Frequency Domain Filtering

- Fourier Series
- Sampling Theorem
- 1-D Fourier Transform
- 2-D Fourier Transform
- 2-D Filters
- Filtering
- Butterworth Filter
- Gaussian Filter

Fourier Series

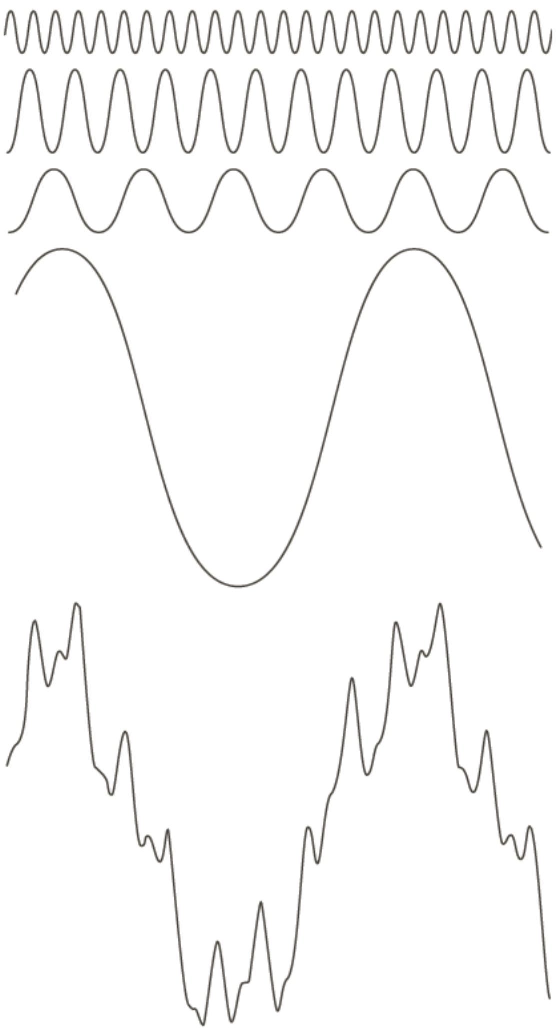


FIGURE 4.1 The function at the bottom is the sum of the four functions above it. Fourier's idea in 1807 that periodic functions could be represented as a weighted sum of sines and cosines was met with skepticism.

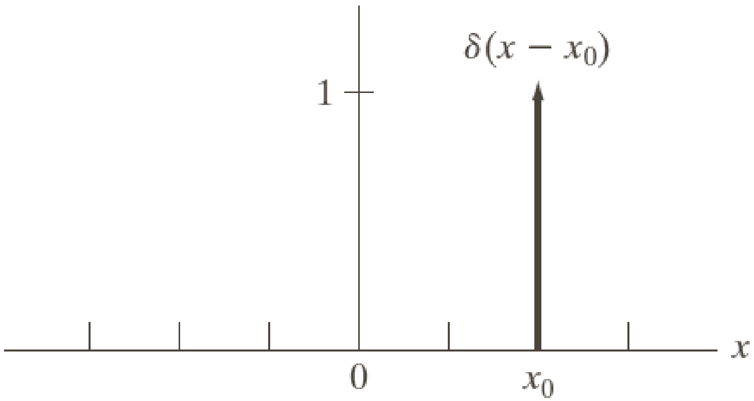


FIGURE 4.2 A unit discrete impulse located at $x = x_0$. Variable x is discrete, and δ is 0 everywhere except at $x = x_0$.

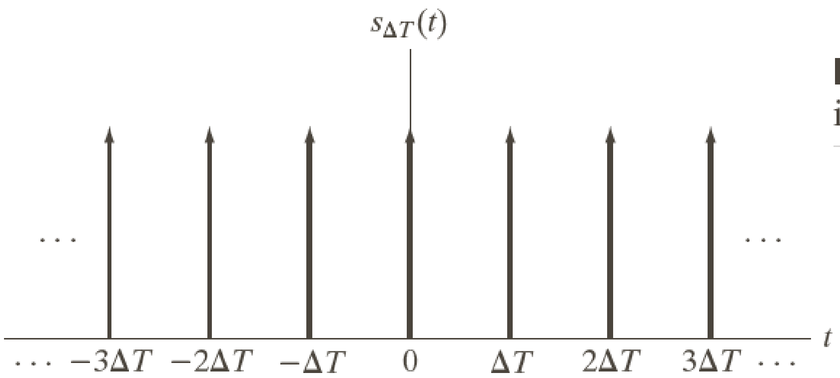
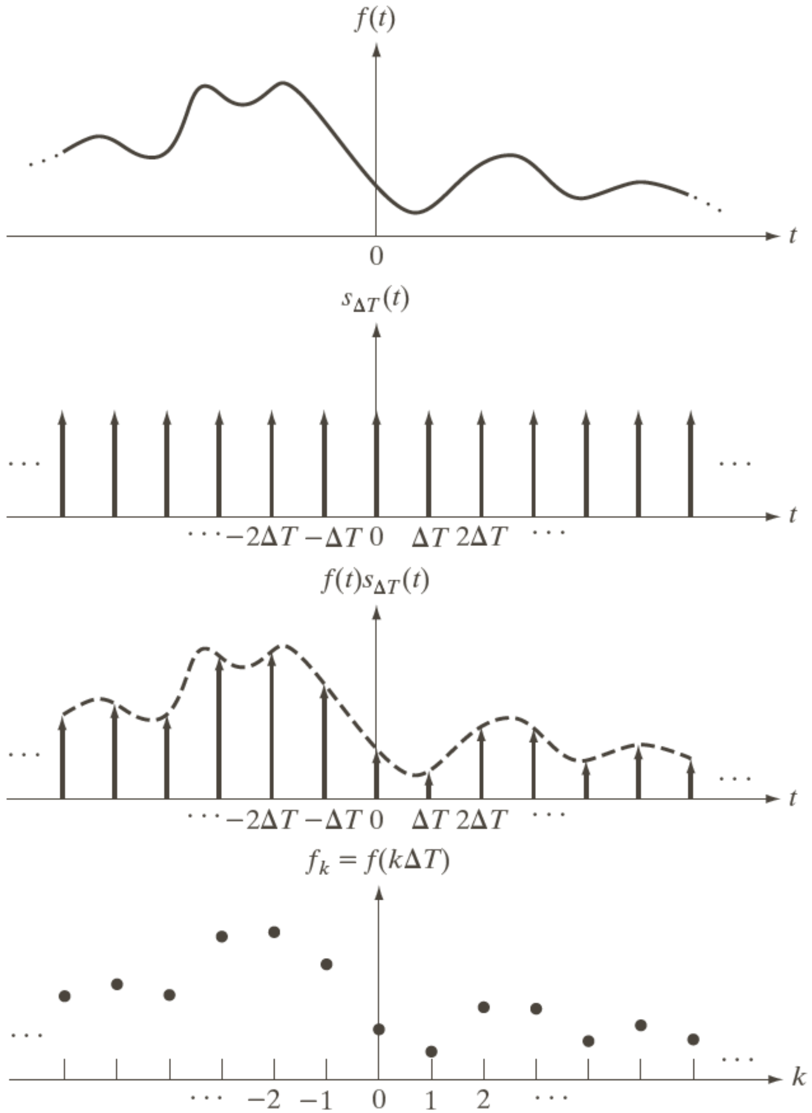


FIGURE 4.3 An impulse train.

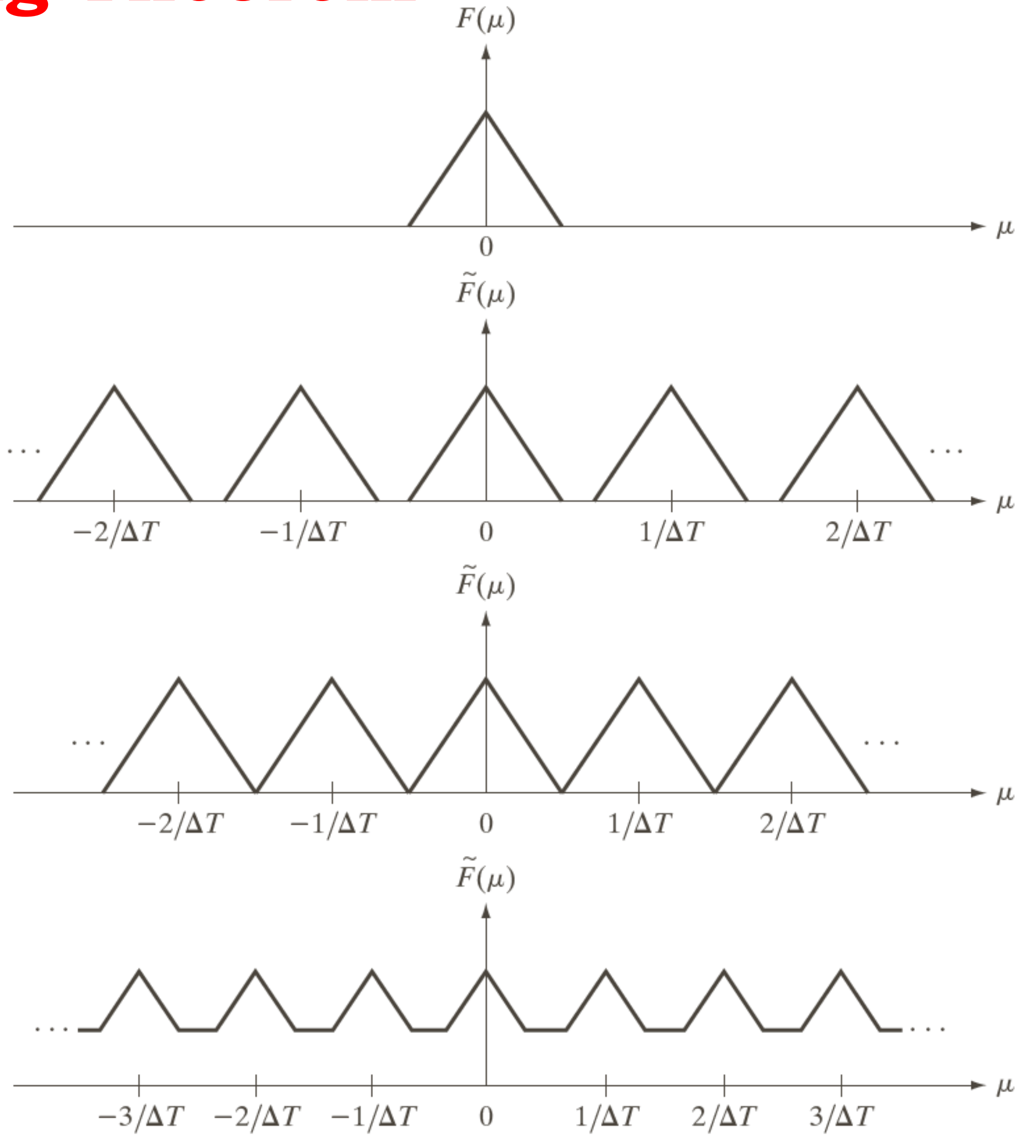
Sampling Theorem



a
b
c
d

FIGURE 4.5
(a) A continuous function. (b) Train of impulses used to model the sampling process. (c) Sampled function formed as the product of (a) and (b). (d) Sample values obtained by integration and using the sifting property of the impulse. (The dashed line in (c) is shown for reference. It is not part of the data.)

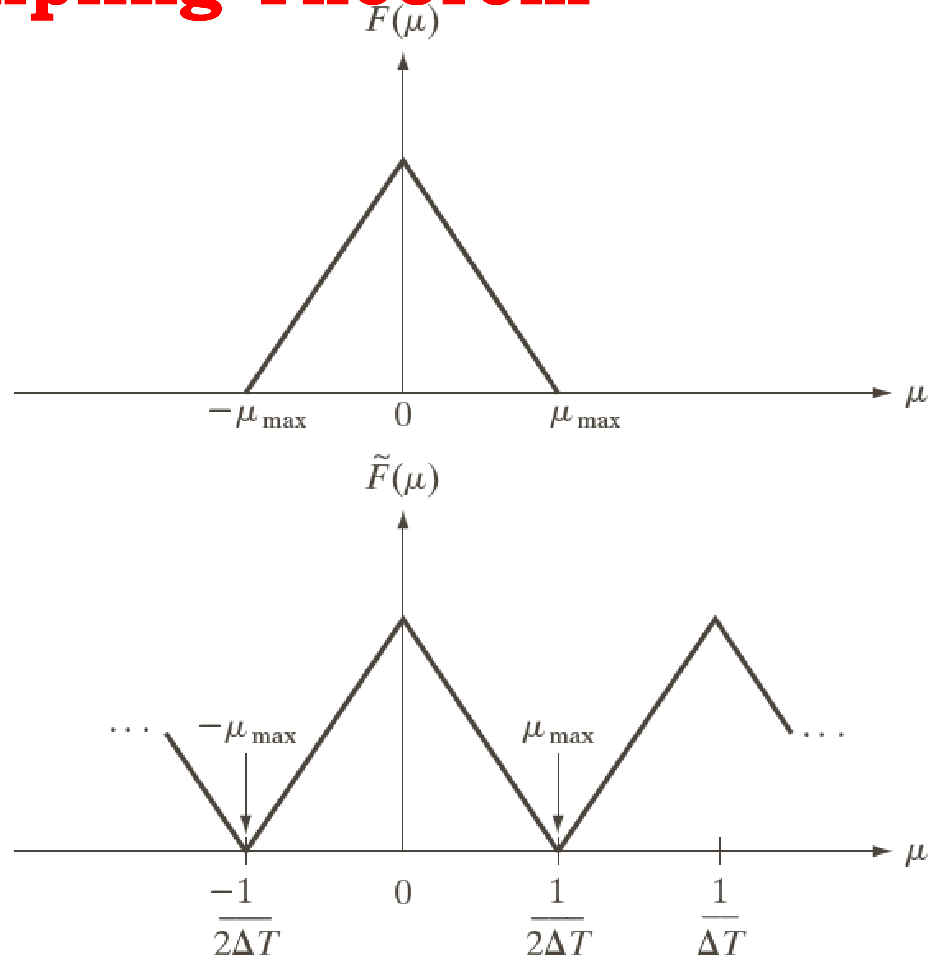
Sampling Theorem



a
b
c
d

FIGURE 4.6
(a) Fourier transform of a band-limited function.
(b)–(d) Transforms of the corresponding sampled function under the conditions of over-sampling, critically-sampling, and under-sampling, respectively.

Sampling Theorem



a
b

FIGURE 4.7
(a) Transform of a band-limited function.
(b) Transform resulting from critically sampling the same function.

