Chapter 3

Image Transform: Frequency Domain Representation and Enhancement 10 CO3

3.1 Introduction , DFT and its properties, radix-2 algorithm(2- DFT), FFT algorithm: divide and conquer approach, Dissemination in Time(DIT)-FFT

3.2 Discrete Cosine Transform, Walsh Transform, Hadamard Transform, Haar Transform, Principal component Analysis(PCA/Hoteling Transform), Introduction to Wavelet Transform

3.3 Low Pass and High Pass Frequency domain filters: Ideal, Butterworth, Homomorphic filter

Self-Learning Topic: Discrete Sine Transform (DST)

Discrete Fourier Transform

- Discrete fourier transform for a finite sequence is given by:

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi \frac{kn}{N}}$$

- Evaluating summation for finite sequence for 0 to N-1.
- k/N corresponds to frequency F
- N corresponds to time for discrete signals

Discrete Fourier Transform

- Magnitude function
- $X(k) = \sqrt{Xr^2(k)+Xi^2(k)}$

- Phase function
- Angle $X(k)=\tan^{-1}[Xi(k)/Xr(k)]$

Inverse Discrete Fourier Transform

The inverse discrete Fourier transform of X(k) is defined as

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{j2\pi kn/N} \qquad 0 \le n \le N-1$$

Where K and n are in the range of 0 ,1,2……N-1 For example, if N=4, K= 0,1,2,3: N=0,1,2,3

Properties of DFT

1. Periodicity

Let x(n) and x(k) be the DFT pair then if

$x(n+N) = x(n)$ for all n then

$X(k+N) = X(k)$ for all k

Properties of DFT

2. Linearity

The linearity property states that if

$$x1(n) \xleftrightarrow[N]{\text{DFT}} X1(k) \quad \text{And}$$

$$x2(n) \xleftrightarrow[N]{\text{DFT}} X2(k) \quad \text{Then}$$

Then

$$a1\,x1(n) + a2\,x2(n) \xleftrightarrow{\text{DFT}} a1\,X1(k) + a2\,X2(k)\;N$$

DFT of linear combination of two or more signals is equal to the same linear combination of DFT of individual signals.

Properties of DFT

5. Circular Convolution

The Circular Convolution property states that if

$$x1(n) \xleftrightarrow[N]{DFT} X1(k) \quad \text{And}$$

$$x2(n) \xleftrightarrow[N]{DFT} X2(k) \quad \text{Then}$$

$$\text{Then } x1(n) \; \textcircled{N} \; x2(n) \xleftrightarrow{DFT} x1(k)\; x2(k)\; N$$

It means that circular convolution of x1(n) & x2(n) is equal to multiplication of their DFT s. Thus circular convolution of two periodic discrete signal with period N is given by

$$\mathbf{y(m) = \sum_{n=0}^{N-1} x1\,(n)\; x2\,(m-n)N} \qquad \mathbf{..........(4)}$$

Properties of DFT

6. Multiplication

The Multiplication property states that if

$$X1(n) \underset{N}{\overset{DFT}{\longleftrightarrow}} x1(k) \text{ And}$$

$$X2(n) \underset{N}{\overset{DFT}{\longleftrightarrow}} x2(k) \text{ Then}$$

$$\text{Then} \quad x1(n)\ x2(n) \overset{DFT}{\longleftrightarrow} 1/N \quad x1(k)\ N \ N \ x2(k)$$

It means that multiplication of two sequences in time domain results in circular convolution of their DFT s in frequency domain.

Properties of DFT

7. Time reversal of a sequence

The Time reversal property states that if

$$X(n) \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} x(k) \quad \text{And}$$

$$\text{Then } x((-n))_N = x(N-n) \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} x((-k))_N = x(N-k)$$

It means that the sequence is circularly folded its DFT is also circularly folded.

Properties of DFT

8. Circular Time shift

The Circular Time shift states that if

$$X(n) \xleftrightarrow[N]{DFT} x(k) \quad \text{And}$$

$$\text{Then} \quad x((n-l))_N \quad \xleftrightarrow[N]{DFT} \quad x(k)\ e^{-j2\prod k\, l / N}$$

Thus shifting the sequence circularly by „l samples is equivalent to multiplying its DFT by $e^{-j2\prod k\, l / N}$

Properties of DFT

- Conjugation
- DFT of complex conjugate of any sequence is equal to complex conjugate of DFT of that sequence , with sequence delayed by k samples in frequency domain.
- DFT{x(n)}=X(k)
- DFT{X*(n)}=X*(N-k)

Fast Fourier Transform

- FFT is method of computing DFT with reduced number of calculations.
- Computational efficiency is achieved if we adopt a divide and conquer approach.
- Approach is based on decomposition of N point DFT into successively smaller DFTs.

Radix-2 FFT

- N=2^m
- m is number of stages
- N point sequence is decimated into 2 point , further from a 2 point sequence 4 point DFT can be computed and so on.

Phase or Twiddle Factor

- Complex valued factor in DFT is defined as W_N

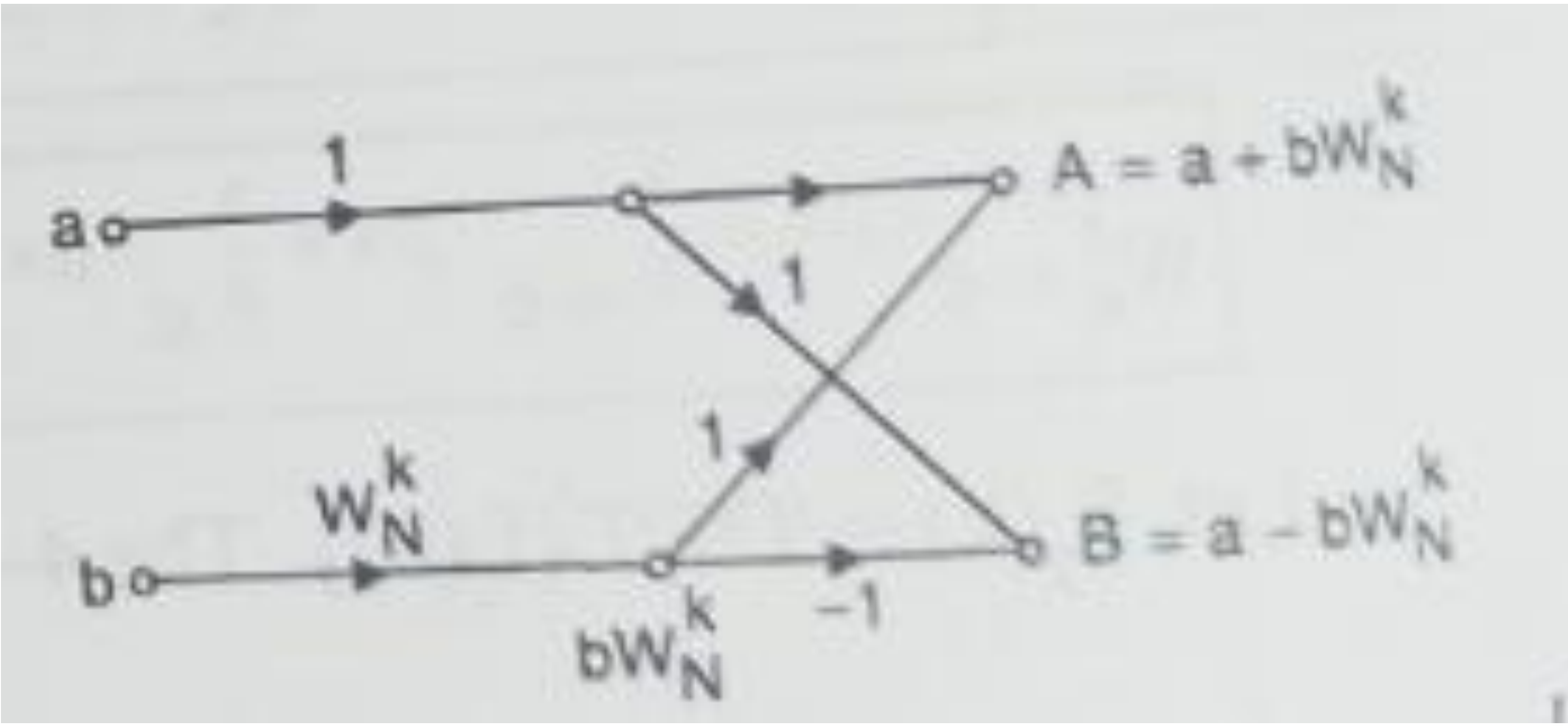
also called as twiddle factor.

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi \frac{kn}{N}}$$

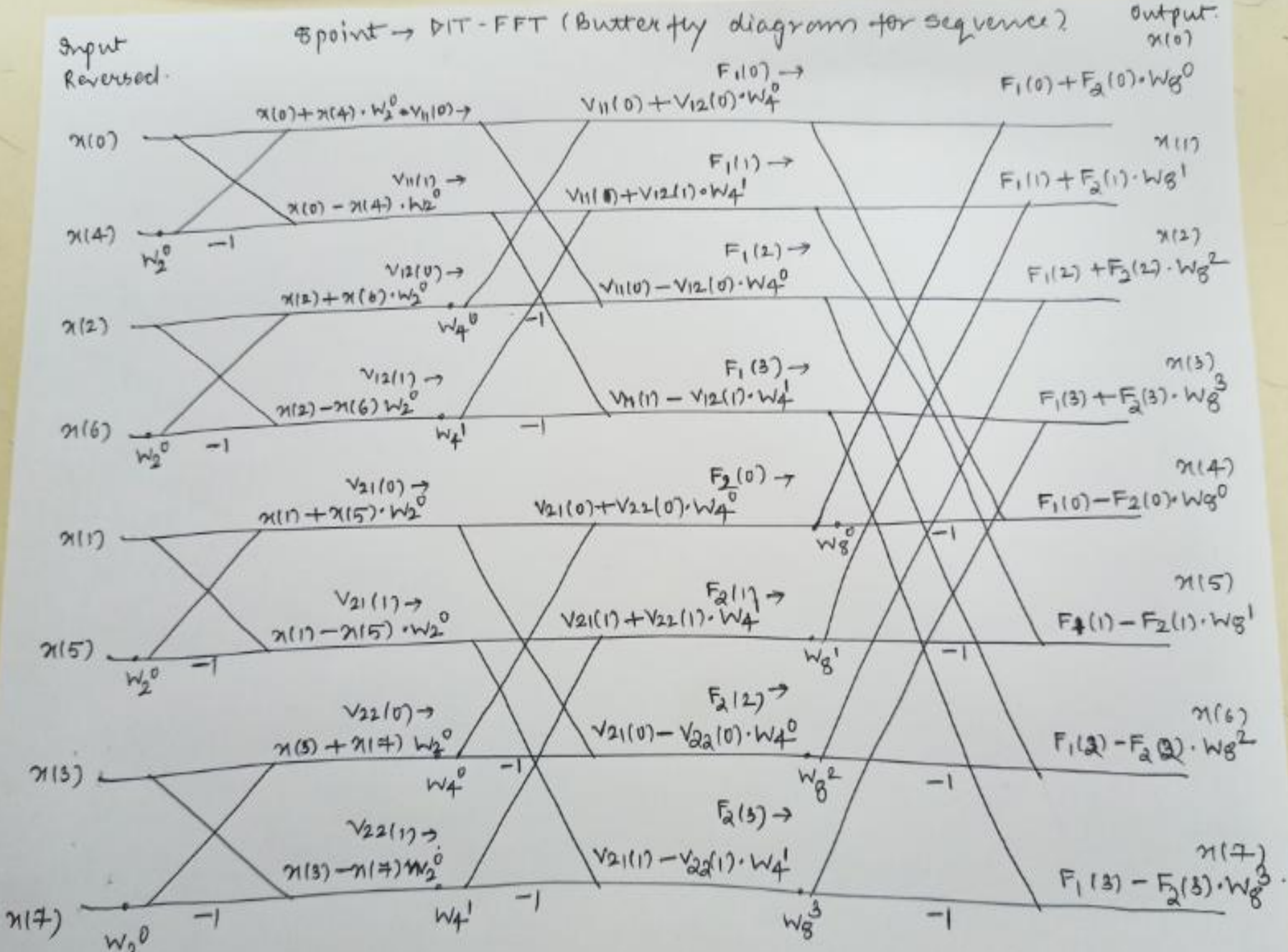
- Twiddle factor is given as
- W = e^(-j2*pi)
- Twiddle factor can be multiplied or divided by any integer.

