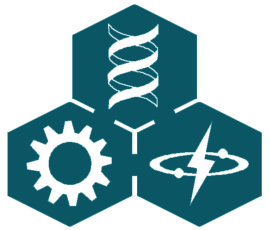
FILTERING IN THE FREQUENCY DOMAIN

- 4.1 Background
- 4.2 Preliminary Concepts
- 4.3 Sampling and the Fourier Transform of Sampled Functions
- 4.4 The Discrete Fourier Transform (DFT) of One Variable
- 4.5 Extension to Functions of Two Variables
- 4.6 Some properties of the 2-D Discrete Fourier Transform

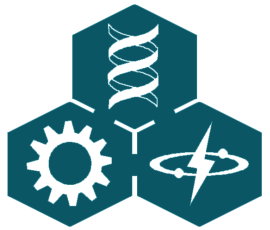
Prof. Ta-Te Lin
Dept. of Biomechatroincs Engineering, National Taiwan University



FILTERING IN THE FREQUENCY DOMAIN

- 4.7 The Basics of Filtering in the Frequency Domain
- 4.8 Image Smoothing Using Frequency Domain Filters
- 4.9 Image Sharpening Using Frequency Domain Filters
- 4.10 Selective Filtering
- 4.11 The Fast Fourier Transform

Prof. Ta-Te Lin
Dept. of Biomechatroincs Engineering, National Taiwan University



4.1 Background

- Jean Baptiste Joseph Fourier (1768~1830)

SOURCE: http://www-gap.dcs.st-and.ac.uk/~history/Mathematicians/Fourier.html

4.1 Background

- Fourier Series and Transform

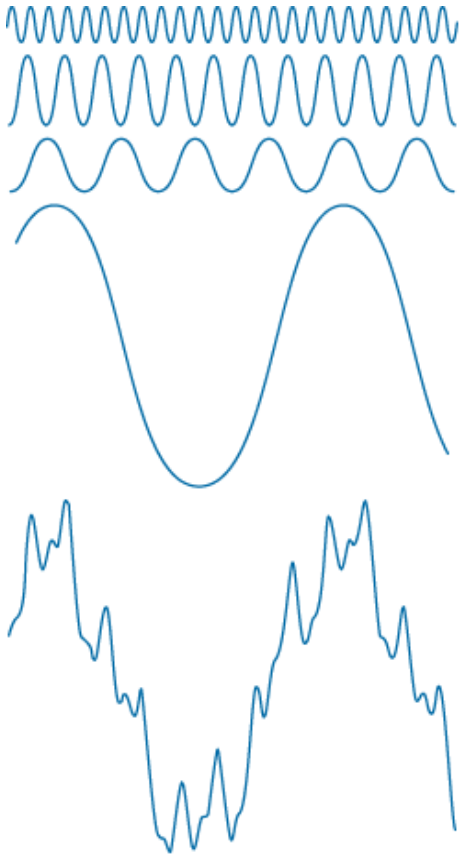


FIGURE 4.1
The function at the bottom is the sum of the four functions above it. Fourier's idea in 1807 that periodic functions could be represented as a weighted sum of sines and cosines was met with skepticism.

4.1 Background

- Fourier Series and Transform

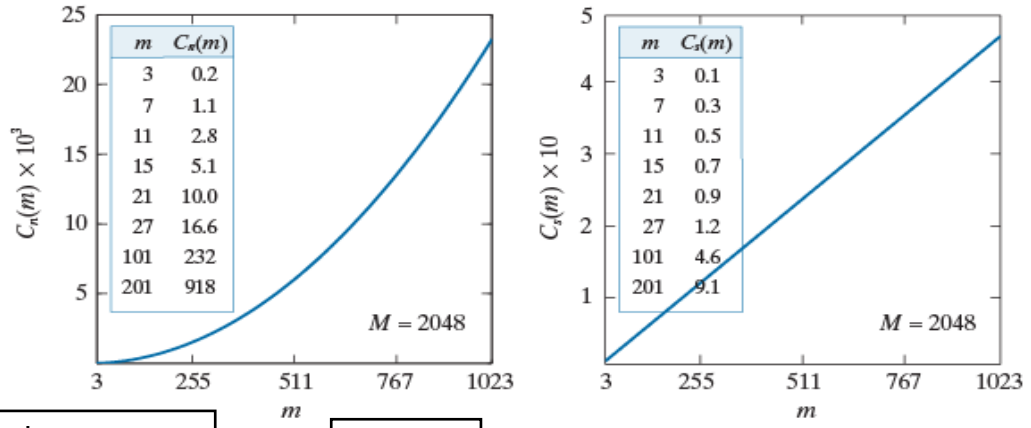
$$C_n(m) = \frac{M^2 m^2}{2M^2 \log_2 M^2} = \frac{m^2}{4 \log_2 M} \qquad (4\text{-}1)$$

If the kernel is separable, the advantage becomes

$$C_s(m) = \frac{2M^2 m}{2M^2 \log_2 M^2} = \frac{m}{2 \log_2 M} \qquad (4\text{-}2)$$

a b

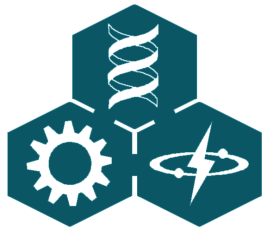
FIGURE 4.2
(a) Computational advantage of the FFT over non-separable spatial kernels.
(b) Advantage over separable kernels. The numbers for $C(m)$ in the inset tables are not to be multiplied by the factors of 10 shown for the curves.



fast fourier transform的效能

影像大小

比代傅立葉公式來的快



4.2 Preliminary Concepts

4.2.1 Complex Numbers

- Definition of Complex Numbers

$$C = R + jI \qquad (4\text{-}3)$$

- Conjugate

$$C^* = R - jI \qquad (4\text{-}4)$$

- Complex numbers in polar coordinates

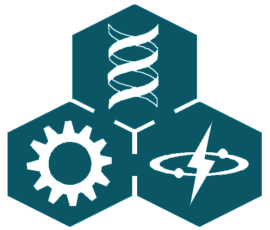
$$C = |C|(\cos\theta + j\sin\theta) \qquad (4\text{-}5)$$

$$|C| = \sqrt{R^2 + I^2}$$

- Euler's Formula

$$e^{j\theta} = \cos\theta + j\sin\theta \qquad (4\text{-}6)$$

$$C = |C| e^{j\theta} \qquad (4\text{-}7)$$



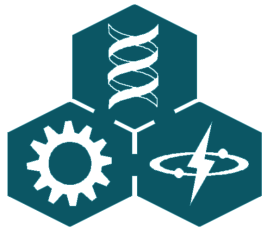
4.2 Preliminary Concepts

4.2.2 Fourier Series

- A function *f(t)* of a continuous variable *t* that is periodic with period, *T*, can be expressed as the sum of sines and cosines multiplied by appropriate coefficients

$$f(t) = \sum_{n=-\infty}^{\infty} c_n e^{j\frac{2\pi n}{T}t} \qquad (4\text{-}8)$$

$$c_n = \frac{1}{T}\int_{-T/2}^{T/2} f(t) e^{j\frac{2\pi n}{T}t} \qquad for\ n = 0, \pm 1, \pm 2, \ldots \qquad (4\text{-}9)$$



4.2 Preliminary Concepts

4.2.3 Impulses and Their Sifting Property

- Definition of *unit impulse*

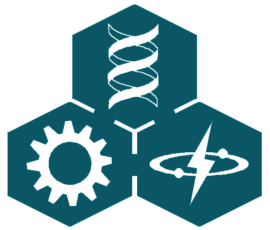
$$\delta(t) = \begin{cases} \infty & if\ t = 0 \\ 0 & if\ t \neq 0 \end{cases} \qquad (4\text{-}10)$$

$$\int_{-\infty}^{\infty} \delta(t)dt = 1 \qquad (4\text{-}11)$$

- Sifting property of *unit impulse*

$$\int_{-\infty}^{\infty} f(t)\delta(t)dt = f(0) \qquad (4\text{-}12)$$

$$\int_{-\infty}^{\infty} f(t)\delta(t-t_0)dt = f(t_0) \qquad (4\text{-}13)$$



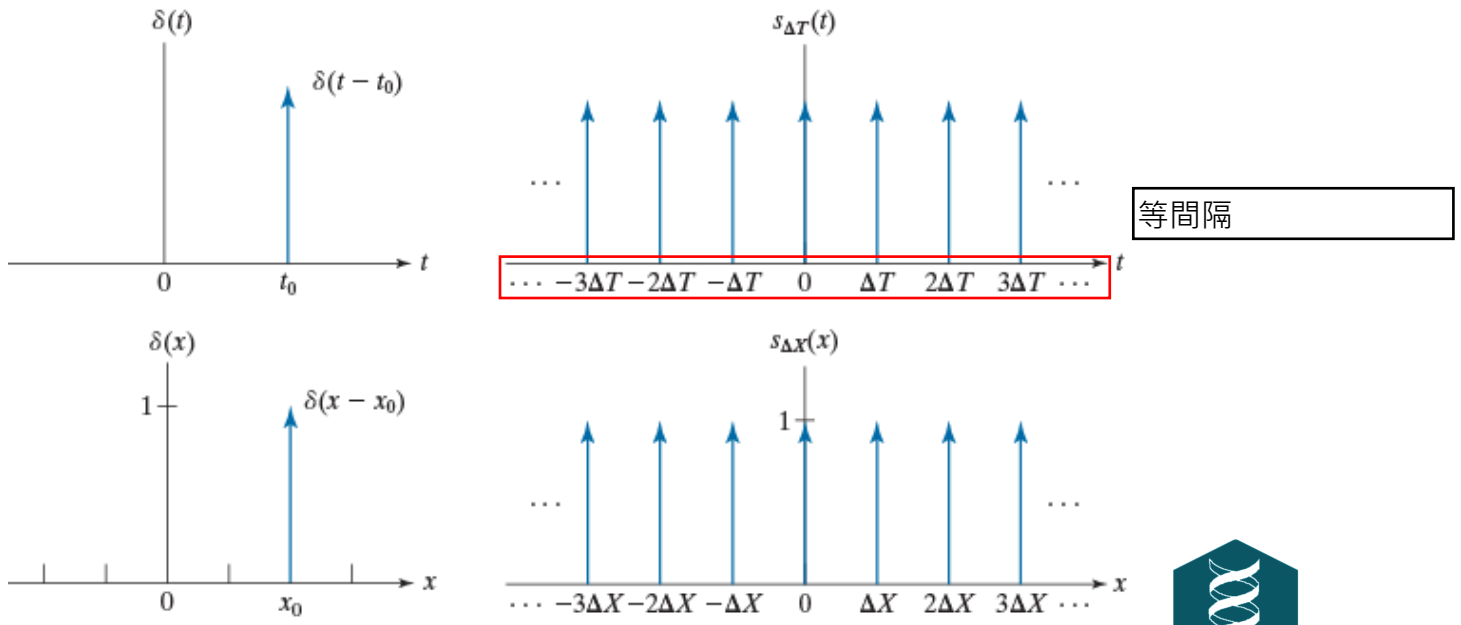
4.2 Preliminary Concepts

4.2.3 Impulses and Their Sifting Property

- **Impulse Train**

用來sample影像

$$s_{\Delta T}(t) = \sum_{n=-\infty}^{\infty} \delta(t - n\Delta T) \qquad (4\text{-}14)$$



a b
c d

FIGURE 4.3 (a) Continuous impulse located at $t = t_0$. (b) An impulse train consisting of continuous impulses. (c) Unit discrete impulse located at $x = x_0$. (d) An impulse train consisting of discrete unit impulses.

等間隔

4.2 Preliminary Concepts

4.2.3 Impulses and Their Sifting Property

- Definition of unit discrete impulse

$$\delta(x) = \begin{cases} 1 & if\ x = 0 \\ 0 & if\ x \neq 0 \end{cases} \qquad (4\text{-}15)$$

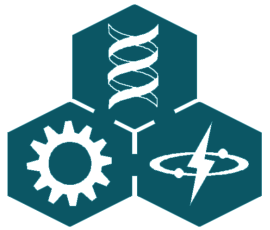
$$\sum_{x=-\infty}^{\infty} \delta(x) = 1 \qquad (4\text{-}16)$$

- Sifting property

離散時的定義

$$\sum_{x=-\infty}^{\infty} f(x)\delta(x) = f(0) \qquad (4\text{-}17)$$

$$\sum_{x=-\infty}^{\infty} f(x)\delta(x - x_0) = f(x_0) \qquad (4\text{-}18)$$



4.2 Preliminary Concepts

4.2.4 The Fourier Transform of Functions of One Continuous Variable

Definition of the Fourier transform of a continuous function *f(t)*

$$\Im\{f(t)\} = \int_{-\infty}^{\infty} f(t)e^{-j2\pi\mu t}dt \qquad (4\text{-}19)$$

硬幹
或轉到freq domain相乘
一樣意思

mu: freq.

$$F(\mu) = \int_{-\infty}^{\infty} f(t)e^{-j2\pi\mu t}dt \qquad (4\text{-}20)$$

Inverse Fourier transform

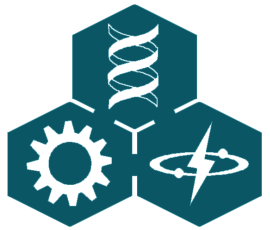
$$f(t) = \int_{-\infty}^{\infty} F(\mu)e^{j2\pi\mu t}d\mu \qquad (4\text{-}21)$$

轉到F: forward
轉道f: inverse

Equations (4-19) and (4-20) comprise the so-called Fourier transform pair, *f*(*t*) is continuous integrable function, *F*(μ) is integrable function. In imaging processing, *f(t)* is real and *F*(μ) is complex in general.

Fourier transform expressed by Euler's formula:

$$F(\mu) = \int_{-\infty}^{\infty} f(t)[\cos(2\pi\mu t) - j\sin(2\pi\mu t)]dt \qquad (4\text{-}22)$$



4.2 Preliminary Concepts

4.2.4 The Fourier Transform of Functions of One Continuous Variable

- Example

$$F(\mu)=\int_{-\infty}^{\infty} f(t)e^{-j2\pi\mu t}dt = \int_{-W/2}^{W/2} Ae^{-j2\pi\mu t}dt$$

$$= \frac{-A}{j2\pi\mu}[e^{-j2\pi\mu t}]_{-W/2}^{W/2} = \frac{-A}{j2\pi\mu}[e^{-j\pi\mu W}-e^{j\pi\mu W}]$$

$$= \frac{A}{j2\pi\mu}[e^{j\pi\mu W}-e^{-j\pi\mu W}]$$

$$= AW\frac{\sin(\pi\mu W)}{(\pi\mu W)}$$

$$\text{sinc}(m)=\frac{\sin(\pi m)}{(\pi m)} \qquad (4\text{-}23)$$

$$|F(\mu)| = AT\left|\frac{\sin(\pi\mu W)}{(\pi\mu W)}\right|$$

lowpass filter

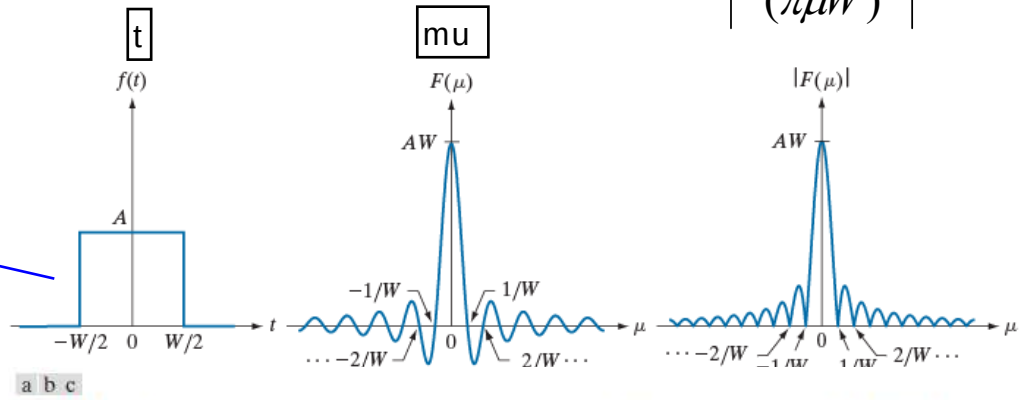


FIGURE 4.4 (a) A box function, (b) its Fourier transform, and (c) its spectrum. All functions extend to infinity in both directions. Note the inverse relationship between the width, W, of the function and the zeros of the transform.

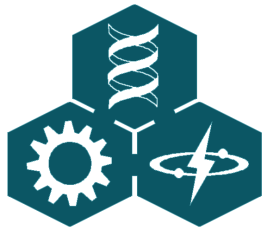
4.2 Preliminary Concepts

4.2.4 The Fourier Transform of Functions of One Continuous Variable

- The Fourier transform of a unit impulse function

$$F(\mu) = \int_{-\infty}^{\infty} \delta(t) e^{-j2\pi\mu t} dt = \int_{-\infty}^{\infty} e^{-j2\pi\mu t} \delta(t) dt$$
$$= e^{-j2\pi\mu 0} = e^{0}$$
$$= 1$$

$$F(\mu) = \int_{-\infty}^{\infty} \delta(t - t_0) e^{-j2\pi\mu t} dt = \int_{-\infty}^{\infty} e^{-j2\pi\mu t} \delta(t - t_0) dt$$
$$= e^{-j2\pi\mu t_0}$$
$$= \cos(2\pi\mu t_0) - j\sin(2\pi\mu t_0)$$



4.2 Preliminary Concepts

4.2.4 The Fourier Transform of Functions of One Continuous Variable

- The Fourier Transform of impulse train

  The impulse train can be expressed as a Fourier series:

$$s_{\Delta T}(t) = \sum_{n=-\infty}^{\infty} c_n e^{j\frac{2\pi n}{\Delta T}t}$$

$$c_n = \frac{1}{\Delta T}\int_{-\Delta T/2}^{\Delta T/2} s_{\Delta T}(t) e^{j\frac{2\pi n}{\Delta T}t} dt = \frac{1}{\Delta T} e^0 = \frac{1}{\Delta T}$$

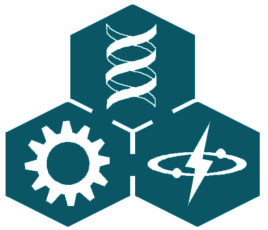
$$s_{\Delta T}(t) = \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty} e^{j\frac{2\pi n}{\Delta T}t}$$

time domain impulse train with time interval delta_t->still impulse train in freq., but the interval would be 1/delta_t

  The Fourier transform of a periodic impulse train:

$$\Im\{e^{j\frac{2\pi n}{\Delta T}t}\} = \delta(\mu - \frac{n}{\Delta T})$$

$$S(\mu) = \Im\{s_{\Delta T}(t)\} = \Im\{\frac{1}{\Delta T}\sum_{n=-\infty}^{\infty} e^{j\frac{2\pi n}{\Delta T}t}\} = \frac{1}{\Delta T}\Im\{\sum_{n=-\infty}^{\infty} e^{j\frac{2\pi n}{\Delta T}t}\} = \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty}\delta(\mu - \frac{n}{\Delta T})$$



4.2 Preliminary Concepts

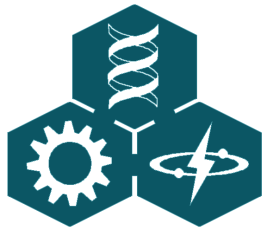
4.2.4 The Fourier Transform of Functions of One Continuous Variable

- The Fourier Transform of some functions

Function	f(x)	F(u)
Gaussian	$e^{-\pi x^2}$	$e^{-\pi u^2}$
Rectangular pulse	$\Pi(x)$	$\frac{\sin(\pi u)}{\pi u}$
Triangular pulse	$\Lambda(x)$	$\frac{\sin^2(\pi u)}{(\pi u)^2}$
Impulse	$\delta(x)$	1
Unit step	$u(x)$	$\frac{1}{2}[\delta(u)-\frac{j}{\pi u}]$
Cosine	$\cos(2\pi fx)$	$\frac{1}{2}[\delta(u+f)+\delta(u-f)]$
Sine	$\sin(2\pi fx)$	$j\frac{1}{2}[\delta(u+f)-\delta(u-f)]$
Complex exponential	$e^{j2\pi fx}$	$\delta(u-f)$

square function

common Fourier transform



4.2 Preliminary Concepts

4.2.5 Convolution

- Definition

convolution in time domain

$$f(t)\star h(t) = \int_{-\infty}^{\infty} f(\tau)h(t-\tau)d\tau \qquad (4\text{-}24)$$

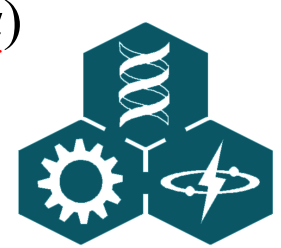
- The Fourier Transform of Convolution

$$\Im\{f(t)\star h(t)\} = \int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty} f(\tau)h(t-\tau)d\tau\right]e^{-j2\pi\mu t}dt$$

$$= \int_{-\infty}^{\infty} f(\tau)\left[\int_{-\infty}^{\infty} h(t-\tau)e^{-j2\pi\mu t}dt\right]d\tau$$

$$\Im\{f(t)\star h(t)\} = \int_{-\infty}^{\infty} f(\tau)[H(\mu)e^{-j2\pi\mu\tau}]d\tau$$

$$= H(\mu)\int_{-\infty}^{\infty} f(\tau)e^{-j2\pi\mu\tau}d\tau = H(\mu)F(\mu)$$

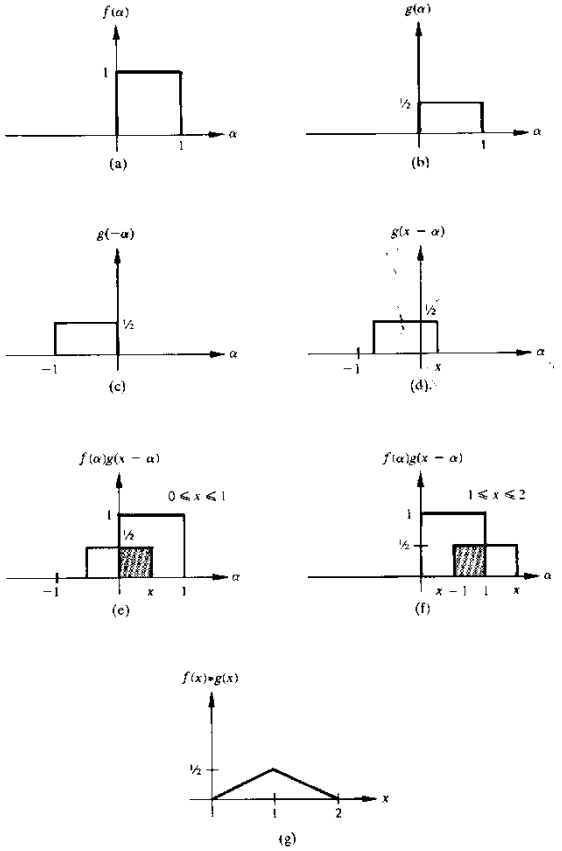


4.2 Preliminary Concepts

4.2.5 Convolution

- Example

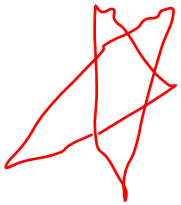
$$f(x)*g(x)=\begin{cases} x/2 & 0\le x\le 1 \\ 1-x/2 & 1< x\le 2 \\ 0 & \text{elsewhere} \end{cases}$$



4.2 Preliminary Concepts

4.2.5 Convolution

- Convolution theorem and multiplication



$$f(t) \star h(t) \quad \Leftrightarrow \quad F(\mu)H(\mu)$$

$$f(t)h(t) \quad \Leftrightarrow \quad F(\mu) \star H(\mu)$$

4.3 Sampling and the Fourier Transform of Sampled Functions

4.3.1 Sampling

取樣後得到的function

$$\tilde{f}(t) = f(t)s_{\Delta T}(t) = \sum_{n=-\infty}^{\infty} f(t)\delta(t - n\Delta T) \qquad (4\text{-}27)$$

$$f_k = \int_{-\infty}^{\infty} f(t)\delta(t - k\Delta T) = f(k\Delta T) \qquad (4\text{-}28)$$

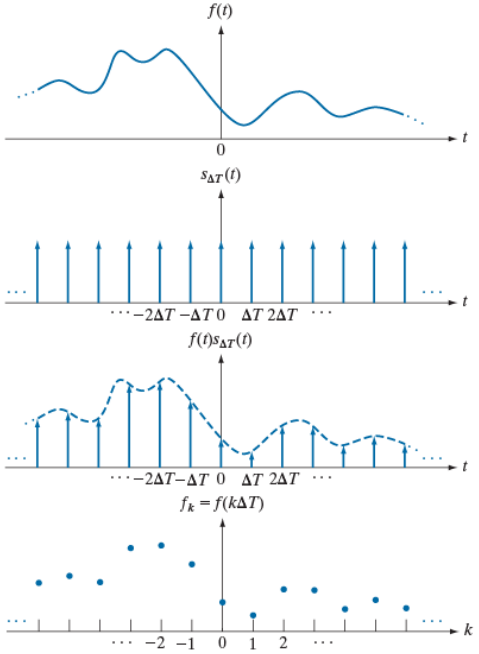


FIGURE 4.5 (a) A continuous function. (b) Train of impulses used to model sampling. (c) Sampled function formed as the product of (a) and (b). (d) Sample values obtained by integration and using the sifting property of impulses. (The dashed line in (c) is shown for reference. It is not part of the data.)