Sampling Theorem

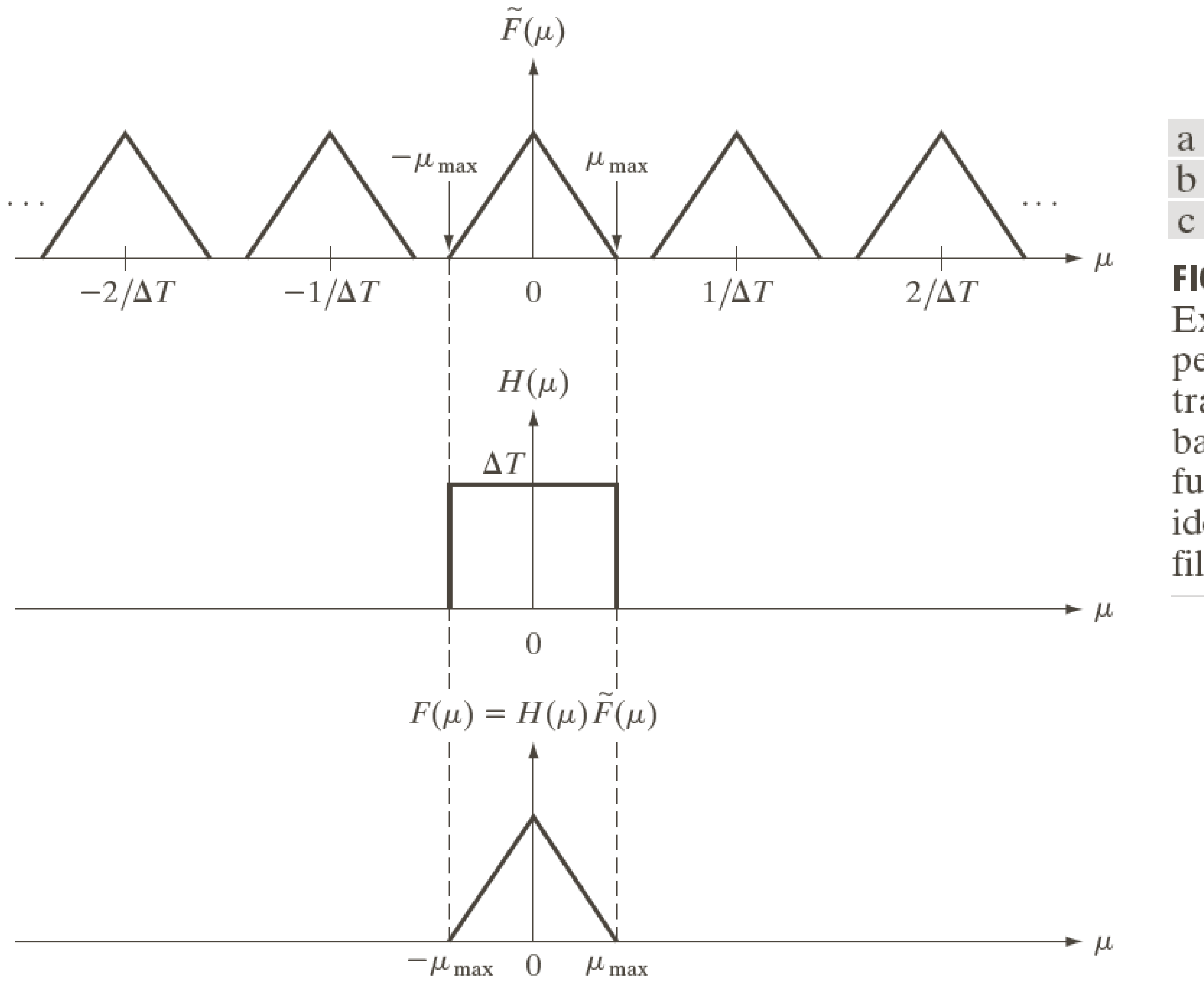
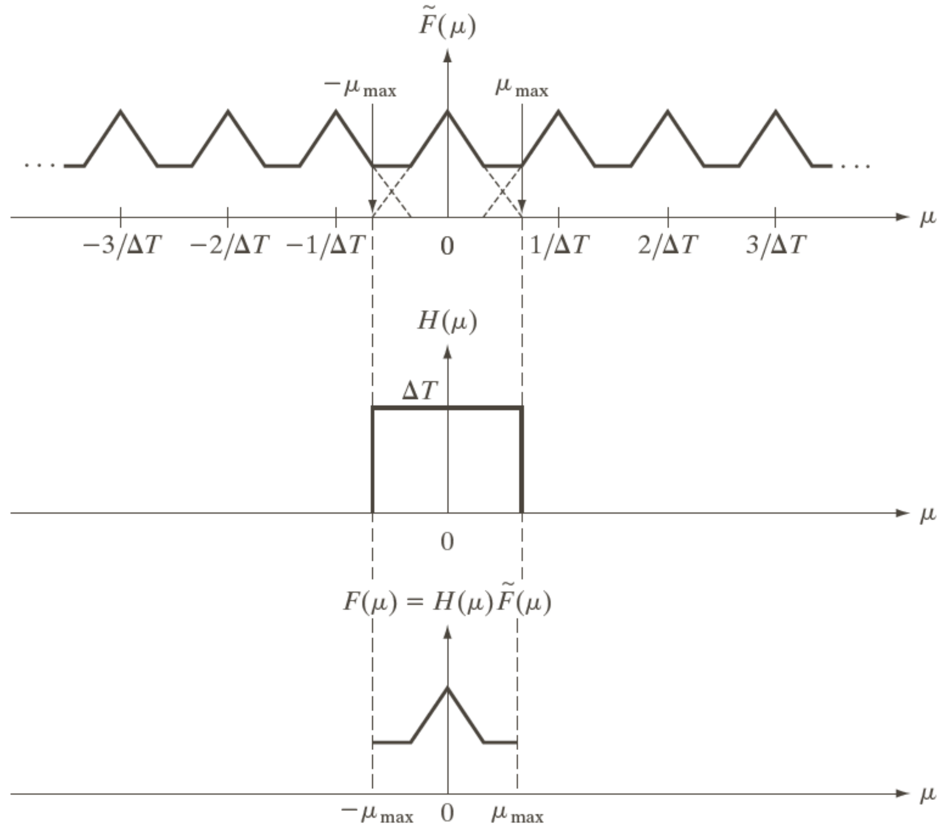


FIGURE 4.8 Extracting one period of the transform of a band-limited function using an ideal lowpass filter.

Sampling Theorem



a
b
c

FIGURE 4.9 (a) Fourier transform of an under-sampled, band-limited function. (Interference from adjacent periods is shown dashed in this figure). (b) The same ideal lowpass filter used in Fig. 4.8(b). (c) The product of (a) and (b). The interference from adjacent periods results in aliasing that prevents perfect recovery of $F(\mu)$ and, therefore, of the original, band-limited continuous function. Compare with Fig. 4.8.

Sampling Theorem

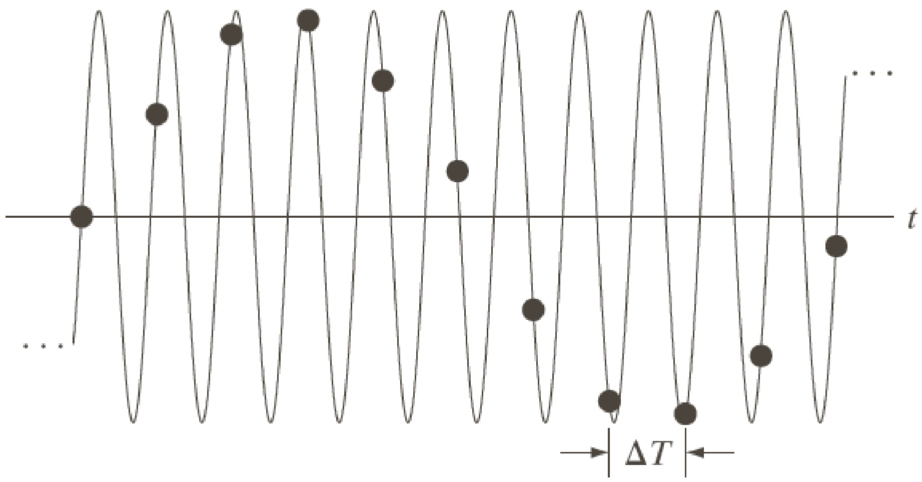


FIGURE 4.10 Illustration of aliasing. The under-sampled function (black dots) looks like a sine wave having a frequency much lower than the frequency of the continuous signal. The period of the sine wave is 2 s, so the zero crossings of the horizontal axis occur every second. ΔT is the separation between samples.

2-D Sampling

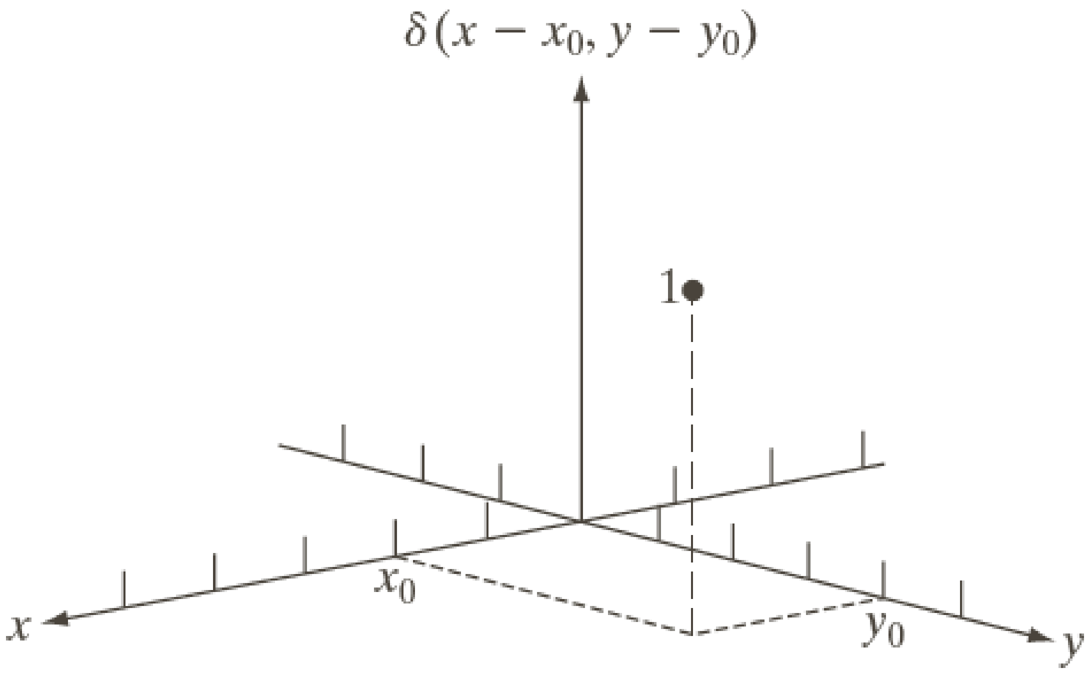


FIGURE 4.12 Two-dimensional unit discrete impulse. Variables x and y are discrete, and δ is zero everywhere except at coordinates (x_0, y_0).

2-D Sampling

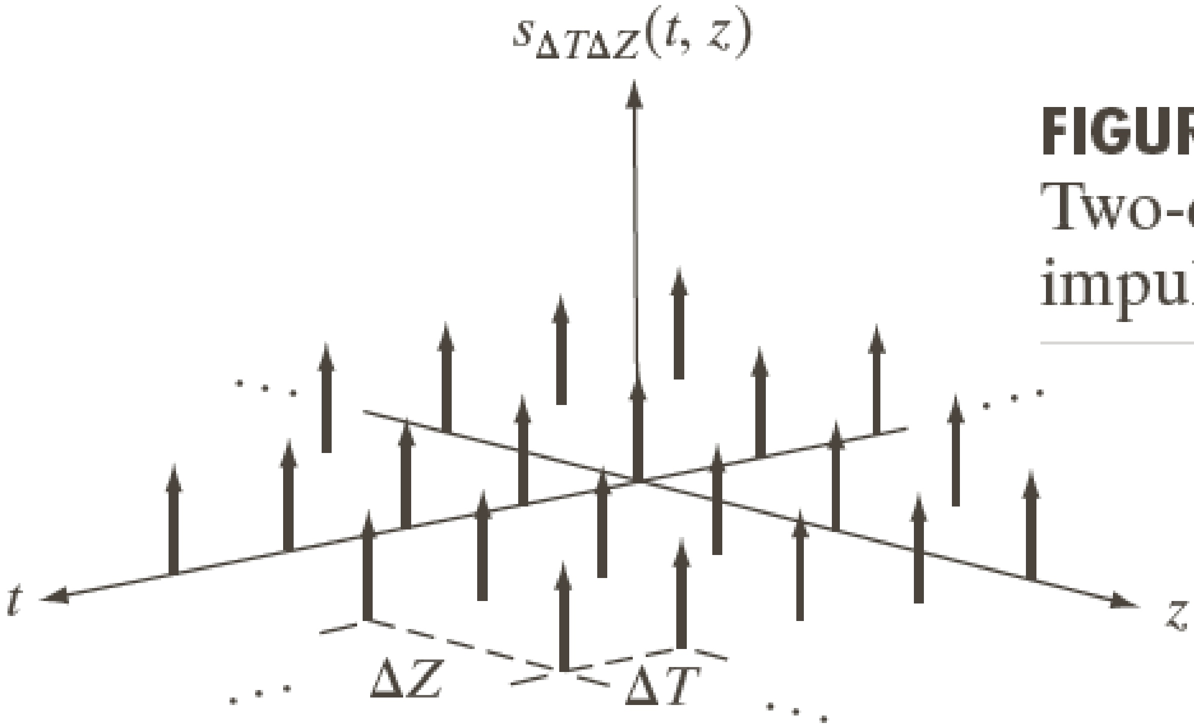
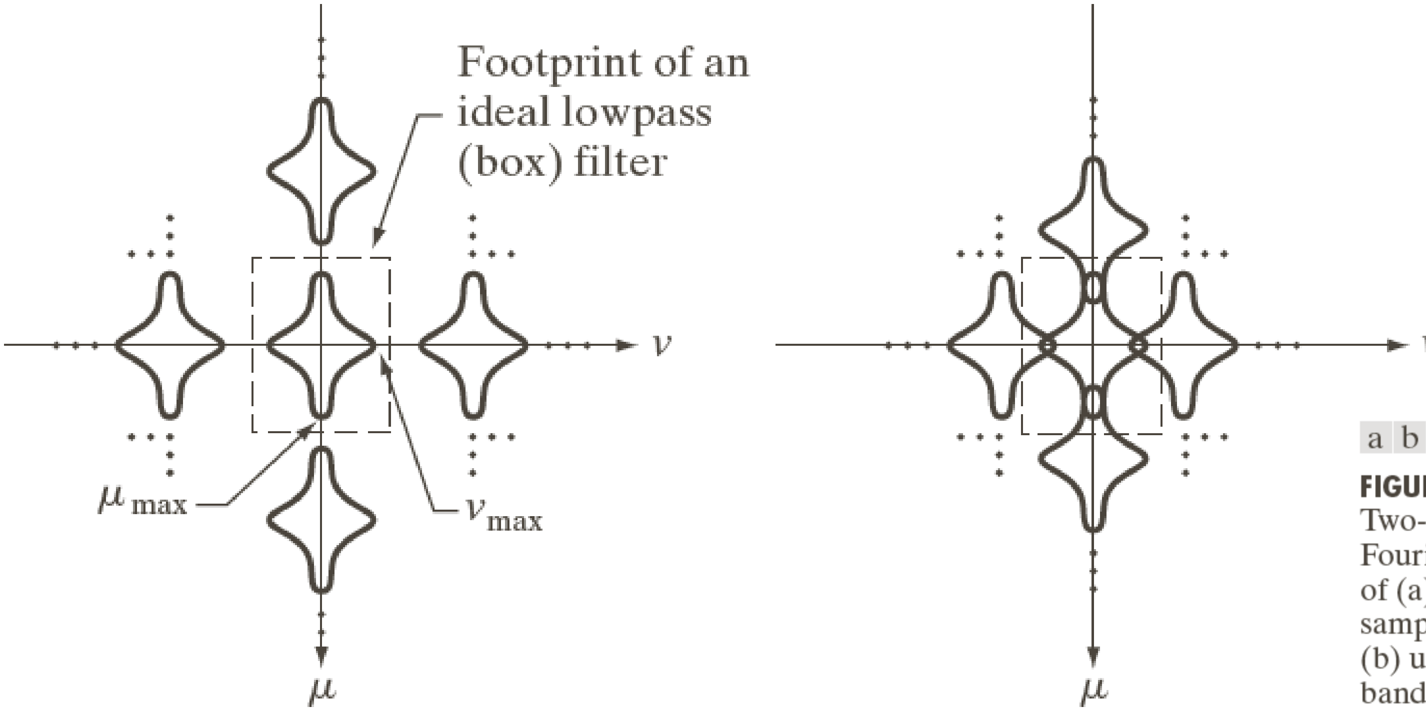


FIGURE 4.14 Two-dimensional impulse train.