DIT-FFT

- Steps to find DFT using Radix 2 DIT FFT:



8point → DIT - FFT (Butterfly diagram for sequence)



2 point | 4 point | 8

$W_2^0 = 1$ $\quad\quad$ $W_4^0 = 1$

$W_4^1 = -j$

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

2.

$$W_2^0 = e^0 = 1$$

$$W_4^0 = e^0 = 1$$

$$W_4^1 = e^{\frac{-j2\pi(1)}{4}}$$

$$e^{-j2\pi/2}$$

$\cos \pi/2 - j\sin \pi/2$

4 point

$W_4^0 = 1$

$W_4^1 = -j$

8 point

$W_8^0 = 1$

$W_8^1 = \frac{1}{\sqrt{2}} - j\frac{1}{\sqrt{2}}$

$W_8^2 = -j$

$W_8^3 = -\frac{1}{\sqrt{2}} - j\frac{1}{\sqrt{2}}$

Table 5.2 : Comparison of Number of Comput

Number of points N	Direct Computation		Radix-2 FFT	
	Complex additions $N(N-1)$	Complex Multiplications N^2	Complex additions $N\log_2 N$	Complex Multiplications $(N/2)\log_2 N$
$4\,(=2^2)$	12	16	$4 \times \log_2 2^2 = 4 \times 2 = 8$	$\frac{4}{2} \times \log_2 2^2 = \frac{4}{2} \times 2 = 4$
$8\,(=2^3)$	56	64	$8 \times \log_2 2^3 = 8 \times 3 = 24$	$\frac{8}{2} \times \log_2 2^3 = \frac{8}{2} \times 3 = 12$
$16\,(=2^4)$	240	256	$16 \times \log_2 2^4 = 16 \times 4 = 64$	$\frac{16}{2} \times \log_2 2^4 = \frac{16}{2} \times 4 = 32$
$32\,(=2^5)$	992	1,024	$32 \times \log_2 2^5 = 32 \times 5 = 160$	$\frac{32}{2} \times \log_2 2^5 = \frac{32}{2} \times 5 = 80$
$64\,(=2^6)$	4,032	4,096	$64 \times \log_2 2^6 = 64 \times 6 = 384$	$\frac{64}{2} \times \log_2 2^6 = \frac{64}{2} \times 6 = 192$
$128\,(=2^7)$	16,256	16,384	$128 \times \log_2 2^7 = 128 \times 7 = 896$	$\frac{128}{2} \times \log_2 2^7 = \frac{128}{2} \times 7 = 448$

Image Transforms

- Image transforms are extensively used in image processing and image analysis.
- Transform is basically a mathematical tool, which allows us to move from one domain to another domain (time domain to the frequency domain).
- migrate from one domain to another domain is to perform the task at hand in an easier manner.

Image Transforms

- Image transforms are useful for fast computation of convolution and correlation.

- The transforms do not change the information content present in the signal.

- Transforms play a significant role in various image-processing applications such as image analysis, image enhancement, image filtering and image compression.

NEED FOR TRANSFORM

(i) Mathematical Convenience

Convolution in time domain $\Longleftrightarrow$ Multiplication in the frequency domain

The complex convolution operation in time domain is equal to simple multiplication operation in the frequency domain.

NEED FOR TRANSFORM

(ii) *To Extract more Information*

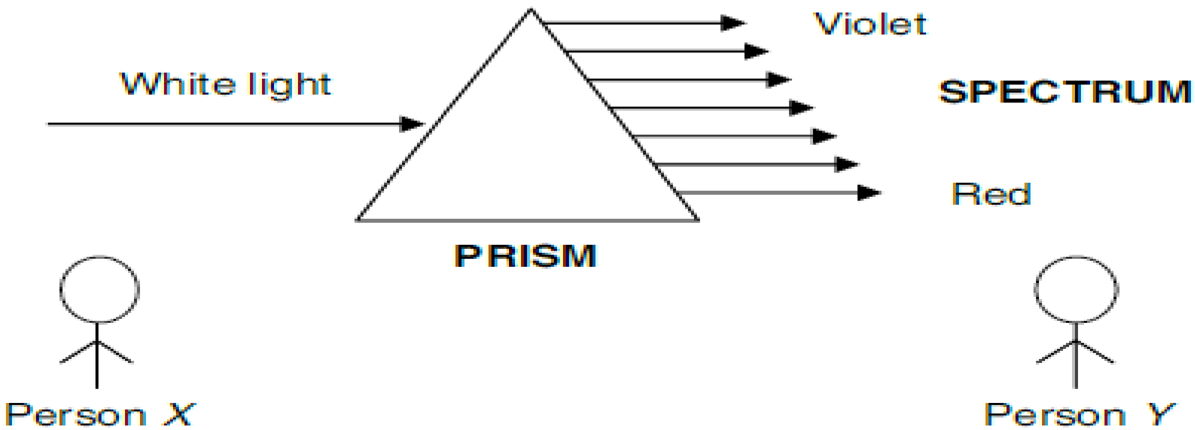


Fig. 4.1 *Spectrum of white light*

Fig. 4.2 *Concept of transformation*

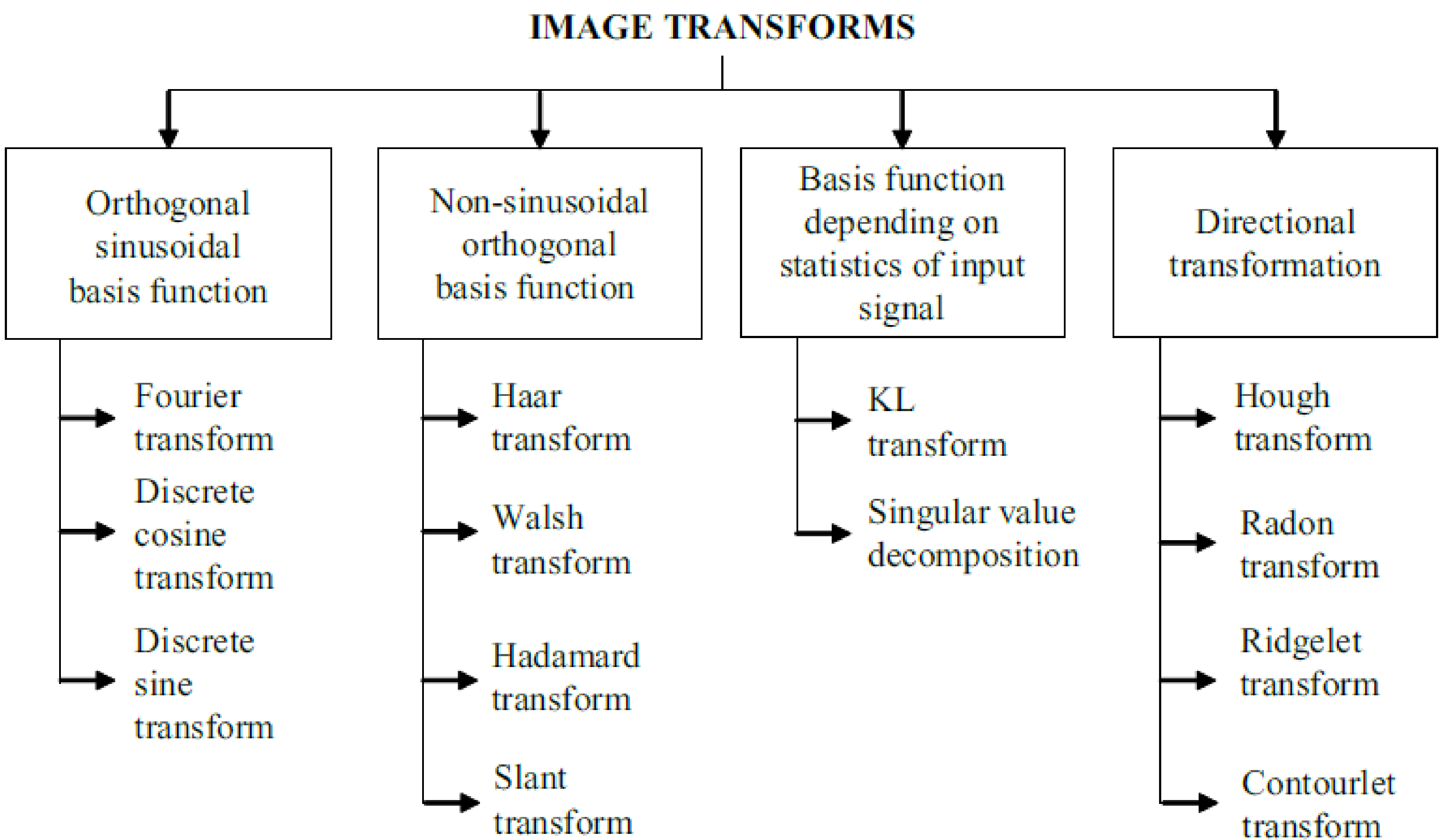


IMAGE TRANSFORMS
Orthogonal sinusoidal basis function
Non-sinusoidal orthogonal basis function
Basis function depending on statistics of input signal
Directional transformation
Fourier transform
Discrete cosine transform
Discrete sine transform
Haar transform
Walsh transform
Hadamard transform
Slant transform
KL transform
Singular value decomposition
Hough transform
Radon transform
Ridgelet transform
Contourlet transform

Image Transform

1) Orthogonal sinusoidal basis function

- One of the most powerful transforms with orthogonal sinusoidal basis function is the Fourier transform.
- The transform which is widely used in the field of **image compression** is discrete cosine transform.

Image Transform

2) Non-sinusoidal orthogonal basis function

- The transforms whose basis functions are non-sinusoidal in nature.

- One of the important advantages of wavelet transform is that signals (images) can be **represented in different resolutions.**

Image Transform

3) Basis function depending on statistics of input signal

- The transform whose basis function depends on the statistics of input signal are KL transform and singular value decomposition.

- The KL transform is considered to be the best among all linear transforms with respect to **energy compaction**.

Image Transform

4) Directional transformation

- The transforms whose basis functions are effective in **representing the directional information of a signal** include Hough transform, Radon transform, Ridgelet transform and Contourlet transform.

Need for transform

- The need for transform is most of the signals or images are time domain signal (ie) signals can be measured with a function of time.
- This representation is not always best.
- For most image processing applications anyone of the mathematical transformation are applied to the signal or images to obtain further information from that signal.

Unitary Transform A discrete linear transform is unitary if its transform matrix conforms to the unitary condition

$$A \times A^H = I \tag{4.11}$$

where A = transformation matrix, A^H represents Hermitian matrix.

$$A^H = A^{*T}$$

I = identity matrix

When the transform matrix A is unitary, the defined transform is called unitary transform.

Orthogonal DFT

- If A is unitary and has only real elements , then it is orthogonal matrix

- A is orthogonal if $A.A' = I$

Image Transform

1) Check whether the DFT matrix is unitary or not.