4.3 Sampling and the Fourier Transform of Sampled Functions

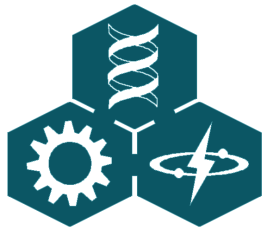
4.3.2 The Fourier Transform of Sampled Functions

$$\tilde{F}(\mu) = \Im\{\tilde{f}(t)\} = \Im\{f(t)s_{\Delta T}(t)\}$$
$$= F(\mu) \star S(\mu) \qquad (4\text{-}29)$$

$$S(\mu) = \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty}\delta(\mu - \frac{n}{\Delta T}) \qquad (4\text{-}30)$$

$$\tilde{F}(\mu) = F(\mu) \star S(\mu)$$
$$= \int_{-\infty}^{\infty} F(\tau)S(\mu - \tau)d\tau$$
$$= \frac{1}{\Delta T}\int_{-\infty}^{\infty} F(\tau)\sum_{n=-\infty}^{\infty}\delta(\mu - \tau - \frac{n}{\Delta T})d\tau \qquad (4\text{-}31)$$
$$= \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty}\int_{-\infty}^{\infty} F(\tau)\delta(\mu - \tau - \frac{n}{\Delta T})d\tau$$
$$= \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty} F(\mu - \frac{n}{\Delta T})$$

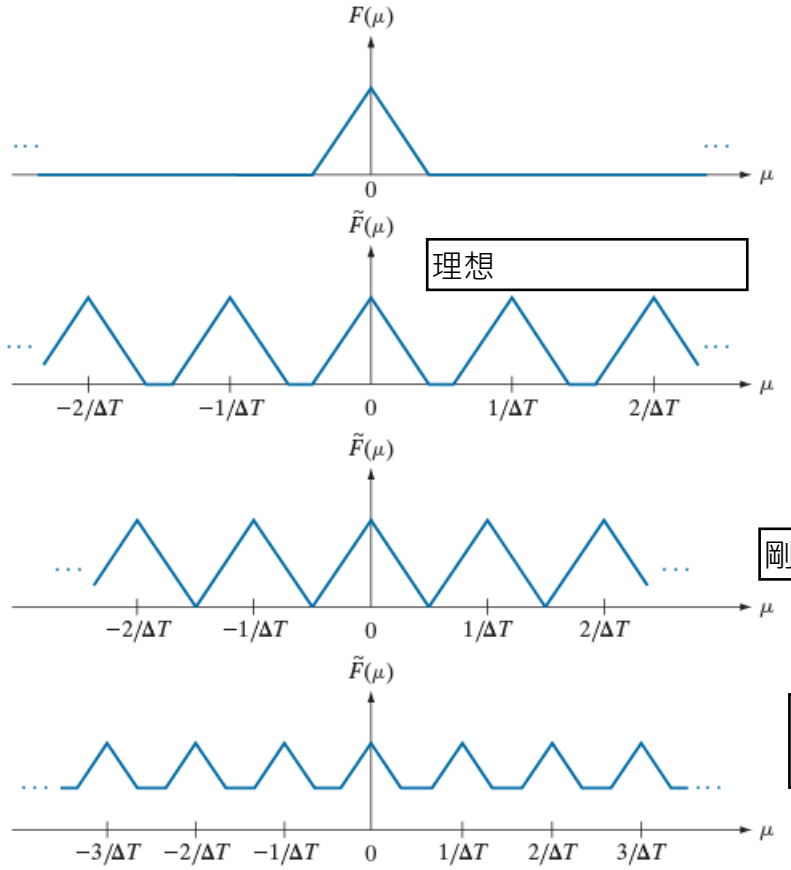
離散訊號的fourier transform



4.3 Sampling and the Fourier Transform of Sampled Functions

4.3.2 The Fourier Transform of Sampled Functions

- Over-sampling
- Critical-sampling
- Under-sampling



a b c d

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

impulse train取樣後fourier transform得到copy效果，但間隔改變（變成1/deltaT）

delta太大，導術後重疊

4.3 Sampling and the Fourier Transform of Sampled Functions

4.3.3 Sampling Theorem

- Band-limited functions

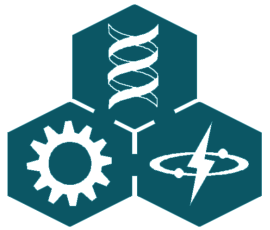
A function f(t) whose Fourier transform is zero for values of frequencies outside a finite interval (band) $[-\mu_{max}, \mu_{max}]$ about the origin is called a **band-limited function**.

- Sampling Theorem

A continuous, band-limited function can be recovered from a set of its samples if the samples are acquired at a rate exceeding twice the highest frequency content of the function.

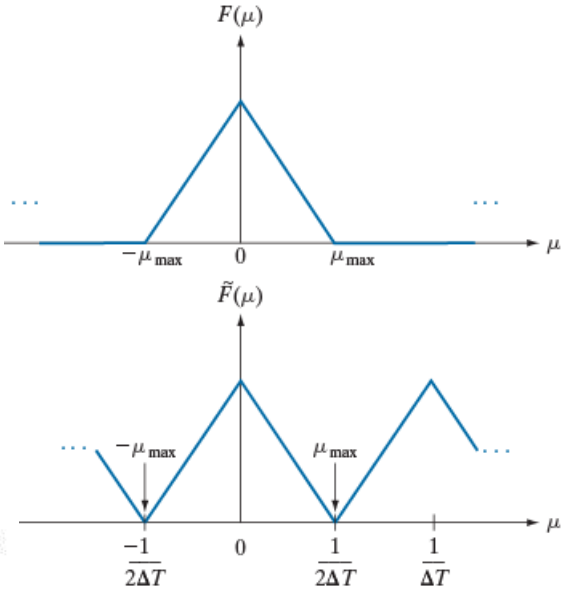
- Nyquist rate

$$\frac{1}{\Delta T} > 2\mu_{max} \qquad (4\text{-}32)$$



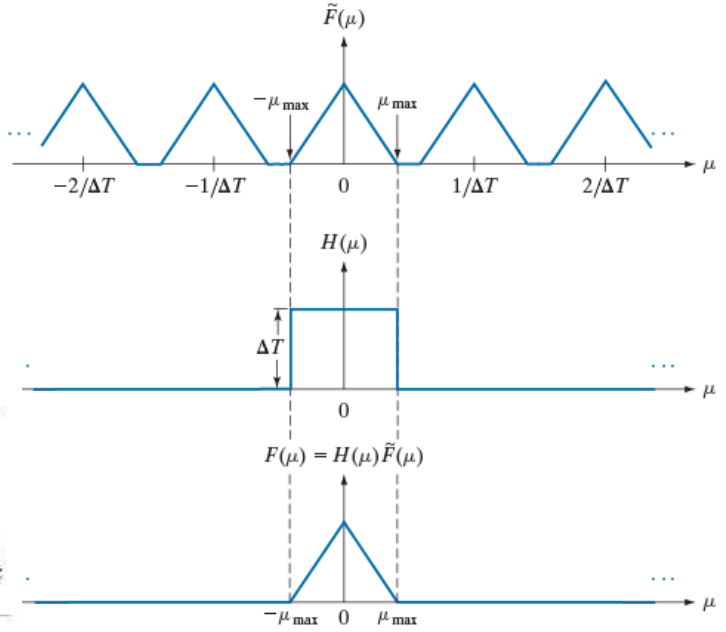
4.3 Sampling and the Fourier Transform of Sampled Functions

4.3.3 Sampling Theorem



a
b

FIGURE 4.7
(a) Illustrative sketch of the Fourier transform of a band-limited function.
(b) Transform resulting from critically sampling that band-limited function.



a
b
c

FIGURE 4.8
(a) Fourier transform of a sampled, band-limited function.
(b) Ideal lowpass filter transfer function.
(c) The product of (b) and (a), used to extract one period of the infinitely periodic sequence in (a).

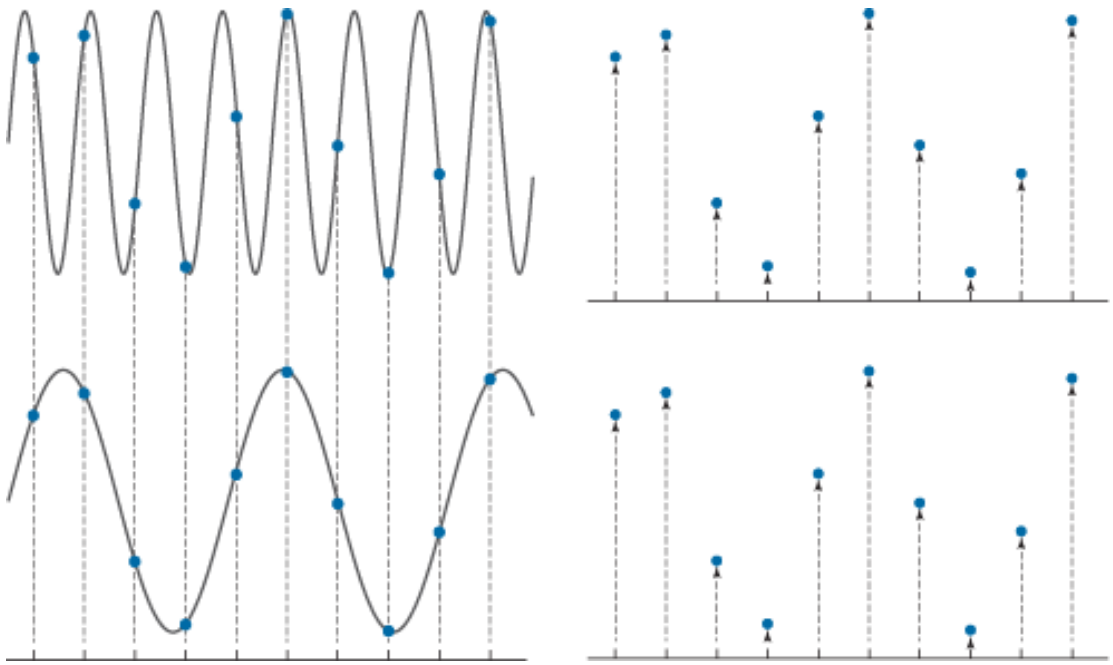
$$H(\mu) = \begin{cases} \Delta T & -\mu_{max} \le \mu \le \mu_{max} \\ 0 & otherwise \end{cases} \qquad (4\text{-}33)$$

$$F(\mu) = H(\mu)\tilde{F}(\mu) \qquad (4\text{-}34)$$

$$f(t) = \int_{-\infty}^{\infty} F(\mu) e^{j2\pi\mu t} d\mu \qquad (4\text{-}35)$$

4.3 Sampling and the Fourier Transform of Sampled Functions

4.3.4 Aliasing

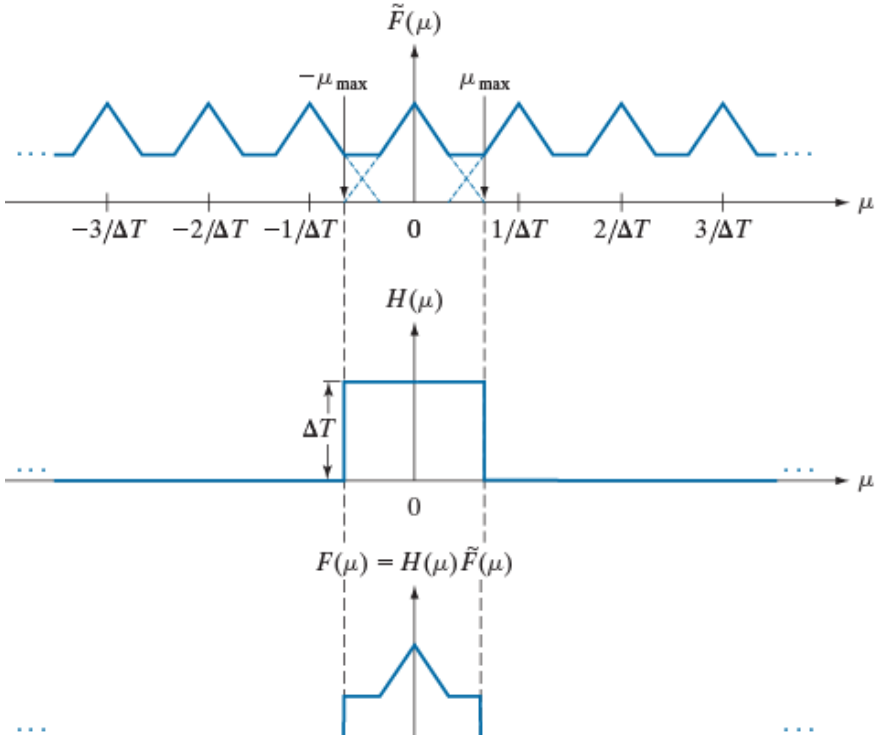


a b
c d

FIGURE 4.9
The functions in (a) and (c) are totally different, but their digitized versions in (b) and (d) are identical. Aliasing occurs when the samples of two or more functions coincide, but the functions are different elsewhere.

4.3 Sampling and the Fourier Transform of Sampled Functions

4.3.4 Aliasing



a
b
c

FIGURE 4.10 (a) Fourier transform of an under-sampled, band-limited function. (Interference between adjacent periods is shown dashed). (b) The same ideal lowpass filter used in Fig. 4.8. (c) The product of (a) and (b).The interference from adjacent periods results in aliasing that prevents perfect recovery of $F(\mu)$ and, consequently, of $f(t)$.

4.3 Sampling and the Fourier Transform of Sampled Functions

4.3.4 Aliasing

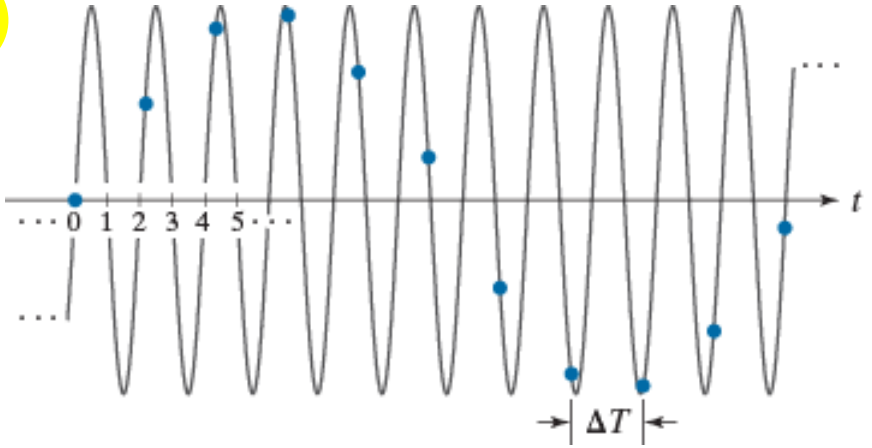


FIGURE 4.11 Illustration of aliasing. The under-sampled function (dots) looks like a sine wave having a frequency much lower than the frequency of the continuous signal. The period of the sine wave is 2 s, so the zero crossings of the horizontal axis occur every second. ΔT is the separation between samples.

- No function of finite duration can be band-limited. Conversely, a function that is band-limited must extend from $-\infty$ to ∞.
- The effects of aliasing can be reduced by smoothing the input function to attenuate its higher frequencies. This process, called anti-aliasing, has to be done before the function is sampled.

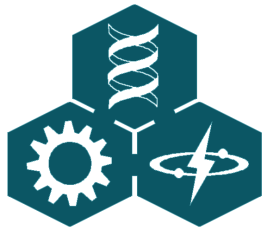
4.3 Sampling and the Fourier Transform of Sampled Functions

4.3.5 Function Reconstruction from Sampled Data

$$\begin{aligned} f(t) &= \Im^{-1}\{F(\mu)\} \\ &= \Im^{-1}\{H(\mu)\tilde{F}(\mu)\} \\ &= h(t) \bigstar \tilde{f}(t) \end{aligned} \qquad (4\text{-}37)$$

$$f(t) = \sum_{n=-\infty}^{\infty} f(n\Delta T)\,\mathrm{sinc}[(t - n\Delta T)/n\Delta T] \qquad (4\text{-}38)$$

- Perfectly reconstructed function is an infinite sum of *sinc* functions weighted by the sample values.
- For any $t = k\,\Delta T$, reconstructed value is equal to sampled value. Between sample points, values of $f(t)$ are *interpolations* formed by the sum of the *sinc* functions.



4.4 The Discrete Fourier Transform of One Variable

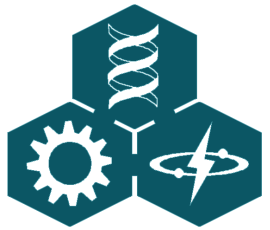
4.4.1 Obtaining the DFT from the Continuous Transform of a Sampled Function

The Fourier Transform of sampled data in terms of the transform of the original function:

$$\tilde{F}(\mu) = \int_{-\infty}^{\infty} \tilde{f}(t) e^{-j2\pi\mu t} dt \qquad (4\text{-}39)$$

By substituting Eq. (4.3-1) for $\tilde{f}(t)$, we obtain

$$\begin{aligned}
\tilde{F}(\mu) &= \int_{-\infty}^{\infty} \tilde{f}(t) e^{-j2\pi\mu t} dt \\
&= \int_{-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(t)\delta(t - n\Delta T) e^{-j2\pi\mu t} dt \qquad (4\text{-}40) \\
&= \sum_{n=-\infty}^{\infty} \int_{-\infty}^{\infty} f(t)\delta(t - n\Delta T) e^{-j2\pi\mu t} dt \\
&= \sum_{n=-\infty}^{\infty} f_n e^{-j2\pi\mu n\Delta T}
\end{aligned}$$



4.4 The Discrete Fourier Transform of One Variable

4.4.1 Obtaining the DFT from the Continuous Transform of a Sampled Function

Taking the following frequencies

$$\mu = \frac{m}{M\Delta T} \qquad m = 0,1,2,\ldots,M-1 \qquad (4\text{-}41)$$

The discrete Fourier transform pair will be

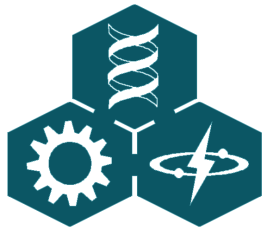
$$F_m = \sum_{n=0}^{M-1} f_n e^{-j2\pi nm/M} \qquad m = 0,1,2,\ldots,M-1 \qquad (4\text{-}42)$$

$$f_n = \frac{1}{M}\sum_{m=0}^{M-1} F_m e^{j2\pi nm/M} \qquad n = 0,1,2,\ldots,M-1 \qquad (4\text{-}43)$$

Expressed by the notation for image coordinated variables x ,u

$$F(u) = \sum_{x=0}^{M-1} f(x) e^{-j2\pi ux/M} \qquad u = 0,1,2,\ldots,M-1 \qquad (4\text{-}44)$$

$$f(x) = \frac{1}{M}\sum_{u=0}^{M-1} F(u) e^{j2\pi ux/M} \qquad x = 0,1,2,\ldots,M-1 \qquad (4\text{-}45)$$



4.4 The Discrete Fourier Transform of One Variable

4.4.1 Obtaining the DFT from the Continuous Transform of a Sampled Function

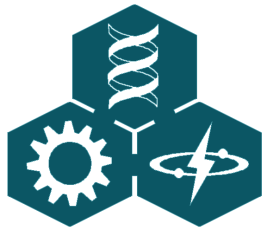
Both the forward and inverse discrete transforms are infinitely periodic, with period *M*

$$F(u) = F(u + kM) \quad \text{以M為周期} \qquad (4\text{-}46)$$

$$f(x) = f(x + kM) \qquad (4\text{-}47)$$

The discrete equivalent of the convolution

$$f(x) \star h(x) = \sum_{m=0}^{M-1} f(m)h(x - m) \qquad (4\text{-}48)$$



4.4 The Discrete Fourier Transform of One Variable

4.4.2 Relationship between the sampling and frequency intervals

If $f(x)$ consists of M samples of a function $f(t)$ taken ΔT units apart, the length of the record is:

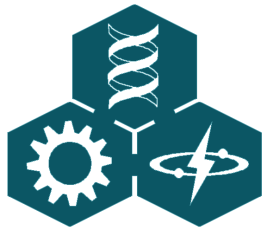
$$T = M\Delta T \tag{4-49}$$

The spacing Δu, in the discrete frequency domain:

$$\Delta u = \frac{1}{M\Delta T} = \frac{1}{T} \tag{4-50}$$

The entire frequency range of the DFT is:

$$\Omega = M\Delta u = \frac{1}{\Delta T} \tag{4-51}$$



4.4 The Discrete Fourier Transform of One Variable

4.4.2 Relationship between the sampling and frequency intervals

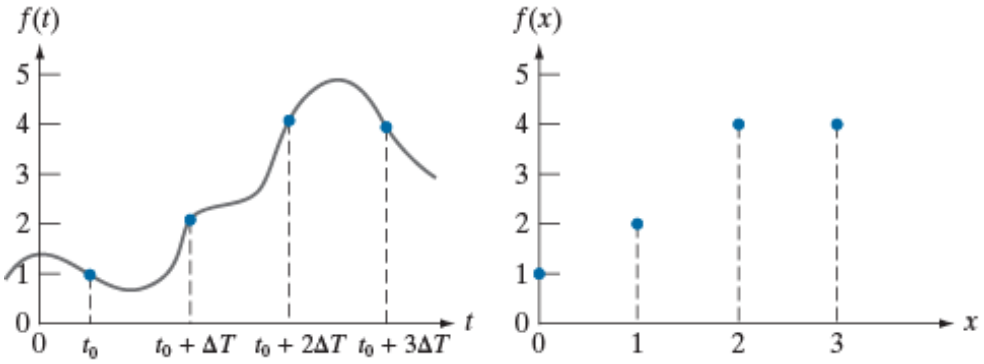
- Example

離散的FOURIER TRANSFORM

$$F(0)=\sum_{x=0}^{3} f(x)=[f(0)+f(1)+f(2)+f(3)]=[1+2+4+4]=11$$

$$F(1)=\sum_{x=0}^{3} f(x)e^{-j2\pi(1)x/4}=1e^{0}+2e^{-j\pi/2}+4e^{-j\pi}+4e^{-j3\pi/2}=-3+2j$$

$$f(0)=\frac{1}{4}\sum_{u=0}^{3} F(u)e^{j2\pi u(0)}=\frac{1}{4}\sum_{u=0}^{3} F(u)=\frac{1}{4}[11-3+2j-1-3-2j]=\frac{1}{4}[4]=1$$



a b

FIGURE 4.12 (a) A continuous function sampled ΔT units apart. (b) Samples in the x-domain. Variable t is continuous, while x is discrete.

4.5 Extension to Functions of Two Variables

4.5.1 The 2-D Impulse and Its Sifting Property

- Continuous impulse function

$$\delta(t,z)=\begin{cases}\infty & if\ t=z=0\\ 0 & otherwise\end{cases} \qquad (4\text{-}52)$$

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\delta(t,z)dtdz=1 \qquad (4\text{-}53)$$

Sifting property under integration

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f(t,z)\delta(t,z)dtdz=f(0,0) \qquad (4\text{-}54)$$

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f(t,z)\delta(t-t_0,z-z_0)dtdz=f(t_0,z_0) \qquad (4\text{-}55)$$

4.5 Extension to Functions of Two Variables

4.5.1 The 2-D Impulse and Its Sifting Property

- Discrete impulse function

$$\delta(t,z)=\begin{cases}1 & if\ t=z=0\\ 0 & otherwise\end{cases} \qquad (4\text{-}56)$$

Sifting property

$$\sum_{x=-\infty}^{\infty}\sum_{y=-\infty}^{\infty} f(x,y)\delta(x,y)=f(0,0) \qquad (4\text{-}57)$$

$$\sum_{x=-\infty}^{\infty}\sum_{y=-\infty}^{\infty} f(x,y)\delta(x-x_0,y-y_0)=f(x_0,y_0) \qquad (4\text{-}58)$$

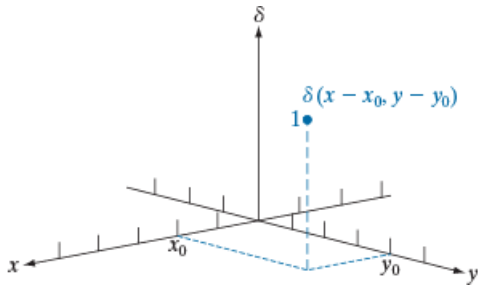


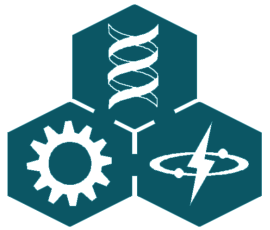
FIGURE 4.13 2-D unit discrete impulse. Variables x and y are discrete, and δ is zero everywhere except at coordinates (x_0, y_0), where its value is 1.

4.5 Extension to Functions of Two Variables

4.5.2 The 2-D Continuous Fourier Transform Pair

$$F(\mu,\nu) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(t,z)e^{-j2\pi(\mu t+\nu z)}dtdz \qquad (4\text{-}59)$$

$$f(t,z) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(\mu,\nu)e^{j2\pi(\mu t+\nu z)}d\mu d\nu \qquad (4\text{-}60)$$



4.5 Extension to Functions of Two Variables

4.5.2 The 2-D Continuous Fourier Transform Pair

$$F(\mu,\nu) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(t,z)e^{-j2\pi(\mu t+\nu z)}dtdz$$

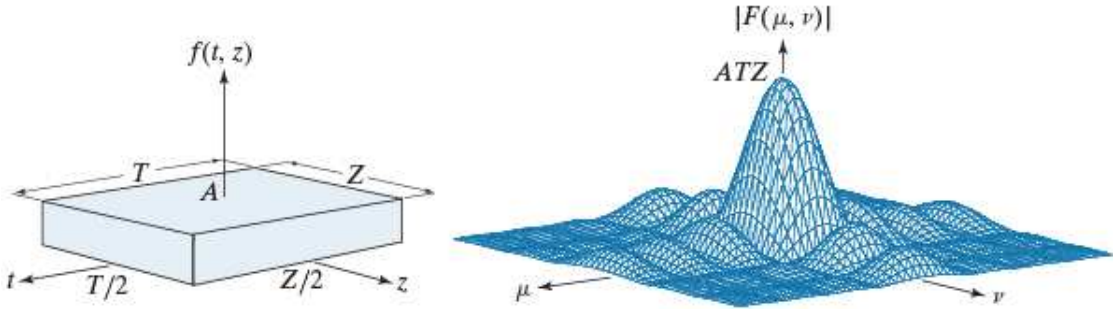
$$= \int_{-T/2}^{T/2}\int_{-Z/2}^{Z/2} f(t,z)e^{-j2\pi(\mu t+\nu z)}dtdz$$

$$= ATZ\left[\frac{\sin(\pi\mu T)}{(\pi\mu T)}\right]\left[\frac{\sin(\pi\nu Z)}{(\pi\nu Z)}\right]$$

$$|F(\mu,\nu)| = ATZ\left|\frac{\sin(\pi\mu T)}{(\pi\mu T)}\right|\left|\frac{\sin(\pi\nu Z)}{(\pi\nu Z)}\right|$$

a b

FIGURE 4.14 (a) A 2-D function and (b) a section of its spectrum. The box is longer along the t-axis, so the spectrum is more contracted along the μ-axis.