2-D Sampling



a b

FIGURE 4.15 Two-dimensional Fourier transforms of (a) an over-sampled, and (b) under-sampled band-limited function.

1-D Fourier Transform

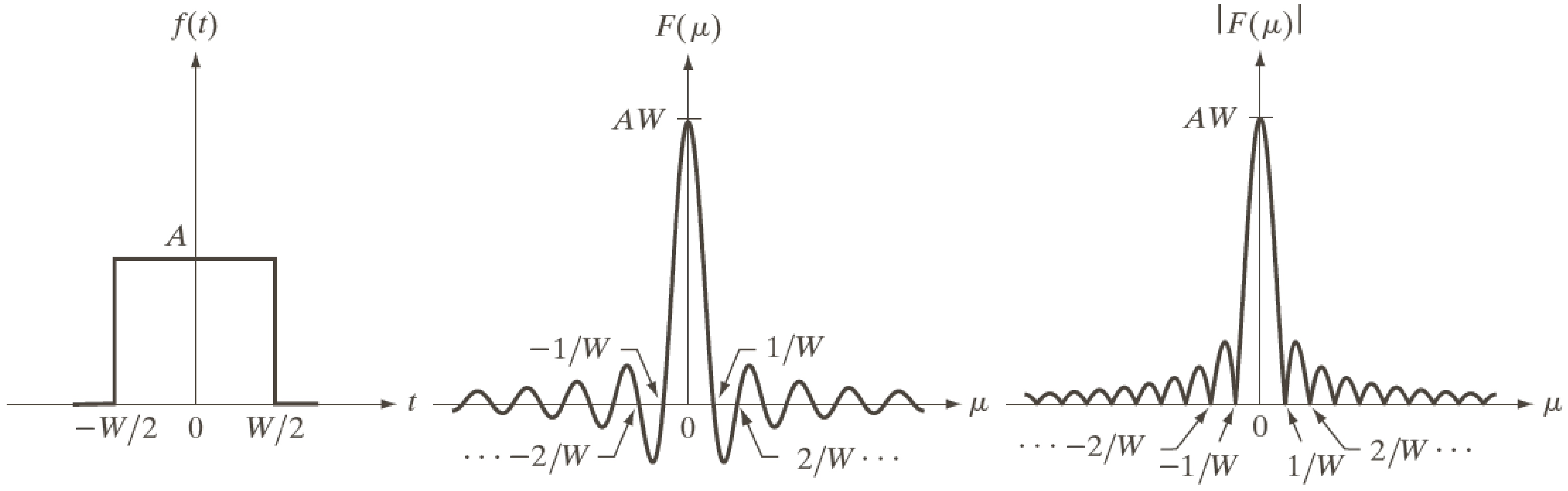


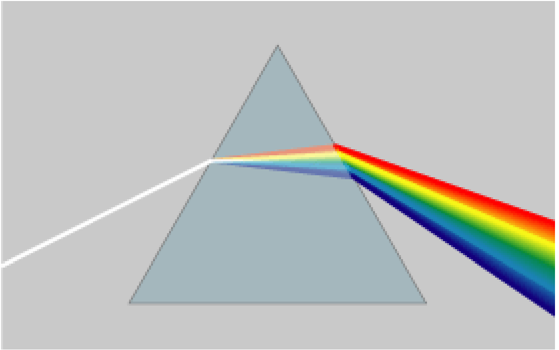
FIGURE 4.4 (a) A simple function; (b) its Fourier transform; and (c) the spectrum. All functions extend to infinity in both directions.

Image Transform

- Representation of the signal
- Basis Function
- Extraction of information from the signal

Role of Transform

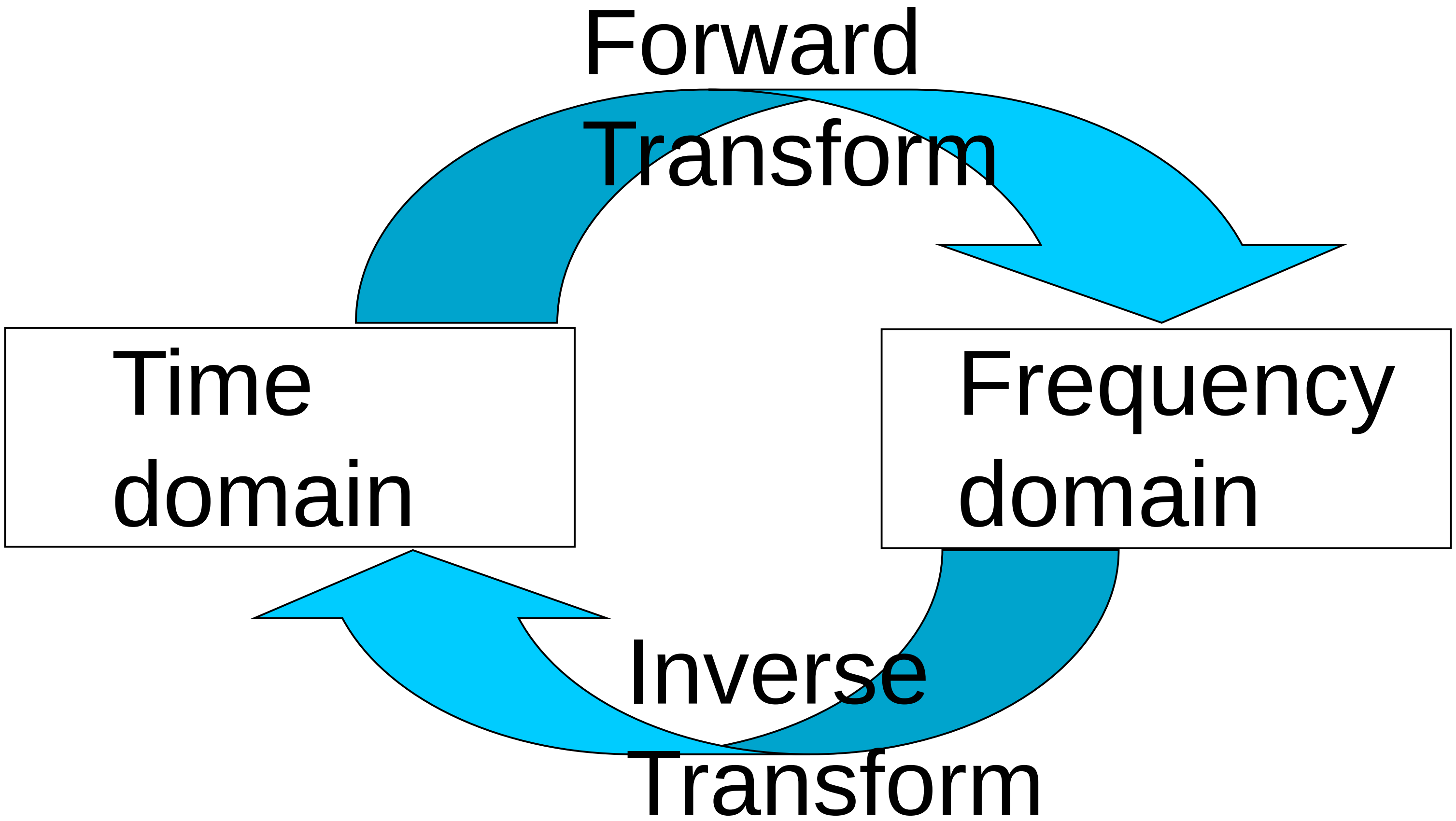
White Light?



VIBGYOR!

Time domain → Transform → Frequency domain

Transform


Forward Transform
Time domain
Frequency domain
Inverse Transform

Basis Function

Finality of Co-efficients

Orthogonality

Basis Function

$$X = \sum_{i=0}^{N} c_i \phi_i$$

X **Input signal**

c_i **Transform coefficient**

ϕ_i **Basis function**

$$c_i = \langle X, \phi_i \rangle$$

$$\hat{X} = \sum_{i=0}^{L<<N} c_i \phi_i$$

Different Image Trasforms

- **Fourier Transform**
- **Walsh Transform**
- **Hadamard Transform**
- **Discrete Cosine Transform**
- **KL Transform**
- **Wavelet Transform**

Fourier Transform

Jean Baptiste Joseph Fourier

French Mathematical Physicist

Idea in 1809

1-D Discrete Fourier Transform

$$x[n] \xrightarrow[\text{Analysis}]{\textbf{Forward Transform}} X[k]$$

$$X[k] \xrightarrow[\text{Synthesis}]{\textbf{Inverse Transform}} x[n]$$

1-D DFT (cont..)

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}kn}, \; k = 0 \text{ to } N-1$$

x[n] represents the signal

X[k] represents the spectrum

2-D DFT

$$x[n] \text{ ——— } f(m,n)$$

1-D signal 2-D signal

$$X[k] \text{ ——— } F[k,l]$$

1-D Spectrum **2-D Spectrum**

2-D DFT

$$X[k] = \sum_{n=0}^{N-1} x[n]\ e^{-j\frac{2\pi}{N}kn}$$

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

2-D Fourier Transform

- Let $f(x,y)$ for x=0,1,2, ..., M-1 and y=1,2, ..., N-1 denote an $M\times N$ image. The 2D DFT of f is given by

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

$$\text{for } u=0,1,2,\ldots,M-1 \text{ and } v=0,1,2,\ldots,N-1$$

- The *frequency domain* is simply the coordinate system spanned by $F(u,v)$ with u and v as frequency variables.

2-D Fourier Transform

- The inverse DFT is given by

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

$$\text{for } x=0,1,2,\ldots,M-1 \text{ and } y=0,1,2,\ldots,N-1$$

- Thus, given $F(u,v)$, we can obtain $f(x,y)$ back by means of the inverse DFT.

2-D Fourier Transform

- The value of the transform at the origin of the frequency domain that is $F(0,0)$ is called the *dc* component of the Fourier transform.
- Even if $f(x,y)$ is real, its transform is complex.
- The principal method of visually analyzing a transform is to compute its spectrum that is the magnitude of $F(u,v)$.

2-D Fourier Transform

- The Fourier spectrum is defined as

$$|F(u,v)| = [R^2(u,v) + I^2(u,v)]^{1/2}$$

- The phase angle is defined as

$$|\phi(u,v)| = \tan^{-1}[\frac{I(u,v)}{R(u,v)}]$$

2-D Fourier Transform

- The polar representation of $F(u,v)$ is defined by

$$F(u,v) = |F(u,v)| e^{j\phi(u,v)}$$

- The power spectrum is defined as

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

2-D Fourier Transform

Name	Expression(s)
1) Discrete Fourier transform (DFT) of $f(x, y)$	$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M + vy/N)}$
2) Inverse discrete Fourier transform (IDFT) of $F(u, v)$	$f(x, y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi(ux/M + vy/N)}$
3) Polar representation	$F(u, v) = \lvert F(u, v) \rvert e^{j\phi(u,v)}$
4) Spectrum	$\lvert F(u, v) \rvert = \left[R^2(u, v) + I^2(u, v)\right]^{1/2}$ $R = \text{Real}(F); \quad I = \text{Imag}(F)$
5) Phase angle	$\phi(u, v) = \tan^{-1}\left[\frac{I(u, v)}{R(u, v)}\right]$
6) Power spectrum	$P(u, v) = \lvert F(u, v) \rvert^2$
7) Average value	$\bar{f}(x, y) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) = \frac{1}{MN} F(0, 0)$

(*Continued*)

TABLE 4.2
Summary of DFT definitions and corresponding expressions.

Name	Expression(s)
8) Periodicity (k_1 and k_2 are integers)	$F(u, v) = F(u + k_1 M, v) = F(u, v + k_2 N)$ $= F(u + k_1 M, v + k_2 N)$ $f(x, y) = f(x + k_1 M, y) = f(x, y + k_2 N)$ $= f(x + k_1 M, y + k_2 N)$
9) Convolution	$f(x, y) \bigstar h(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) h(x - m, y - n)$
10) Correlation	$f(x, y) \star h(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f^*(m, n) h(x + m, y + n)$
11) Separability	The 2-D DFT can be computed by computing 1-D DFT transforms along the rows (columns) of the image, followed by 1-D transforms along the columns (rows) of the result. See Section 4.11.1.
12) Obtaining the inverse Fourier transform using a forward transform algorithm.	$MNf^*(x, y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F^*(u, v) e^{-j2\pi(ux/M + vy/N)}$ This equation indicates that inputting $F^*(u, v)$ into an algorithm that computes the forward transform (right side of above equation) yields $MNf^*(x, y)$. Taking the complex conjugate and dividing by MN gives the desired inverse. See Section 4.11.2.