Solution

Step 1 **Determination of the matrix A**

Finding 4-point DFT (where $N = 4$)

The formula to compute a DFT matrix of order 4 is given below.

$$X(K) = \sum_{n=0}^{3} x(n) e^{-j\frac{2\pi}{4}kn} \quad \text{where } k = 0, 1\ldots, 3$$

1. *Finding* $X(0)$

$$X(0) = \sum_{n=0}^{3} x(n) = x(0) + x(1) + x(2) + x(3)$$

2. *Finding* $X(1)$

$$X(1)=\sum_{n=0}^{3}x(n)e^{-j\frac{\pi}{2}n}$$

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

$$X(1)=x(0)-jx(1)-x(2)+jx(3)$$

3. *Finding* $X(2)$

$$X(2)=\sum_{n=0}^{3}x(n)e^{-j\pi n}$$

$$=x(0)+x(1)e^{-j\pi}+x(2)e^{-j2\pi}+x(3)e^{-j3\pi}$$

$$X(2)=x(0)-x(1)+x(2)-x(3)$$

4. *Finding* $X(3)$

$$X(3) = \sum_{n=0}^{3} x(n) e^{-j\frac{3\pi}{2}n}$$

$$= x(0) + x(1)e^{-j\frac{3\pi}{2}} + x(2)e^{-j3\pi} + x(3)e^{-j\frac{9\pi}{2}}$$

$$\boldsymbol{X}(3) = x(0) + jx(1) - x(2) - jx(3)$$

Collecting the coefficients of $X(0)$, $X(1)$, $X(2)$ and $X(3)$, we get

$$X[k] = A = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

Step 2 Computation of A^H

To determine A^H, first determine the conjugate and then take its transpose.

$$A \xrightarrow{\text{Conjugate}} A^* \xrightarrow{\text{Transpose}} A^H$$

Step 2a Computation of conjugate A^*

$$A^* = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix}$$

Step 2b Determination of transpose of A^*

$$\left(A^*\right)^T = A^{\mathrm{H}} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix}$$

***Step 3* Determination of $A \times A^H$**

$$A\times A^{\mathrm{H}} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} = \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix} = 4\times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The result is the identity matrix, which shows that Fourier transform satisfies unitary condition.

Unitary Transform

- unitary transformation preserves the **signal energy**

- Most unitary transforms pack a **large fraction of the energy of the image into relatively few of the transform coefficients.**

- Few of the transform coefficients have significant values and these are the coefficients that are close to the origin

DFT of Image Matrix (2D)

Example 4.4 *Compute the 2D DFT of the 4 × 4 grayscale image given below.*

$$f[m,n] = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Solution The 2D DFT of the image $f[m, n]$ is represented as $F[k, l]$.

$$F[k,l] = \text{kernel} \times f[m,n] \times (\text{kernel})^T$$

The kernel or basis of the Fourier transform for $N = 4$ is given by

$$\text{The DFT basis for } N = 4 \text{ is given by} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

DFT of Image Matrix (2D)

$$F(k,l) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$F(k,l) = \begin{bmatrix} 4 & 4 & 4 & 4 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} = \begin{bmatrix} 16 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Example 4.5 *Compute the inverse 2D DFT of the transform coefficients given by*

$$F[k,l] = \begin{bmatrix} 16 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Solution The inverse 2D DFT of the Fourier coefficients $F[k, l]$ is given by $f[m, n]$ as

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

In this example, $N = 4$

$$f[m,n] = \frac{1}{16} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \times \begin{bmatrix} 16 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$f[m,n] = \frac{1}{16} \times \begin{bmatrix} 16 & 0 & 0 & 0 \\ 16 & 0 & 0 & 0 \\ 16 & 0 & 0 & 0 \\ 16 & 0 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

$$f[m,n] = \frac{1}{16} \times \begin{bmatrix} 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \end{bmatrix}$$

$$f[m,n]=\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Discrete Cosine Transform (DCT)

- The discrete cosine transforms are the members of a family of real-valued discrete sinusoidal unitary transforms.

- A discrete cosine transform consists of a set of basis vectors that are sampled cosine functions.

- DCT is a technique for converting a signal into elementary frequency components and it is widely used in image compression.

Thus, the kernel of a one-dimensional discrete cosine transform is given by

$$X[k] = \alpha(k)\sum_{n=0}^{N-1} x[n]\cos\left\{\frac{(2n+1)\pi k}{2N}\right\} \text{, where } 0 \le k \le N-1$$

$$\alpha(k) = \sqrt{\frac{1}{N}} \text{ if } k = 0$$

$$\alpha(k) = \sqrt{\frac{2}{N}} \text{ if } k \neq 0$$

Example 4.10 *Compute the discrete cosine transform (DCT) matrix for N = 4.*

Solution The formula to compute the DCT matrix is given by

$$X[k] = \alpha(k)\sum_{n=0}^{N-1} x[n]\cos\left\{\frac{(2n+1)\pi k}{2N}\right\}, \text{where } 0 \leq k \leq N-1$$

where

$$\alpha(k) = \sqrt{\frac{1}{N}} \text{ if } k = 0$$

$$\alpha(k) = \sqrt{\frac{2}{N}} \text{ if } k \neq 0$$

In our case, the value of $N = 4$. Substituting $N = 4$ in the expression of $X[k]$ we get

$$X[k] = \alpha(k)\sum_{n=0}^{3} x[n]\cos\left[\frac{(2n+1)\pi k}{8}\right]$$

Substituting $k = 0$ in Eq. (4.82), we get

$$X[0] = \sqrt{\frac{1}{4}} \times \sum_{n=0}^{3} x[n]\cos\left[\frac{(2n+1)\pi \times 0}{8}\right]$$

$$= \frac{1}{2} \times \sum_{n=0}^{3} x[n]\cos(0) = \frac{1}{2} \times \sum_{n=0}^{3} x[n] \times 1$$

$$= \frac{1}{2} \times \sum_{n=0}^{3} x[n]$$

$$= \frac{1}{2} \times \{x(0) + x(1) + x(2) + x(3)\}$$

$$X[0] = \frac{1}{2}x(0) + \frac{1}{2}x(1) + \frac{1}{2}x(2) + \frac{1}{2}x(3)$$

Substituting $k = 1$ in Eq. (4.82), we get

$$X[1] = \sqrt{\frac{2}{4}} \times \sum_{n=0}^{3} x[n] \cos\left[\frac{(2n+1)\pi \times 1}{8}\right]$$

$$= \sqrt{\frac{1}{2}} \times \sum_{n=0}^{3} x[n] \cos\left[\frac{(2n+1)\pi}{8}\right]$$

$$= \sqrt{\frac{1}{2}} \times \left\{x(0) \times \cos\left(\frac{\pi}{8}\right) + x(1) \times \cos\left(\frac{3\pi}{8}\right) + x(2) \times \cos\left(\frac{5\pi}{8}\right) + x(3) \times \cos\left(\frac{7\pi}{8}\right)\right\}$$

$$= 0.707 \times \{x(0) \times 0.9239 + x(1) \times 0.3827 + x(2) \times (-0.3827) + x(3) \times (-0.9239)\}$$

$$X(1) = 0.6532x(0) + 0.2706x(1) - 0.2706x(2) - 0.6532x(3)$$

Substituting $k = 2$ in Eq. (4.82), we get

$$X(2) = \sqrt{\frac{2}{4}}\sum_{n=0}^{3} x[n]\cos\left[\frac{(2n+1)\pi\times 2}{8}\right]$$

$$= \sqrt{\frac{1}{2}}\times\sum_{n=0}^{3} x[n]\cos\left[\frac{(2n+1)\pi}{4}\right]$$

$$= \sqrt{\frac{1}{2}}\times\left\{x(0)\times\cos\left(\frac{\pi}{4}\right)+x(1)\times\cos\left(\frac{3\pi}{4}\right)+x(2)\times\cos\left(\frac{5\pi}{4}\right)+x(3)\times\cos\left(\frac{7\pi}{4}\right)\right\}$$

$$= 0.7071\times\{x(0)\times 0.7071 + x(1)\times(-0.7071) + x(2)\times(-0.7071) + x(3)\times(0.7071)\}$$

$$X(2) = 0.5x(0) - 0.5x(1) - 0.5x(2) + 0.5x(3)$$

Substituting $k = 3$ in Eq. (4.82), we get

$$X(3) = \sqrt{\frac{2}{4}}\sum_{n=0}^{3} x[n]\cos\left[\frac{(2n+1)\pi\times 3}{8}\right]$$

$$= \sqrt{\frac{1}{2}}\times\sum_{n=0}^{3} x[n]\cos\left[\frac{(2n+1)\times 3\pi}{8}\right]$$

$$= \sqrt{\frac{1}{2}}\times\left\{x(0)\times\cos\left(\frac{3\pi}{8}\right)+x(1)\times\cos\left(\frac{9\pi}{8}\right)+x(2)\times\cos\left(\frac{15\pi}{8}\right)+x(3)\times\cos\left(\frac{21\pi}{8}\right)\right\}$$

$$= 0.7071\times\left\{x(0)\times 0.3827 + x(1)\times(-0.9239) + x(2)\times(0.9239) + x(3)\times(-0.3827)\right\}$$

$$X(3) = 0.2706x(0) - 0.6533x(1) + 0.6533x(2) - 0.2706x(3)$$

Collecting the coefficients of $x(0)$, $x(1)$, $x(2)$ and $x(3)$ from $X(0)$, $X(1)$, $X(2)$ and $X(3)$, we get

$$\begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ X[3] \end{bmatrix} = \begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ 0.6532 & 0.2706 & -0.2706 & -0.6532 \\ 0.5 & -0.5 & -0.5 & 0.5 \\ 0.2706 & -0.6533 & 0.6533 & -0.2706 \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \end{bmatrix}$$

IDCT

$$x[n] = \alpha(k) \sum_{k=0}^{N-1} X[k] \cos\left[\frac{(2n+1)\pi k}{2N}\right], 0 \leq n \leq N-1$$

Symmetric and Asymmetric

- **Transformation Matrix – T**
- **Image Matrix – f**
- **If Transformation matrix is symmetric:**
- **F = TfT**

- **If Transformation matrix is asymmetric:**
- **F = TfT'**

Hadamard Transform

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

The Hadamard matrix of order $2N$ can be generated by Kronecker product operation:

$$H_{2N} = \begin{bmatrix} H_N & H_N \\ H_N & -H_N \end{bmatrix}$$

Substituting $N = 2$ in Eq. (4.59), we get

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

Similarly, substituting $N = 4$ in Eq. (4.59), we get

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

Hadamard Transform

- Hadamard of Sequence
- **X[n]={H(N).x(n)}**
- Inverse Hadamard of Sequence
- **X[n] = 1/N{H(N).x(n)}**

- Hadamard of image (2D)
- Hadamard is symmetric transform
- **F = TfT**

$$H_8 = \begin{bmatrix} H_4 & H_4 \\ H_4 & -H_4 \end{bmatrix} = \begin{array}{c} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix} \end{array} \begin{array}{c} 0 \\ 7 \\ 3 \\ 4 \\ 1 \\ 6 \\ 2 \\ 5 \end{array} \Rightarrow$$

$$W(8) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix} \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array}$$

Walsh Transform

$$W(8) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix},$$

Haar Transform

Haar Transform

- For N=2 dan N=4:

$$\mathbf{Hr}_2 = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad \mathbf{Hr}_4 = \frac{1}{\sqrt{4}}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -1 \\ \sqrt{2} & -\sqrt{2} & 0 & 0 \\ 0 & 0 & \sqrt{2} & -\sqrt{2} \end{bmatrix}$$

- Properties of Haar Transform:
  1. Real and orthogonal: $\mathbf{Hr} = \mathbf{Hr}^*$ dan $\mathbf{Hr}^{-1} = \mathbf{Hr}^T$
  2. Very fast transform : O(N) operation on Nx1 vector.
  3. Poor energy compaction for images

Haar Transform

$$\mathbf{H}_3 = \frac{1}{\sqrt{8}} \begin{array}{c} \left[\begin{array}{cccccccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} \\ 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 \end{array}\right] \end{array} \begin{array}{l} h_0 \\ h_1 \\ h_{2,1} \\ h_{2,2} \\ h_{3,1} \\ h_{3,2} \\ h_{3,3} \\ h_{3,4} \end{array}$$

IMAGE ENHANCEMENT IN THE FREQUENCY DOMAIN

- If we multiply each element of the Fourier coefficient by a suitably chosen weighting function then we can **accentuate certain frequency components and attenuate others.**
- This **selective enhancement or suppression of frequency components** is termed Fourier filtering or frequency domain filtering.
- The spatial representation of image data **describes the adjacency relationship between the pixels.**
- On the other hand, the frequency domain **representation clusters the image data according to their frequency distribution.**
- In frequency domain filtering, the **image data is dissected into various spectral bands**, where each band depicts a specific range of details within the image.
- The **process of selective frequency inclusion or exclusion is termed frequency domain filtering.**
- It is possible to perform filtering in the frequency domain by specifying the frequencies that should be kept and the frequencies that should be discarded.