4.5 Extension to Functions of Two Variables

4.5.3 2-D Sampling and the 2-D Sampling Theorem

- Sampling

$$s_{\Delta T \Delta Z}(t,z) = \sum_{-\infty}^{\infty}\sum_{-\infty}^{\infty}\delta(t - m\Delta T,\ z - n\Delta Z) \qquad (4\text{-}61)$$

- Band-limited functions

$$F(\mu,\nu) = 0 \quad for \quad |\mu| \geq \mu_{\max} \ and \quad |\nu| \geq \nu_{\max} \qquad (4\text{-}62)$$

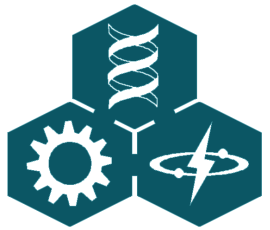
- Sampling Theorem

$$\Delta T < \frac{1}{2\mu_{\max}} \qquad (4\text{-}63)$$

$$\Delta Z < \frac{1}{2\nu_{\max}} \qquad (4\text{-}64)$$

$$\frac{1}{\Delta T} > 2\mu_{\max} \qquad (4\text{-}65)$$

$$\frac{1}{\Delta Z} > 2\nu_{\max} \qquad (4\text{-}66)$$



4.5 Extension to Functions of Two Variables

4.5.3 2-D Sampling and the 2-D Sampling Theorem

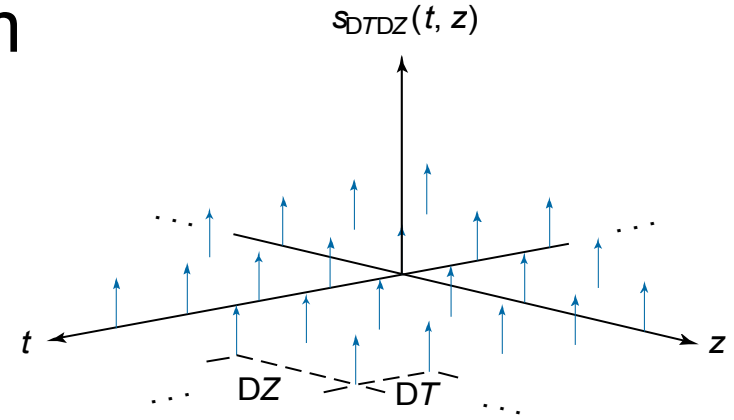
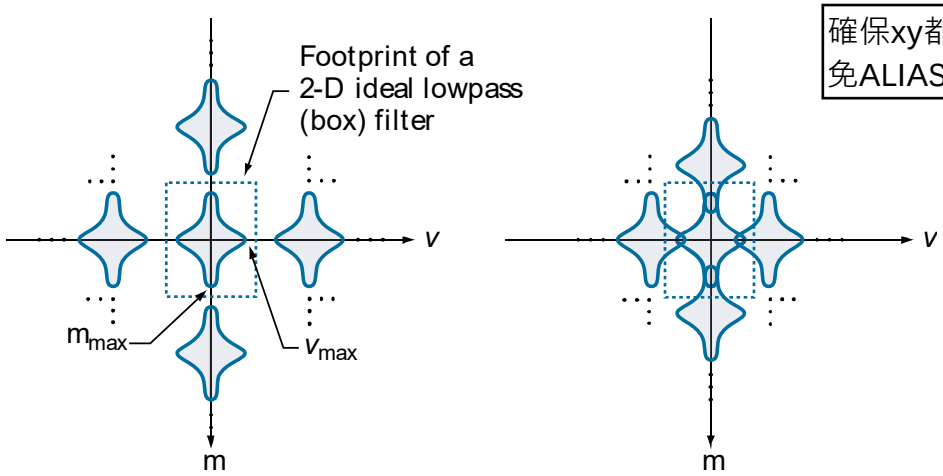


FIGURE 4.15
2-D impulse train.



確保xy都沒重疊避免ALIASING

a b

FIGURE 4.16
Two-dimensional Fourier transforms of (a) an over-sampled, and (b) an under-sampled, band-limited function.

4.5 Extension to Functions of Two Variables

4.5.4 Aliasing in Images

- Example

格點：取樣區
不同的待測平率

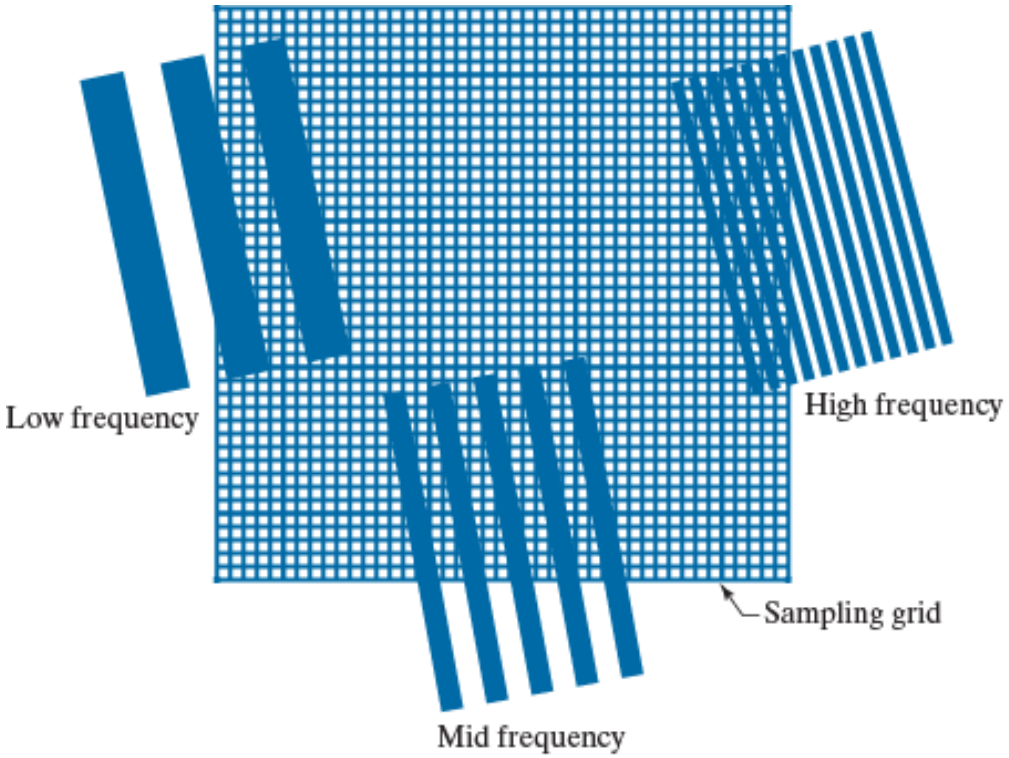
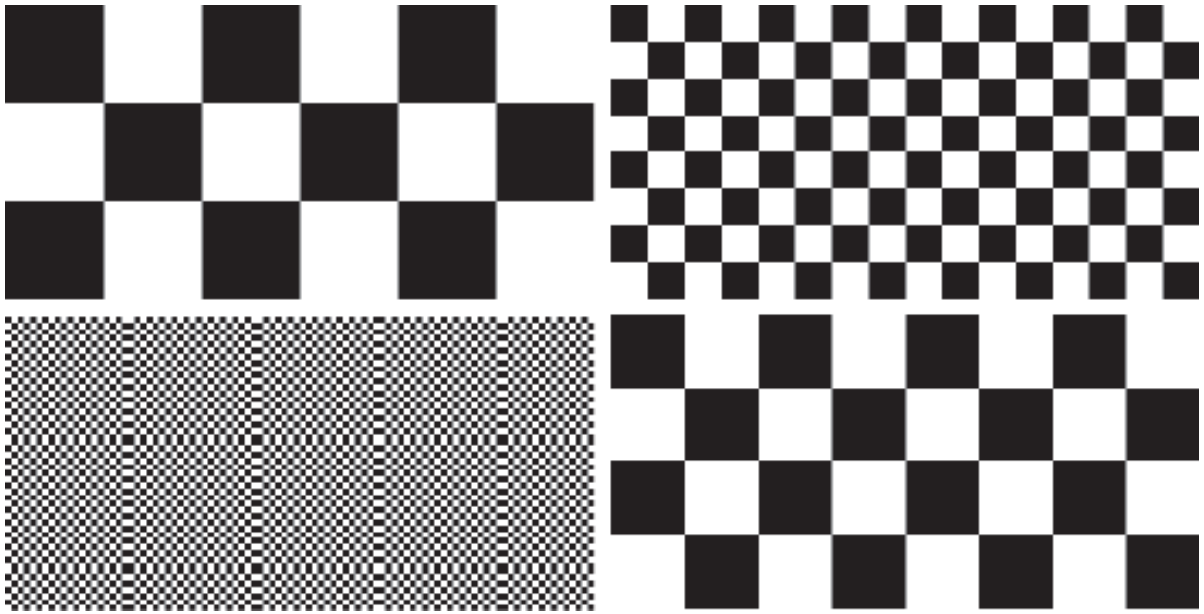


FIGURE 4.17 Various aliasing effects resulting from the interaction between the frequency of 2-D signals and the sampling rate used to digitize them. The regions outside the sampling grid are continuous and free of aliasing.

4.5 Extension to Functions of Two Variables

4.5.4 Aliasing in Images

- Example



a b
c d

FIGURE 4.18
Aliasing. In (a) and (b) the squares are of sizes 16 and 6 pixels on the side. In (c) and (d) the squares are of sizes 0.95 and 0.48 pixels, respectively. Each small square in (c) is one pixel. Both (c) and (d) are aliased. Note how (d) masquerades as a "normal" image.

4.5 Extension to Functions of Two Variables

4.5.4 Aliasing in Images

- Example: interpolation and resampling of an image

a b c

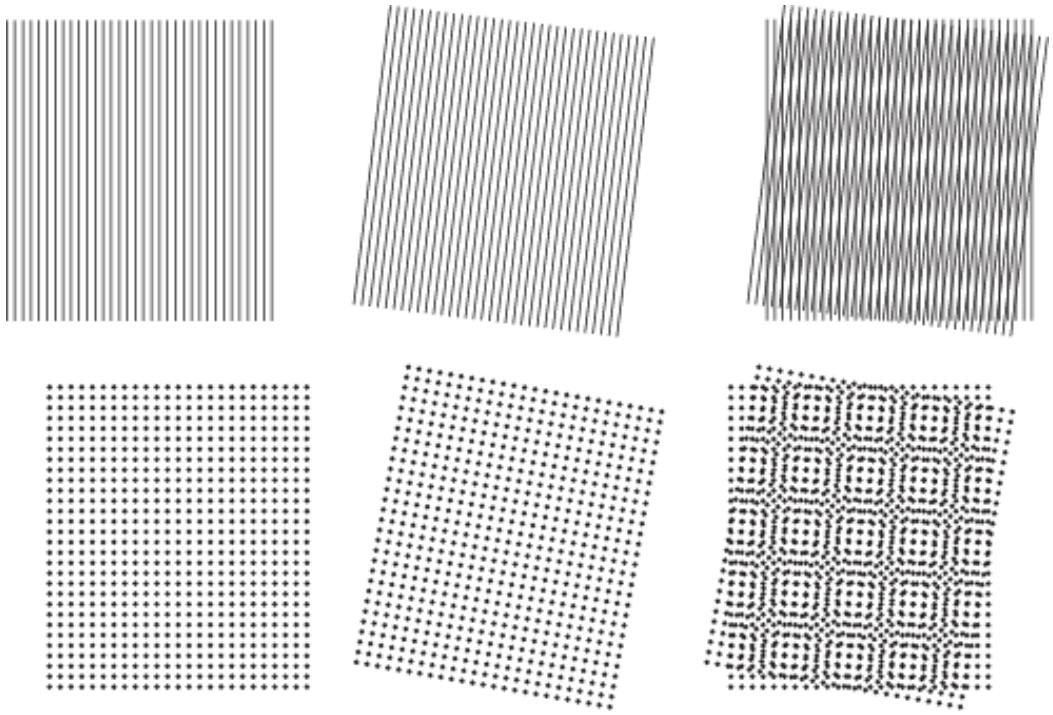
FIGURE 4.19 Illustration of aliasing on resampled natural images. (a) A digital image of size 772 × 548 pixels w… ally negligible aliasing. (b) Result of resizing the image to 33% of its original size by pixel deletion and then restoring it to its original size by pixel replication. Aliasing is clearly visible. (c) Result of blurring the image in (a) with an averaging filter prior to resizing. The image is slightly more blurred than (b), but aliasing is not longer objectionable. (Original image courtesy of the Signal Compression Laboratory, University of California, Santa Barbara.)

先低通平滑在取樣
ALIASING遍布明顯

4.5 Extension to Functions of Two Variables

4.5.4 Aliasing in Images

- Moiré patterns



a b c
d e f

FIGURE 4.20
Examples of the moiré effect. These are vector drawings, not digitized patterns. Superimposing one pattern on the other is analogous to multiplying the patterns.

4.5 Extension to Functions of Two Variables

4.5.4 Aliasing in Images

- Moiré patterns

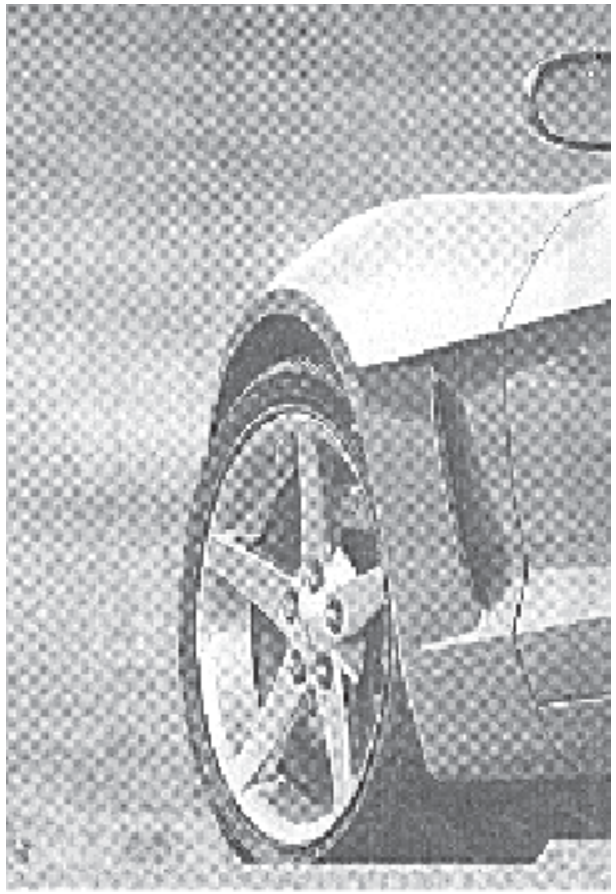
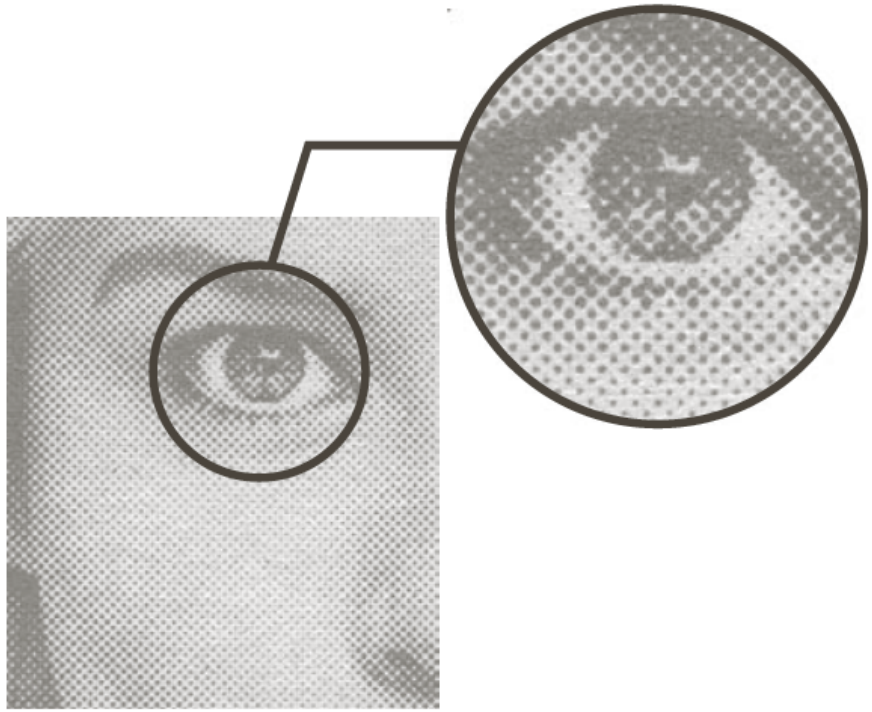


FIGURE 4.21
A newspaper image digitized at 75 dpi. Note the moiré-like pattern resulting from the interaction between the ±45° orientation of the half-tone dots and the north-south orientation of the sampling elements used to digitized the image.

4.5 Extension to Functions of Two Variables

4.5.4 Aliasing in Images

- Moiré patterns



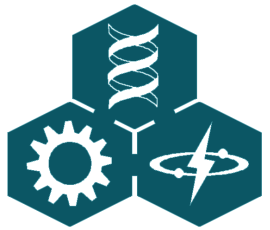
A newspaper image and an enlargement showing how halftone dots are arranged to render shades of gray.

4.5 Extension to Functions of Two Variables

4.5.5 The 2-D Discrete Fourier Transform

$$F(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(ux/M+vy/N)} \qquad (4\text{-}67)$$

$$f(x,y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)e^{j2\pi(ux/M+vy/N)} \qquad (4\text{-}68)$$

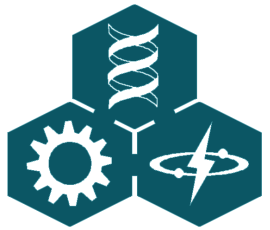


4.6 Properties of the 2-D Discrete Fourier Transform

4.6.1 Relationships Between Spatial and Frequency Intervals

$$\Delta u = \frac{1}{M\Delta T} \tag{4-69}$$

$$\Delta v = \frac{1}{N\Delta T} \tag{4-70}$$



4.6 Properties of the 2-D Discrete Fourier Transform

4.6.2 Translation and Rotation

- Translation

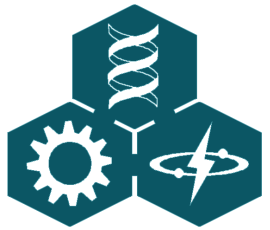
$$f(x,y)e^{j2\pi(u_0x/M+v_0y/N)} \Leftrightarrow F(u-u_0,v-v_0) \qquad (4\text{-}71)$$

$$f(x-x_0,y-y_0) \Leftrightarrow F(u,v)\, e^{-j2\pi(x_0u/M+y_0v/N)} \qquad (4\text{-}72)$$

Translation has no effect on the magnitude of *F(u, v)*

- Rotation

$$f(r,\theta+\theta_0) \Leftrightarrow F(\omega,\phi+\theta_0) \qquad (4\text{-}73)$$



4.6 Properties of the 2-D Discrete Fourier Transform

4.6.3 Periodicity

$$F(u,v) = F(u + k_1 M, v) = F(u, v + k_2 N) = F(u + k_1 M, v + k_2 N) \qquad (4\text{-}74)$$

$$f(x,y) = f(x + k_1 M, y) = f(x, y + k_2 N) = f(x + k_1 M, y + k_2 N) \qquad (4\text{-}75)$$

Applying the periodicity and translation properties, it's able to use the following equations to shift the transform result to the center ($M/2, N/2$):

$$f(x)e^{j2\pi(u_0 x/M)} \Leftrightarrow F(u - u_0)$$

Let $u_0 = \mathrm{M}/2$, $e^{j\pi x} = (-1)^x$, then

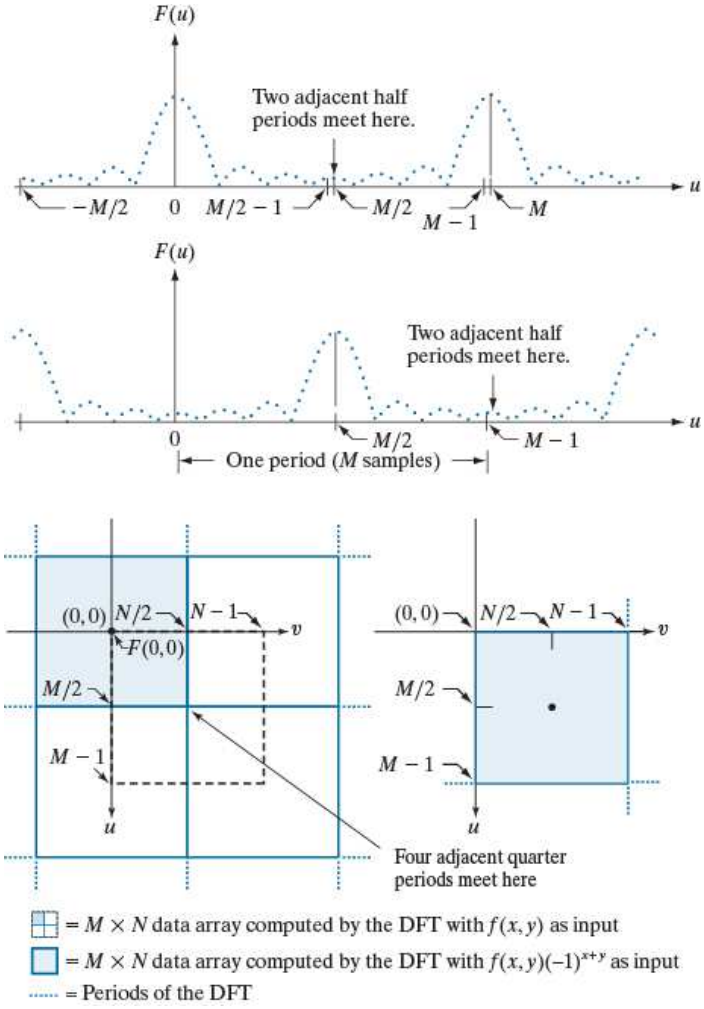
$$f(x)(-1)^x \Leftrightarrow F(u - M/2)$$

In 2-D case:

$$f(x,y)(-1)^{x+y} \Leftrightarrow F(u - M/2, v - N/2) \qquad (4\text{-}76)$$

4.6 Properties of the 2-D Discrete Fourier Transform

4.6.3 Periodicity



a
b
c d

FIGURE 4.22 Centering the Fourier transform. (a) A 1-D DFT showing an infinite number of periods. (b) Shifted DFT obtained by multiplying $f(x)$ by $(-1)^x$ before computing $F(u)$. (c) A 2-D DFT showing an infinite number of periods. The area within the dashed rectangle is the data array, $F(u,v)$, obtained with Eq. (4-67) with an image $f(x, y)$ as the input. This array consists of four quarter periods. (d) Shifted array obtained by multiplying $f(x,y)$ by $(-1)^{x+y}$ before computing $F(u, v)$. The data now contains one complete, centered period, as in (b).

4.6 Properties of the 2-D Discrete Fourier Transform

4.6.4 Symmetry Property

- Odd Function and Even Function

$$w(x,y) = w_e(x,y) + w_o(x,y) \tag{4-77}$$

$$w_e(x,y) = \frac{w(x,y) + w(-x,-y)}{2} \tag{4-78}$$

$$w_o(x,y) = \frac{w(x,y) - w(-x,-y)}{2} \tag{4-79}$$

$$w_e(x,y) = w_e(-x,-y) \tag{4-80}$$

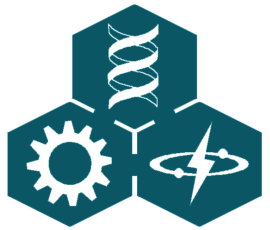
$$w_o(x,y) = -w_o(-x,-y) \tag{4-81}$$

For DFT and IDFT:

$$w_e(x,y) = w_e(M-x, N-y) \tag{4-82}$$

$$w_o(x,y) = -w_o(M-x, N-y) \tag{4-83}$$

$$\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} w_e(x,y)w_o(x,y) = 0 \tag{4-84}$$



4.6 Properties of the 2-D Discrete Fourier Transform

4.6.4 Symmetry Property

- Conjugate Symmetry

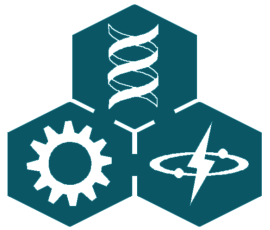
  The Fourier transform of a real function, f(x, y), is conjugate symmetric :

  $$F^*(u,v) = F(-u,-v) \qquad (4\text{-}85)$$

- Conjugate Antisymmetry

  The Fourier transform of an imaginary function, f(x, y), is conjugate antisymmetric :

  $$F^*(-u,-v) = -F(u,v)$$



4.6 Properties of the 2-D Discrete Fourier Transform

4.6.4 Symmetry Property

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. Use of the word *complex* indicates that a function has nonzero real and imaginary parts.

	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. As before, "$\Leftrightarrow$" indicates a Fourier transform pair.

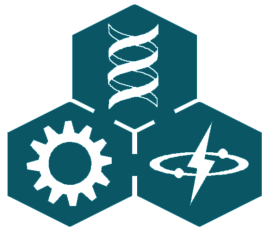
4.6 Properties of the 2-D Discrete Fourier Transform

4.6.4 Symmetry Property

- Example

TABLE 4.2
1-D examples of some of the properties in Table 4.1.