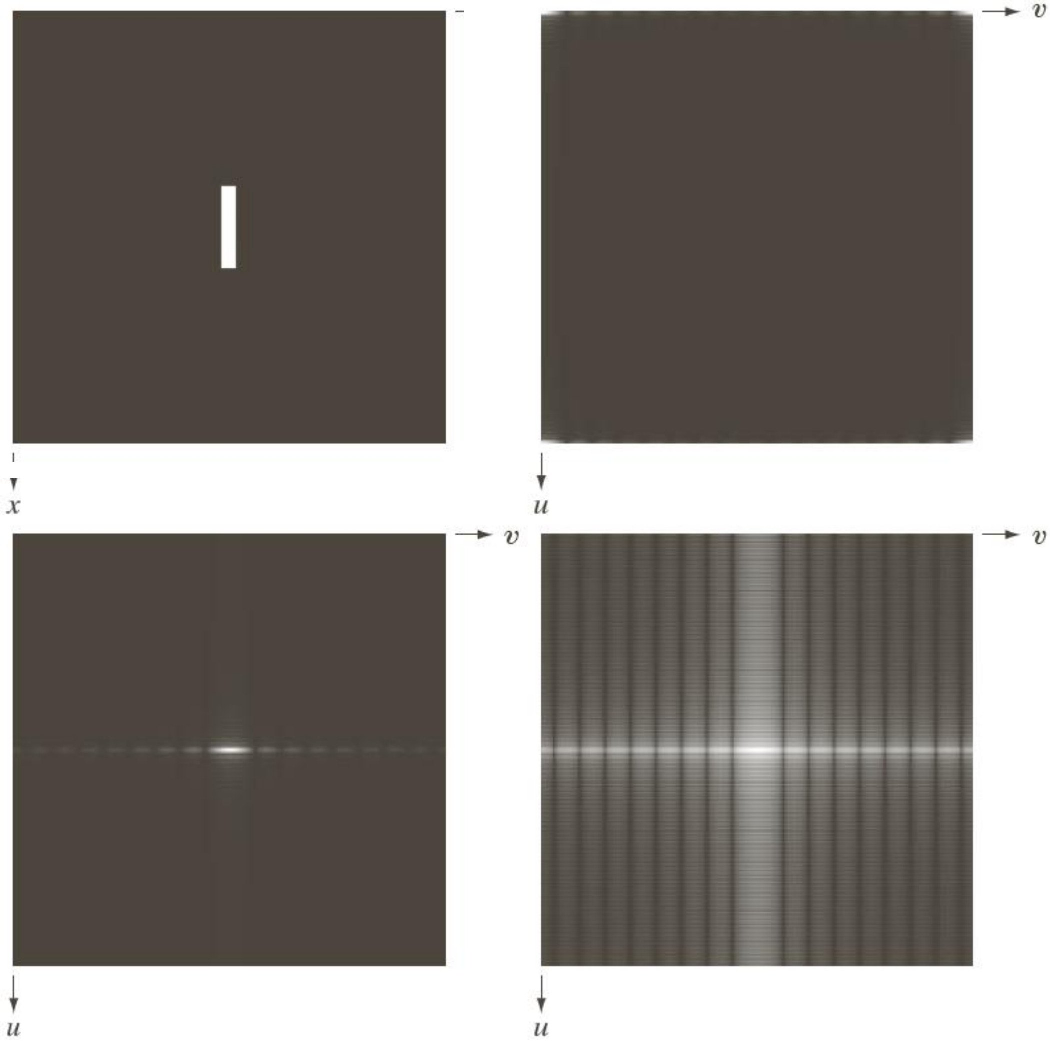
2-D Fourier Transform

	Spatial Domain†		Frequency Domain†
1)	$f(x, y)$ real	$\Leftrightarrow$	$F^*(u, v) = F(-u, -v)$
2)	$f(x, y)$ imaginary	$\Leftrightarrow$	$F^*(-u, -v) = -F(u, v)$
3)	$f(x, y)$ real	$\Leftrightarrow$	$R(u, v)$ even; $I(u, v)$ odd
4)	$f(x, y)$ imaginary	$\Leftrightarrow$	$R(u, v)$ odd; $I(u, v)$ even
5)	$f(-x, -y)$ real	$\Leftrightarrow$	$F^*(u, v)$ complex
6)	$f(-x, -y)$ complex	$\Leftrightarrow$	$F(-u, -v)$ complex
7)	$f^*(x, y)$ complex	$\Leftrightarrow$	$F^*(-u - v)$ complex
8)	$f(x, y)$ real and even	$\Leftrightarrow$	$F(u, v)$ real and even
9)	$f(x, y)$ real and odd	$\Leftrightarrow$	$F(u, v)$ imaginary and odd
10)	$f(x, y)$ imaginary and even	$\Leftrightarrow$	$F(u, v)$ imaginary and even
11)	$f(x, y)$ imaginary and odd	$\Leftrightarrow$	$F(u, v)$ real and odd
12)	$f(x, y)$ complex and even	$\Leftrightarrow$	$F(u, v)$ complex and even
13)	$f(x, y)$ complex and odd	$\Leftrightarrow$	$F(u, v)$ complex and odd

†Recall that x, y, u, and v are *discrete* (integer) variables, with x and u in the range $[0, M - 1]$, and y, and v in the range $[0, N - 1]$. To say that a complex function is *even* means that its real *and* imaginary parts are even, and similarly for an odd complex function.

TABLE 4.1 Some symmetry properties of the 2-D DFT and its inverse. $R(u, v)$ and $I(u, v)$ are the real and imaginary parts of $F(u, v)$, respectively. The term *complex* indicates that a function has nonzero real and imaginary parts.

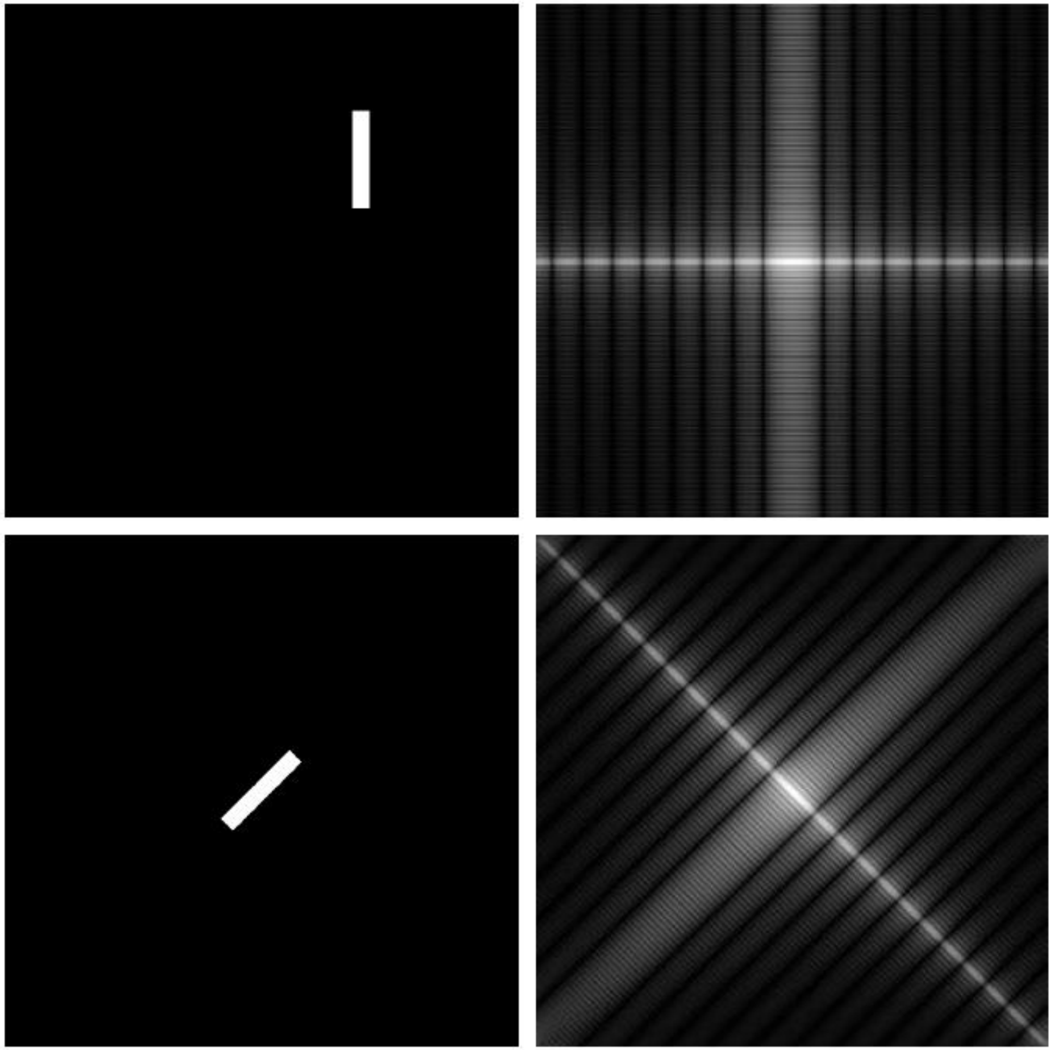
2-D Fourier Transform



a b
c d

FIGURE 4.24 (a) Image. (b) Spectrum showing bright spots in the four corners. (c) Centered spectrum. (d) Result showing increased detail after a log transformation. The zero crossings of the spectrum are closer in the vertical direction because the rectangle in (a) is longer in that direction. The coordinate convention used throughout the book places the origin of the spatial and frequency domains at the top left.

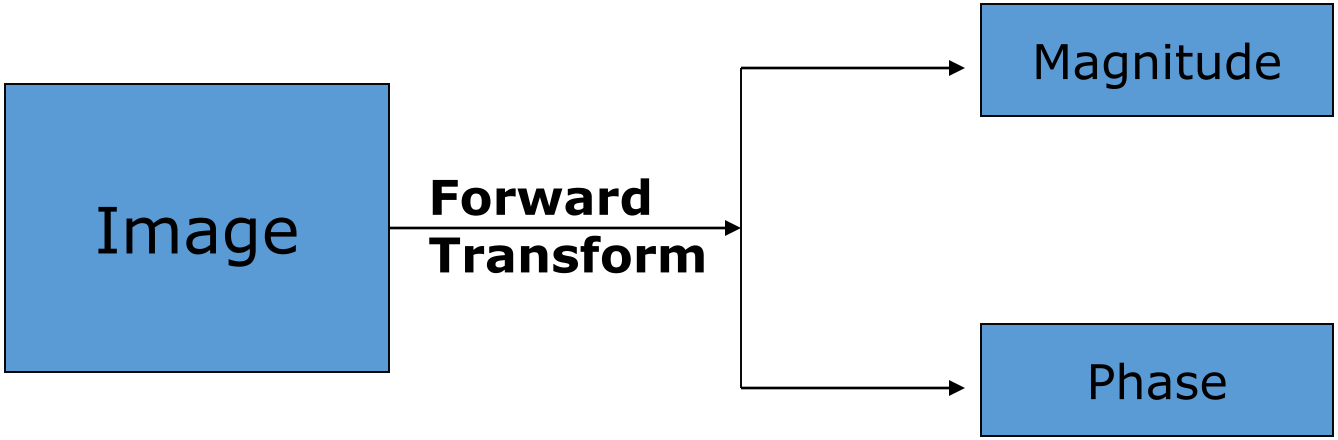
2-D Fourier Transform



a b
c d

FIGURE 4.25 (a) The rectangle in Fig. 4.24(a) translated, and (b) the corresponding spectrum. (c) Rotated rectangle, and (d) the corresponding spectrum. The spectrum corresponding to the translated rectangle is identical to the spectrum corresponding to the original image in Fig. 4.24(a).

2-D DFT



- The representation of intensity as a function of frequency is called ***spectrum***
- The coordinates of the Fourier spectrum are ***spatial frequencies*** and *not special distances or spatial positions*.
- The F.T is a process that always results in *positive* and *negative* coefficients of a ***complex number***.
- The positive coefficients are ---- ***white***
- Negative are– ***Black***
- Difficult to visualize the spectrum, *so* ***Magnitude of the F. Spectrum is used*** .

- The ***special position information*** of the image data is encoded as ***the difference between the coefficients of the real and imaginary data***.
- The difference is called the phase angle.
- Phase information represents ***Edge or the boundary*** information of the object that are present in an image.

2-Dim. DFT (cont.)

Properties of 2D DFT

Separability

$$F(k,l) = \frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} W_N^{km} \left(\frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} f(m,n) W_N^{ln} \right), \quad k,l = 0,\cdots,N-1$$

$$f(m,n) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} W_N^{-km} \left(\frac{1}{\sqrt{N}} \sum_{l=0}^{N-1} F(k,l) W_N^{-ln} \right), \quad m,n = 0,\cdots,N-1$$

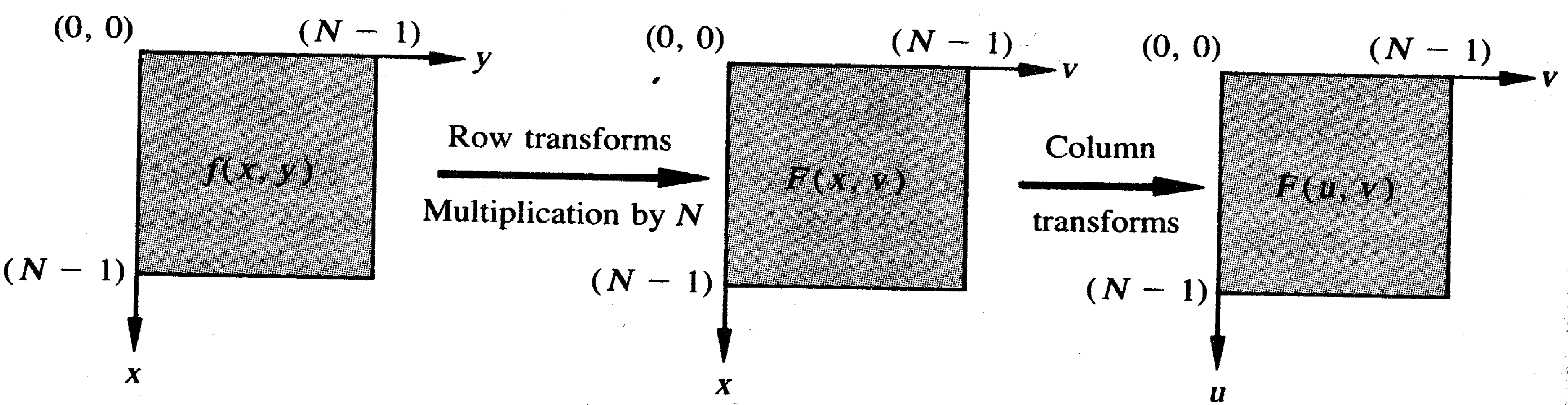
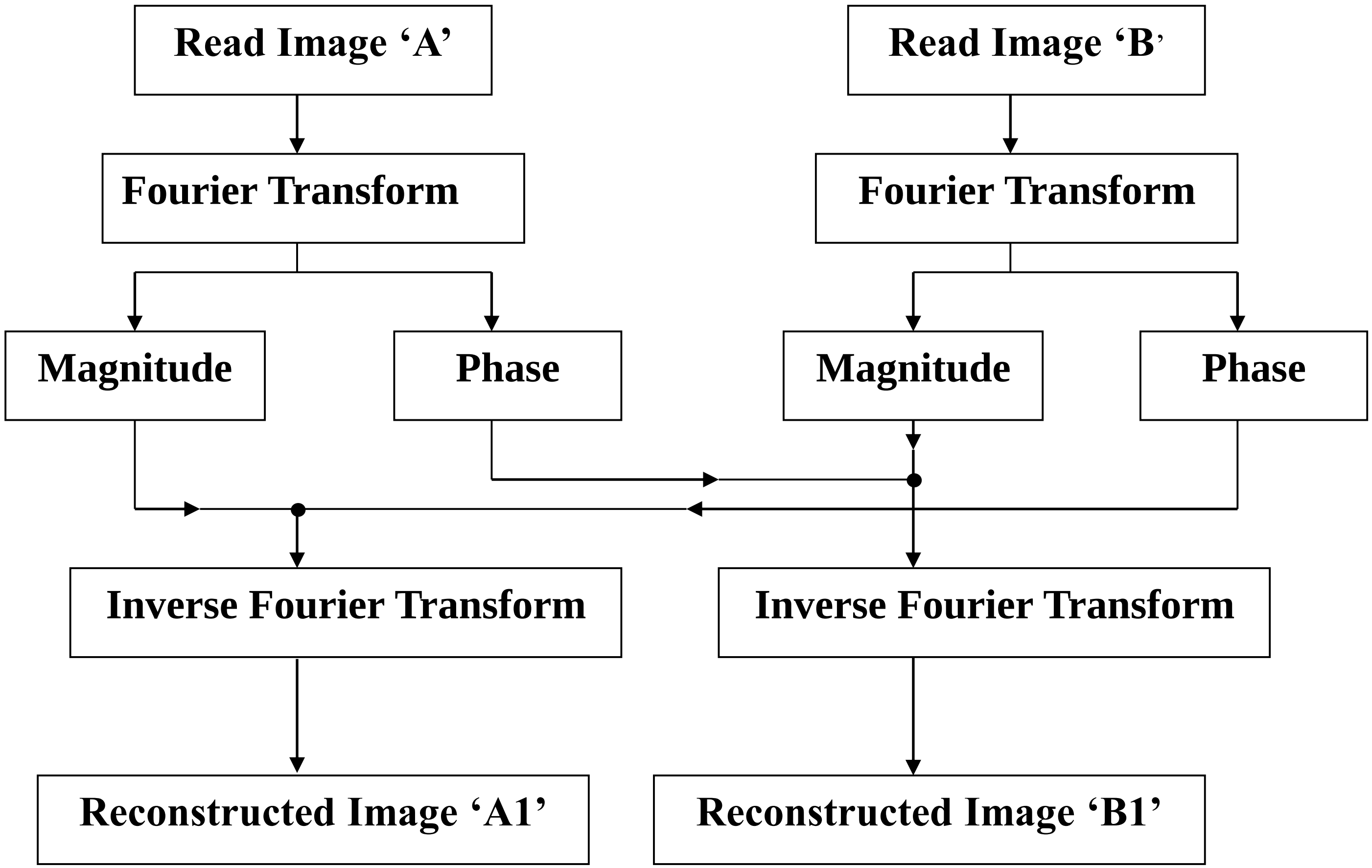


