine transform is not real part of Unitary DFT.

$\therefore T^{-1} = T^{*T}$

③ It is a fast transform. The cosine transform of N-elements can be calculated in $O[N\log_2 N]$ operation via N-point FFT

④ It has excellent energy compaction for highly correlated data.

3. Write the properties of Hadamard Transform.
   a. The hadamard transform is real, symmetric and orthogonal.

$$\mathbf{H} = \mathbf{H}^* = \mathbf{H}^T = \mathbf{H}^{-1}$$

   b. It is fast transform and can be implemented in order of O[$N\log_2 N$] addition and subtraction.
   c. It has only binary valus +1 and -1 in its kernel matrix.
   d. It has very good energy compaction for highly co-related image.

4. Write the properties of Haar transform.

1. It is symmetric, separable unitary transform that uses Haar function for its basis.
2. It is orthogonal and real.

$$\underbrace{T^{-1} = T^{T}}_{\text{matrix inverse = matrix transpose}} \qquad \underbrace{H_r = H_r^{*}}_{\text{Conjugate of HT = HT}}$$

3. It is a fast transform and can be implemented in O(N) operations; where N = no. of samples.
4. It has poor energy compaction property.
5. It exists for $N = 2^n$ where 'n' is an integer.