Property	$f(x)$		$F(u)$
3	$\{1, 2, 3, 4\}$	$\Leftrightarrow$	$\{(10+0j), (-2+2j), (-2+0j), (-2-2j)\}$
4	$\{1j, 2j, 3j, 4j\}$	$\Leftrightarrow$	$\{(0+2.5j), (.5-.5j), (0-.5j), (-.5-.5j)\}$
8	$\{2, 1, 1, 1\}$	$\Leftrightarrow$	$\{5, 1, 1, 1\}$
9	$\{0, -1, 0, 1\}$	$\Leftrightarrow$	$\{(0+0j), (0+2j), (0+0j), (0-2j)\}$
10	$\{2j, 1j, 1j, 1j\}$	$\Leftrightarrow$	$\{5j, j, j, j\}$
11	$\{0j, -1j, 0j, 1j\}$	$\Leftrightarrow$	$\{0, -2, 0, 2\}$
12	$\{(4+4j), (3+2j), (0+2j), (3+2j)\}$	$\Leftrightarrow$	$\{(10+10j), (4+2j), (-2+2j), (4+2j)\}$
13	$\{(0+0j), (1+1j), (0+0j), (-1-j)\}$	$\Leftrightarrow$	$\{(0+0j), (2-2j), (0+0j), (-2+2j)\}$



4.6 Properties of the 2-D Discrete Fourier Transform

4.6.5 Fourier Spectrum and Phase Angle

- Definition

$$F(u,v) = |F(u,v)|e^{j\varphi(u,v)} \quad (4\text{-}86)$$

$$|F(u,v)| = [R^2(u,v) + I^2(u,v)]^{1/2} \quad (4\text{-}87)$$

$$\varphi(u,v) = \arctan\left[\frac{I(u,v)}{R(u,v)}\right] \quad (4\text{-}88)$$

$$P(u,v) = |F(u,v)|^2 = R^2(u,v) + I^2(u,v) \quad (4\text{-}89)$$

- Properties

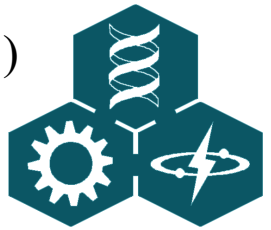
$$|F(u,v)| = |F(-u,-v)| \quad (4\text{-}90)$$

$$\varphi(u,v) = -\varphi(-u,-v) \quad (4\text{-}91)$$

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

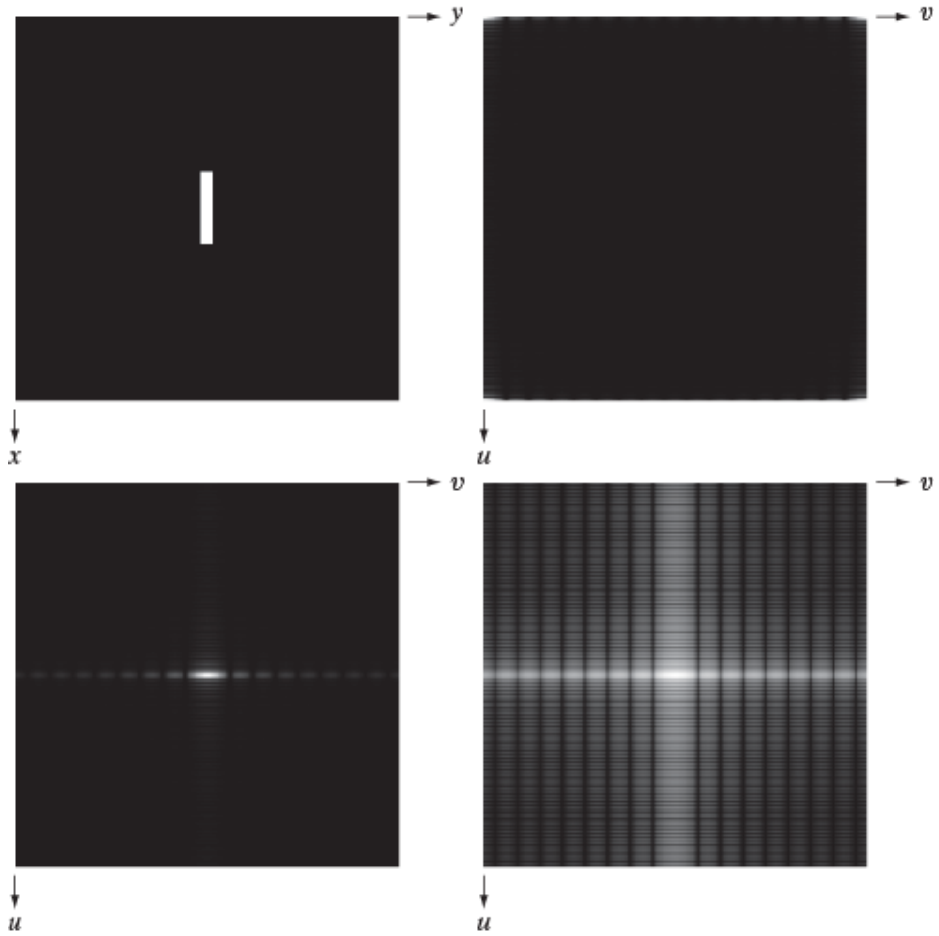
$$F(0,0) = MN\frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y) = MN\bar{f}(x,y) \quad (4\text{-}92)$$

$$|F(0,0)| = MN|\bar{f}(x,y)| \quad (4\text{-}93)$$



4.6 Properties of the 2-D Discrete Fourier Transform

4.6.5 Spectrum and Phase Angle

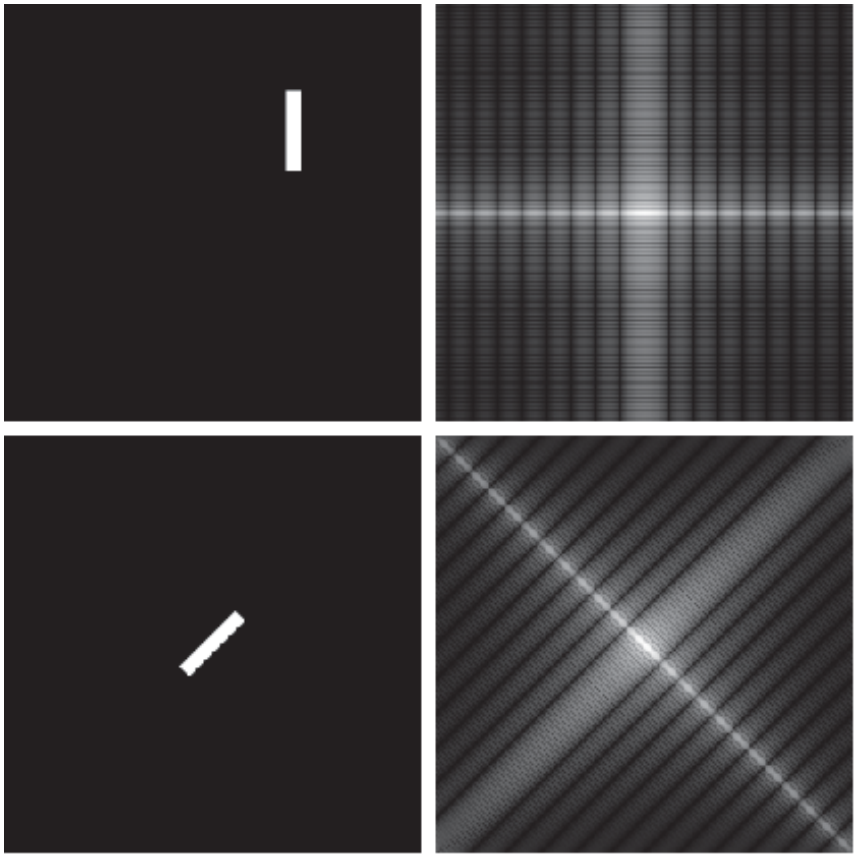


a b
c d

FIGURE 4.23
(a) Image.
(b) Spectrum, showing small, bright areas in the four corners (you have to look carefully to see them).
(c) Centered spectrum.
(d) Result 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 right-handed coordinate convention used in the book places the origin of the spatial and frequency domains at the top left (see Fig. 2.19).

4.6 Properties of the 2-D Discrete Fourier Transform

4.6.5 Spectrum and Phase Angle



a b
c d

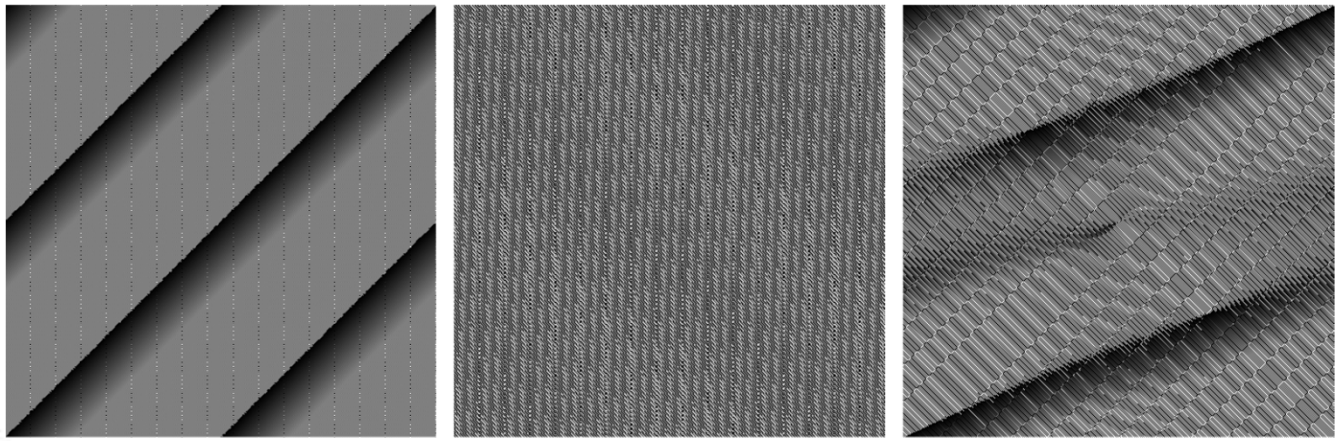
FIGURE 4.24
(a) The rectangle in Fig. 4.23(a) translated.
(b) Corresponding spectrum.
(c) Rotated rectangle.
(d) Corresponding spectrum.
The spectrum of the translated rectangle is identical to the spectrum of the original image in Fig. 4.23(a).

4.6 Properties of the 2-D Discrete Fourier Transform

4.6.5 Spectrum and Phase Angle

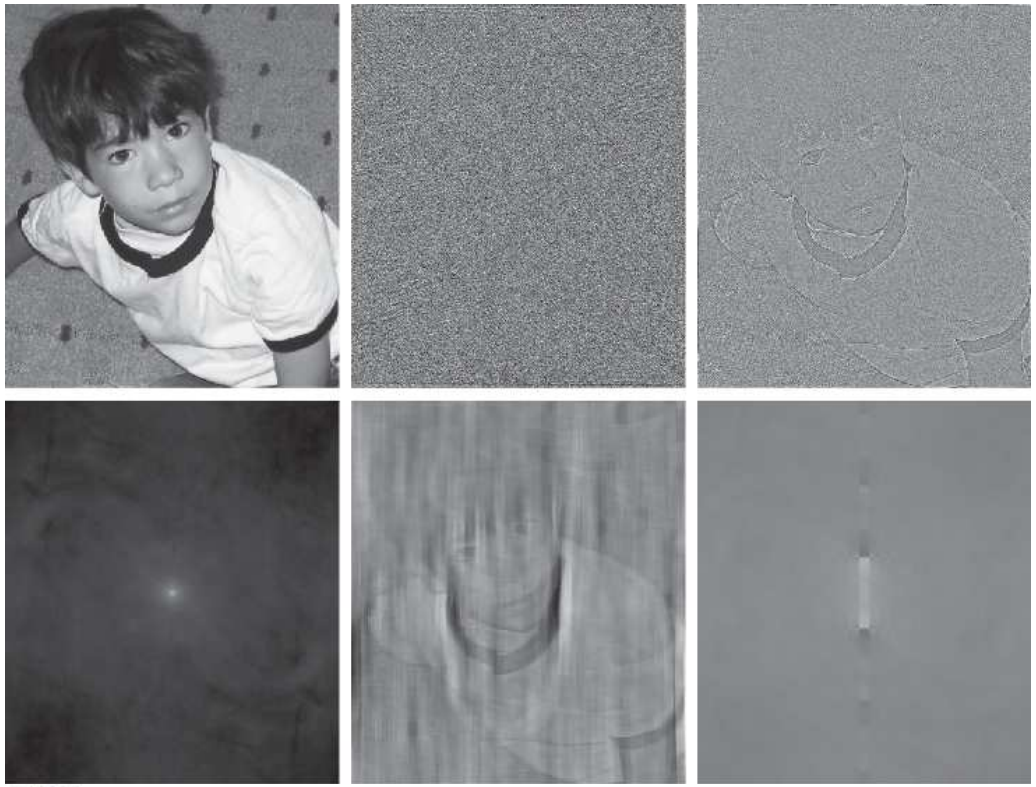
a b c

FIGURE 4.25 Phase angle images of (a) centered, (b) translated, and (c) rotated rectangles.



4.6 Properties of the 2-D Discrete Fourier Transform

4.6.5 Spectrum and Phase Angle

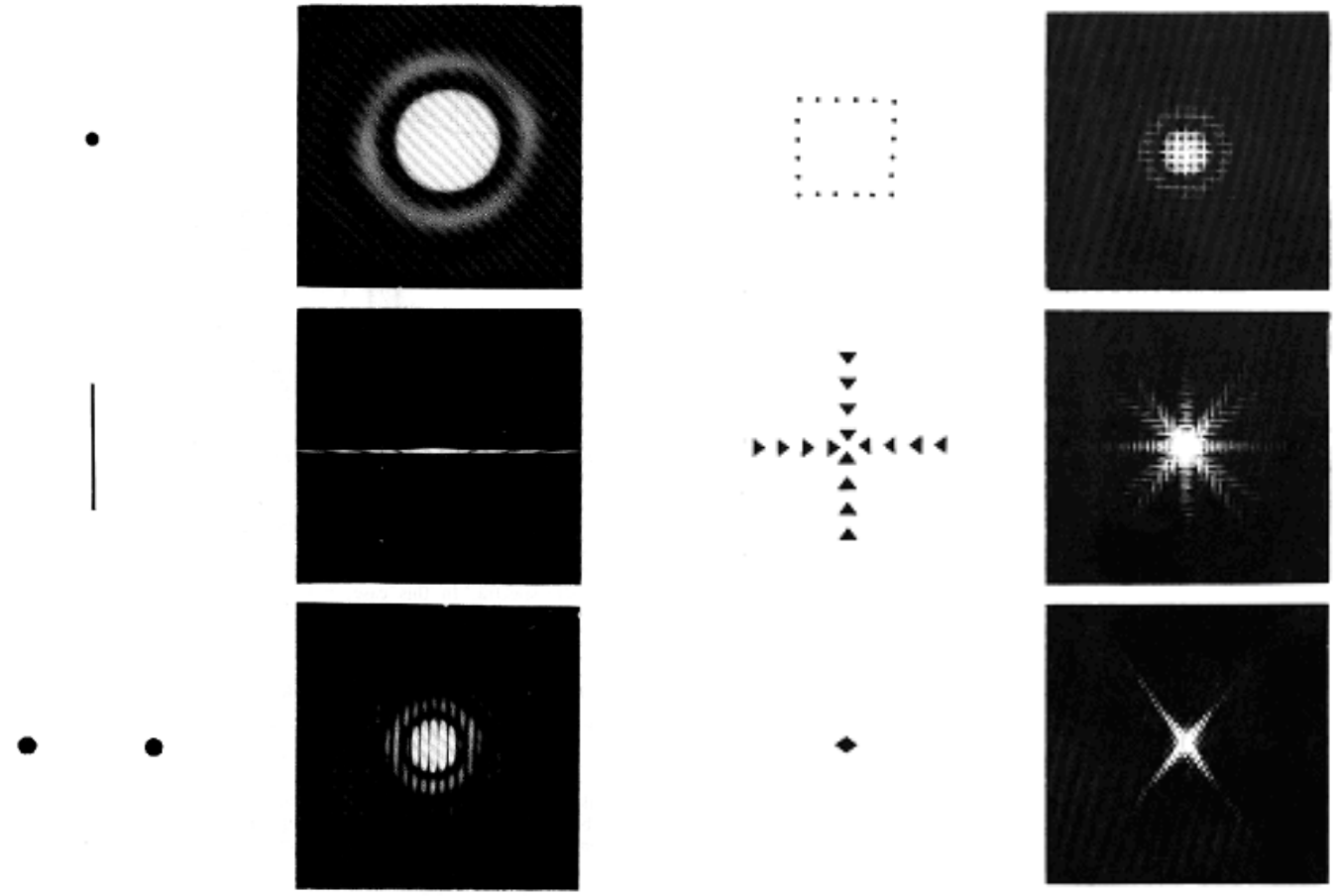


a b c
d e f

FIGURE 4.26 (a) Boy image. (b) Phase angle. (c) Boy image reconstructed using only its phase angle (all shape features are there, but the intensity information is missing because the spectrum was not used in the reconstruction). (d) Boy image reconstructed using only its spectrum. (e) Boy image reconstructed using its phase angle and the spectrum of the rectangle in Fig. 4.23(a). (f) Rectangle image reconstructed using its phase and the spectrum of the boy's image.

4.6 Properties of the 2-D Discrete Fourier Transform

4.6.5 Spectrum and Phase Angle



4.6 Properties of the 2-D Discrete Fourier Transform

4.6.6 The 2-D Convolution Theorem

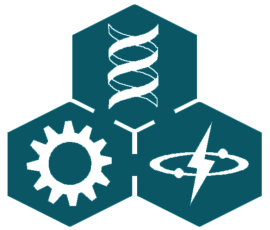
- 2-D Convolution

$$f(x,y)\star h(x,y)=\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}f(m,n)h(x-m,y-n) \qquad (4\text{-}94)$$

- 2-D Convolution Theorem

$$f(x,y)\star h(x,y)\Leftrightarrow F(u,v)H(u,v) \qquad (4\text{-}95)$$

$$f(x,y)h(x,y)\Leftrightarrow F(u,v)\star H(u,v) \qquad (4\text{-}96)$$



4.6 Properties of the 2-D Discrete Fourier Transform

4.6.6 The 2-D Convolution Theorem

a f
b g
c h
d i
e j

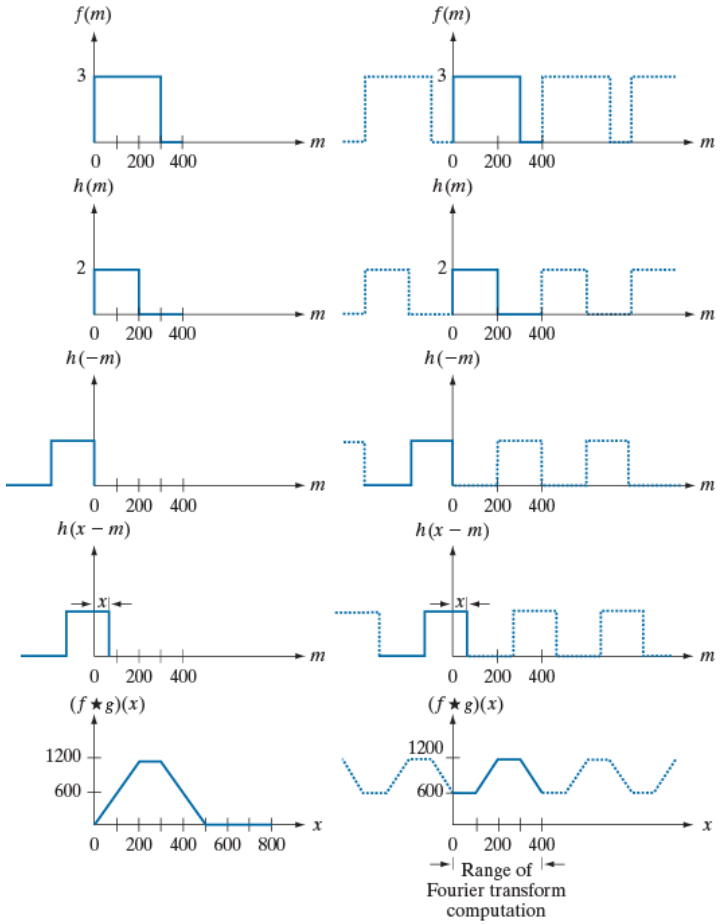


FIGURE 4.27 Left column: Spatial convolution computed with Eq. (3-44), using the approach discussed in Section 3.4. Right column: Circular convolution. The solid line in (j) is the result we would obtain using the DFT, or, equivalently, Eq. (4-48). This erroneous result can be remedied by using zero padding.

4.6 Properties of the 2-D Discrete Fourier Transform

4.6.6 The 2-D Convolution Theorem

- The period of discrete convolution

Period requirement for 1-D discrete convolution

$$P \geq A + B - 1 \tag{4-97}$$

Extend the period of the function: zero padding

Period requirement for 2-D discrete convolution

$$f_p(x,y) = \begin{cases} f(x,y) & 0 \leq x \leq A-1 \text{ and } 0 \leq y \leq B-1 \\ 0 & A \leq x \leq P \text{ or } B \leq y \leq Q \end{cases} \tag{4-98}$$

$$h_p(x,y) = \begin{cases} h(x,y) & 0 \leq x \leq C-1 \text{ and } 0 \leq y \leq D-1 \\ 0 & C \leq x \leq P \text{ or } D \leq y \leq Q \end{cases} \tag{4-99}$$

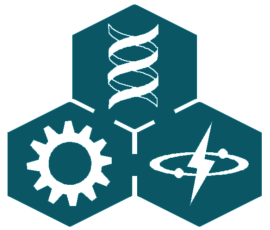
$$P \geq A + C - 1 \tag{4-100}$$

$$Q \geq B + D - 1 \tag{4-101}$$

If both arrays are of the same size, $M \times N$, then

$$P \geq 2M - 1 \tag{4-102}$$

$$Q \geq 2N - 1 \tag{4-103}$$

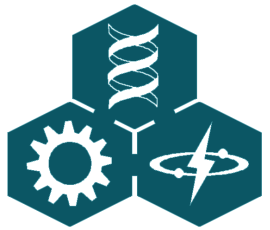


4.6 Properties of the 2-D Discrete Fourier Transform

4.6.7 Summary of 2-D Discrete Fourier Transform Properties

TABLE 4.3 Summary of DFT definitions and corresponding expressions.

	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)	Spectrum	$\lvert F(u,v)\rvert = \left[R^2(u,v) + I^2(u,v)\right]^{1/2}$ $R = \text{Real}(F)$; $I = \text{Imag}(F)$
4)	Phase angle	$\phi(u,v) = \tan^{-1}\left[\frac{I(u,v)}{R(u,v)}\right]$
5)	Polar representation	$F(u,v) = \lvert F(u,v)\rvert e^{j\phi(u,v)}$
6)	Power spectrum	$P(u,v) = \lvert F(u,v)\rvert^2$
7)	Average value	$\overline{f} = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y) = \frac{1}{MN}F(0,0)$
8)	Periodicity (k_1 and k_2 are integers)	$F(u,v) = F(u+k_1M,v) = F(u,v+k_2N)$ $= F(u+k_1,v+k_2N)$ $f(x,y) = f(x+k_1M,y) = f(x,y+k_2N)$ $= f(x+k_1M,y+k_2N)$
9)	Convolution	$(f \star h)(x,y) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)h(x-m,y-n)$
10)	Correlation	$(f \star h)(x,y) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f^*(m,n)h(x+m,y+n)$
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.
12)	Obtaining the IDFT using a DFT 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.



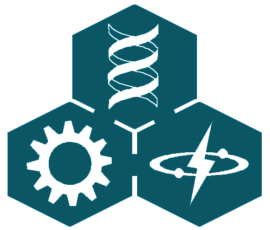
4.6 Properties of the 2-D Discrete Fourier Transform

4.6.7 Summary of 2-D Discrete Fourier Transform Properties

TABLE 4.4 Summmary 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 continuous expressions.

	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	$f(r, \theta+\theta_0) \Leftrightarrow F(\omega, \varphi+\theta_0)$ $r = \sqrt{x^2+y^2} \quad \theta = \tan^{-1}(y/x) \quad \omega = \sqrt{u^2+v^2} \quad \varphi = \tan^{-1}(v/u)$
6)	Convolution theorem†	$f \star h(x,y) \Leftrightarrow (F \bullet H)(u,v)$ $(f \bullet h)(x,y) \Leftrightarrow (1/MN)[(F \star H)(u,v)]$
7)	Correlation theorem†	$(f \star h)(x,y) \Leftrightarrow (F^* \bullet H)(u,v)$ $(f^* \bullet h)(x,y) \Leftrightarrow (1/MN)[(F \star H)(u,v)]$
8)	Discrete unit impulse	$\delta(x,y) \Leftrightarrow 1$ $1 \Leftrightarrow MN\delta(u,v)$
9)	Rectangle	$\text{rec}[a,b] \Leftrightarrow ab\frac{\sin(\pi ua)}{(\pi ua)}\frac{\sin(\pi vb)}{(\pi vb)}e^{-j\pi(ua+vb)}$
10)	Sine	$\sin(2\pi u_0x/M + 2\pi v_0y/N) \Leftrightarrow \frac{jMN}{2}[\delta(u+u_0, v+v_0) - \delta(u-u_0, v-v_0)]$
11)	Cosine	$\cos(2\pi u_0x/M + 2\pi v_0y/N) \Leftrightarrow \frac{1}{2}[\delta(u+u_0, v+v_0) + \delta(u-u_0, v-v_0)]$
	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 $f(\pm\infty, \pm\infty) = 0$.)	$\left(\frac{\partial}{\partial t}\right)^m\left(\frac{\partial}{\partial z}\right)^n f(t,z) \Leftrightarrow (j2\pi\mu)^m(j2\pi\nu)^n F(\mu,\nu)$ $\frac{\partial^m f(t,z)}{\partial t^m} \Leftrightarrow (j2\pi\mu)^m F(\mu,\nu); \quad \frac{\partial^n f(t,z)}{\partial z^m} \Leftrightarrow (j2\pi\nu)^n F(\mu,\nu)$
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)

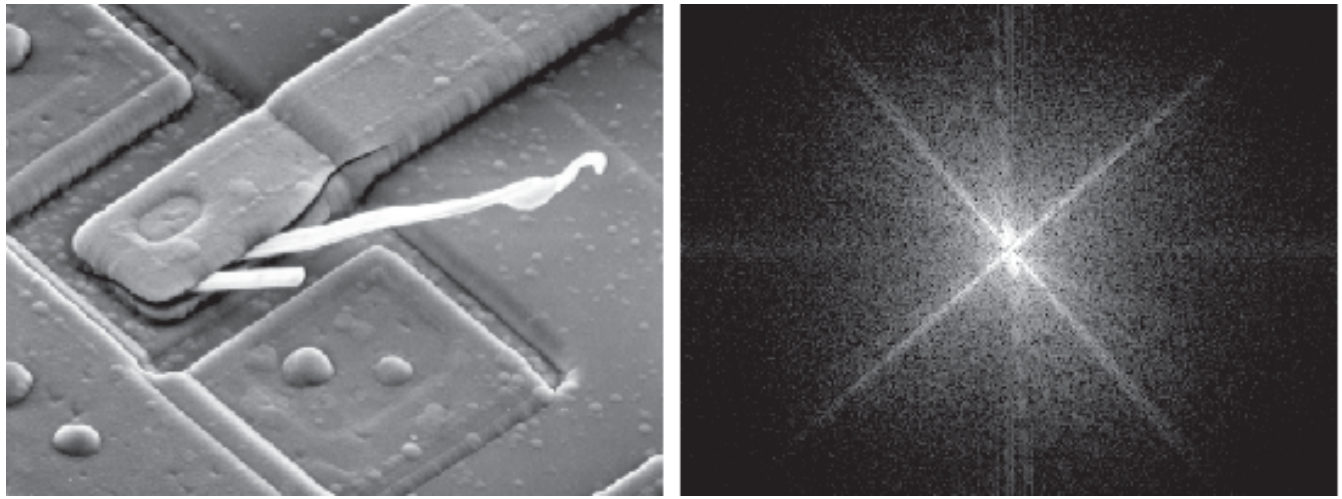
†Assumes that $f(x,y)$ and $h(x,y)$ have been properly padded. Convolution is associative, commutative, and distributive. Correlation is distributive (see Table 3.5). The products are elementwise products (see Section 2.6).



4.7 The Basics of Filtering in the Frequency Domain

4.7.1 Additional Characteristics of the Frequency Domain

- Relationship between the frequency and varying intensity of an image



a b

FIGURE 4.28 (a) SEM image of a damaged integrated circuit. (b) Fourier spectrum of (a). (Original image courtesy of Dr. J. M. Hudak, Brockhouse Institute for Materials Research, McMaster University, Hamilton, Ontario, Canada.)