Figure 3.7 *Computation of the 2-D Fourier transform as a series of 1-D transforms.*

Importance of Phase



Importance of Phase

Originalimage

OriginalImage

Cameraman image after phase reversal

Lenna image after phase reversal

2-D Fourier Transform

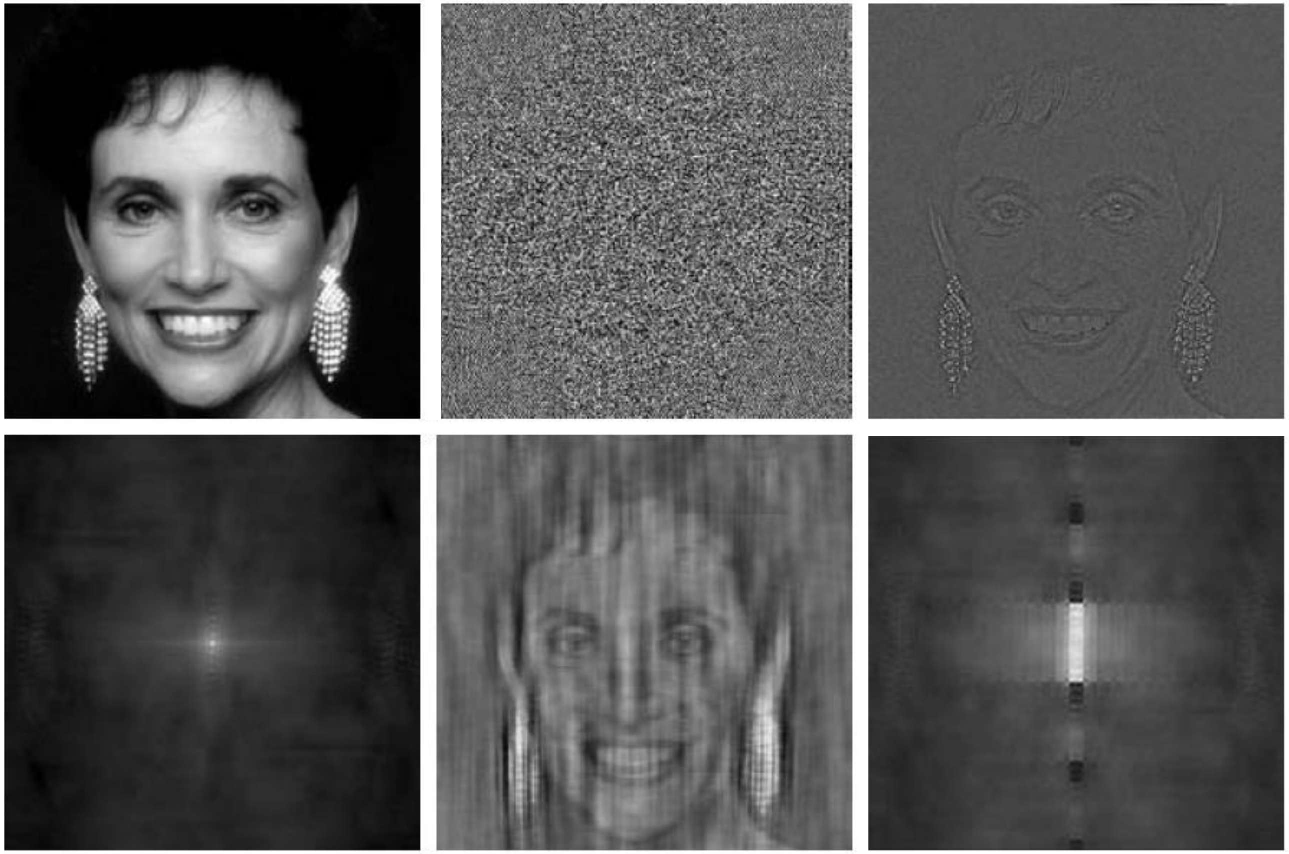


FIGURE 4.27 (a) Woman. (b) Phase angle. (c) Woman reconstructed using only the phase angle. (d) Woman reconstructed using only the spectrum. (e) Reconstruction using the phase angle corresponding to the woman and the spectrum corresponding to the rectangle in Fig. 4.24(a). (f) Reconstruction using the phase of the rectangle and the spectrum of the woman.

Transformations

2-D Fourier Transform Properties

Name	DFT Pairs
1) Symmetry properties	See Table 4.1
2) Linearity	$af_1(x, y) + bf_2(x, y) \Leftrightarrow aF_1(u, v) + bF_2(u, v)$
3) Translation (general)	$f(x, y)e^{j2\pi(u_0x/M + v_0y/N)} \Leftrightarrow F(u - u_0, v - v_0)$ $f(x - x_0, y - y_0) \Leftrightarrow F(u, v)e^{-j2\pi(ux_0/M + vy_0/N)}$
4) Translation to center of the frequency rectangle, $(M/2, N/2)$	$f(x, y)(-1)^{x+y} \Leftrightarrow F(u - M/2, v - N/2)$ $f(x - M/2, y - N/2) \Leftrightarrow F(u, v)(-1)^{u+v}$
5) Rotation	$f(r, \theta + \theta_0) \Leftrightarrow F(\omega, \varphi + \theta_0)$ $x = r\cos\theta \quad y = r\sin\theta \quad u = \omega\cos\varphi \quad v = \omega\sin\varphi$
6) Convolution theorem†	$f(x, y) \star h(x, y) \Leftrightarrow F(u, v)H(u, v)$ $f(x, y)h(x, y) \Leftrightarrow F(u, v) \star H(u, v)$

(Continued)

TABLE 4.3 Summary of DFT pairs. The closed-form expressions in 12 and 13 are valid only for continuous variables. They can be used with discrete variables by sampling the closed-form, continuous expressions.

2-D Fourier Transform Properties

Name	DFT Pairs
7) Correlation theorem†	$f(x, y) \star h(x, y) \Leftrightarrow F^*(u, v)\, H(u, v)$ $f^*(x, y) h(x, y) \Leftrightarrow F(u, v) \star H(u, v)$
8) Discrete unit impulse	$\delta(x, y) \Leftrightarrow 1$
9) Rectangle	$\text{rect}[a, b] \Leftrightarrow ab \dfrac{\sin(\pi ua)}{(\pi ua)} \dfrac{\sin(\pi vb)}{(\pi vb)} e^{-j\pi(ua+vb)}$
10) Sine	$\sin(2\pi u_0 x + 2\pi v_0 y) \Leftrightarrow$ $j\frac{1}{2}\left[\delta(u + Mu_0, v + Nv_0) - \delta(u - Mu_0, v - Nv_0)\right]$
11) Cosine	$\cos(2\pi u_0 x + 2\pi v_0 y) \Leftrightarrow$ $\frac{1}{2}\left[\delta(u + Mu_0, v + Nv_0) + \delta(u - Mu_0, v - Nv_0)\right]$
The following Fourier transform pairs are derivable only for continuous variables, denoted as before by t and z for spatial variables and by μ and ν for frequency variables. These results can be used for DFT work by sampling the continuous forms.	
12) *Differentiation* (The expressions on the right assume that $f(\pm\infty, \pm\infty) = 0$.)	$\left(\frac{\partial}{\partial t}\right)^m \left(\frac{\partial}{\partial z}\right)^n f(t, z) \Leftrightarrow (j2\pi\mu)^m (j2\pi\nu)^n F(\mu, \nu)$ $\dfrac{\partial^m f(t, z)}{\partial t^m} \Leftrightarrow (j2\pi\mu)^m F(\mu, \nu); \dfrac{\partial^n f(t, z)}{\partial z^n} \Leftrightarrow (j2\pi\nu)^n F(\mu, \nu)$
13) *Gaussian*	$A2\pi\sigma^2 e^{-2\pi^2\sigma^2(t^2+z^2)} \Leftrightarrow Ae^{-(\mu^2+\nu^2)/2\sigma^2}$ (A is a constant)

†Assumes that the functions have been extended by zero padding. Convolution and correlation are associative, commutative, and distributive.

Application of Fourier Transform in Image Processing

Image Filtering

$$f(m,n) \longrightarrow \boxed{h(m,n)} \longrightarrow g(m,n)$$

$f(m,n)$ $\longrightarrow$ **Input image**

$h(m,n)$ $\longrightarrow$ **Impulse response of the filter**

$g(m,n)$ $\longrightarrow$ **Output image**

$$g(m,n) = f(m,n) * h(m,n)$$

Application of Fourier Transform in Image Processing

$$f(m,n) \rightarrow \boxed{\text{DFT}} \xrightarrow{F(k,l)} \boxed{H(k,l)} \xrightarrow{G(k,l)} \boxed{\text{IDFT}} \rightarrow g(m,n)$$

$F(k,l) \longrightarrow$ Spectrum of input signal

$H(k,l) \longrightarrow$ Filter function

$g(m,n) \longrightarrow$ Output image

Convolution in time domain = Multiplication in frequency domain

DISCRETE COSINE TRANSFORM

$$F[k,l] = \alpha(k)\alpha(l)\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} f(m,n)\cos\left[\frac{(2m+1)\pi k}{2N}\right]\cos\left[\frac{(2n+1)\pi l}{2N}\right]$$

$$\alpha(k) = \begin{cases} \sqrt{\dfrac{1}{N}} & if\ \ k = 0 \\ \sqrt{\dfrac{2}{N}} & if\ k \neq 0 \end{cases}$$

$$\alpha(l) = \begin{cases} \sqrt{\dfrac{1}{N}} & if\ \ l = 0 \\ \sqrt{\dfrac{2}{N}} & if\ l \neq 0 \end{cases}$$

Discrete cosine transform

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

$$\alpha(k) = \left\{ \begin{array}{ll} \sqrt{\frac{1}{N}} & \text{if } k=0 \\ \sqrt{\frac{2}{N}} & \text{otherwise} \end{array} \right\}.$$

The strength of the ‘u’ sinusoid is given by C(u)

- Project f onto the basis function
- All samples of f contribute the coefficient
- C(0) is the zero-frequency component – the average value!

Consider a digital image such that one row has the following samples

Index	0	1	2	3	4	5	6	7
Value	20	12	18	56	83	10	104	114

There are 8 samples so N=8

u is in [0, N-1] or [0, 7]

Must compute 8 DCT coefficients: C(0), C(1), …, C(7)

Start with C(0)

$$C(0) = \sqrt{\frac{1}{N}} \sum_{x}^{N-1} f(x)$$

$$
\begin{aligned}
C(0) &= \sqrt{\tfrac{1}{8}}\sum_{x=0}^{7} f(x)\cos\left(\frac{(2x+1)\cdot 0\pi}{2\cdot 8}\right) \\
&= \sqrt{\tfrac{1}{8}}\sum_{x=0}^{7} f(x)\cos(0) \\
&= \sqrt{\tfrac{1}{8}}\cdot\{f(0)+f(1)+f(2)+f(3)+f(4)+f(5)+f(6)+f(7)\} \\
&= .35\cdot\{20+12+18+56+83+110+104+115\} \\
&= 183.14
\end{aligned}
$$

Repeating the computation for all u we obtain the following coefficients

Spatial domain

f(0)	f(1)	f(2)	f(3)	f(4)	f(5)	f(6)	f(7)
20	12	18	56	83	110	104	114

Frequency domain

$C(0)$	$C(1)$	$C(2)$	$C(3)$	$C(4)$	$C(5)$	$C(6)$	$C(7)$
183.1	-113.0	-4.2	22.1	10.6	-1.5	4.8	-8.7

DCT (2D)

The 2D DCT is given below where the definition for alpha is the same as before

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

For an MxN image there are MxN coefficients

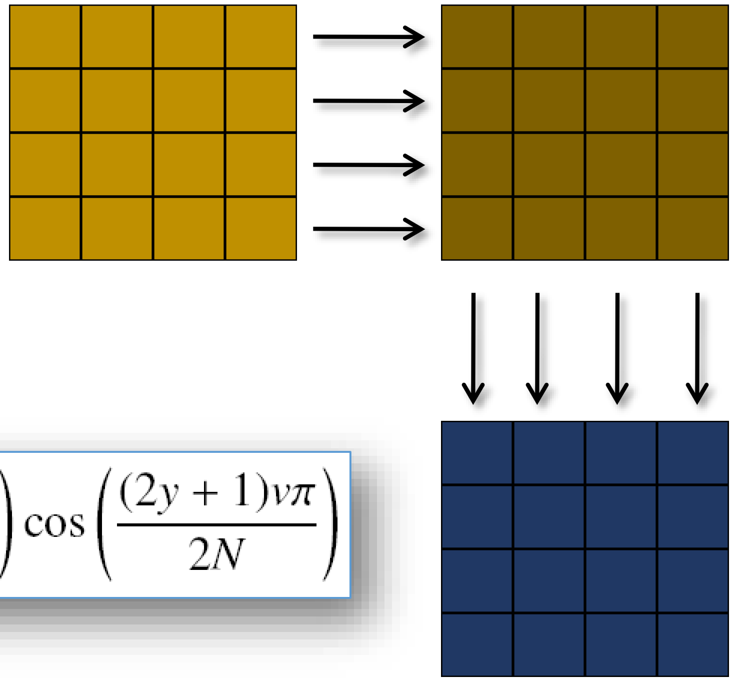
Each image sample contributes to each coefficient