IMAGE ENHANCEMENT IN THE FREQUENCY DOMAIN

$$\boxed{\textbf{Convolution in spatial domain = Multiplication in the frequency domain}}$$

$$\text{Filtering in spatial domain} = f(m,n) * h(m,n)$$

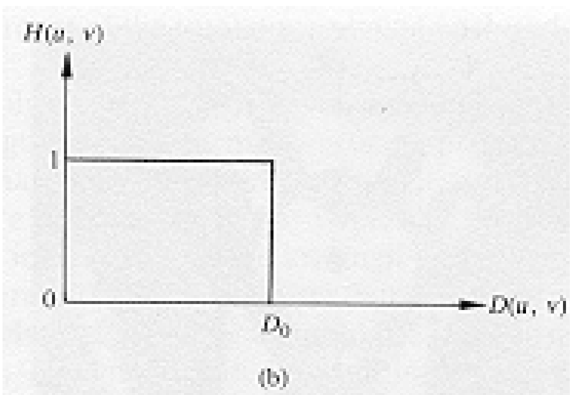
$$\text{Filtering in the frequency domain} = F(k,l) \times H(k,l)$$

f(m,n) – original image
F(k,l) – FT of original image
h(m,n) – Filtering mask
H(k,l) – FT of filtering mask

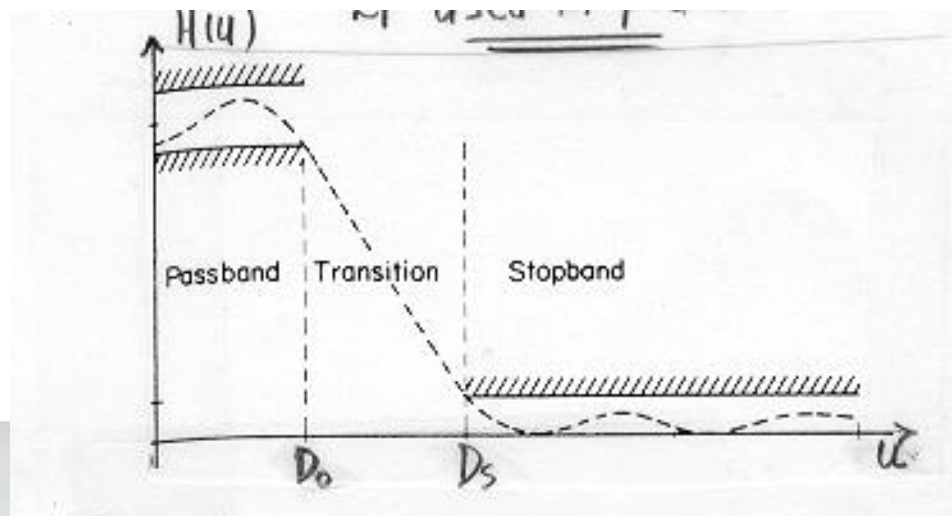
Low-pass (LP) filtering

- Preserves low frequencies, attenuates high frequencies.

ideal



in practice



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

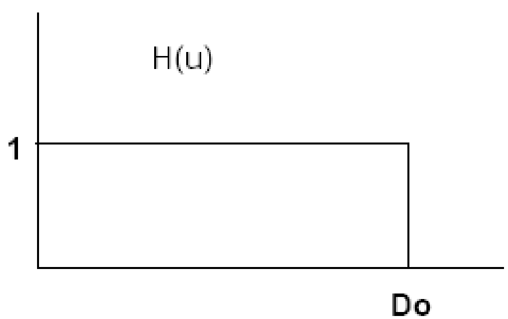
D_0: cut-off frequency

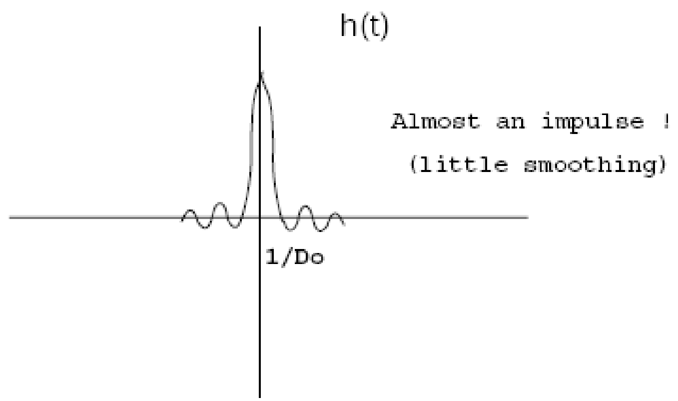
Low-pass (LP) filtering

- D0 is the cut-off frequency of the low-pass filter.
- The cut-off frequency determines the amount of frequency components passed by the filter.
- The smaller the value of D0 , the greater the number of image components eliminated by the filter.
- The value of D0 is chosen in such a way that the components of interest are passed through.

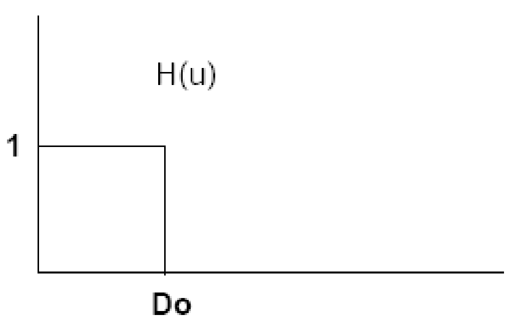
How does D_0 control smoothing? (cont'd)

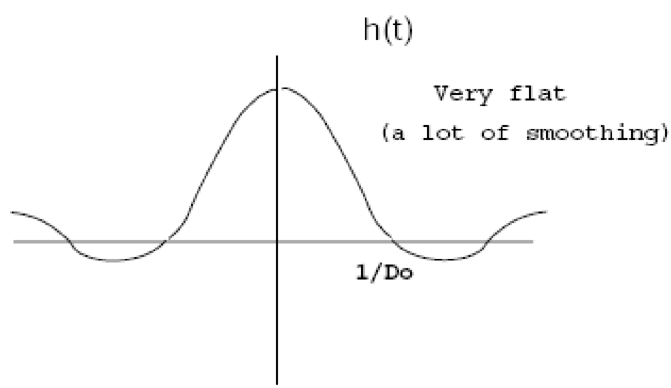
- D_0 controls the amount of blurring





r=78 (99%)



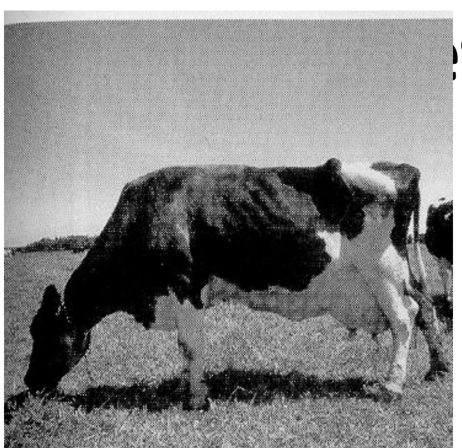


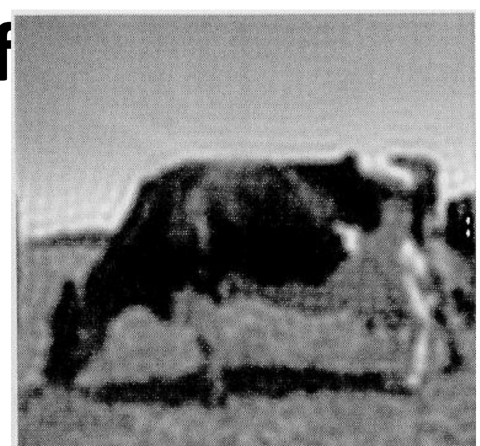
r=8 (90%)

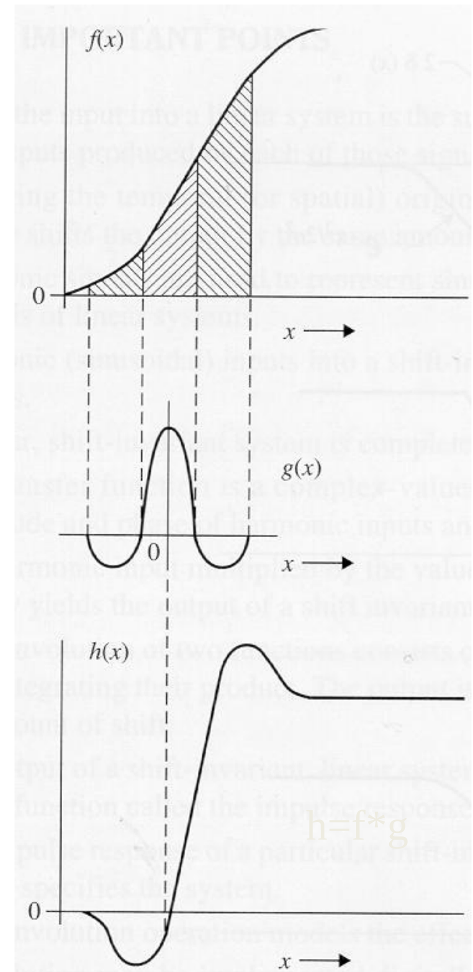
Ringing Effect

- Sharp cutoff frequencies produce an overshoot of image features whose frequency is close to the cutoff frequencies

eff







Low Pass (LP) Filters

- Ideal low-pass filter (ILPF)
- Butterworth low-pass filter (BLPF)
- Gaussian low-pass filter (GLPF)

Butterworth Low-pass Filter The transfer function of a two-dimensional Butterworth low-pass filter is given by

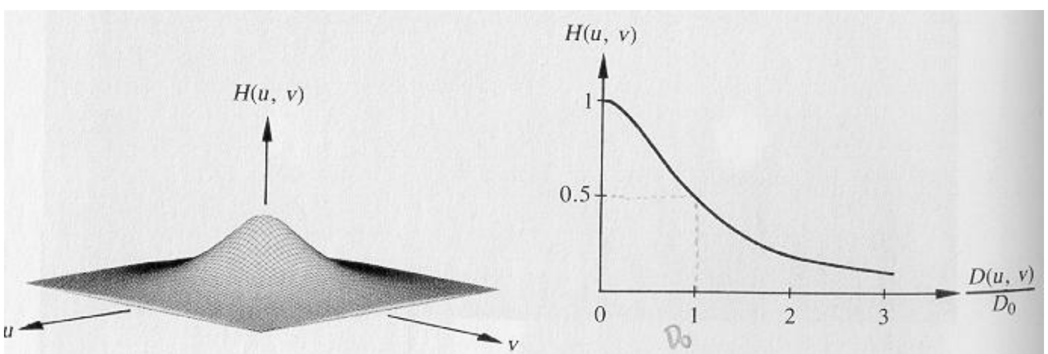
$$H(k,l)=\frac{1}{1+\left[\frac{\sqrt{k^2+l^2}}{D_0}\right]^{2n}} \tag{5.18}$$

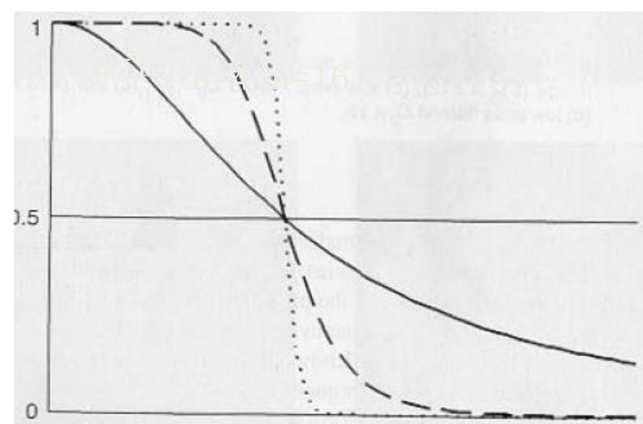
In the above expression, n refers to the filter order and D_0 represents the cut-off frequency.