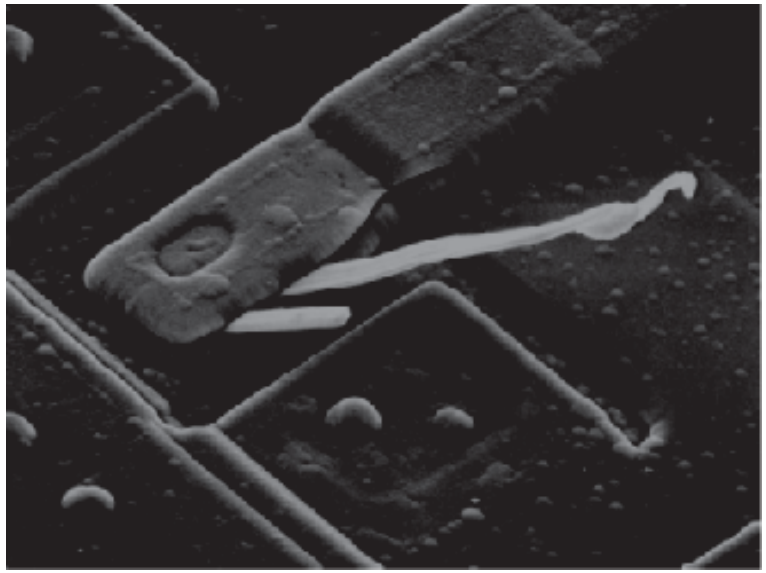
4.7 The Basics of Filtering in the Frequency Domain

4.7.2 Frequency Domain Filtering Fundamentals

$$g(x,y) = \text{Real}\left\{\Im^{-1}[H(u,v)F(u,v)]\right\} \qquad (4\text{-}104)$$

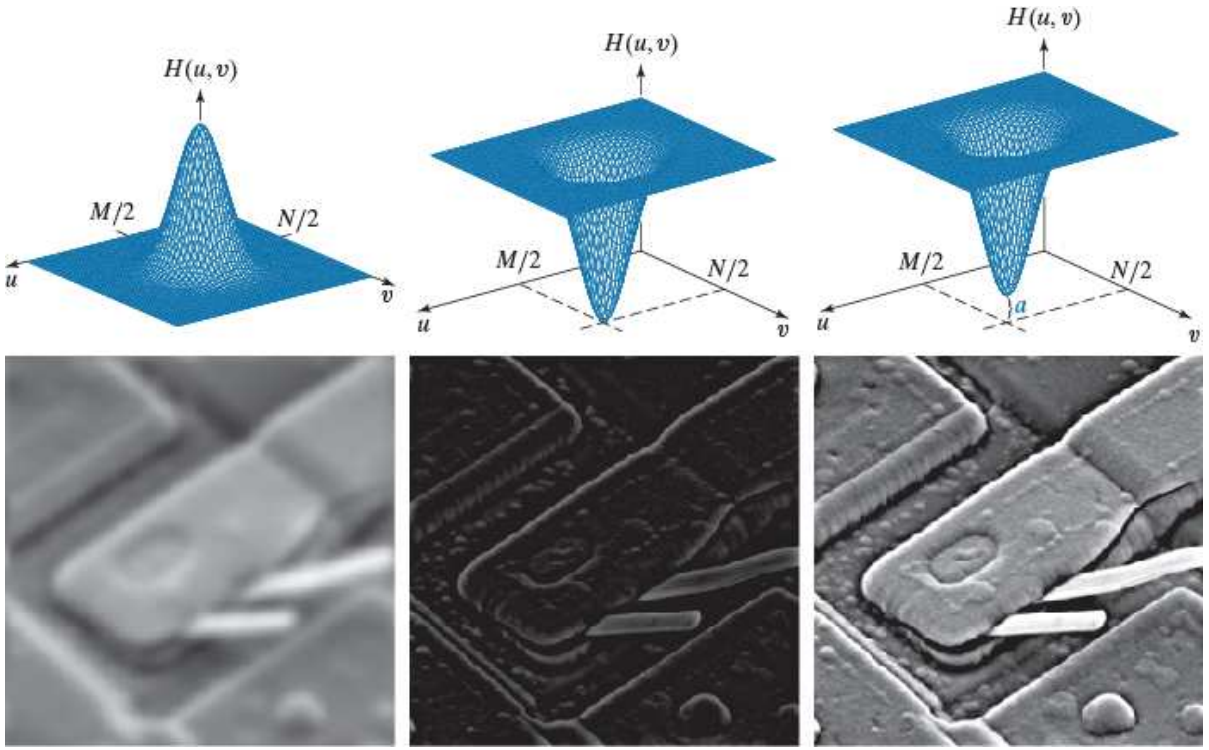
$$H(u,v) = \begin{cases} 0 & \text{if } (u,v) = (M/2, N/2) \\ 1 & \text{otherwise} \end{cases}$$

FIGURE 4.29 Result of filtering the image in Fig. 4.28(a) with a filter transfer function that sets to 0 the dc term, $F(P/2,Q/2)$, in the centered Fourier transform, while leaving all other transform terms unchanged.



4.7 The Basics of Filtering in the Frequency Domain

4.7.2 Frequency Domain Filtering Fundamentals



a b c
d e f

FIGURE 4.30 Top row: Frequency domain filter transfer functions of (a) a lowpass filter, (b) a highpass filter, and (c) an offset highpass filter. Bottom row: Corresponding filtered images obtained using Eq. (4-104). The offset in (c) is $a = 0.85$, and the height of $H(u,v)$ is 1. Compare (f) with Fig. 4.28(a).

4.7 The Basics of Filtering in the Frequency Domain

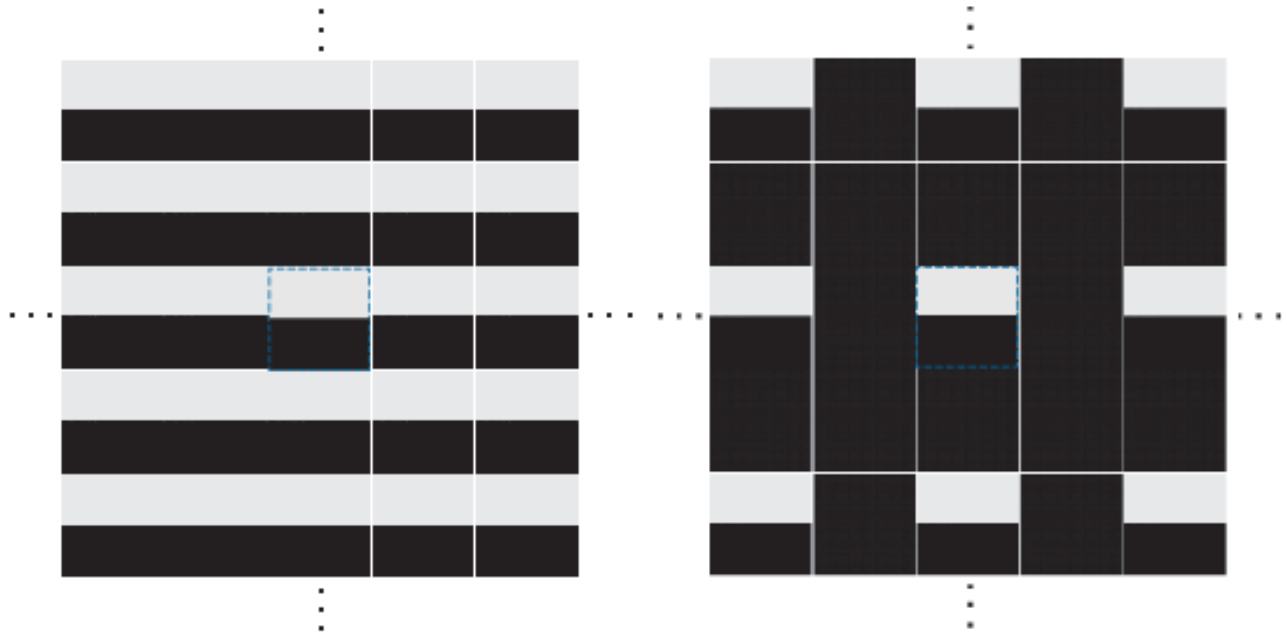
4.7.2 Frequency Domain Filtering Fundamentals

a b c

FIGURE 4.31 (a) A simple image. (b) Result of blurring with a Gaussian lowpass filter without padding. (c) Result of lowpass filtering with zero padding. Compare the vertical edges in (b) and (c).

4.7 The Basics of Filtering in the Frequency Domain

4.7.2 Frequency Domain Filtering Fundamentals



a b

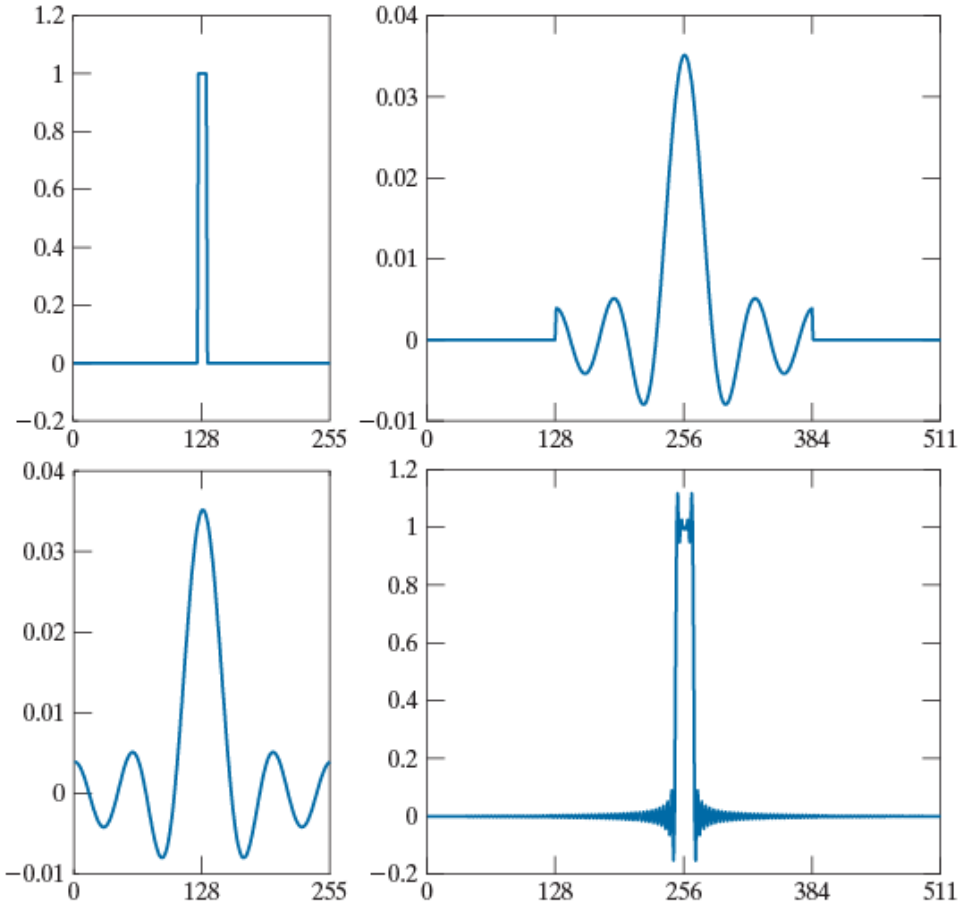
FIGURE 4.32 (a) Image periodicity without image padding. (b) Periodicity after padding with 0's (black). The dashed areas in the center correspond to the image in Fig. 4.31(a). Periodicity is inherent when using the DFT. (The thin white lines in both images are superimposed for clarity; they are not part of the data.)

4.7 The Basics of Filtering in the Frequency Domain

4.7.2 Frequency Domain Filtering Fundamentals

a c
b d

FIGURE 4.33
(a) Filter transfer function specified in the (centered) frequency domain. (b) Spatial representation (filter kernel) obtained by computing the IDFT of (a). (c) Result of padding (b) to twice its length (note the discontinuities). (d) Corresponding filter in the frequency domain obtained by computing the DFT of (c). Note the ringing caused by the discontinuities in (c). Part (b) of the figure is below (a), and (d) is below (c).



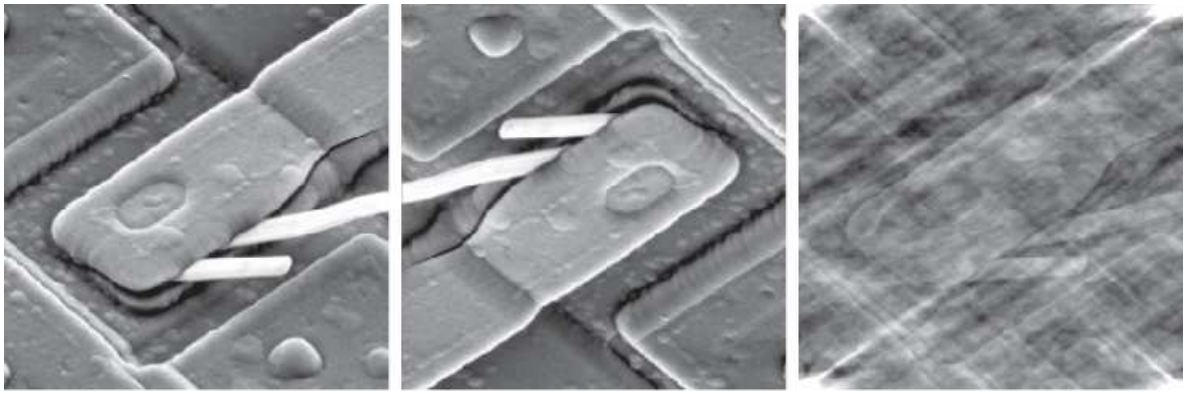
4.7 The Basics of Filtering in the Frequency Domain

4.7.2 Frequency Domain Filtering Fundamentals

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

$$g(x,y) = \Im^{-1}[H(u,v)R(u,v) + jH(u,v)I(u,v)] \quad (4\text{-}105)$$

Filters that affect the real and imaginary parts equally, and thus have no effect on the phase, are appropriately called ***zero-phase-shift filters***.



a b c

FIGURE 4.34 (a) Original image. (b) Image obtained by multiplying the phase angle array by –1 in Eq. (4-86) and computing the IDFT. (c) Result of multiplying the phase angle by 0.25 and computing the IDFT. The magnitude of the transform, $|F(u,v)|$, used in (b) and (c) was the same.

4.7 The Basics of Filtering in the Frequency Domain

4.7.3 Summary of Steps for Filtering in the Frequency Domain

(1) Given an input image $f(x, y)$ of size $M \times N$, obtain the padding parameters P and Q from Eqs. (4-102) and (4-103).

(2) Form a padded image depend on P and Q.

(3) Multiply by $(-1)^{x+y}$ to center its transform.

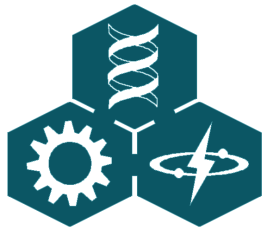
(4) Compute the DFT, $F(u, v)$, of the image from step (3).

(5) Generate a real, symmetric filter function, $H(u, v)$, of size $P \times Q$. Form the product $G(u,v) = H(u,v)F(u,v)$。

(6) Compute the IDFT of the result from step (5).

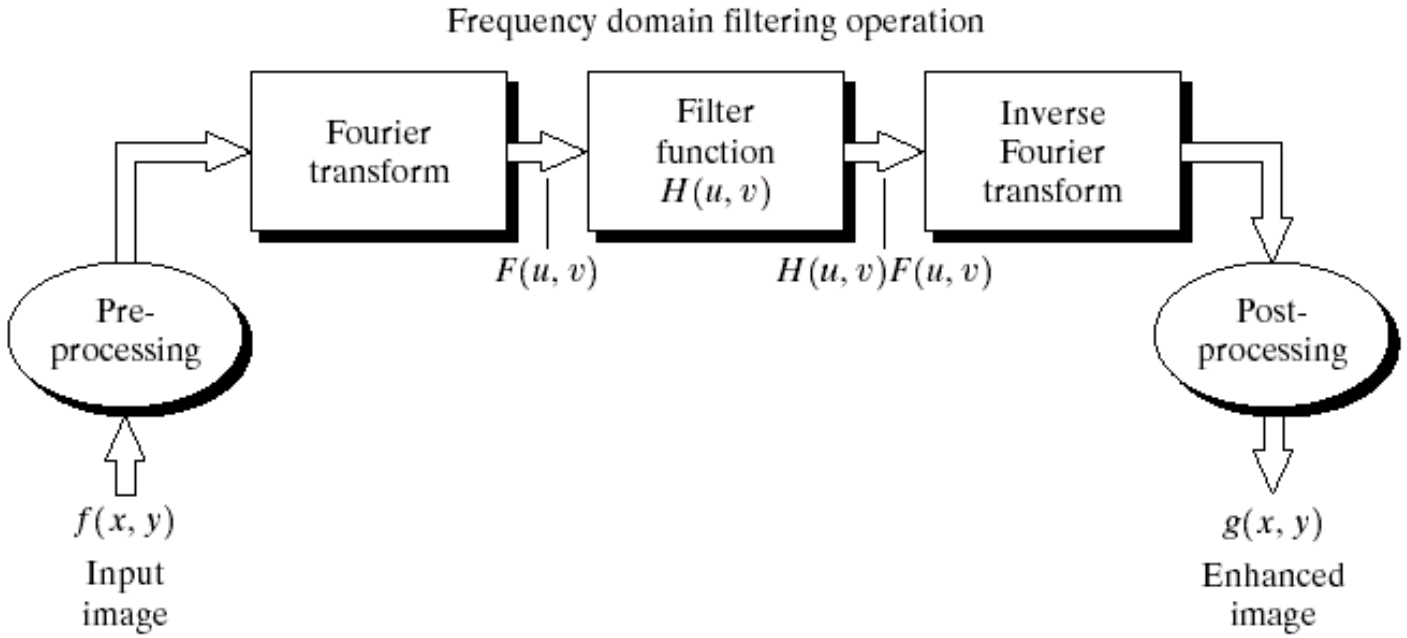
$$g_p(x,y) = \{real[\Im^{-1}[G(u,v)]]\}(-1)^{x+y}$$

(7) Extracting the $M \times N$ region from the top, left quadrant of $g_p(x, y)$.



4.7 The Basics of Filtering in the Frequency Domain

4.7.3 Summary of Steps for Filtering in the Frequency Domain



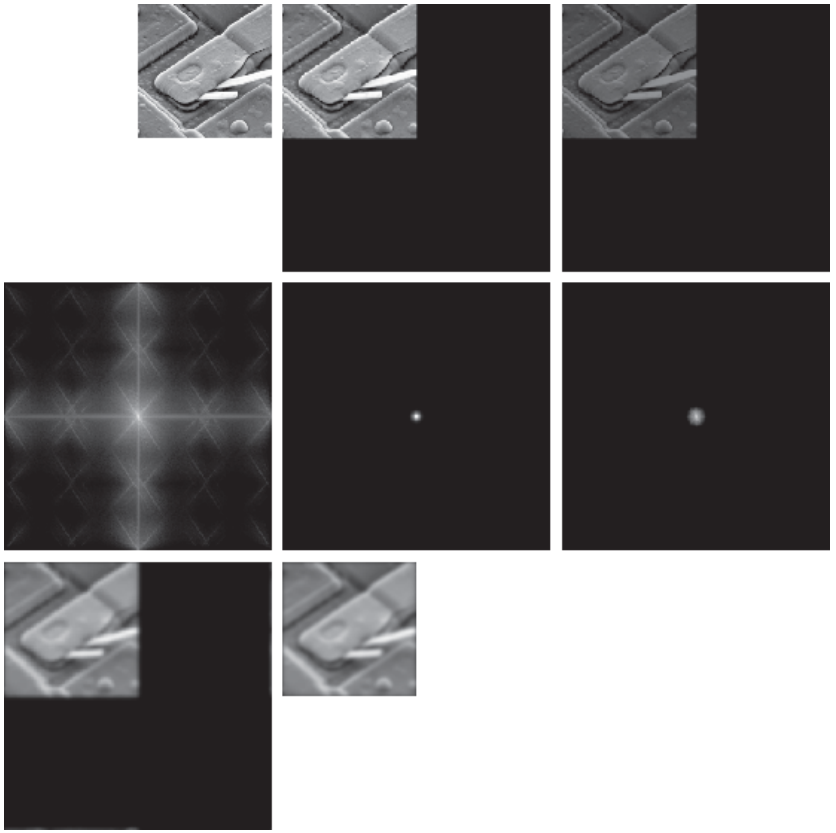
Basic steps for filtering in the frequency domain.

4.7 The Basics of Filtering in the Frequency Domain

4.7.3 Summary of Steps for Filtering in the Frequency Domain

a b c
d e f
g h

FIGURE 4.35
(a) An $M \times N$ image, f.
(b) Padded image, f_p of size $P \times Q$.
(c) Result of multiplying f_p by $(-1)^{x+y}$.
(d) Spectrum of F. (e) Centered Gaussian lowpass filter transfer function, H, of size $P \times Q$.
(f) Spectrum of the product HF.
(g) Image g_p, the real part of the IDFT of HF, multiplied by $(-1)^{x+y}$.
(h) Final result, g, obtained by extracting the first M rows and N columns of g_p.



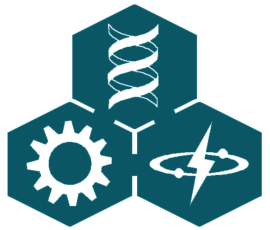
4.7 The Basics of Filtering in the Frequency Domain

4.7.4 Correspondence between Filtering in the Spatial and Frequency Domains

- Filtering in the Spatial and Frequency Domains

  Based on the convolution theorem and impulse response, the frequency domain filter and the spatial filter form a Fourier transform pair:

$$
\begin{aligned}
f(x,y) * h(x,y) &\Leftrightarrow F(u,v)H(u,v) \\
\delta(x,y) * h(x,y) &\Leftrightarrow \Im[\delta(x,y)]H(u,v) \\
h(x,y) &\Leftrightarrow H(u,v)
\end{aligned}
\qquad (4\text{-}106)
$$



4.7 The Basics of Filtering in the Frequency Domain

4.7.4 Correspondence Between Filtering in the Spatial and Frequency Domains

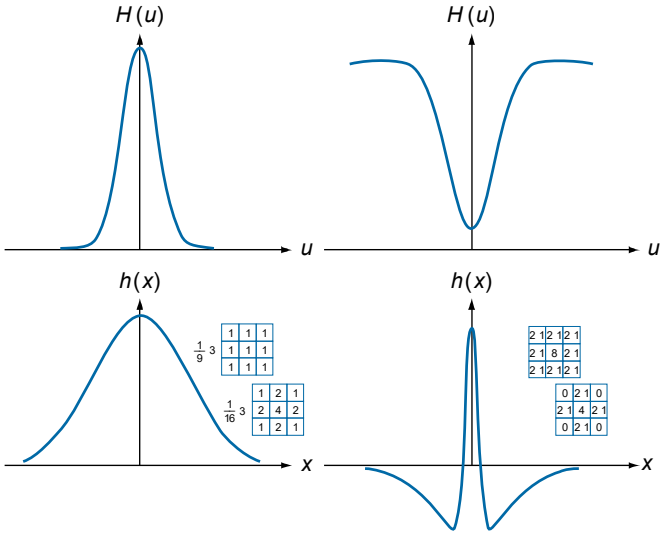
- Gaussian filter

$$H(u) = Ae^{-u^2/2\sigma^2} \qquad (4\text{-}107)$$

$$h(x) = \sqrt{2\pi}\sigma A e^{-2\pi^2\sigma^2 x^2} \qquad (4\text{-}108)$$

$$H(u) = Ae^{-u^2/2\sigma_1^2} - Be^{-u^2/2\sigma_2^2} \qquad (4\text{-}109)$$

$$h(x) = \sqrt{2\pi}\sigma_1 A e^{-2\pi^2\sigma_1^2 x^2} - \sqrt{2\pi}\sigma_2 A e^{-2\pi^2\sigma_2^2 x^2} \qquad (4\text{-}110)$$



a c
b d

FIGURE 4.36
(a) A 1-D Gaussian lowpass transfer function in the frequency domain. (b) Corresponding kernel in the spatial domain. (c) Gaussian highpass transfer function in the frequency domain. (d) Corresponding kernel. The small 2-D kernels shown are kernels we used in Chapter 3.

4.7 The Basics of Filtering in the Frequency Domain

4.7.4 Correspondence Between Filtering in the Spatial and Frequency Domains

a b

FIGURE 4.37
(a) Image of a building, and (b) its Fourier spectrum.

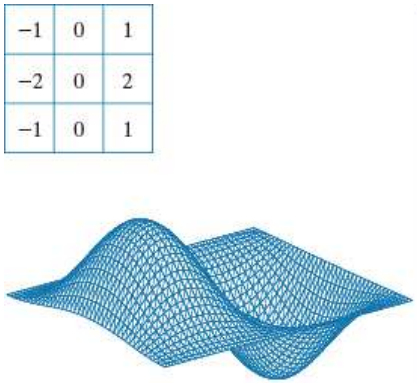
4.7 The Basics of Filtering in the Frequency Domain

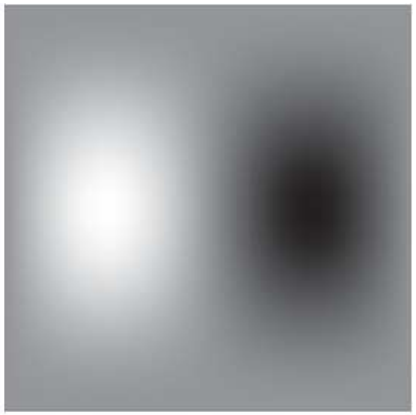
4.7.4 Correspondence Between Filtering in the Spatial and Frequency Domains

a b
c d

FIGURE 4.38
(a) A spatial kernel and perspective plot of its corresponding frequency domain filter transfer function.
(b) Transfer function shown as an image.
(c) Result of filtering Fig. 4.37(a) in the frequency domain with the transfer function in (b).
(d) Result of filtering the same image in the spatial domain with the kernel in (a). The results are identical.