Each (u,v) pair corresponds to a ‘pattern’ or ‘basis function’

Separability

The DCT is separable

- The coefficients can be obtained by computing the 1D coefficients for each row
- Using the row-coefficients to compute the coefficients of each column (using the 1D forward transform)



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

Invertability

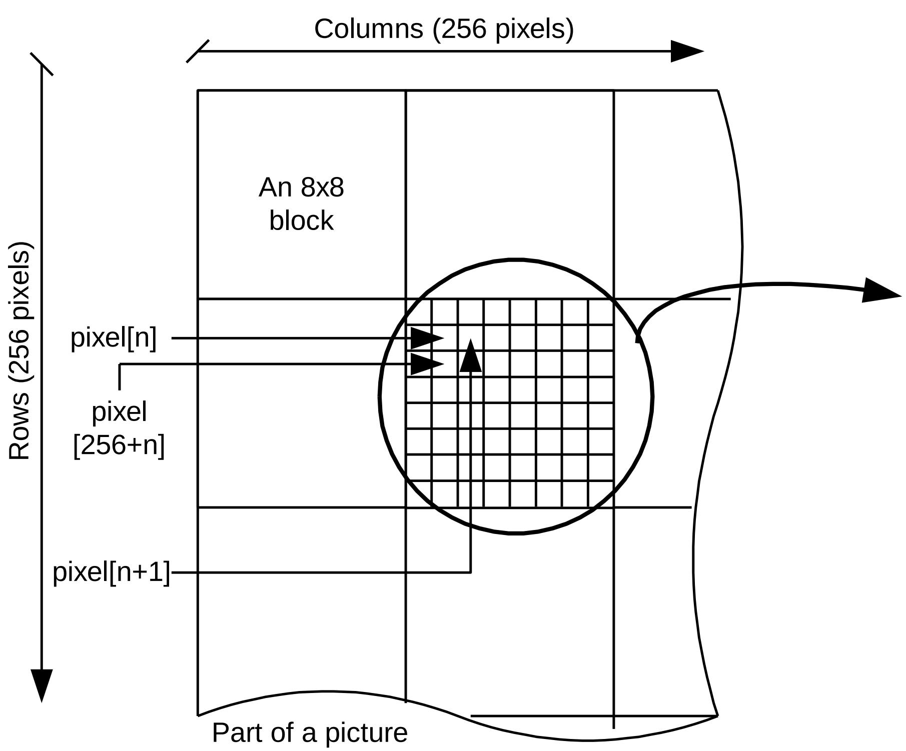
The DCT is invertible

Spatial samples can be recovered from the DCT coefficients

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

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

Transformations



Raw Block of Pixels

128	127	130	128	134	130	128	128
128	127	130	128	134	130	128	128
125	126	130	127	130	131	128	127
124	128	127	126	129	130	127	127
131	132	130	129	132	128	129	131
129	131	134	131	133	129	131	131
129	134	134	136	136	137	134	132
133	135	136	138	137	140	135	132

Result after DCT by rows

365	-1	-4	1	1	-3	1	3
365	-1	-4	1	1	-3	1	3
362	-3	-4	0	-3	-1	2	1
359	-3	-3	0	-3	-3	0	-1
368	1	1	-1	1	-3	-2	1
370	0	-3	-3	-1	-1	-1	2
378	-3	-6	0	-3	-2	-1	-2
383	-1	-6	2	-3	0	0	-3

Final result after DCT by columns

1024	-4	-11	0	-4	-6	0	1
-18	-1	1	0	2	-2	2	4
11	-1	-5	2	0	0	1	-1
0	2	2	-3	3	-2	-2	3
0	1	1	1	0	-1	-1	-2
-4	-3	-3	-1	-3	-1	0	-2
-2	0	-1	-2	-1	1	1	0
2	0	1	0	0	-1	-1	-1

Summary of DCT

The DCT provides energy compaction

- Low frequency coefficients have larger magnitude (typically)
- High frequency coefficients have smaller magnitude (typically)
- Most information is compacted into the lower frequency coefficients (those coefficients at the 'upper-left')

Compaction can be leveraged for compression

- Use the DCT coefficients to store image data but discard a certain percentage of the high-frequency coefficients!
- JPEG does this

DCT can be obtained from DFT

Remark: It is possible to extract DCT from DFT. This becomes rather obvious from the following presentation:

$$C(0) = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} f(x)$$

and

$$C(u) = \sqrt{\frac{2}{N}} \operatorname{Re}\left\{ \left[e^{\frac{-ju\pi}{2N}} \right] \sum_{x=0}^{2N-1} f(x) e^{\frac{-jux}{2N}} \right\}$$

This can be achieved by taking 2N-point discrete Fourier Transform.

Discrete Sine Transform(DST)

$$v(k)=\sqrt{\frac{2}{N+2}}\sum_{n=0}^{N-1}u(n)\sin\left[\frac{\pi(k+1)(n+1)}{N+1}\right], \quad 0\le k\le N-1$$

$$u(n)=\sqrt{\frac{2}{N+1}}\sum_{k=0}^{N-1}v(k)\sin\left[\frac{\pi(k+1)(n+1)}{N+1}\right], \quad 0\le n\le N-1$$

$$\phi=\varphi(k,n)=\left\{\sqrt{\frac{2}{N+1}}\sin\frac{\pi(k+1)(n+1)}{N+1}\right\}, \qquad 0\le k,n\le N-1$$

- $S^{*}=S=S^{T}=S^{-1}$
- fast algorithm
- DST is close to the KLT, provided that $|\rho|\le 0.5$

Similar to DCT.

Walsh Transform

Walsh basis function:

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

$$m = \log_2 N$$

n **time index**

k **frequency index**

N **Order**

Walsh

Walsh basis for N=4

Step 1: To compute ‘m’

$$m = \log_2 N \qquad m = \log_2 4 = 2$$

Step 2: Substituting N=4 and m=2 in ***g(n,k)***

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

$$g(n,k) = \frac{1}{4}\prod_{i=0}^{2-1}(-1)^{b_i(n)b_{2-1-i}(k)}$$

$$\boxed{g(n,k) = \frac{1}{4}\prod_{i=0}^{1}(-1)^{b_i(n)b_{1-i}(k)}}$$

Walsh basis for N=4

Step 3: Substituting n=0 and k=0 in g(n,k)

$$g(n,k) = \frac{1}{4}\prod_{i=0}^{1}(-1)^{b_i(n)b_{1-i}(k)}$$

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

Walsh basis for N=4

Step 4: Substituting i = 0 and i = 1 in g(0,0)

$$g(0,0) = \frac{1}{4}\left\{(-1)^{b_0(0)b_1(0)} \times (-1)^{b_1(0)b_0(0)}\right\}$$

$b_0(0)$ **Zeroth bit position in the binary value of zero**

$b_1(0)$ **First bit position in the binary value of zero**

0 $\xrightarrow{\text{binary representation}}$ 0 0

Zeroth bit position

First bit position

Walsh basis for N=4

Step 5: Substitute $b_0(0)=0$ and $b_1(0)=0$ in g(0,0)

$$g(0,0)=\frac{1}{4}\left\{(-1)^0\times(-1)^0\right\}=\frac{1}{4}$$

To compute g(2,1)

$$g(n,k)=\frac{1}{4}\prod_{i=0}^{1}(-1)^{b_i(n)b_{1-i}(k)}$$

$$g(2,1)=\frac{1}{4}\prod_{i=0}^{1}(-1)^{b_i(2)b_{1-i}(1)}$$

Walsh basis for N=4

$$g(2,1) = \frac{1}{4}\prod_{i=0}^{1}(-1)^{b_i(2)b_{1-i}(1)}$$

Substituting i=0 and i=1 in the expression of g(2,1)

$$g(2,1) = \frac{1}{4}\left\{(-1)^{b_0(2)b_1(1)} \times (-1)^{b_1(2)b_0(1)}\right\}$$

$b_0(2)$	Zeroth bit position in the binary value of 2
$b_1(1)$	First bit position in the binary value of 1
$b_1(2)$	First bit position in the binary value of 2
$b_0(1)$	Zeroth bit position in the binary value of 1

Walsh basis for N=4

Decimal number	Binary representation
0	0 0
1	0 1
2	1 0
3	1 1

Zeroth bit position

First bit position

$b_0(2) = 0$

$b_1(1) = 0$

$b_1(2) = 1$

$b_0(1) = 1$

$$g(2,1) = \frac{1}{4}\left\{(-1)^{b_0(2)b_1(1)} \times (-1)^{b_1(2)b_0(1)}\right\}$$

$$g(2,1) = \frac{1}{4}\left\{(-1)^{0} \times (-1)^{1}\right\} = -\frac{1}{4}$$

Transformations

Walsh basis for N=4

$$g(n,k)=\begin{pmatrix} +\frac{1}{4} & +\frac{1}{4} & +\frac{1}{4} & +\frac{1}{4} \\ +\frac{1}{4} & +\frac{1}{4} & -\frac{1}{4} & -\frac{1}{4} \\ +\frac{1}{4} & -\frac{1}{4} & +\frac{1}{4} & -\frac{1}{4} \\ +\frac{1}{4} & -\frac{1}{4} & -\frac{1}{4} & +\frac{1}{4} \end{pmatrix}\begin{matrix} \rightarrow \text{No sign change} \\ \rightarrow \text{One sign change} \\ \rightarrow \text{Three sign changes} \\ \rightarrow \text{Two sign changes} \end{matrix}$$

No sign change

One sign change

Three sign changes

Two sign changes

Walsh Basis for N=4

$$g(n,k) = \begin{pmatrix} +\frac{1}{4} & +\frac{1}{4} & +\frac{1}{4} & +\frac{1}{4} \\ +\frac{1}{4} & +\frac{1}{4} & -\frac{1}{4} & -\frac{1}{4} \\ +\frac{1}{4} & -\frac{1}{4} & +\frac{1}{4} & -\frac{1}{4} \\ +\frac{1}{4} & -\frac{1}{4} & -\frac{1}{4} & +\frac{1}{4} \end{pmatrix}$$

n=1 and k=2

$n=1 \longrightarrow 0\ 1$

$k=2 \xrightarrow{\text{original}} 1\ 0$

$k=2 \xrightarrow{\text{reversal}} 0\ 1$

$\begin{matrix} 0 & 1 \\ 0 & 1 \end{matrix} \longrightarrow$ No. of overlap is odd

No. of overlap of one is odd then sign is negative

Transformations

Values of the 1-D Walsh Transformation Kernel for N=8

$$\frac{1}{8}g(x,u)$$

u \ x	0	1	2	3	4	5	6	7
0	+	+	+	+	+	+	+	+
1	+	+	+	+	-	-	-	-
2	+	+	-	-	+	+	-	-
3	+	+	-	-	-	-	+	+
4	+	-	+	-	+	-	+	-
5	+	-	+	-	-	+	-	+
6	+	-	-	+	+	-	-	+
7	+	-	-	+	-	+	+	-

We have done it earlier in different ways

Here we calculate the matrix of Walsh coefficients

Symmetry of Walsh

Looking at this 8 x 8 matrix, realize that it is a symmetric one.

The inverse kernel transform is given by:

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

which results

$$f(x)=\sum_{u=0}^{N-1}W(u)\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}$$

Think about other transforms that you know, are they symmetric?

Two-Dimensional Walsh Transform

Note that the only difference between the forward and inverse transform is $\boxed{1/N \text{ constant.}}$ Now, we go to 2-dimensional transformation we will have:

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

and

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

Two-dimensional Walsh

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

and

Inverse Two-dimensional Walsh

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

Properties of Walsh Transforms

Three observations:

❶ By splitting constant 1/N between the forward and inverse 2-dimensional kernels, we have made them identical which works perfectly for image processing applications.

❷ It is possible to separate two-dimensional transform into 2 one-dimensional ones.

Result of these two observations suggests that to take 2-D Walsh Transform, it is possible to take two 1-D Walsh Transform and get the same result. Further more, they suggest that same algorithm can be used to perform inverse Walsh Transform.

Here is the separable 2-Dim Inverse Walsh

$$g(x,y,u,v) = g_1(x,u)g_1(y,v) = h_1(x,u)h_1(y,v)$$

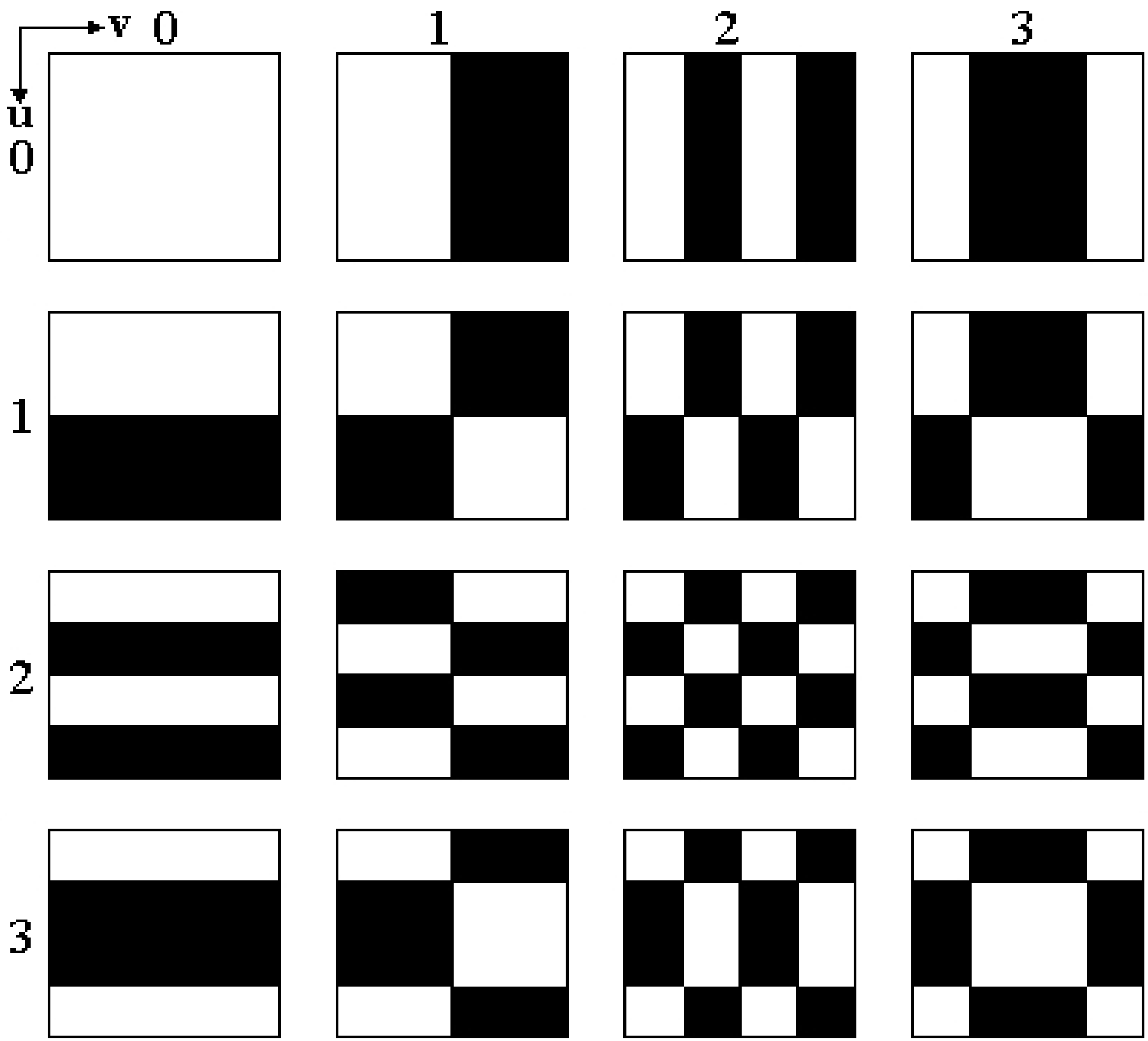
$$= \left[\frac{1}{\sqrt{N}}\prod_{i=o}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}\right]\left[\frac{1}{\sqrt{N}}\prod_{i=o}^{n-1}(-1)^{b_i(y)b_{n-1-i}(v)}\right]$$

❸ Because the Kernel matrix is symmetric, it is possible to use the same fast algorithm derived for Fourier on the base of successive-doubling method with the following modification. Let $W_N = \exp(-j2\pi/N)$ be equal to ± 1.