Butterworth LP filter (BLPF)

- In practice, we use filters that <u>attenuate high frequencies smoothly</u> (e.g., **Butterworth** LP filter) → less ringing effect

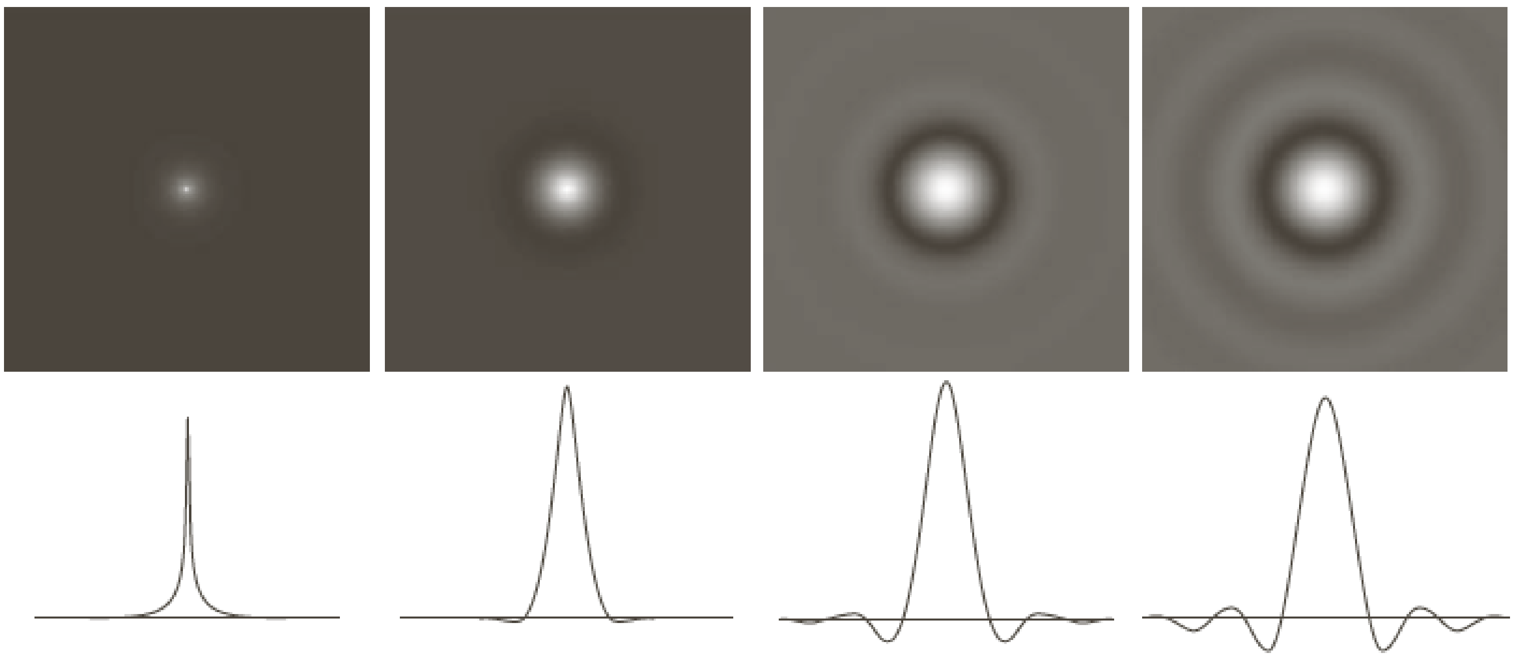
$$H(u,v) = \frac{1}{1 + [\sqrt{u^2 + v^2}/D_0]^{2n}}$$





Spatial Representation of BLPFs

n=1 n=2 n=5 n=20



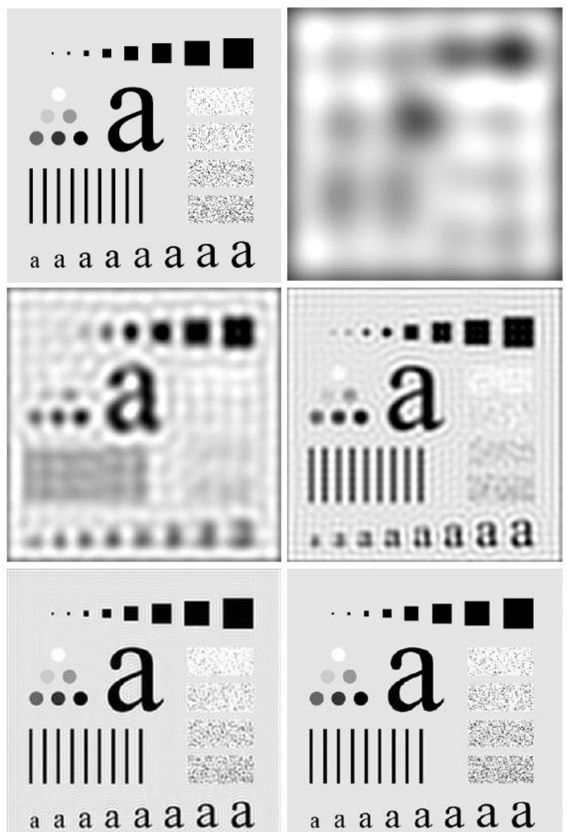
a b c d

FIGURE 4.46 (a)–(d) Spatial representation of BLPFs of order 1, 2, 5, and 20, and corresponding intensity profiles through the center of the filters (the size in all cases is 1000 × 1000 and the cutoff frequency is 5). Observe how ringing increases as a function of filter order.

Comparison: Ideal LP and BLPF

ILPF

D_0=10, 30, 60, 160, 460



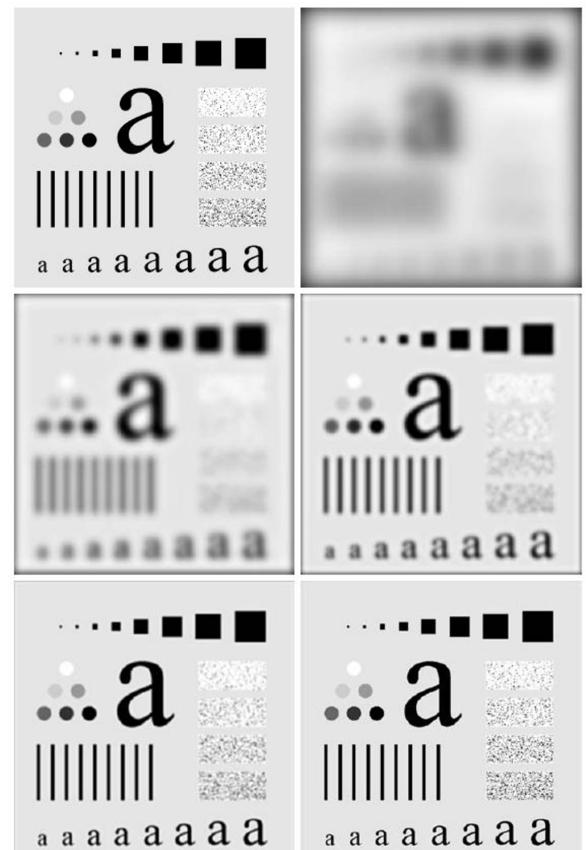
a b
c d
e f

FIGURE 4.42 (a) Original image. (b)–(f) Results of filtering using ILPFs with cutoff frequencies set at radii values 10, 30, 60, 160, and 460, as shown in Fig. 4.41(b). The power removed by these filters was 13, 6.9, 4.3, 2.2, and 0.8% of the total, respectively.

BLPF

D_0=10, 30, 60, 160, 460

n=2



a b
c d
e f

FIGURE 4.45 (a) Original image. (b)–(f) Results of filtering using BLPFs of order 2, with cutoff frequencies at the radii shown in Fig. 4.41. Compare with Fig. 4.42.

Gaussian LP filter (GLPF)

Gaussian Lowpass Filters (GLPF) in two dimensions is given

$$H(u,v) = e^{-(u^2+v^2)/2\sigma^2}$$

By letting $\sigma = D_0$

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

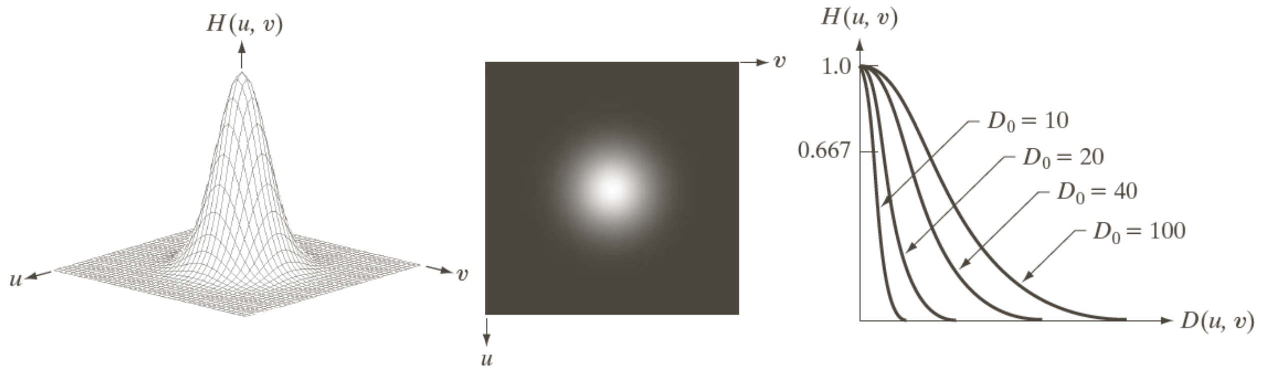
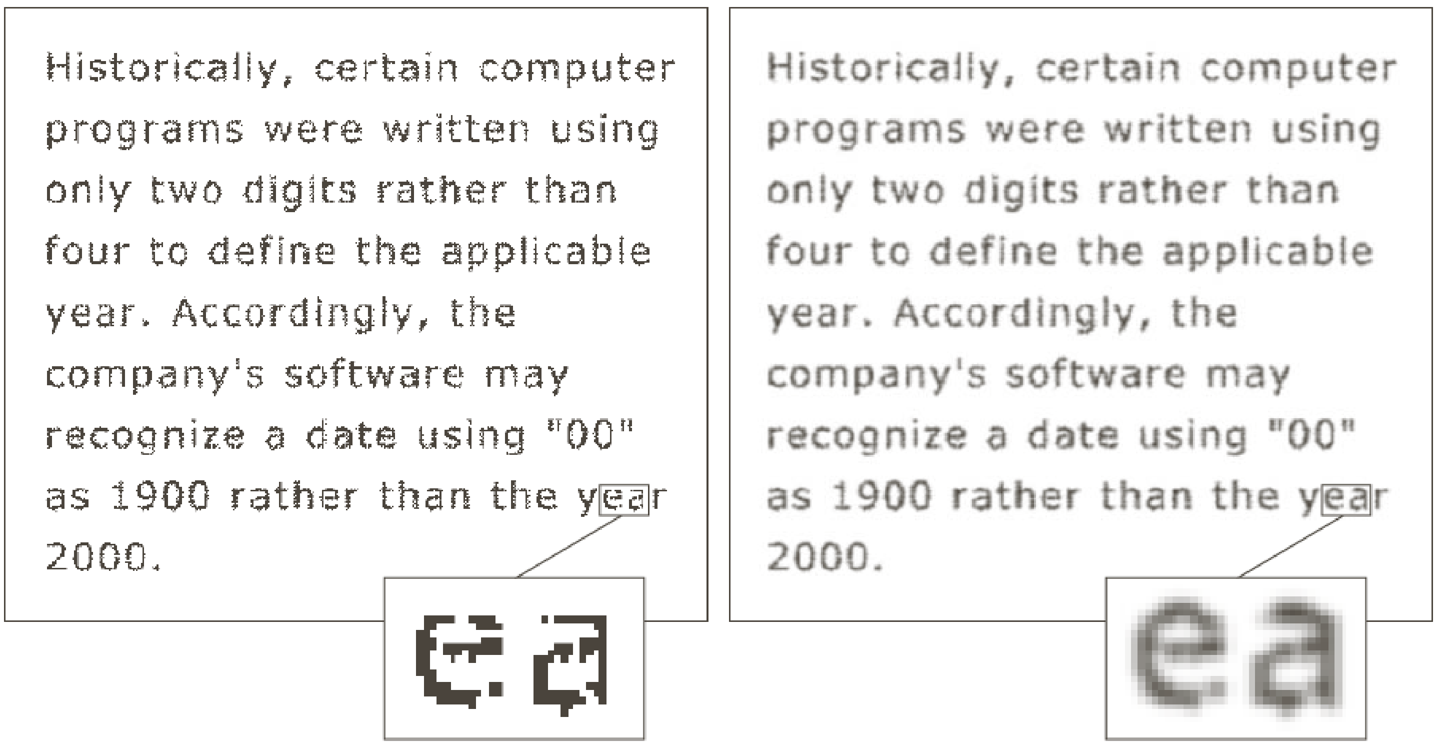