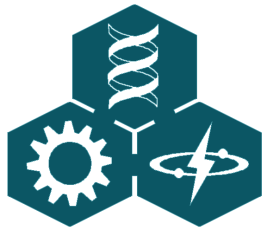
−1	0	1
−2	0	2
−1	0	1





4.8 Image Smoothing Using Frequency Domain Filters

- $G(u,v)=H(u,v)F(u,v)$
  - $H(u,v)$ is Zero-phase-shift Filter
- Ideal Lowpass Filters
- Butterworth Lowpass Filters
- Gaussian Lowpass Filters



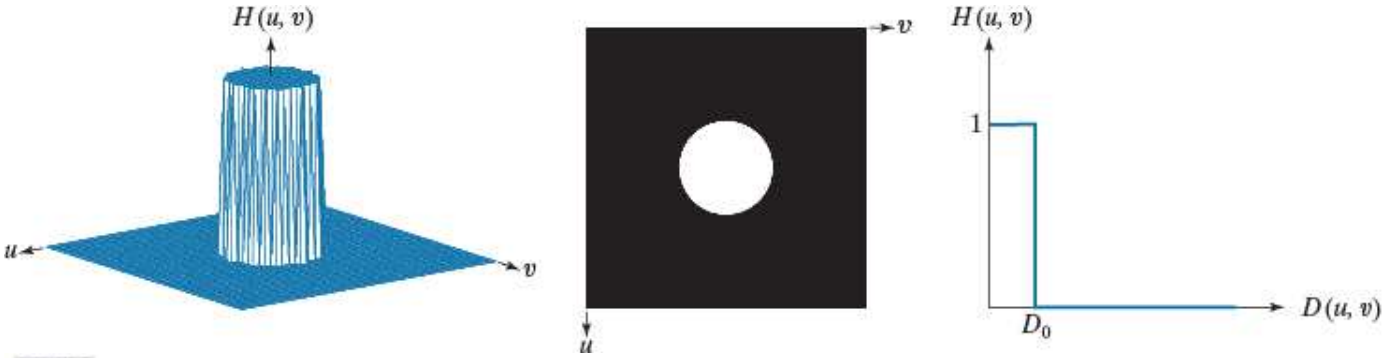
4.8 Image Smoothing Using Frequency Domain Filters

4.8.1 Ideal Lowpass Filter (ILPF)

$$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \le D_0 \\ 0 & \text{if } D(u,v) > D_0 \end{cases} \qquad (4\text{-}111)$$

$$D(u,v) = [(u - P/2)^2 + (v - Q/2)^2]^{1/2} \qquad (4\text{-}112)$$

D_0: Cutoff Frequency



a b c

FIGURE 4.39 (a) Perspective plot of an ideal lowpass-filter transfer function. (b) Function displayed as an image. (c) Radial cross section.

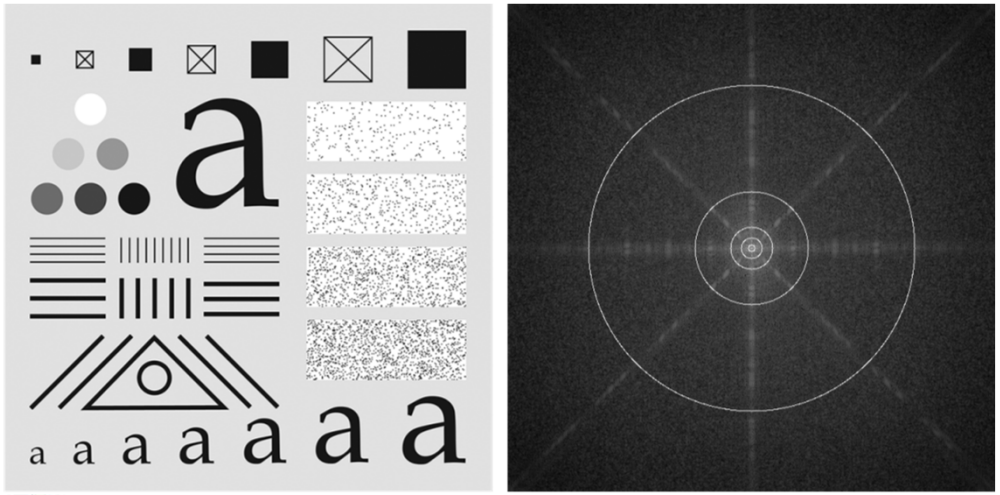
4.8 Image Smoothing Using Frequency Domain Filters

4.8.1 Ideal Lowpass Filter (ILPF)

- Image Power

$$P_T = \sum_{u=0}^{P-1}\sum_{v=0}^{Q-1} P(u,v) \qquad (4\text{-}113)$$

$$\alpha = 100\left[\sum_{u}\sum_{v} P(u,v) / P_T\right] \qquad (4\text{-}114)$$

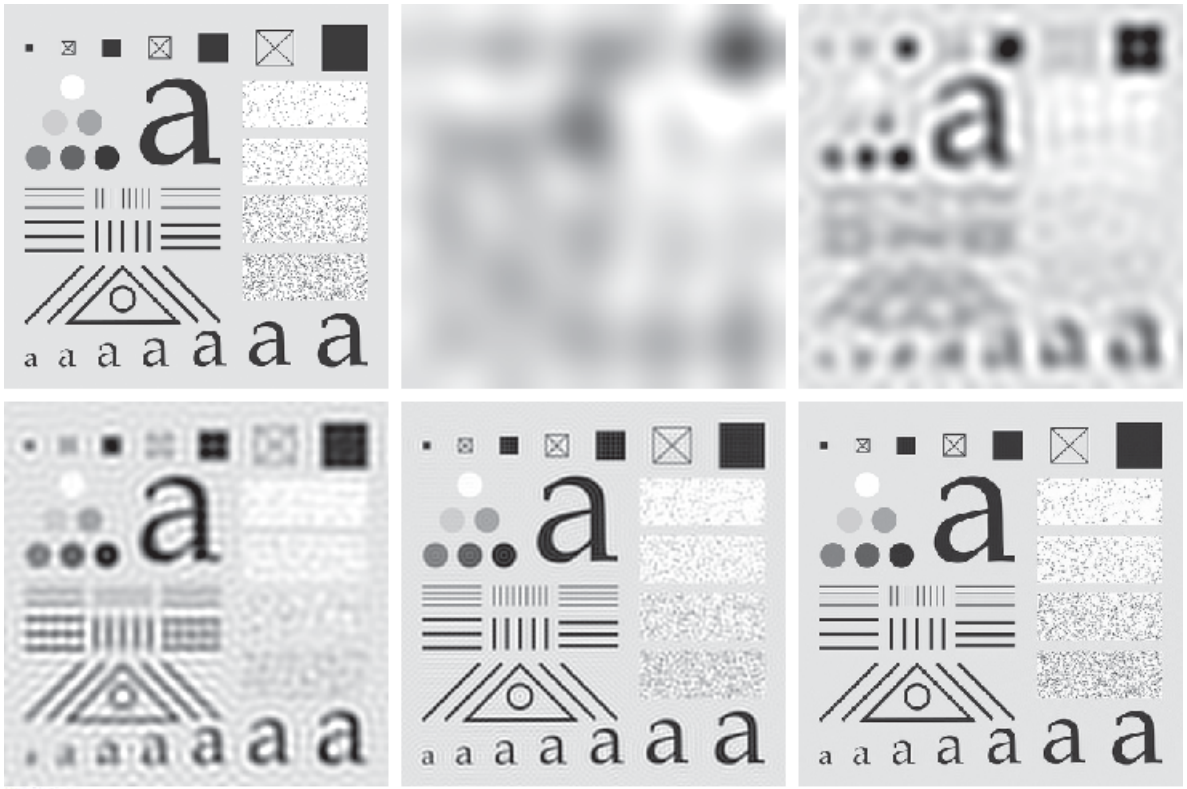


a b

FIGURE 4.40 (a) Test pattern of size 688 × 688 pixels, and (b) its spectrum. The spectrum is double the image size as a result of padding, but is shown half size to fit. The circles have radii of 10, 30, 60, 160, and 460 pixels with respect to the full-size spectrum. The radii enclose 86.9, 92.8, 95.1, 97.6, and 99.4% of the padded image power, respectively.

4.8 Image Smoothing Using Frequency Domain Filters

4.8.1 Ideal Lowpass Filter (ILPF)



a b c
d e f

FIGURE 4.41 (a) Original image of size 688 × 688 pixels. (b)–(f) Results of filtering using ILPFs with cutoff frequencies set at radii values 10, 30, 60, 160, and 460, as shown in Fig. 4.40(b). The power removed by these filters was 13.1, 7.2, 4.9, 2.4, and 0.6% of the total, respectively. We used mirror padding to avoid the black borders characteristic of zero padding, as illustrated in Fig. 4.31(c).

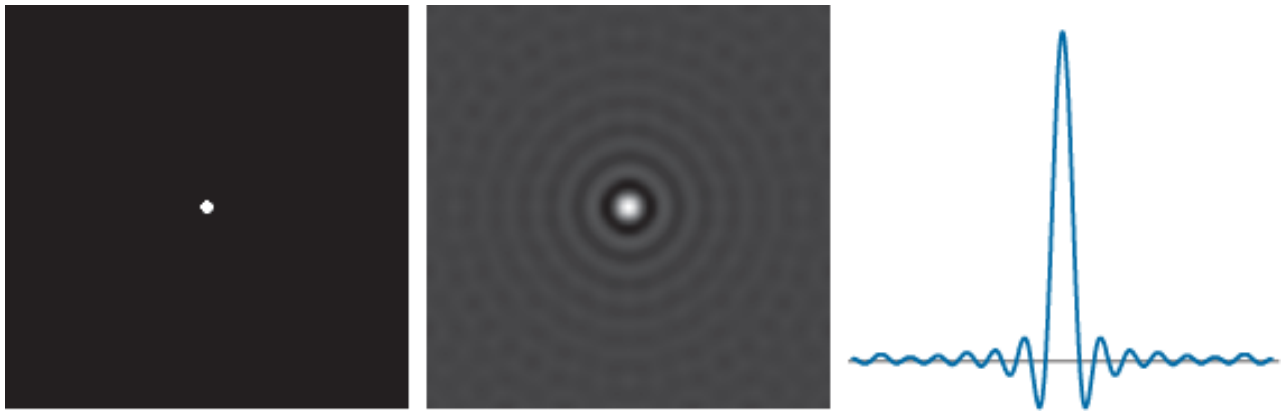
4.8 Image Smoothing Using Frequency Domain Filters

4.8.1 Ideal Lowpass Filter (ILPF)

- Correspondence in Spatial and Frequency Domain

a b c

FIGURE 4.42
(a) Frequency domain ILPF transfer function. (b) Corresponding spatial domain kernel function. (c) Intensity profile of a horizontal line through the center of (b).



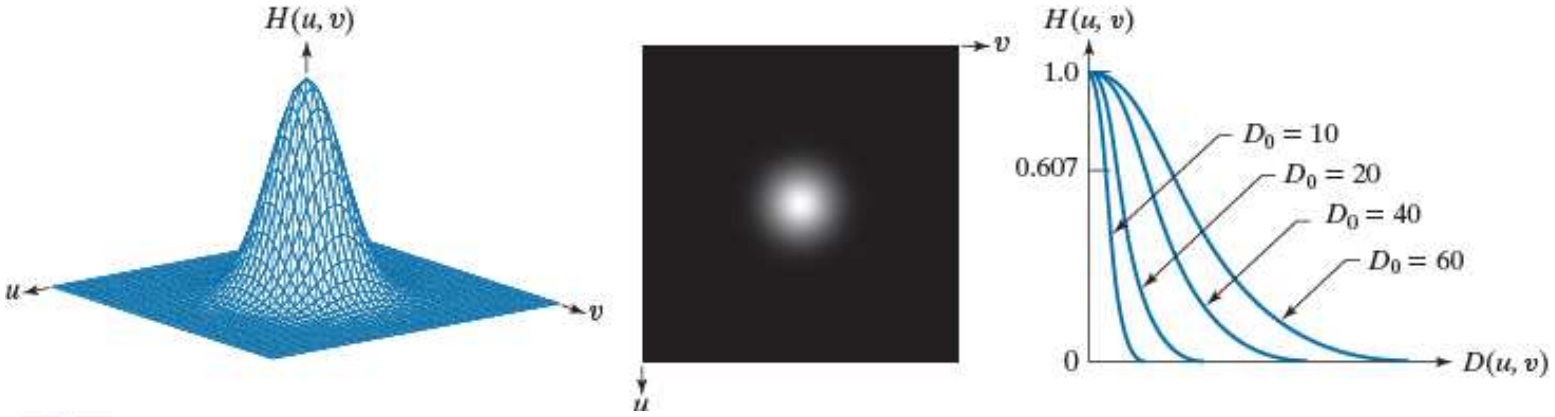
4.8 Image Smoothing Using Frequency Domain Filters

4.8.2 Gaussian Lowpass Filters

$$H(u,v) = e^{-D^2(u,v)/2\sigma^2} \qquad (4\text{-}115)$$

$$H(u,v) = e^{-D^2(u,v)/2D_0^2} \qquad (4\text{-}116)$$

- Gaussian Filter has no ringing effect

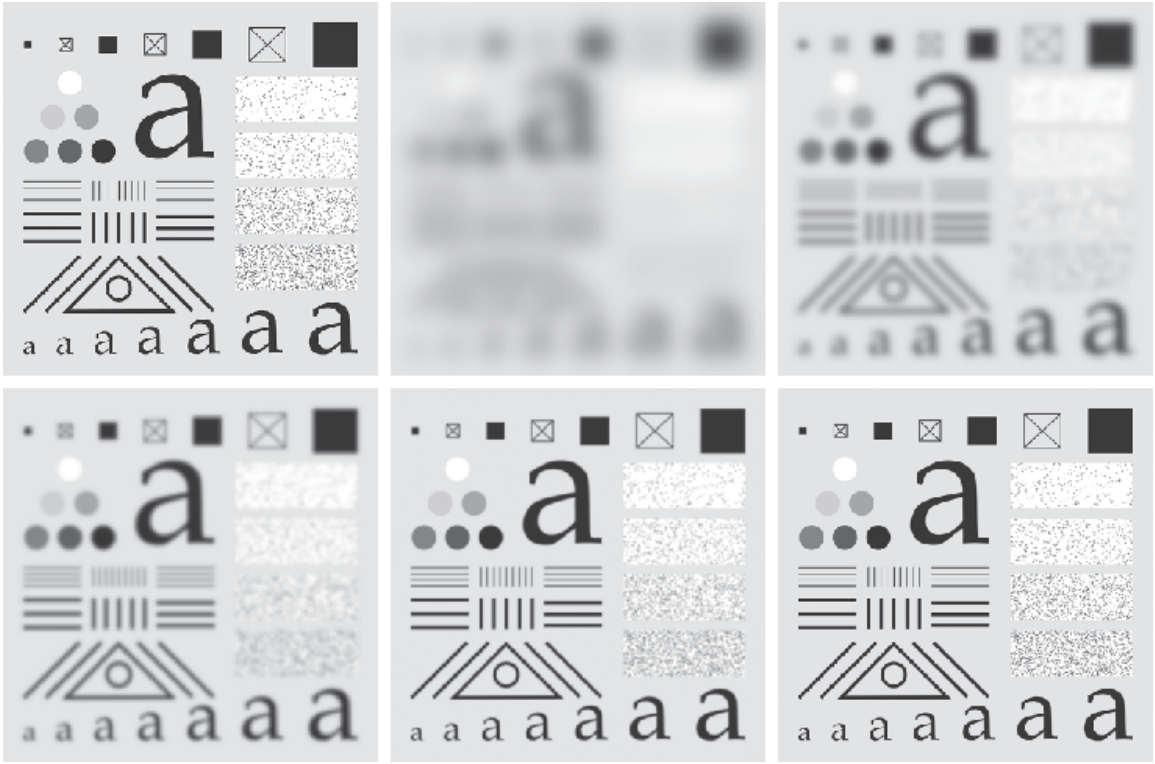


a b c

FIGURE 4.43 (a) Perspective plot of a GLPF transfer function. (b) Function displayed as an image. (c) Radial cross sections for various values of D_0.

4.8 Image Smoothing Using Frequency Domain Filters

4.8.2 Gaussian Lowpass Filters



a b c
d e f

FIGURE 4.44 (a) Original image of size 688 × 688 pixels. (b)–(f) Results of filtering using GLPFs with cutoff frequencies at the radii shown in Fig. 4.40. Compare with Fig. 4.41. We used mirror padding to avoid the black borders characteristic of zero padding.

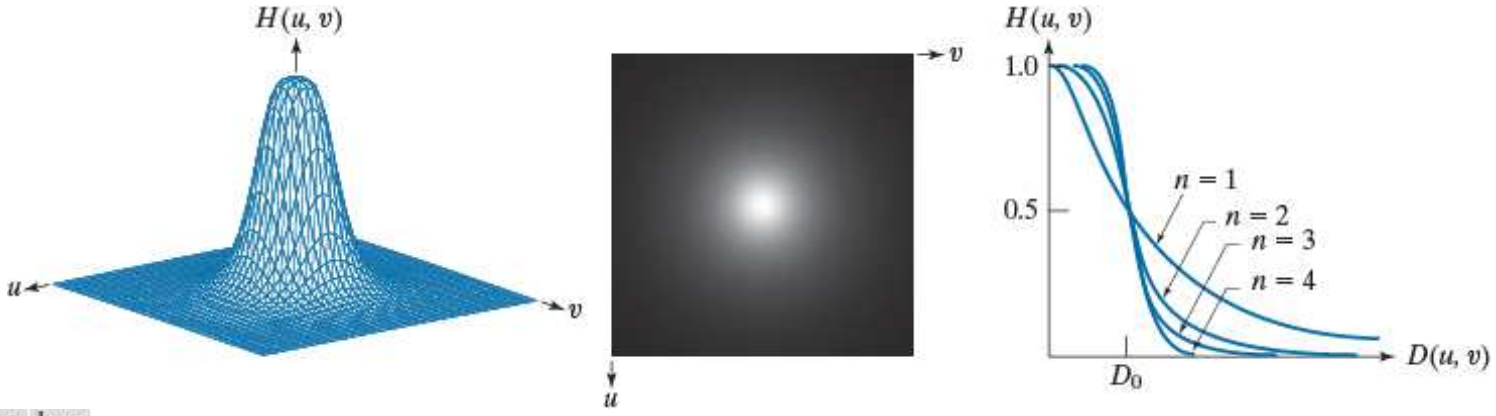
4.8 Image Smoothing Using Frequency Domain Filters

4.8.3 Butterworth Lowpass Filters

$$H(u,v) = \frac{1}{1+[D(u,v)/D_0]^{2n}} \qquad (4\text{-}117)$$

Cutoff frequency is defined at point for which

$H(u,v) = 0.5$ when $D(u,v) = D_0$

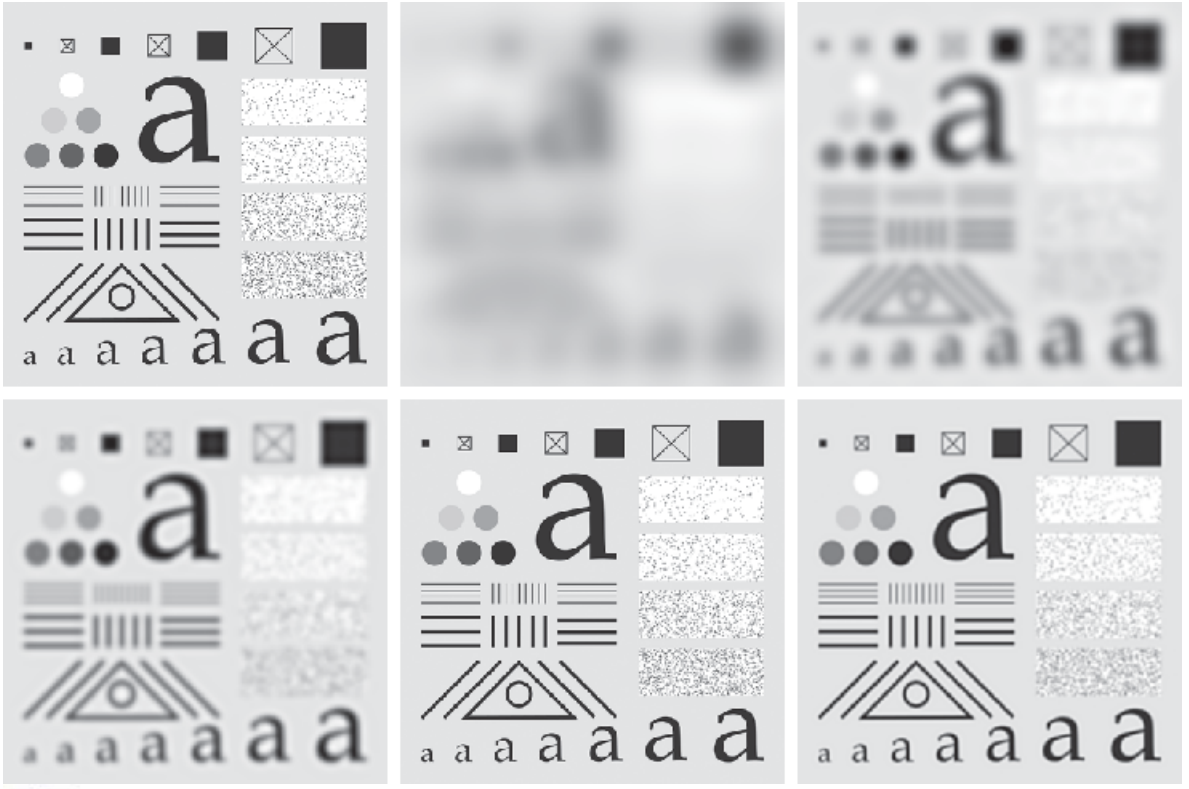


a b c

FIGURE 4.45 (a) Perspective plot of a Butterworth lowpass-filter transfer function. (b) Function displayed as an image. (c) Radial cross sections of BLPFs of orders 1 through 4.

4.8 Image Smoothing Using Frequency Domain Filters

4.8.3 Butterworth Lowpass Filters



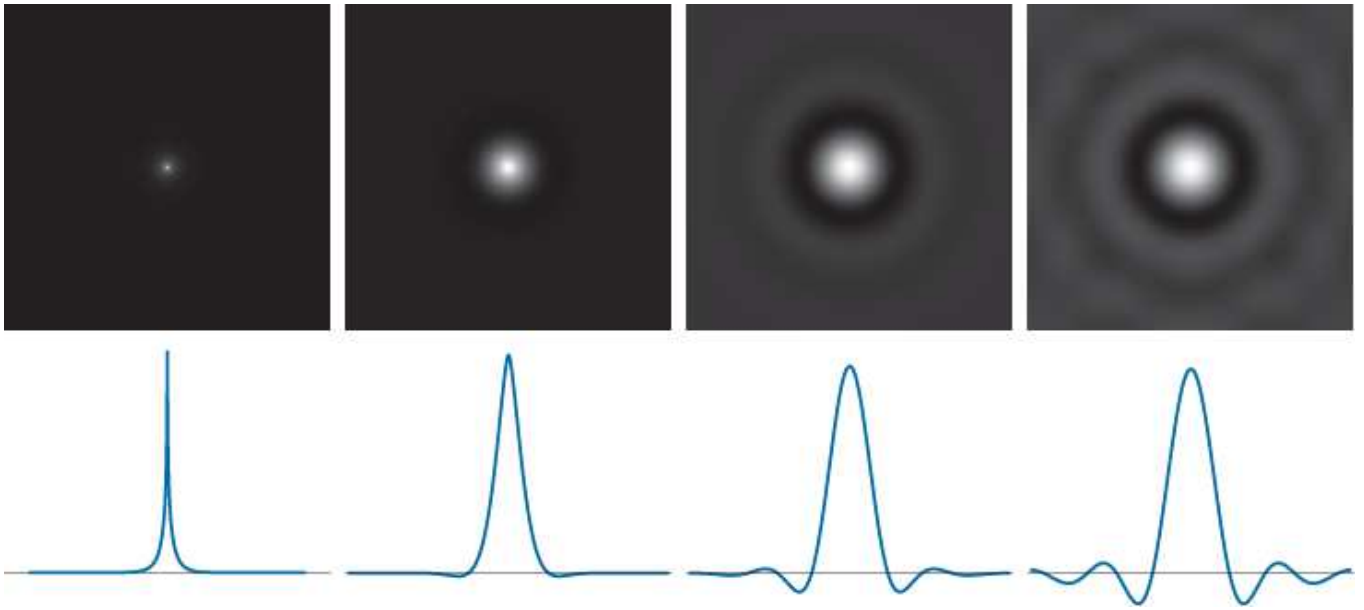
a b c
d e f

FIGURE 4.46 (a) Original image of size 688 × 688 pixels. (b)–(f) Results of filtering using BLPFs with cutoff frequencies at the radii shown in Fig. 4.40 and $n = 2.25$. Compare with Figs. 4.41 and 4.44. We used mirror padding to avoid the black borders characteristic of zero padding.

4.8 Image Smoothing Using Frequency Domain Filters

4.8.3 Butterworth Lowpass Filters

- Ringing effect in filters of higher orders



a b c d
e f g h

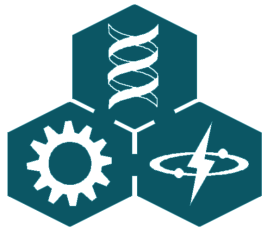
FIGURE 4.47 (a)–(d) Spatial representations (i.e., spatial kernels) corresponding to BLPF transfer functions of 1000×1000 pixels, cut-off frequency of 5, and order 1, 2, 5, and 20, respectively. (e)–(h) Corresponding inten profiles through the center of the filter functions.

4.8 Image Smoothing Using Frequency Domain Filters

4.8.3 Butterworth Lowpass Filters

TABLE 4.5
Lowpass filter transfer functions. D_0 is the cutoff frequency, and n is the order of the Butterworth filter.

Ideal	Gaussian	Butterworth
$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \le D_0 \\ 0 & \text{if } D(u,v) > D_0 \end{cases}$	$H(u,v) = e^{-D^2(u,v)/2D_0^2}$	$H(u,v) = \dfrac{1}{1 + [D(u,v)/D_0]^{2n}}$

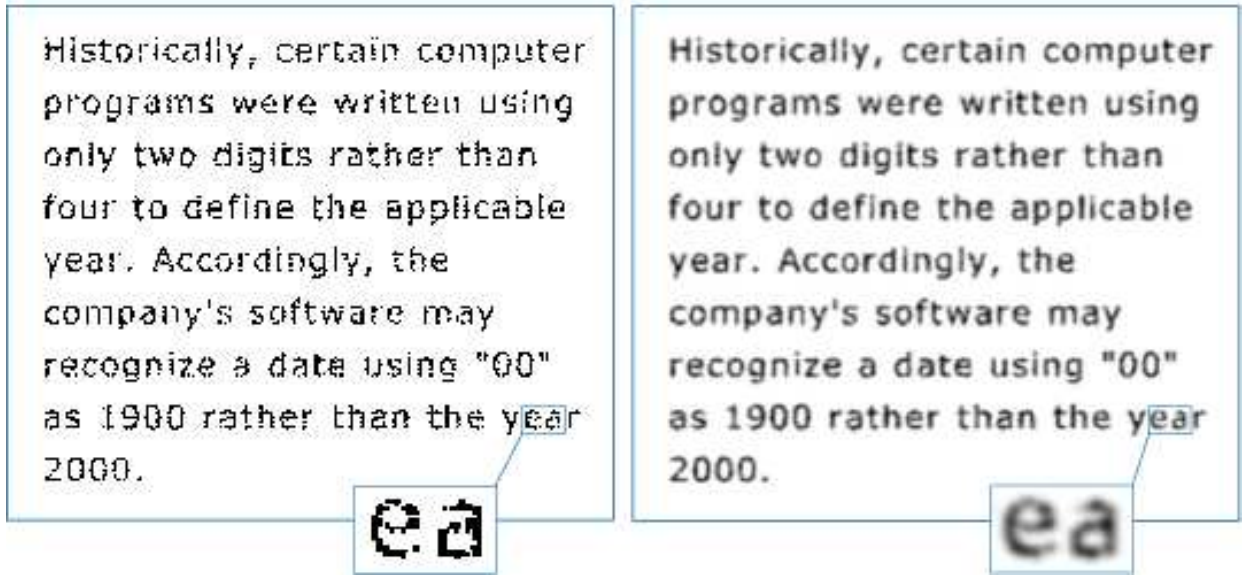


4.8 Image Smoothing Using Frequency Domain Filters

4.8.4 Additional Examples of Lowpass Filtering

a b

FIGURE 4.48
(a) Sample text of low resolution (note the broken characters in the magnified view). (b) Result of filtering with a GLPF, showing that gaps in the broken characters were joined.