In a sense, the same principle of splitting sequence into odd and even is used.

$$\begin{cases} W(u) = \frac{1}{2}[W_{even}(u) + W_{odd}(u)] \\ W(u+\frac{N}{2}) = \frac{1}{2}[W_{even}(u) - W_{odd}(u)] \end{cases} , 0 \leq u \leq \frac{N}{2} - 1$$

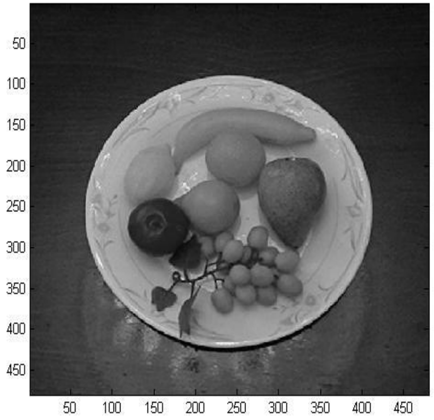
Walsh basis functions for N = 4. Each block consist of 4 x 4 elements, corresponding to x and y varying from 0 to 3.



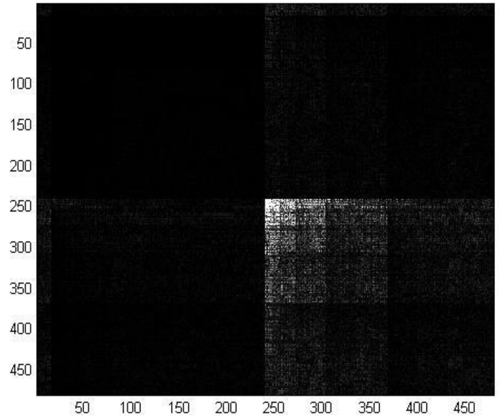
Discuss the importance of this figure

Walsh Transform

Example

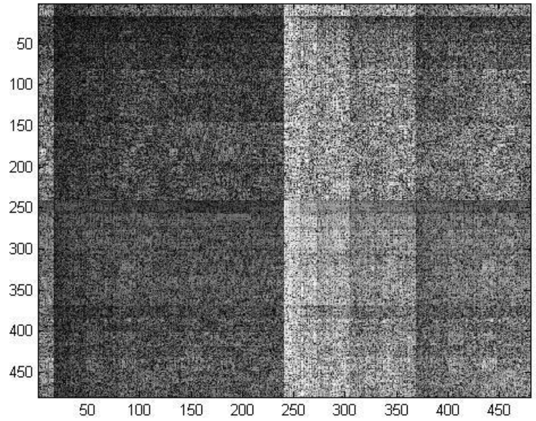


Main Image (Gray Level)



WT of Main image

(Walsh spectrum)



Logarithmic scaled

of the Walsh spectrum

HADAMARD TRANSFORM

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

Jacques Salomon Hadamard

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

Hadamard Transform

Transformation kernel is

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

Hence;

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

where $N = 2^n$

The Hadamard kernel is an orthogonal matrix $\Rightarrow$

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

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

Two-dimensional Hadamard transforms are

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

and,

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

Again notice that coefficient $1/N^2$ is divided between the pair so, one can use the transform for inverse transform and also just like other orthogonal functions kernel is separable.

$$g(x,y,u,v)=g_1(x,u)\,g_1(y,v)=h_1(x,u)\,h_1(y,v)$$

$$=\left[\frac{1}{\sqrt{N}}(-1)^{\sum_{i=0}^{n-1}b_i(x)b_i(u)}\right]\left[\frac{1}{\sqrt{N}}(-1)^{\sum_{i=0}^{n-1}b_i(y)b_i(v)}\right]$$

separabl

Two important facts:

❶ All the summation on the power is based on modulo 2.

❷ The formulation of Hadamard, the way it is done, is only good for $N = 2^n$, for values of N other than 2^n, up to N = 200 it is shown that Hadamard transform exist.

Lets look at kernel matrix for N = 8 and disregard $1/\sqrt{N}$.

$$g(0,0) = (-1)^{\sum_{i=0}^{2} b_i(0)b_i(0)} = (-1)^{(0+0+0)} = (+)$$

$$g(0,1) = (-1)^{\sum_{i=0}^{2} b_i(0)b_i(1)} = (-1)^{(0+0+0)} = (+)$$

Example of calculating Hadamard coefficients – analogous to what was before

$$g(1,0)=(-1)^{\sum_{i=0}^{2} b_i(1)b_i(0)}=(-1)^{(0+0+0)}=(+)$$

$$\vdots$$

$$g(1,1)=(-1)^{\sum_{i=0}^{2} b_i(1)b_i(1)}=(-1)^{(1+0+0)}=(-)$$

$$\vdots$$

$$g(7,1)=(-1)^{\sum_{i=0}^{2} b_i(7)b_i(1)}=(-1)^{(1+0+0)}=(-)$$

$$\vdots$$

$$g(7,7)=(-1)^{\sum_{i=0}^{2} b_i(7)b_i(7)}=(-1)^{(1+1+1)}=(-)$$

Values of the 1-D Hadamard Transformation Kernel for N=8

$$\frac{1}{\sqrt{8}} g_H(x,u)$$

u \ x	0	1	2	3	4	5	6	7
0	+	+	+	+	+	+	+	+
1	+	-	+	-	+	-	+	-
2	+	+	-	-	+	+	-	-
3	+	-	-	+	+	-	-	+
4	+	+	+	+	-	-	-	-
5	+	-	+	-	-	+	-	+
6	+	+	-	-	-	-	+	+
7	+	-	-	+	-	+	+	-

- If you look closely to Hadamard kernel, it is exactly the same as Walsh, but some of the rows are intermixed.
- In image processing area, in some cases, they are called Walsh-Hadamard transform.
- Successive doubling format allows computation of the FWT by a straight forward modification of the FFT algorithm.
- The FHT is the extension of FWT by taking into account the difference in ordering.
- Although the Hadamard ordering has disadvantages in terms of a successive doubling, it leads to a simple recursive relationship for generating the transformation matrices needed .

The Hadamard matrix of lowest order (N = 2) is

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

Then letting $\mathbf{H}_N$ represent the matrix of order N, the recursive relationship is given by the expression:

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

Using matrix format $\boldsymbol{H} = \boldsymbol{AFA}$ suggests that $A = \frac{1}{\sqrt{N}} H_N$.

Fact! The number of sign changes along a column of the Hadamard matrix is often called the ***sequency*** of that column. For instance, $\mathbf{H_8}$ has a sequency of

0, 7, 3, 4, 1, 6, 2, 5

- It is important to order the sequency such that it increase as u increases.
- For one-dimensional Hadamard this ordering can be accomplished through the following relation:

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

Where, $p_0(u) = b_{n-1}(u)$

$$p_1(u) = b_{n-1}(u) + b_{n-2}(u)$$

$$p_2(u) = b_{n-2}(u) + b_{n-3}(u)$$

$$\vdots$$

$$p_{n-1}(u) = b_1(u) + b_0(u)$$

and recursive formula is $\Rightarrow \; p_i(u) = b_{n-i}(u) + b_{n-i-1}(u)$

$$\begin{cases} H(u) = \dfrac{1}{N}\displaystyle\sum_{x=0}^{N-1} f(x)(-1)^{\sum_{i=0}^{n-1} b_i(u)p_i(u)} \\ f(x) = \dfrac{1}{N}\displaystyle\sum_{x=0}^{N-1} H(u)(-1)^{\sum_{i=0}^{n-1} b_i(u)p_i(u)} \end{cases}$$

In the same manner, the 2-D transform pair can be written as :

$$\begin{cases} H(u,v) = \dfrac{1}{N}\displaystyle\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)(-1)^{\sum_{i=0}^{n-1}[b_i(x)p_i(u)+b_i(y)p_i(v)]} \\ f(x,y) = \dfrac{1}{N}\displaystyle\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} H(u,v)(-1)^{\sum_{i=0}^{n-1}[b_i(x)p_i(u)+b_i(y)p_i(v)]} \end{cases}$$

How does the ordering look for one dimensional kernel?

Standard Trivial Functions for Hadamard

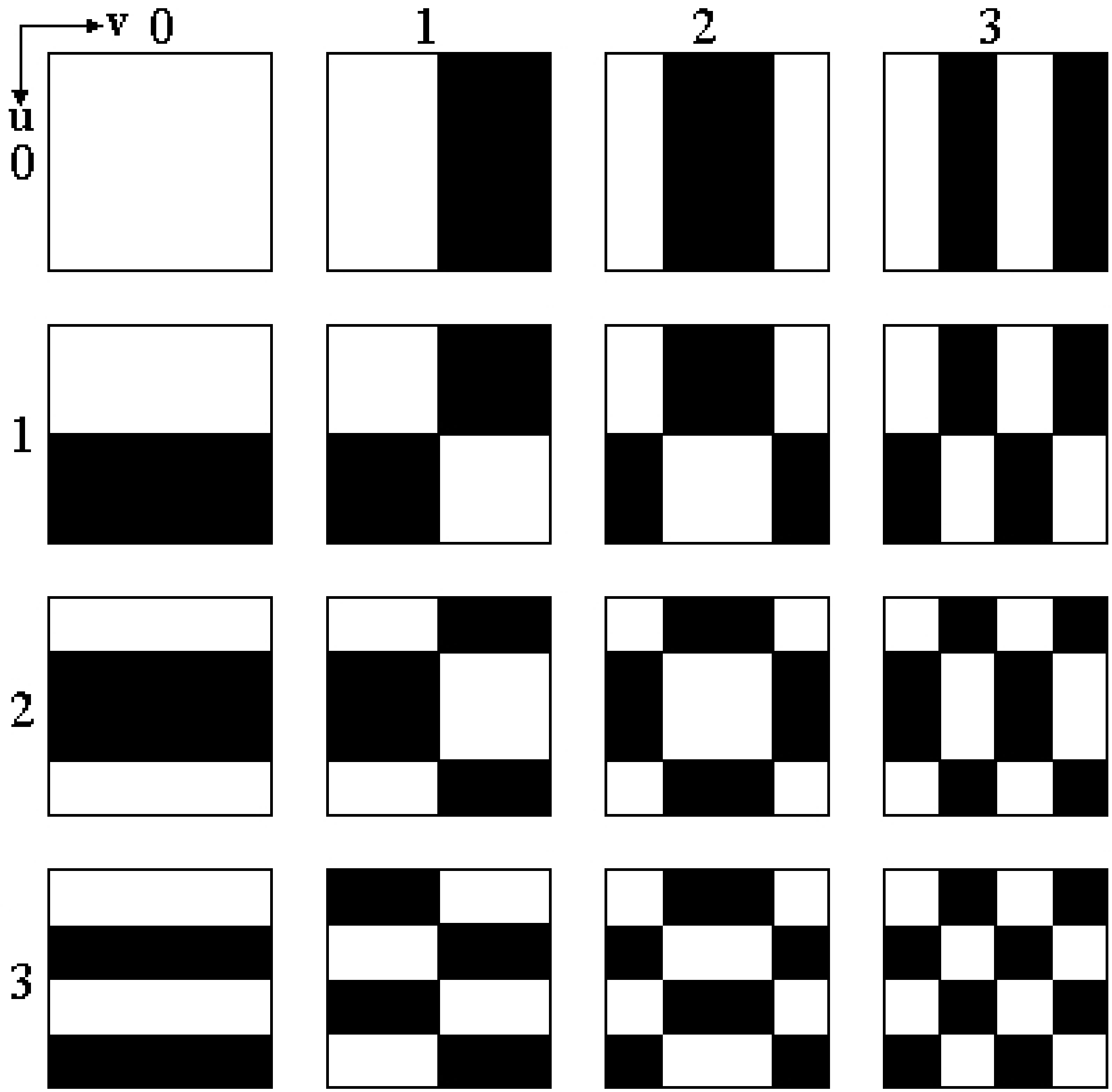
Values of the 1-D Ordered Hadamard Kernel for N=8

u \ x	0	1	2	3	4	5	6	7
0	+	+	+	+	+	+	+	+
1	+	+	+	+	-	-	-	-
2	+	+	-	-	-	-	+	+
3	+	+	-	-	+	+	-	+
4	+	-	-	+	+	-	-	+
5	+	-	-	+	-	+	+	-
6	+	-	+	-	-	+	-	+
7	+	-	+	-	+	-	+	-

One change

two changes

Ordered Hadamard basis functions for N = 4. Each block consist of 4 x 4 elements, corresponding to x and y varying from 0 to 3.