FIGURE 4.47 (a) Perspective plot of a GLPF transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections for various values of D_0.

Example: smoothing by GLPF (1)



a b

FIGURE 4.49 (a) Sample text of low resolution (note broken characters in magnified view). (b) Result of filtering with a GLPF (broken character segments were joined).

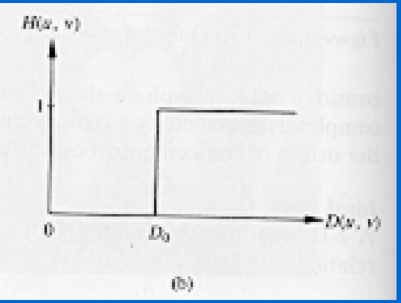
Examples of smoothing by GLPF (2)

FIGURE 4.50 (a) Original image (784 × 732 pixels). (b) Result of filtering using a GLPF with $D_0 = 100$. (c) Result of filtering using a GLPF with $D_0 = 80$. Note the reduction in fine skin lines in the magnified sections in (b) and (c).

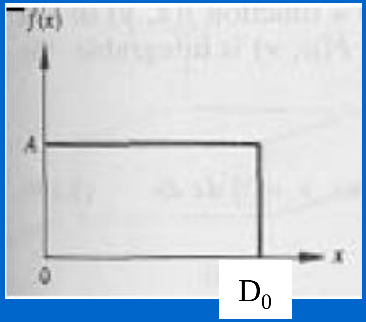
High-Pass filtering

- A high-pass filter can be obtained from a low-pass filter using:

$$H_{HP}(u,v) = 1 - H_{LP}(u,v)$$

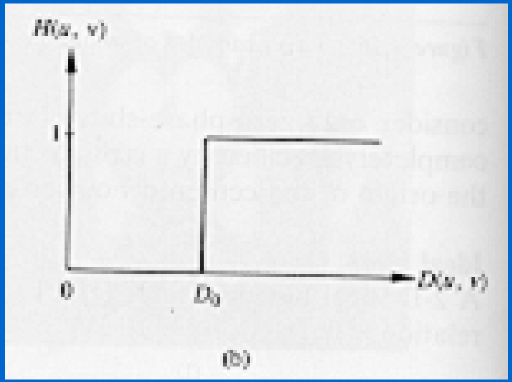


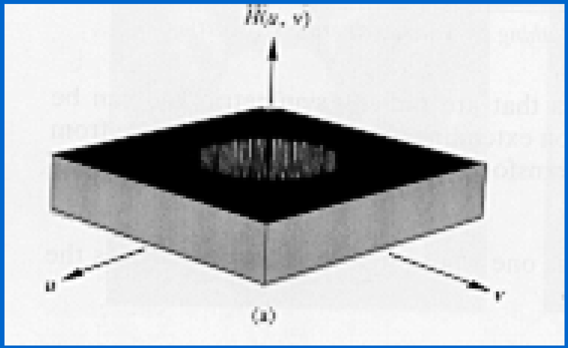
= 1 -



High-pass filtering (cont'd)

- Preserves high frequencies, attenuates low frequencies.





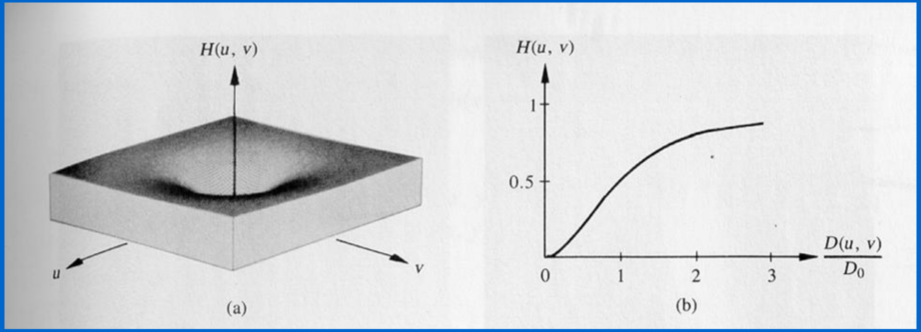
$$H(u,v) = \begin{cases} 1 & \text{if} u \geq D_0 \\ 0 & otherwise \end{cases}$$

$$H(u,v) = \begin{cases} 1 & \text{if} u^2 + v^2 \geq D_0{}^2 \\ 0 & otherwise \end{cases}$$

Butterworth high pass filter (BHPF)

- In practice, we use filters that attenuate low frequencies smoothly (e.g., **Butterworth** HP filter) → less ringing effect

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



Comparison: IHPF and BHPF

IHPF

a b c

FIGURE 4.54 Results of highpass filtering the image in Fig. 4.41(a) using an IHPF with $D_0 = 30$, 60, and 160.

D_0=30,60,160

BHPF

a b c

FIGURE 4.55 Results of highpass filtering the image in Fig. 4.41(a) using a BHPF of order 2 with $D_0 = 30$, 60, and 160, corresponding to the circles in Fig. 4.41(b). These results are much smoother than those obtained with an IHPF.

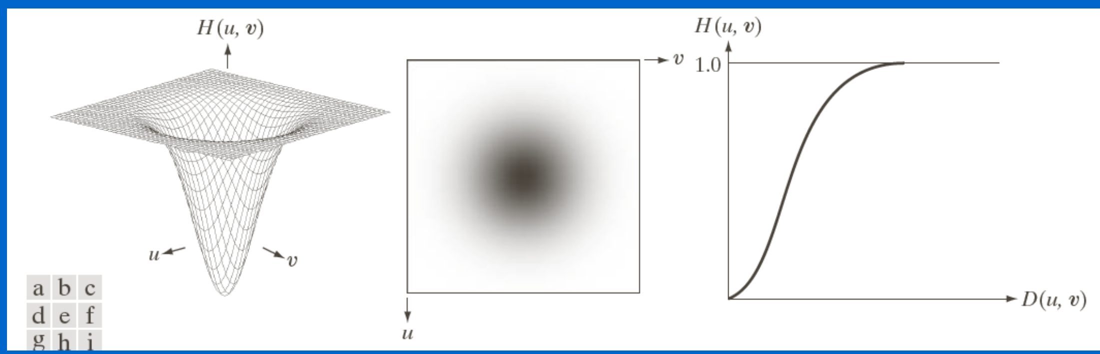
D_0=30,60,160

n=2

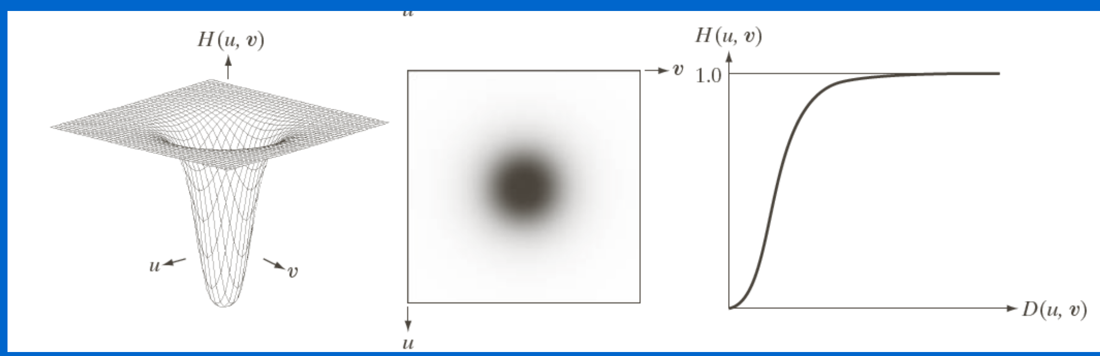
Gaussian HP filter

A 2-D Gaussian highpass filter (GHPL) is defined as

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



GHPF



BHPF

Comparison: BHPF and GHPF

BHPF

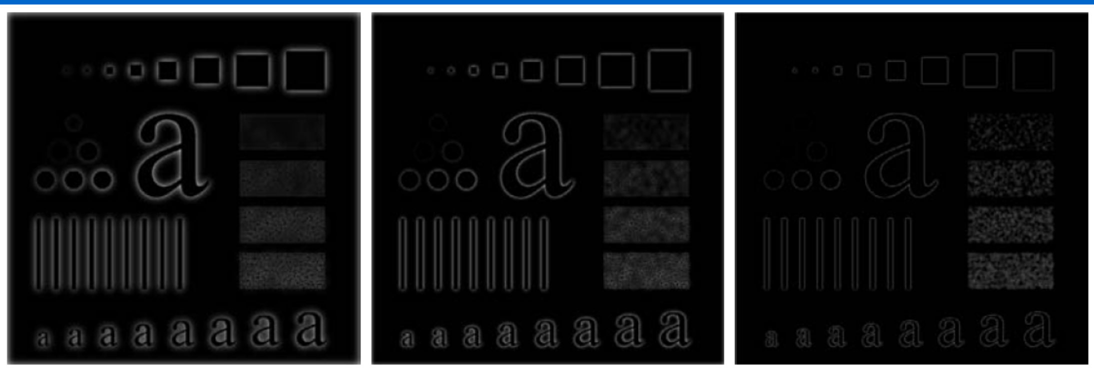
a b c

FIGURE 4.55 Results of highpass filtering the image in Fig. 4.41(a) using a BHPF of order 2 with $D_0 = 30, 60,$ and 160, corresponding to the circles in Fig. 4.41(b). These results are much smoother than those obtained with an IHPF.