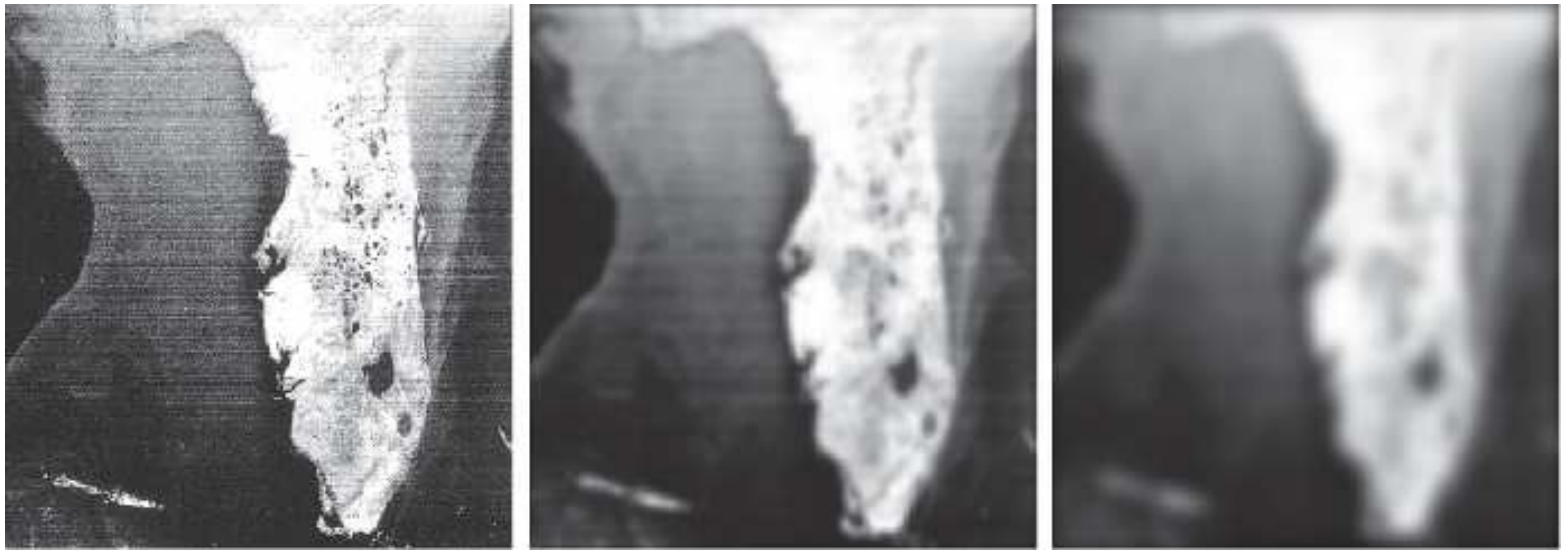
4.8 Image Smoothing Using Frequency Domain Filters

4.8.4 Additional Examples of Lowpass Filtering

FIGURE 4.49 (a) Original 785×732 image. (b) Result of filtering using a GLPF with $D_0 = 150$. (c) Result of filtering using a GLPF with $D_0 = 130$. Note the reduction in fine skin lines in the magnified sections in (b) and (c).

4.8 Image Smoothing Using Frequency Domain Filters

4.8.4 Additional Examples of Lowpass Filtering



a b c

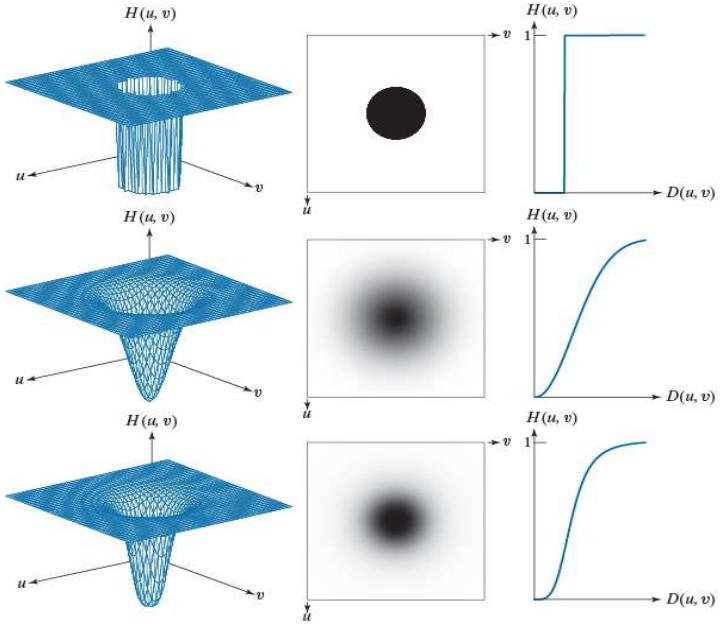
FIGURE 4.50 (a) 808×754 satellite image showing prominent horizontal scan lines. (b) Result of filtering using a GLPF with $D_0 = 50$. (c) Result of using a GLPF with $D_0 = 20$. (Original image courtesy of NOAA.)

4.9 Image Sharpening Using Frequency Domain Filters

4.9.1 Ideal, Gaussian, and Butterworth Highpass Filters from Lowpass Filters

$$H_{HP}(u,v) = 1 - H_{LP}(u,v) \qquad (4\text{-}118)$$

$$H(u,v) = \begin{cases} 0 & \text{if } D(u,v) \le D_0 \\ 1 & \text{if } D(u,v) > D_0 \end{cases} \qquad (4\text{-}119)$$

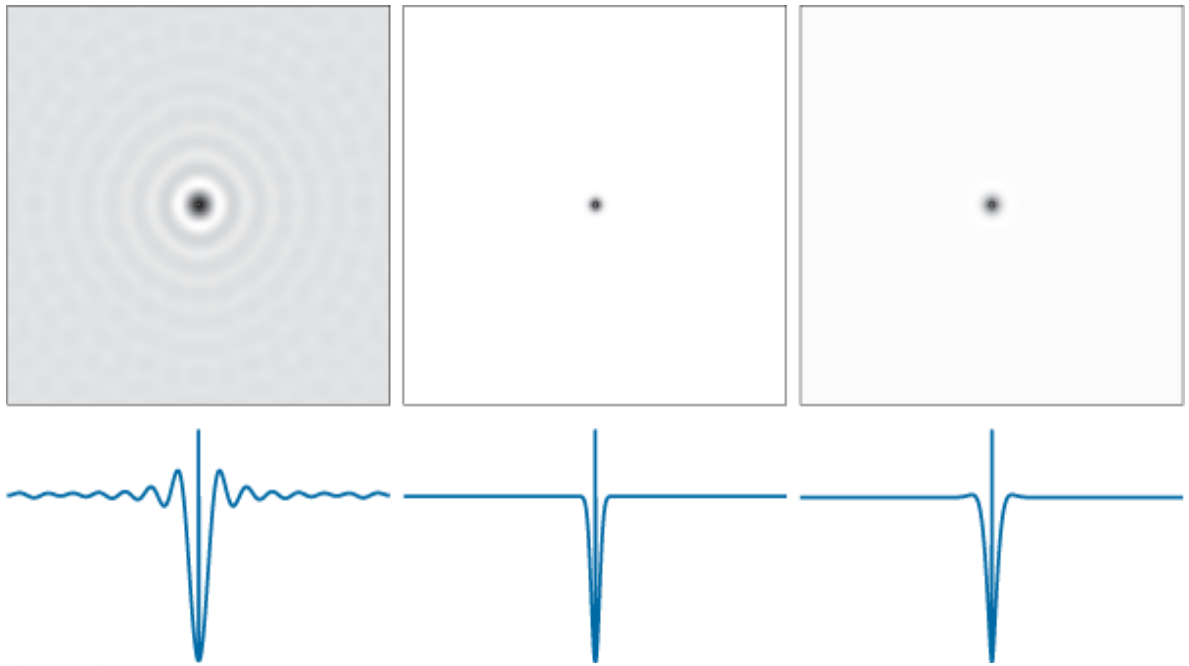


a b c
d e f
g h i

FIGURE 4.51 Top row: Perspective plot, image, and, radial cross section of an IHPF transfer function. Middle and bottom rows: The same sequence for GHPF and BHPF transfer functions. (The thin image borders were added for clarity. They are not part of the data.)

4.9 Image Sharpening Using Frequency Domain Filters

4.9.1 Ideal, Gaussian, and Butterworth Highpass Filters from Lowpass Filters



a b c
d e f

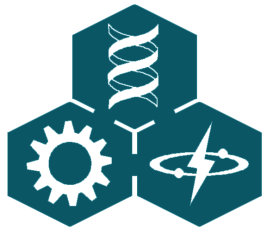
FIGURE 4.52 (a)–(c): Ideal, Gaussian, and Butterworth highpass spatial kernels obtained from IHPF, GHPF, and BHPF frequency-domain transfer functions. (The thin image borders are not part of the data.) (d)–(f): Horizontal intensity profiles through the centers of the kernels.

4.9 Image Sharpening Using Frequency Domain Filters

4.9.1 Ideal, Gaussian, and Butterworth Highpass Filters from Lowpass Filters

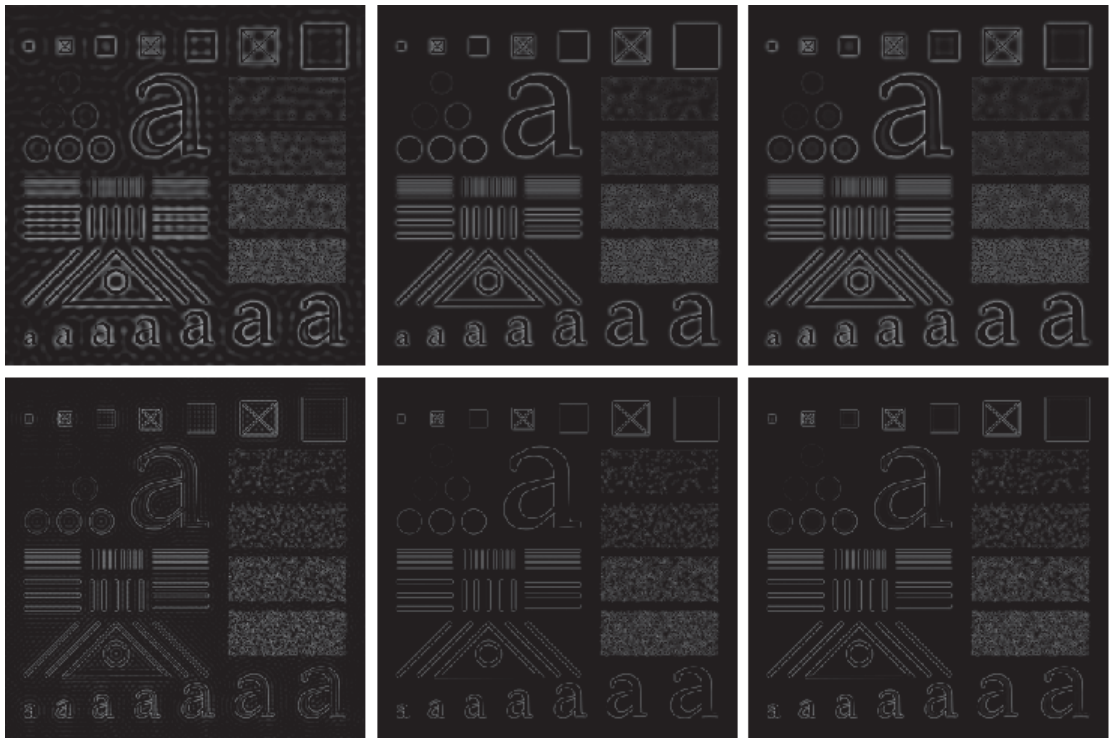
TABLE 4.6
Highpass filter transfer functions. D_0 is the cutoff frequency and n is the order of the Butterworth transfer function.

Ideal	Gaussian	Butterworth
$H(u,v) = \begin{cases} 0 & \text{if } D(u,v) \le D_0 \\ 1 & \text{if } D(u,v) > D_0 \end{cases}$	$H(u,v) = 1 - e^{-D^2(u,v)/2D_0^2}$	$H(u,v) = \dfrac{1}{1 + [D_0/D(u,v)]^{2n}}$



4.9 Image Sharpening Using Frequency Domain Filters

4.9.1 Ideal, Gaussian, and Butterworth Highpass Filters from Lowpass Filters

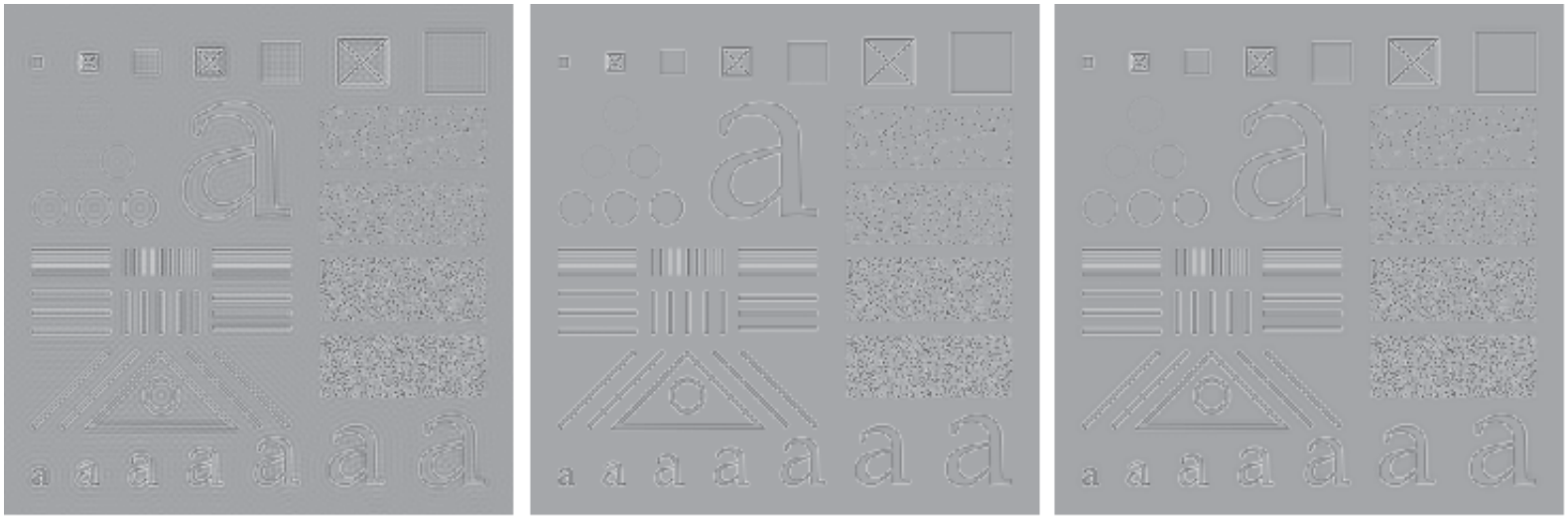


a b c
d e f

FIGURE 4.53 Top row: The image from Fig. 4.40(a) filtered with IHPF, GHPF, and BHPF transfer functions using $D_0 = 60$ in all cases ($n = 2$ for the BHPF). Second row: Same sequence, but using $D_0 = 160$.

4.9 Image Sharpening Using Frequency Domain Filters

4.9.1 Ideal, Gaussian, and Butterworth Highpass Filters from Lowpass Filters



a b c

FIGURE 4.54 The images from the second row of Fig. 4.53 scaled using Eqs. (2-31) and (2-32) to show both positive and negative values.

4.9 Image Sharpening Using Frequency Domain Filters

4.9.1 Ideal, Gaussian, and Butterworth Highpass Filters from Lowpass Filters

a b c

FIGURE 4.55 (a) Smudged thumbprint. (b) Result of highpass filtering (a). (c) Result of thresholding (b). (Original image courtesy of the U.S. National Institute of Standards and Technology.)

4.9 Image Sharpening Using Frequency Domain Filters

4.9.2 The Laplacian in the Frequency Domain

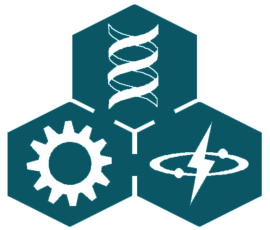
$$H(u,v) = -4\pi^2(u^2+v^2) \tag{4-123}$$

$$H(u,v) = -4\pi^2[(u-P/2)^2+(v-Q/2)^2)] = -4\pi^2 D^2(u,v) \tag{4-124}$$

$$\nabla^2 f(x,y) = \Im^{-1}\{H(u,v)F(u,v)\} \tag{4-125}$$

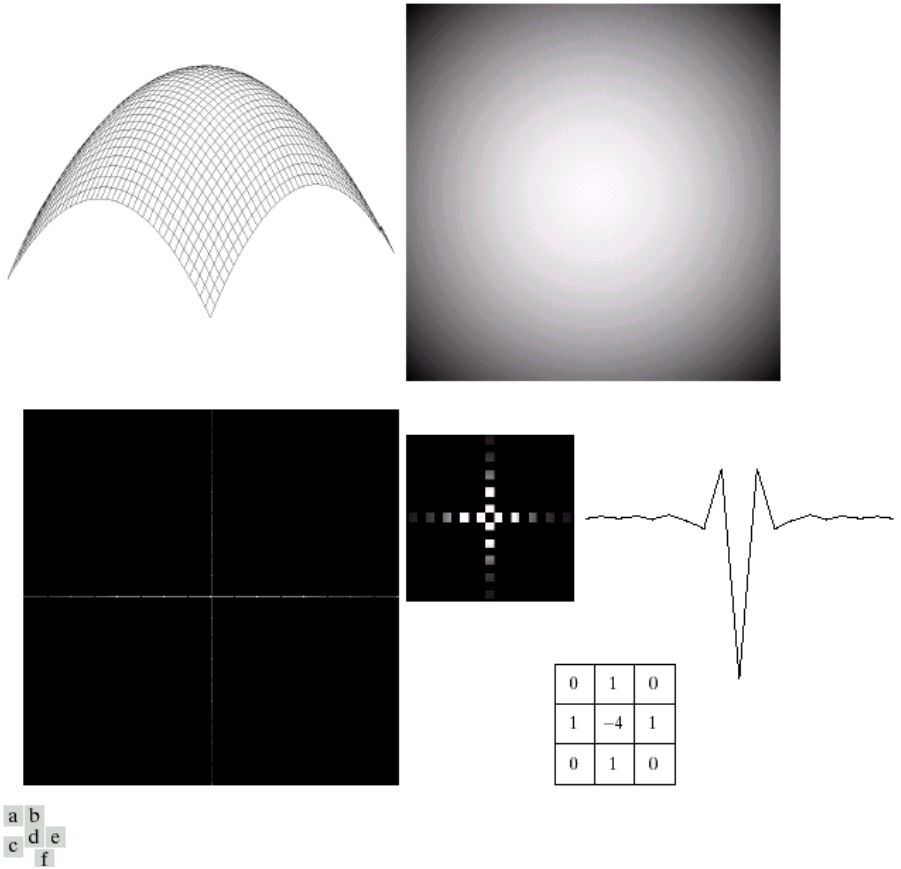
$$g(x,y) = f(x,y) + c\nabla^2 f(x,y) \tag{4-126}$$

$$\begin{aligned} g(x,y) &= \Im^{-1}\{F(u,v) - H(u,v)F(u,v)\} \\ &= \Im^{-1}\{[1-H(u,v)]F(u,v)\} \\ &= \Im^{-1}\{[1+4\pi^2 D^2(u,v)]F(u,v)\} \end{aligned} \tag{4-127}$$



4.9 Image Sharpening Using Frequency Domain Filters

4.9.2 The Laplacian in the Frequency Domain



a b
c d e
f

(a) 3-D plot of Laplacian in the frequency domain. (b) Image representation of (a). (c) Laplacian in the spatial domain obtained from the inverse DFT of (b). (d) Zoomed section of the origin of (c). (e) Gray-level profile through the center of (d). (f) Laplacian mask used in Section 3.7.

4.9 Image Sharpening Using Frequency Domain Filters

4.9.2 The Laplacian in the Frequency Domain

a b

FIGURE 4.56
(a) Original, blurry image. (b) Image enhanced using the Laplacian in the frequency domain. Compare with Fig. 3.52(d). (Original image courtesy of NASA.)

4.9 Image Sharpening Using Frequency Domain Filters

4.9.3 Unsharp Masking, High-Boost and High-Frequency Emphasis Filtering

- Unsharp Masking and High-Boost Filtering

$$g_{mask}(x,y) = f(x,y) - f_{LP}(x,y) \quad (4\text{-}128)$$

$$f_{LP}(x,y) = \Im^{-1}[H_{LP}(u,v)F(u,v)] \quad (4\text{-}129)$$

$$g(x,y) = f(x,y) + k * g_{mask}(x,y) \quad (4\text{-}130)$$

unsharp masking when $k = 1$

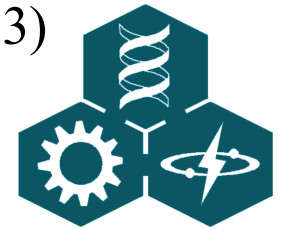
highboost filtering when $k > 1$

- High-Frequency Emphasis Filtering

$$g(x,y) = \Im^{-1}\{[1 + k*[1 - H_{LP}(u,v)]]F(u,v)\} \quad (4\text{-}131)$$

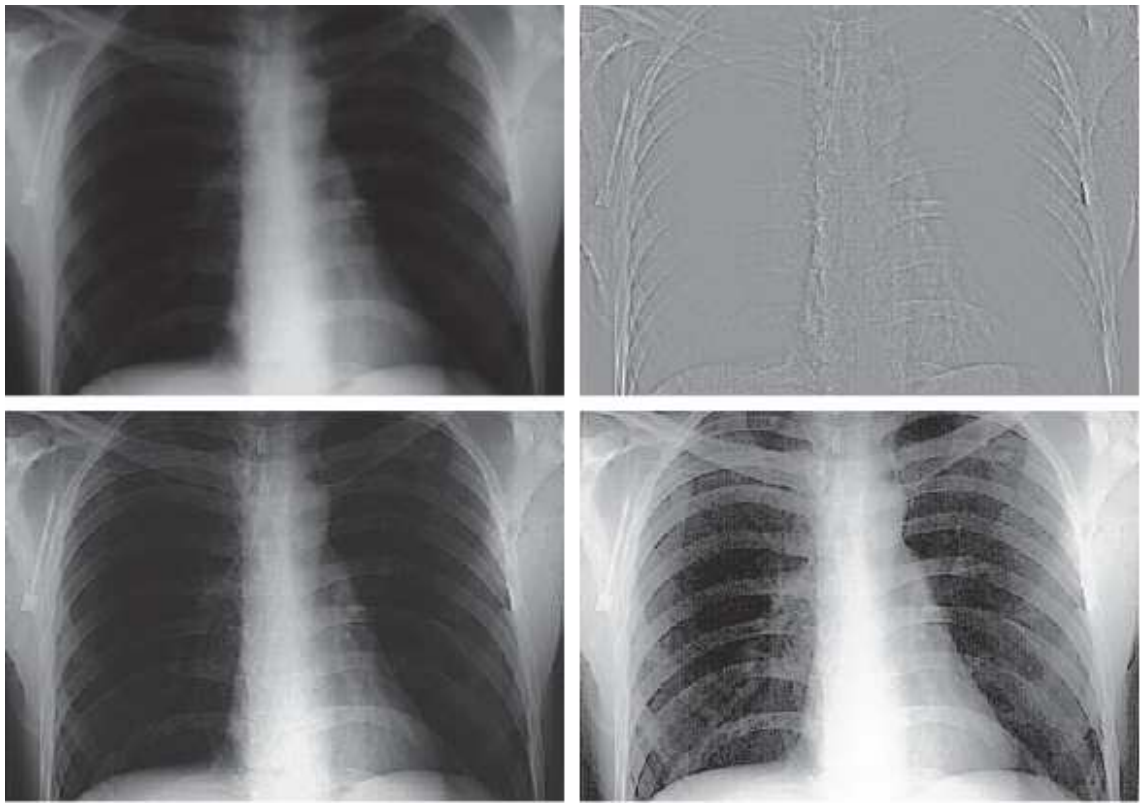
$$g(x,y) = \Im^{-1}\{[1 + k*H_{HP}(u,v)]]F(u,v)\} \quad (4\text{-}132)$$

$$g(x,y) = \Im^{-1}\{[k_1 + k_2*H_{HP}(u,v)]]F(u,v)\} \quad (4\text{-}133)$$



4.9 Image Sharpening Using Frequency Domain Filters

4.9.3 Unsharp Masking, High-Boost and High-Frequency Emphasis Filtering



a b
c d

FIGURE 4.57
(a) A chest X-ray. (b) Result of filtering with a GHPF function. (c) Result of high-frequency-emphasis filtering using the same GHPF. (d) Result of performing histogram equalization on (c). (Original image courtesy of Dr. Thomas R. Gest, Division of Anatomical Sciences, University of Michigan Medical School.)