Discrete Walsh-Hadamard transform

1) Hadamard transform

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

$$H_n = H_{n-1} \otimes H_1 = \frac{1}{\sqrt{2}}\left[\begin{array}{c|c} H_{n-1} & H_{n-1} \\ \hline H_{n-1} & -H_{n-1} \end{array}\right]$$

$\#$ of sign changes

$\Rightarrow$ sequency

$$H_2 = \frac{1}{2}\begin{array}{c} \left[\begin{array}{cc|cc} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ \hline 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{array}\right] \end{array} \begin{array}{c} 0 \\ 3 \\ 1 \\ 2 \end{array}$$

$$H_3 = \frac{1}{\sqrt{8}}\left[\begin{array}{c|c} H_2 & H_2 \\ \hline H_2 & -H_2 \end{array}\right] = \frac{1}{\sqrt{8}}\begin{array}{c} \left[\begin{array}{rrrr|rrrr} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ \hline 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{array}\right] \end{array} \begin{array}{c} \textit{sequency} \\ 0 \\ 7 \\ 3 \\ 4 \\ 1 \\ 6 \\ 2 \\ 5 \end{array}$$

$\Rightarrow$ natural or Hadamard order

$\Rightarrow$ also can be generated by sampling the Walsh function (1923, Walsh)

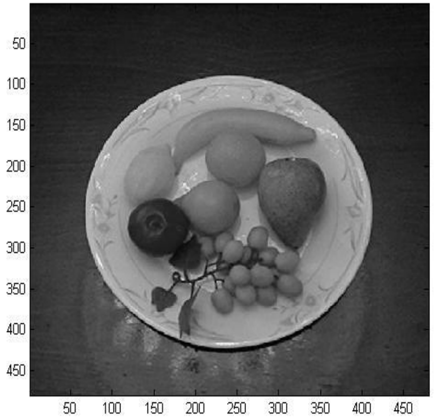
$\Rightarrow$ Walsh - Hadamard transform

$$
\begin{array}{cc}
 & \text{sequency} \\
H_3 = \frac{1}{\sqrt{8}}\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\
1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\
1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\
1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\
1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\
1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1
\end{bmatrix} & \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array}
\end{array}
$$

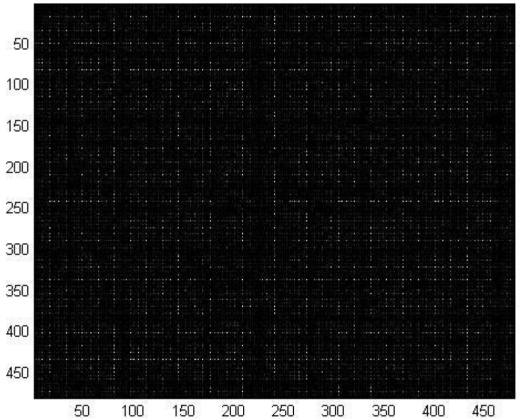
$\Rightarrow$ sequency or Walsh order

Hadamard Transform

Example

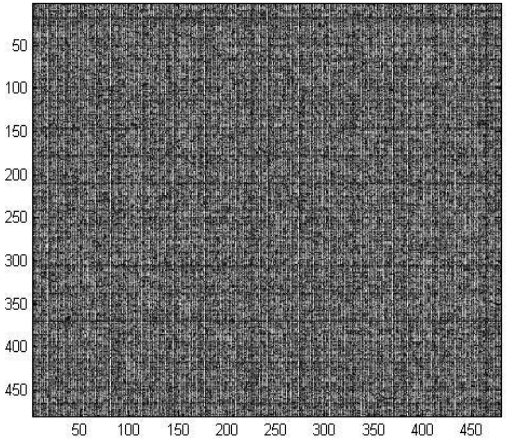


Main Image (Gray Level)



HT of Main image

(Hadamard spectrum)



Logarithmic scaled

of the Hadamard spectrum

Walsh-Hadamard Transform (WHT)

- An image $f(x, y)$ of size NxN (N=2^k)

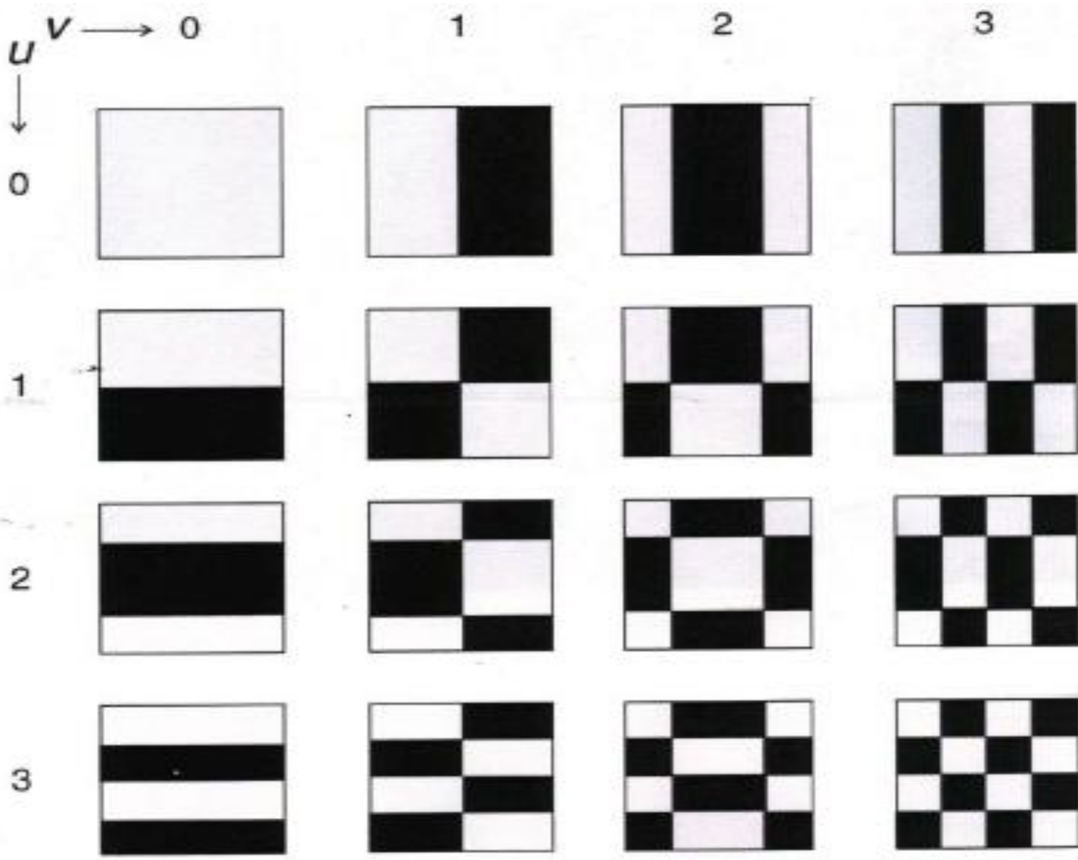
$$WH(u,v)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)(-1)^{\sum_{i=0}^{k-1}[b_i(x)p_i(u)+b_i(y)p_i(v)]}$$

  - The exponent on the (-1) is performed in modular 2 arithmetic.
  - $b_i(x)$ is the ith bit of the binary number of x.

$$k=3, N=2^3=8, \quad x=4=(100)_2 \quad : \quad b_2(x)=1, b_1(x)=0, b_0(x)=0$$

$$k=4, N=2^4=16, \quad x=2=(0010)_2 \quad : \quad b_3(x)=0, b_2(x)=0, b_1(x)=1, b_0(x)=0$$

2-D WHT Kernel Functions



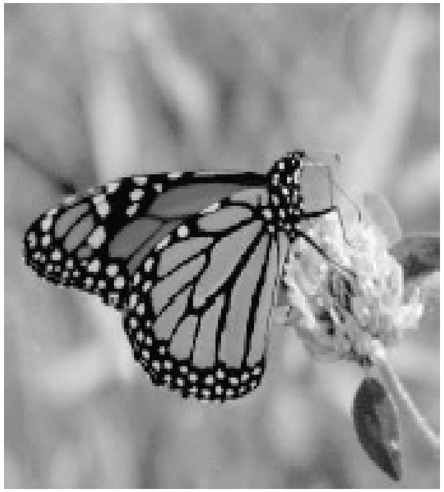
2D Walsh-Hadamard kernel functions (N=4)

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

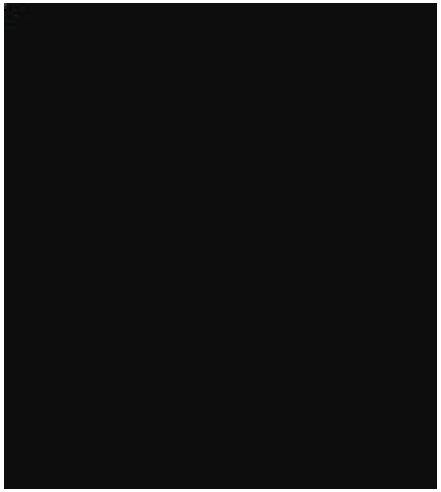
WHT and Fourier Transform

- The Walsh-Hadamard transform kernel functions are not sinusoids.
- Only square or rectangular waves with peaks of ± 1 are used in Walsh-Hadamard transform.
- One advantage of the Walsh-Hadamard transform is that the computations are very simple.

WHT Example



Original image



Walsh-Hadamard Transform linearly mapped

Walsh-Hadamard Transform logarithm remapped

Haar Transform

- The Haar transform is based on the Haar functions, $h_k(z)$
- Haar defined over the continuous, closed interval [0,1] for z
- Harr Tf. Based on the class of orthogonal matrices, whose elements are -1, 0, 1 multiplied by power of $\sqrt{2}$.
- $k=0,1,2,\ldots,N-1$, where $N=2^n$. The first step in generating the Haar transform is to note that the integer *k* can be decomposed uniquely as $k=2^p+q-1$
- where $0 \le p \le n-1$,
- $q=0$ or 1 for $p=0$
- $1 \le q \le 2^p$ for $p \ne 0$.

Haar Transform

$$h_0(z) \stackrel{\Delta}{=} h_{00}(z) = \frac{1}{\sqrt{N}} \qquad \text{for } z \in [0,1]$$

and

$$h_k(z) \stackrel{\Delta}{=} h_{00}(z) = \frac{1}{\sqrt{N}} \begin{cases} 2^{p/2} & \frac{q-1}{2^p} \leq z \prec \frac{q-1/2}{2^p} \\ -2^{p/2} & \frac{q-1/2}{2^p} \leq z \prec \frac{q}{2^p} \\ 0 & \text{otherwise for } z \in [0,1] \end{cases}$$

Haar transform matrix for sizes $N=2,4,8$

$$\mathrm{Hr}_2 = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

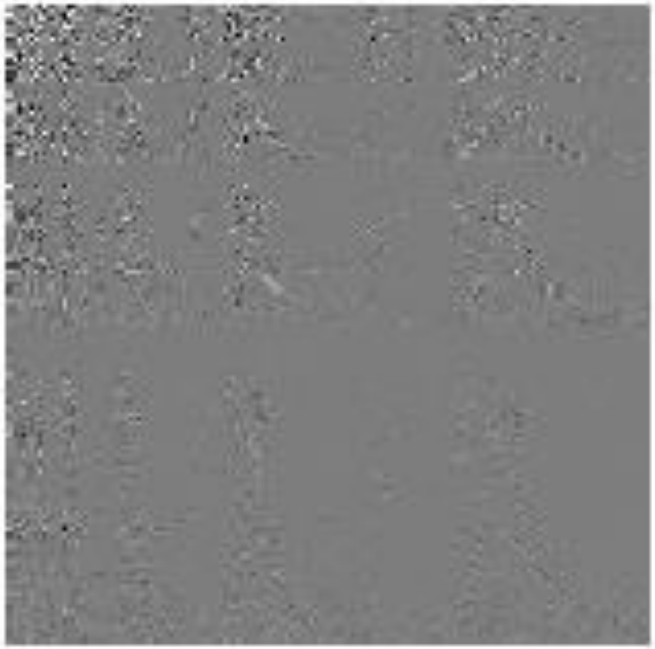
$$\mathrm{Hr}_4 = \frac{1}{\sqrt{4}}\begin{bmatrix} 1 & 1 & \sqrt{2} & 0 \\ 1 & 1 & -\sqrt{2} & 0 \\ 1 & -1 & 0 & \sqrt{2} \\ 1 & -1 & 0 & -\sqrt{2} \end{bmatrix}$$

$$\mathrm{Hr}_8 = \frac{1}{\sqrt{8}}\begin{bmatrix} 1 & 1 & \sqrt{2} & 0 & 2 & 0 & 0 & 0 \\ 1 & 1 & \sqrt{2} & 0 & -2 & 0 & 0 & 0 \\ 1 & 1 & -\sqrt{2} & 0 & 0 & 2 & 0 & 0 \\ 1 & 1 & -\sqrt{2} & 0 & 0 & -2 & 0 & 0 \\ 1 & -1 & 0 & \sqrt{2} & 0 & 0 & 2 & 0 \\ 1 & -1 & 0 & \sqrt{2} & 0 & 0 & -2 & 0 \\ 1 & -1 & 0 & -\sqrt{2} & 0 & 0 & 0 & 2 \\ 1 & -1 & 0 & -\sqrt{2} & 0 & 0 & 0 & -2 \end{bmatrix}$$

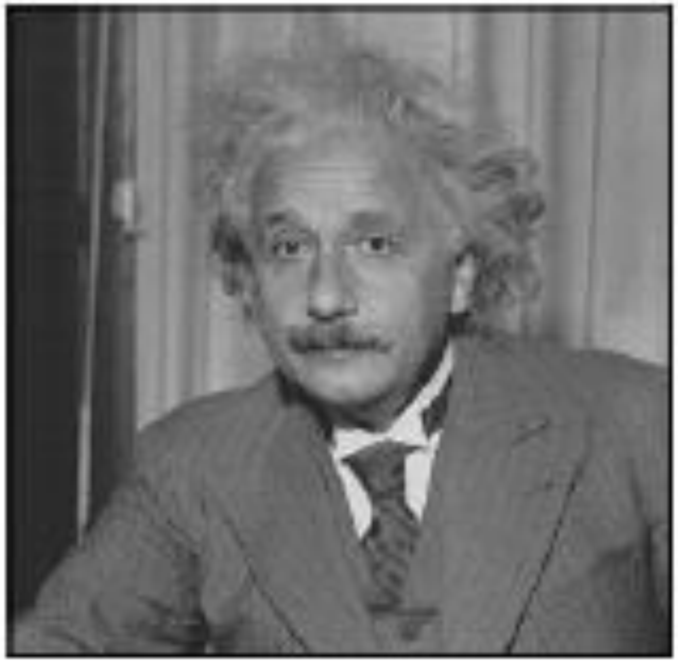
Can be computed by taking sums and differences.

Fast algorithms by recursively applying $\mathbf{Hr_2}$.

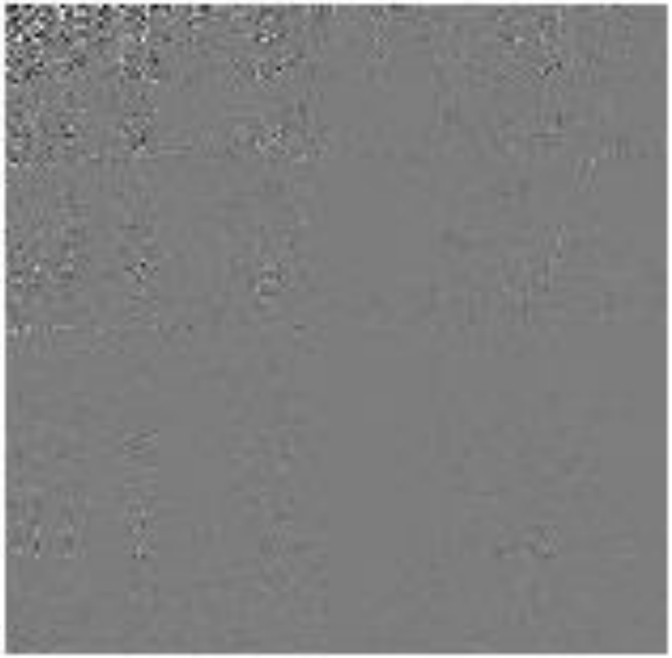
Original *Cameraman*
256x256



256x256 Haar transform
of *Cameraman*



Original *Einstein*
256x256



256x256 Haar transform
of *Einstein*