D_0=30,60,160

n=2

GHPF



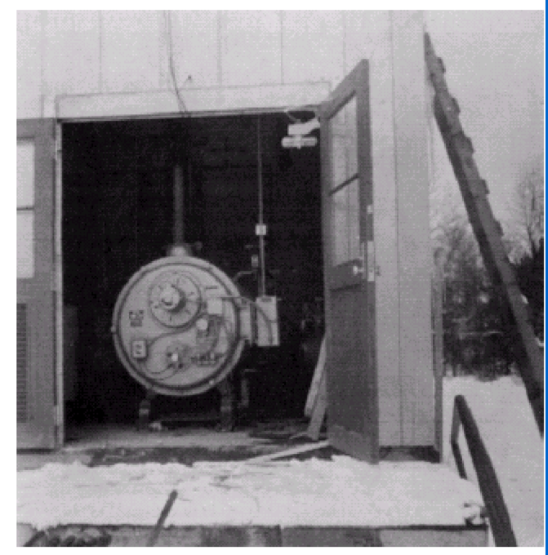
a b c

FIGURE 4.56 Results of highpass filtering the image in Fig. 4.41(a) using a GHPF with $D_0 = 30, 60,$ and 160, corresponding to the circles in Fig. 4.41(b). Compare with Figs. 4.54 and 4.55.

D_0=30,60,160

Homomorphic filtering

- Many times, we want to remove shading effects from an image (i.e., due to uneven illumination)
  - Enhance high frequencies
  - Attenuate low frequencies but preserve fine detail.



Homomorphic Filtering (cont'd)

- Consider the following model of image formation:

$$f(x,y) = i(x,y)\, r(x,y)$$

i(x,y): illumination
r(x,y): reflection

- In general, the illumination component i(x,y) varies **slowly** and affects **low** frequencies mostly.
- In general, the reflection component r(x,y) varies **faster** and affects **high** frequencies mostly.

IDEA: separate low frequencies due to i(x,y) from high frequencies due to r(x,y)

How are frequencies mixed together?

- Low and high frequencies from **i(x,y)** and **r(x,y)** are mixed together.

$$f(x,y) = i(x,y)\,r(x,y) \Rightarrow F(u,v) = I(u,v) * R(u,v)$$

- When applying filtering, it is difficult to handle low/high frequencies separately.

$$F(u,v)H(u,v) = [I(u,v) * R(u,v)]H(u,v)$$

Can we separate them?

- Idea:

Take the ln( ) of $f(x, y) = i(x, y)\, r(x, y)$

$$ln(f(x, y)) = ln(i(x, y)) + ln(r(x, y))$$

Steps of Homomorphic Filtering

(1) Take $ln(f(x,y)) = ln(i(x,y)) + ln(r(x,y))$

(2) Apply FT: $\mathcal{F}(ln(f(x,y))) = \mathcal{F}(ln(i(x,y))) + \mathcal{F}(ln(r(x,y)))$

or $Z(u,v) = Illum(u,v) + Refl(u,v)$

(3) Apply H(u,v)

$$Z(u,v)H(u,v) = Illum(u,v)H(u,v) + Refl(u,v)H(u,v)$$

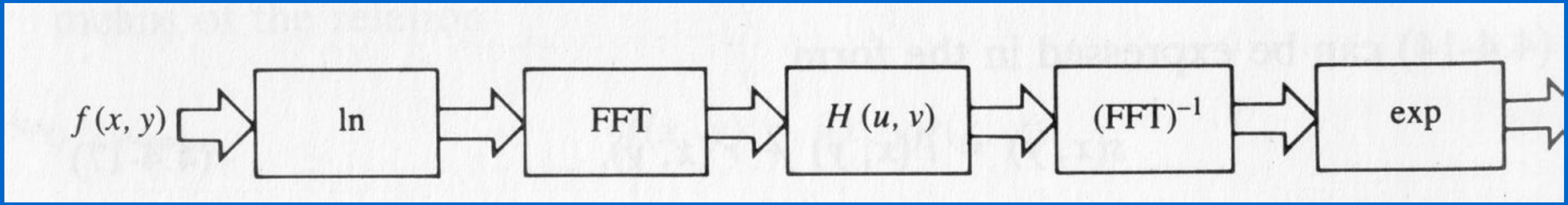
Steps of Homomorphic Filtering (cont'd)

(4) Take Inverse FT:

$$F^{-1}(Z(u,v)H(u,v)) = F^{-1}(Illum(u,v)H(u,v)) + F^{-1}(Refl(u,v)H(u,v))$$

or $s(x,y) = i'(x,y) + r'(x,y)$

(5) Take exp( ) $e^{s(x,y)} = e^{i'(x,y)} e^{r'(x,y)}$ or $g(x,y) = i_0(x,y) r_0(x,y)$



Example using high-frequency emphasis

$$H(u,v) = (\gamma_H - \gamma_L)\left[1 - e^{-c\left[(u^2+v^2)/D_0^2\right]}\right] + \gamma_L$$

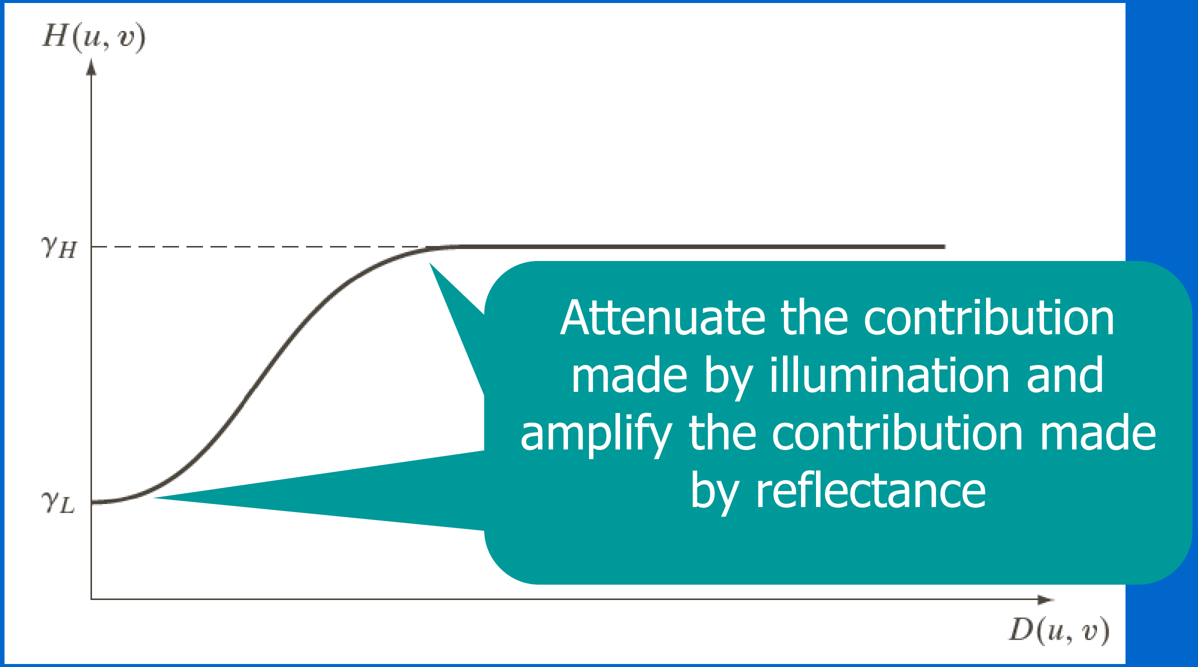
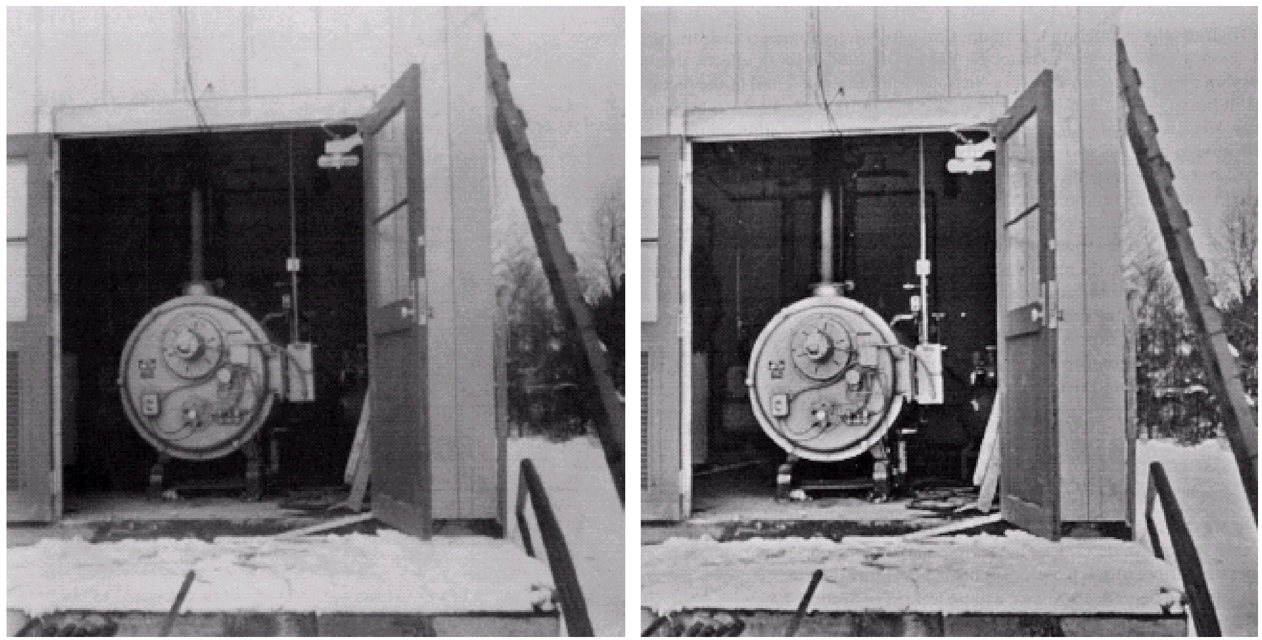


FIGURE 4.61 Radial cross section of a circularly symmetric homomorphic filter function. The vertical axis is at the center of the frequency rectangle and $D(u, v)$ is the distance from the center.

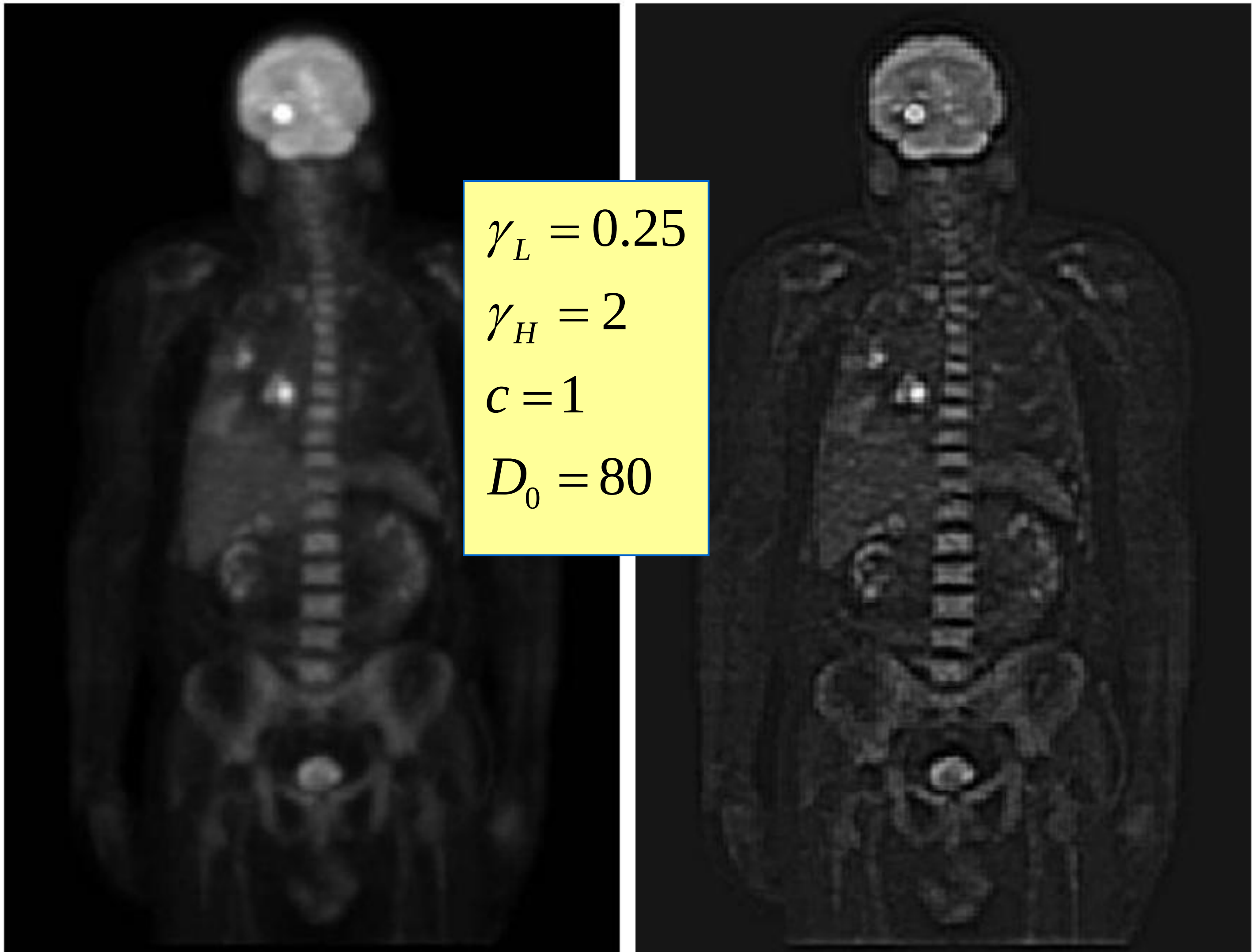
Homomorphic Filtering: Example



a b

FIGURE 4.33 (a) Original image. (b) Image processed by homomorphic filtering (note details inside shelter). (Stockham.)

Homomorphic Filtering: Example



$$\gamma_L = 0.25$$
$$\gamma_H = 2$$
$$c = 1$$
$$D_0 = 80$$

a b

FIGURE 4.62 (a) Full body PET scan. (b) Image enhanced using homomorphic filtering. (Original image courtesy of Dr. Michael E. Casey, CTI PET Systems.)

Karhren Loeve Transform (Hotelling Transfrom)

- Also called as Eigen Vector Transform.
- Based on statistical properties of image and has several important properties that make it useful for image processing – image compression.
- Data from neighbouring pixels is highly correlated , so achieving image compression without losing the quality of image is a challenge.

Karhren Loeve Transform (Hotelling Transfrom)

- By de-correlating this data more compression can be achieved.

- KL transform de-correlates the data which facilitates high degrees of compression

Steps to compute KL transform

To summarise, the process

(i) Find the mean vector and the covariance matrix of x.

(ii) Find the eigenvalues and then the eigenvectors of the covariance matrix.

(iii) Create the transformation matrix T, such that the rows of T (basis functions) are the eigenvectors.

(iv) $X = T[C_x - m]$.

the concepts that are stated

The mean vector of f is given by the formula Equation (11.28).

$$m_f = \frac{1}{N}\sum_{k=1}^{N} f_k$$

Where N is the number of columns (in this case N = 4).

$$\therefore m_f = \frac{1}{4}[f_1 + f_2 + f_3 + f_4] = \frac{1}{4}\begin{bmatrix}4\\4\end{bmatrix}$$

Now,

$$m_f \cdot m_f' = \frac{1}{4}\begin{bmatrix}4\\4\end{bmatrix}\cdot\frac{1}{4}[4 \quad 4] = \frac{1}{16}\begin{bmatrix}16 & 16\\16 & 16\end{bmatrix}$$

Similarly,

$$\sum_{k=1}^{N} f_K \cdot f_K' = f_1 \cdot f_1' + f_2 \cdot f_2' + f_3 \cdot f_3' + f_4 \cdot f_4'$$

Now,

$$f_1 \cdot f_1' = \begin{bmatrix} 1 \\ 2 \end{bmatrix} [1 \;\; 2] = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$$

$$f_2 \cdot f_2' = \begin{bmatrix} 2 \\ -1 \end{bmatrix} [2 \;\; -1] = \begin{bmatrix} 4 & -2 \\ -2 & 1 \end{bmatrix}$$

$$f_3 \cdot f_3 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} [1 \quad 1] = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

$$f_4 \cdot f_4 = \begin{bmatrix} 0 \\ 2 \end{bmatrix} [0 \quad 2] = \begin{bmatrix} 0 & 0 \\ 0 & 4 \end{bmatrix}$$

The covariance matrix of f is given by the formula Equation (11.30).

$$C_f = \frac{1}{N}\sum_{k=1}^{N} f_k \cdot f_k' - m_f \cdot m_f'$$

$$C_f = \frac{1}{N}[(f_1 \cdot f_1' - m_f \cdot m_f') + (f_2 \cdot f_2' - m_f \cdot m_f') + (f_3 \cdot f_3' - m_f \cdot m_f') + (f_4 \cdot f_4' - m_f \cdot m_f')]$$

$$\therefore C_f = \begin{bmatrix} 0.5 & -0.75 \\ -0.75 & 1.5 \end{bmatrix}$$

Eigen Value

$$|C_r - \lambda I| = 0$$

$$\left|\begin{bmatrix} 0.5 & -0.75 \\ -0.75 & 1.5 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right| = 0$$

$$\therefore \begin{vmatrix} 0.5-\lambda & -0.75 \\ -0.75 & 1.5-\lambda \end{vmatrix} = 0$$

$$(0.5-\lambda)(1.5-\lambda) - 0.5625 = 0$$

$$\therefore 0.75 - 0.5\lambda - 1.5\lambda + \lambda^2 - 0.5625 = 0$$

$$\lambda^2 - 2\lambda + 0.1875 = 0$$

$\lambda_1 = 0.0986$ and $\lambda_2 = 1.9014$.

We find the first eigenvector (v_1) corresponding to the first eigenvalue $\lambda_1 = 0.0986$

We use the equation

$$C_T v_1 = \lambda v_1$$

$$\begin{bmatrix} 0.5 & -0.75 \\ -0.75 & 1.5 \end{bmatrix} \begin{bmatrix} v_{10} \\ v_{11} \end{bmatrix} = 0.0986 \begin{bmatrix} v_{10} \\ v_{11} \end{bmatrix}$$

$$\text{where } v_1 = \begin{bmatrix} v_{10} \\ v_{11} \end{bmatrix}$$

$$\therefore\ 0.5v_{10} - 0.75v_{11} = 0.0986v_{10}$$

$$-0.75v_{10} + 1.5v_{11} = 0.0986v_{11}$$

Rearranging (a) and (b), we get,

$$0.4014v_{10} - 0.75v_{11} = 0$$

$$-0.75v_{10} + 1.4014v_{11} = 0$$

We could use either (c) or (d) to get a relationship between v_{10} and v_{11}. L
ion (c),

$$\therefore\ 0.4014v_{10} = 0.75v_{11}$$

$$\therefore\ v_{10} = 1.8684v_{11}$$

Equation (d) would have given the same result.

Therefore eigenvector v_1 corresponding to eigenvalue λ_1 is

$$v_1 = \begin{bmatrix} 1.8685 \\ 1 \end{bmatrix}$$

Similarly we calculate the second eigenvector v_2 corresponding to eigenvalue $\lambda_2 = 1.9014$. We use the same equation i.e.,

$$C_f v_2 = \lambda_2 v_2$$

$$\begin{bmatrix} 0.5 & -0.75 \\ -0.75 & 1.5 \end{bmatrix} \begin{bmatrix} v_{20} \\ v_{21} \end{bmatrix} = 1.9014 \begin{bmatrix} v_{20} \\ v_{21} \end{bmatrix}$$

$$\text{where } v_2 = \begin{bmatrix} v_{20} \\ v_{21} \end{bmatrix}$$

$$\therefore 0.5v_{20} - 0.75v_{21} = 1.9014v_{20} \qquad \text{...(e)}$$

$$-0.75v_{20} + 1.5v_{21} = 1.9014v_{21}$$

...(f)

Rearranging (a) and (b), we get,

$$1.4014v_{20} - 0.75v_{21} = 0$$

...(g)

$$-0.75v_{20} + 0.4014v_{21} = 0$$

...(h)

We could use either (g) or (h) to get a relationship between v_{20} and v_{21}. Let's use Equation (g),

$$-1.4014\,v_{20} = 0.75v_{21}$$

$$\therefore v_{20} = -0.5352v_{21}$$

Equation (h) would have given the same result.

$$v_2 = \begin{bmatrix} -0.5352 \\ 1 \end{bmatrix}$$

Hence the two eigenvectors are

$$v_1 = \begin{bmatrix} 1.8685 \\ 1 \end{bmatrix} \text{ and } \qquad v_2 = \begin{bmatrix} -0.5352 \\ 1 \end{bmatrix}$$

These eigenvectors need to be normalized.

$$v_{1N} = \frac{v_1}{\| v_1 \|}$$

Where v_{1N} is the normalised eigenvector and $\| \cdot \|$ is the Norm

$$v_{1N} = \frac{1}{\sqrt{(1.8685)^2 + (1)^2}} \begin{bmatrix} 1.8685 \\ 1 \end{bmatrix}$$

$$= 0.47186 \begin{bmatrix} 1.8685 \\ 1 \end{bmatrix}$$

$$\therefore v_{1N} = \begin{bmatrix} 0.8817 \\ 0.47186 \end{bmatrix}$$

Similarly we normalise v_2,

$$v_{2N} = \frac{v_2}{\| v_2 \|}$$

$$v_{2N} = \frac{1}{\sqrt{(-0.5352)^2 + (1)^2}} \begin{bmatrix} -0.5352 \\ 1 \end{bmatrix}$$

$$= 0.8817 \begin{bmatrix} -0.5352 \\ 1 \end{bmatrix}$$

$$\therefore v_{2N} = \begin{bmatrix} -0.4719 \\ 0.8817 \end{bmatrix}$$

$$\therefore \quad \mathbf{T} = \begin{bmatrix} -0.4719 & 0.8817 \\ 0.8817 & 0.4719 \end{bmatrix}$$

$$F = T[f - m_f]$$

where f is every column of the original image.

$$F_1 = T[f_1 - m_f]$$

$$F_1 = \begin{bmatrix} -0.4719 & 0.8817 \\ 0.8817 & 0.4719 \end{bmatrix} \left(\underset{f_1}{\begin{bmatrix} 1 \\ 2 \end{bmatrix}} - \underset{m_f}{\begin{bmatrix} 1 \\ 1 \end{bmatrix}} \right)$$

$$F_1 = \begin{bmatrix} 0.8817 \\ 0.4719 \end{bmatrix}$$

Similarly,

$$F_2 = T \cdot [f_2 - m_f] = \begin{bmatrix} -2.2352 \\ -0.0621 \end{bmatrix}$$

$$F_3 = T \cdot [f_3 - m_f] = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$F_4 = T \cdot [f_4 - m_f] = \begin{bmatrix} 1.3535 \\ -0.4098 \end{bmatrix}$$

Here the image, $f = \begin{bmatrix} 1 & 2 & 1 & 0 \\ 2 & -1 & 1 & 2 \end{bmatrix}$ is

transformed to $F = \begin{bmatrix} 0.8817 & -2.2352 & 0 & 1.3535 \\ 0.4719 & -0.0621 & 0 & -0.4098 \end{bmatrix}$