4.9 Image Sharpening Using Frequency Domain Filters

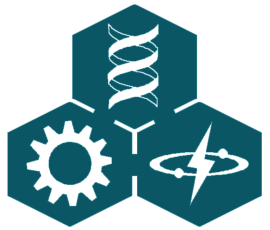
4.9.4 Homomorphic Filtering

- Simultaneously doing brightness range compression and contrast enhancement in frequency domain using illumination-reflectance model

$$f(x,y) = i(x,y)\, r(x,y) \qquad (4\text{-}134)$$

$i(x,y)$: slow spatial variation low frequencies
dynamic range

$r(x,y)$: abrupt spatial variation high frequency
contrast



4.9 Image Sharpening Using Frequency Domain Filters

4.9.4 Homomorphic Filtering

- Mathematical derivation

Because $\Im[f(x,y)] \neq \Im\{i(x,y)\}\Im\{r(x,y)\}$ (4-135)

Suppose $z(x,y) = \ln f(x,y) = \ln i(x,y) + \ln r(x,y)$ (4-136)

Then $\Im\{z(x,y)\} = \Im\{\ln f(x,y)\}$

$= \Im\{\ln i(x,y)\} + \Im\{\ln r(x,y)\}$ (4-137)

or $Z(u,v) = F_i(u,v) + F_r(u,v)$ (4-138)

We can filter $Z(u,v)$ using a filter $H(u, v)$ so that

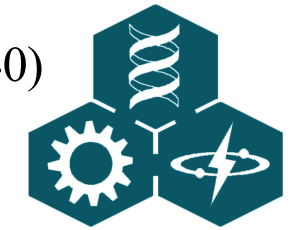
$$S(u,v) = H(u,v)\,Z(u,v)$$

$$= H(u,v)F_i(u,v) + H(u,v)\,F_r(u,v) \quad (4\text{-}139)$$

Therefore, in the spatial domain

$$s(x,y) = \Im^{-1}\{S(u,v)\}$$

$$= \Im^{-1}\{H(u,v)F_i(u,v)\} + \Im^{-1}\{H(u,v)\,F_r(u,v)\} \quad (4\text{-}140)$$



4.9 Image Sharpening Using Frequency Domain Filters

4.9.4 Homomorphic Filtering

- Mathematical derivation

Let $i'(x,y) = \Im^{-1}\{H(u,v)\,F_i(u,v)\}$ (4-141)

$r'(x,y) = \Im^{-1}\{H(u,v)\,F_r(u,v)\}$ (4-142)

Then $(x,y) = i'(x,y) + r'(x,y)$ (4-143)

The output image $g(x, y)$

$$g(x,y) = e^{\,s(x,y)}$$

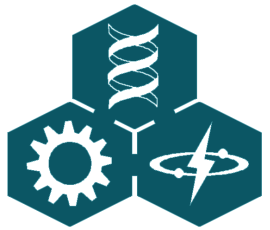
$$= e^{\,i'(x,y)} \cdot e^{r'(x,y)}$$

$$= i_0(x,y)\,r_0(x,y) \quad (4\text{-}144)$$

where

$$i_0(x,y) = e^{\,i'(x,y)} \quad (4\text{-}145)$$

$$r_0(x,y) = e^{\,r'(x,y)} \quad (4\text{-}146)$$



4.9 Image Sharpening Using Frequency Domain Filters

4.9.4 Homomorphic Filtering

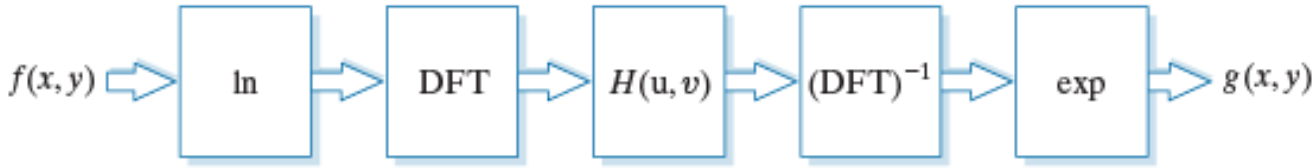


FIGURE 4.58 Summary of steps in homomorphic filtering.

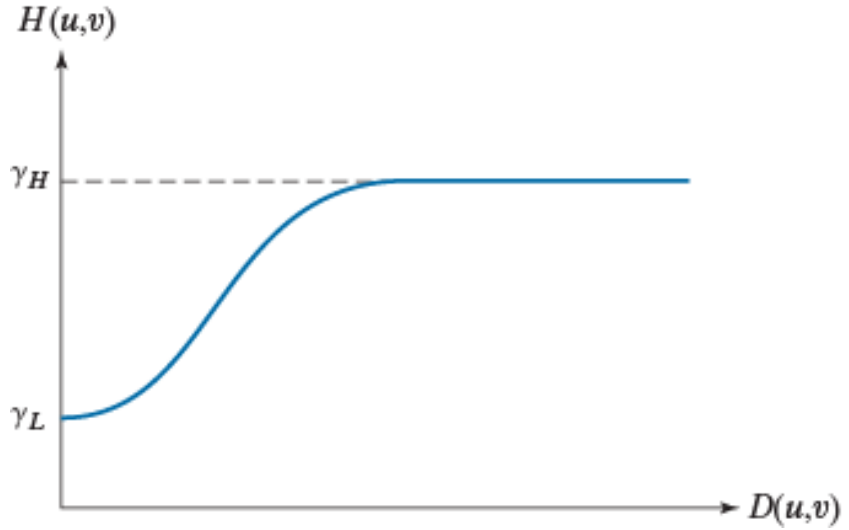
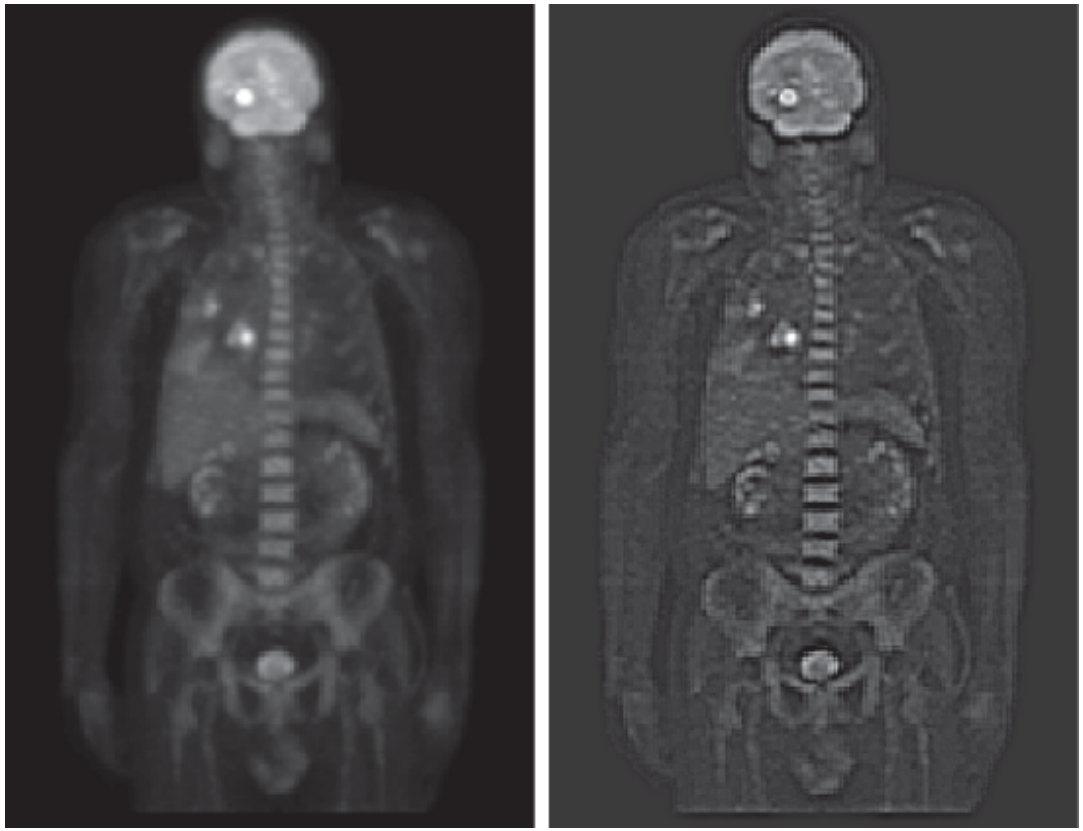


FIGURE 4.59 Radial cross section of a homomorphic filter transfer function.

4.9 Image Sharpening Using Frequency Domain Filters

4.9.4 Homomorphic Filtering



a b

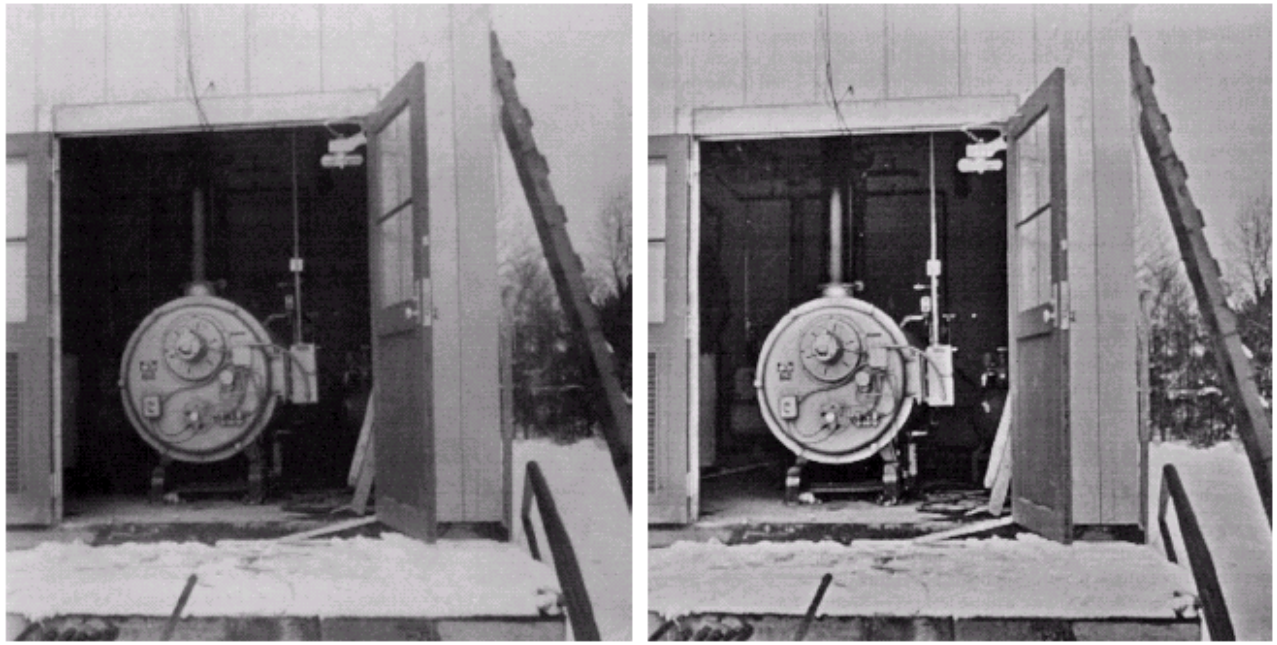
FIGURE 4.60
(a) Full body PET scan. (b) Image enhanced using homomorphic filtering. (Original image courtesy of Dr. Michael E. Casey, CTI Pet Systems.)

4.9 Image Sharpening Using Frequency Domain Filters

4.9.4 Homomorphic Filtering

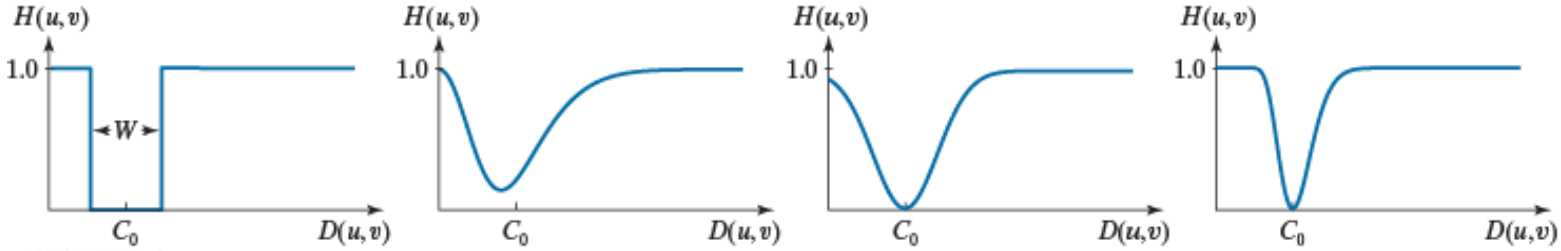
a b

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



4.10 Selective Filtering

4.10.1 Bandreject and Bandpass Filters



a b c d

FIGURE 4.61 Radial cross sections. (a) Ideal bandreject filter transfer function. (b) Bandreject transfer function formed by the sum of Gaussian lowpass and highpass filter functions. (The minimum is not 0 and does not align with C_0.) (c) Radial plot of Eq. (4-149). (The minimum is 0 and is properly aligned with C_0, but the value at the origin is not 1.) (d) Radial plot of Eq. (4-150); this Gaussian-shape plot meets all the requirements of a bandreject filter transfer function.

$$H_{BP}(u,v) = 1 - H_{BR}(u,v) \qquad (4\text{-}148)$$

TABLE 4.7
Bandreject filter transfer functions. C_0 is the center of the band, W is the width of the band, and $D(u,v)$ is the distance from the center of the transfer function to a point (u,v) in the frequency rectangle.

Ideal (IBRF)	Gaussian (GBRF)	Butterworth (BBRF)
$H(u,v) = \begin{cases} 0 & \text{if } C_0 - \frac{W}{2} \le D(u,v) \le C_0 + \frac{W}{2} \\ 1 & \text{otherwise} \end{cases}$	$H(u,v) = 1 - e^{-\left[\frac{D^2(u,v) - C_0^2}{D(u,v)W}\right]^2}$	$H(u,v) = \frac{1}{1 + \left[\frac{D(u,v)W}{D^2(u,v) - C_0^2}\right]^{2n}}$

4.10 Selective Filtering

4.10.2 Notch Filters

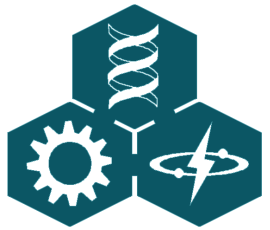
$$H_{NR}(u,v) = \prod_{k=1}^{Q} H_k(u,v) H_{-k}(u,v) \qquad (4\text{-}151)$$

$$D_k(u,v) = [(u - M/2 - u_k)^2 + (v - N/2 - v_k)^2]^{1/2} \qquad (4\text{-}152)$$

$$D_{-k}(u,v) = [(u - M/2 + u_k)^2 + (v - N/2 + v_k)^2]^{1/2} \qquad (4\text{-}153)$$

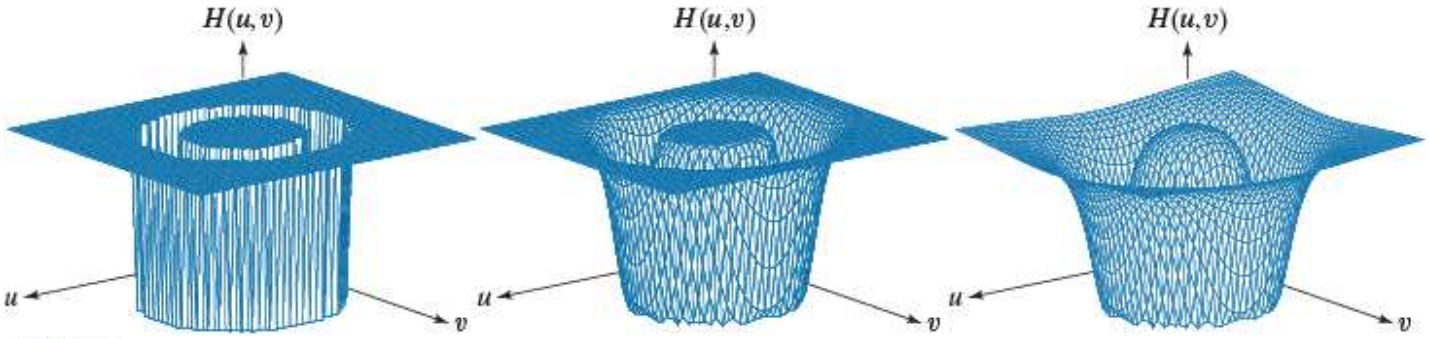
$$H_{NR}(u,v) = \prod_{k=1}^{3} \left[\frac{1}{1 + [D_{0k}/D_k(u,v)]^{2n}} \right] \left[\frac{1}{1 + [D_{0k}/D_{-k}(u,v)]^{2n}} \right] \qquad (4\text{-}154)$$

$$H_{NP}(u,v) = 1 - H_{NR}(u,v) \qquad (4\text{-}155)$$



4.10 Selective Filtering

4.10.2 Notch Filters

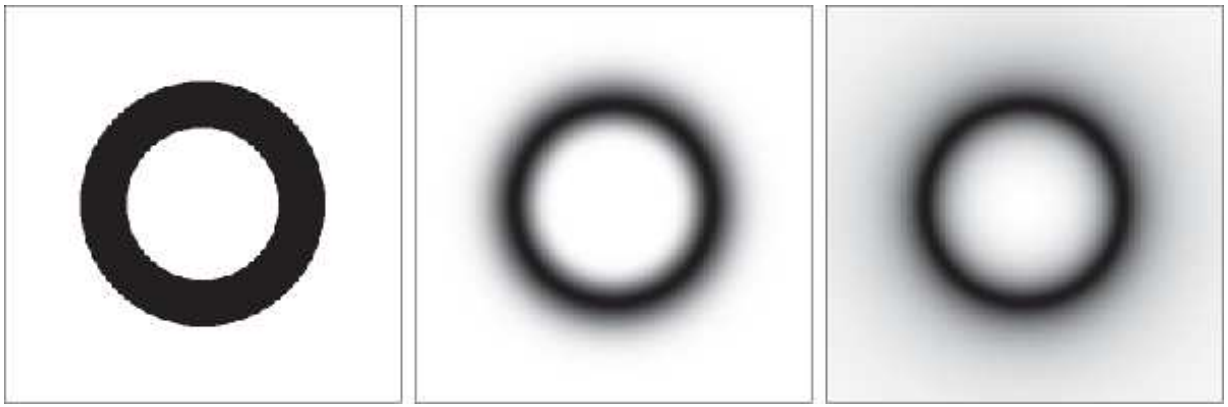


a b c

FIGURE 4.62 Perspective plots of (a) ideal, (b) modified Gaussian, and (c) modified Butterworth (of order 1) bandreject filter transfer functions from Table 4.7. All transfer functions are of size 512×512 elements, with $C_0 = 128$ and $W = 60$.

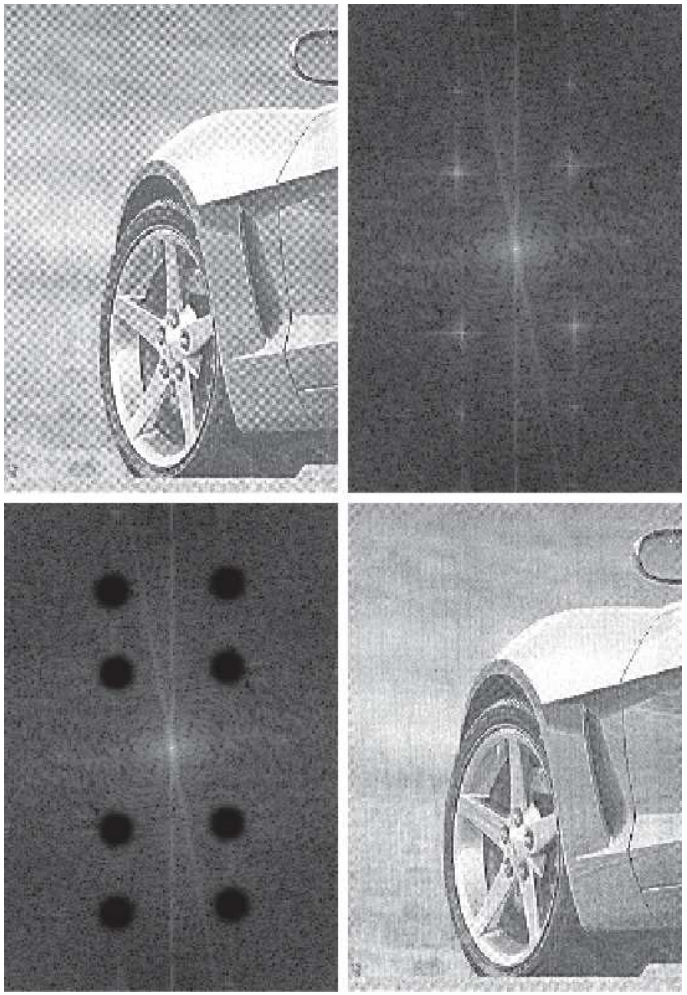
a b c

FIGURE 4.63 (a) The ideal, (b) Gaussian, and (c) Butterworth bandpass transfer functions from Fig. 4.62, shown as images. (The thin border lines are not part of the image data.)



4.10 Selective Filtering

4.10.2 Notch Filters

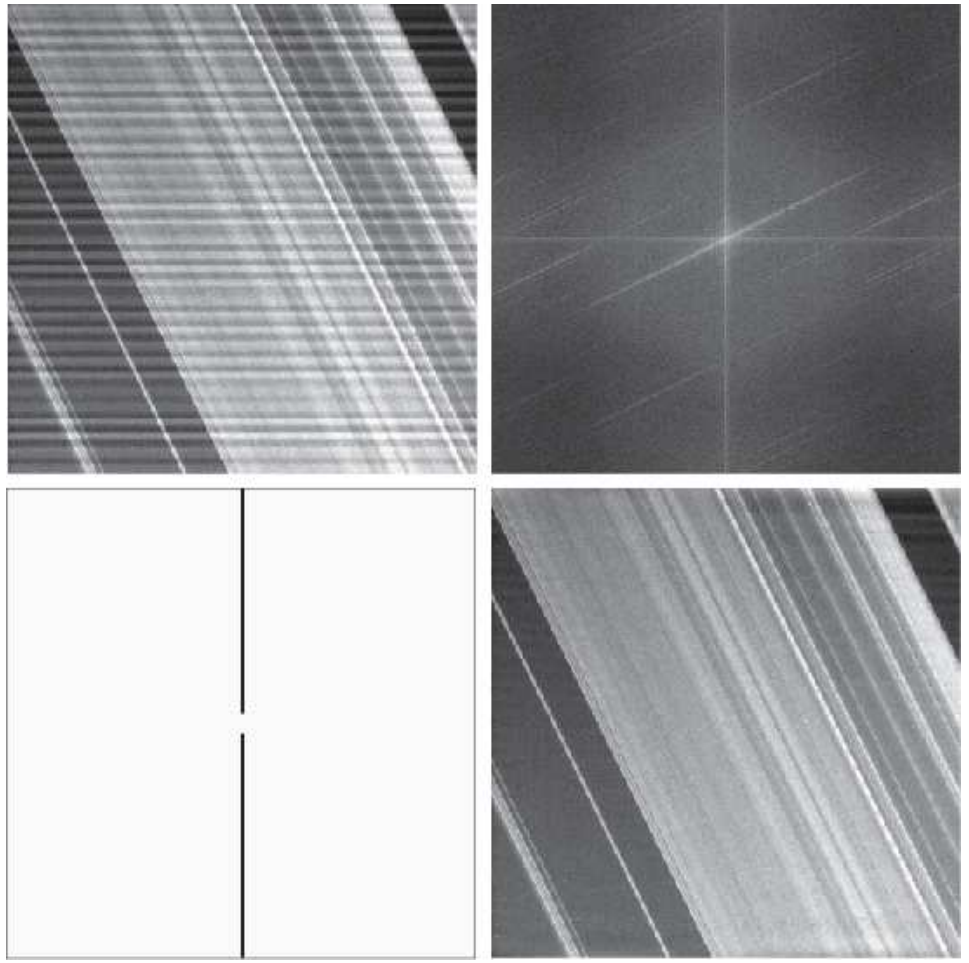


a b
c d

FIGURE 4.64
(a) Sampled newspaper image showing a moiré pattern. (b) Spectrum. (c) Fourier transform multiplied by a Butterworth notch reject filter transfer function. (d) Filtered image.

4.10 Selective Filtering

4.10.2 Notch Filters



a b
c d

FIGURE 4.65
(a) Image of Saturn rings showing nearly periodic interference. (b) Spectrum. (The bursts of energy in the vertical axis near the origin correspond to the interference pattern). (c) A vertical notch reject filter transfer function. (d) Result of filtering. (The thin black border in (c) is not part of the data.) (Original image courtesy of Dr. Robert A. West, NASA/ JPL.)

4.10 Selective Filtering

4.10.2 Notch Filters

a b

FIGURE 4.66
(a) Notch pass filter function used to isolate the vertical axis of the DFT of Fig. 4.65(a).
(b) Spatial pattern obtained by computing the IDFT of (a).

4.11 The Fast Fourier Transform

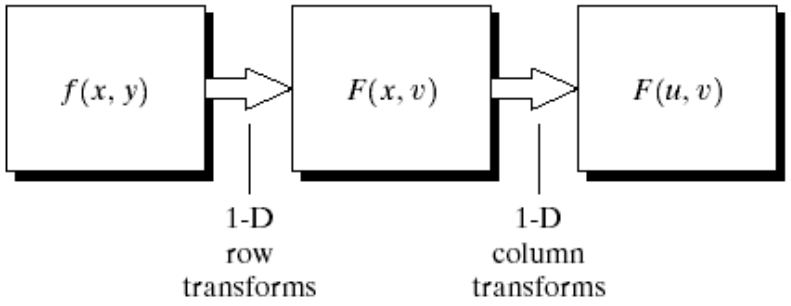
4.11.1 Separability of the 2-D DFT

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

$$= \sum_{x=0}^{M-1} F(x,v) e^{-j2\pi ux/M} \qquad (4\text{-}156)$$

where $$F(x,v) = \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi vy/N} \qquad (4\text{-}157)$$

2-D DFT and IDFT can be obtained by computing the 1-D DFT and IDFT of each row and column



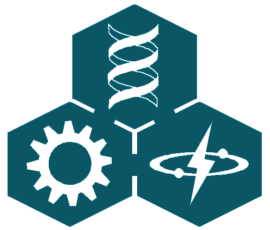
Computation of the 2-D Fourier transform as a series of 1-D transforms.

4.11 The Fast Fourier Transform

4.11.2 Computing the IDFT Using a DFT Algorithm

IDFT can be computed using a DFT algorithm. If we substitute $F^*(u, v)$ into the algorithm, the result will be $MNf^*(x, y)$. Taking the complex conjugate and dividing this result by MN yields $f(x, y)$, which is the inverse of $F(u, v)$.

$$MNf^*(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F^*(u,v)e^{-j2\pi(ux/M+vy/N)} \qquad (4\text{-}158)$$



4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

- Kernel Expression

$$F(u) = \frac{1}{M} \sum_{x=0}^{M-1} f(x) \cdot W_M^{ux} \qquad (4\text{-}159)$$

where

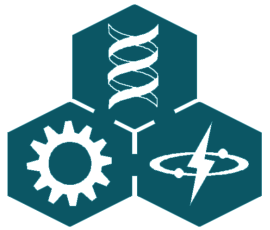
$$W_M = e^{-j2\pi/M} \qquad (4\text{-}160)$$

- Sucessive Doubling Method

Assume $M = 2^n$ (4-161)

Let $M = 2K$ (4-162)

Then the relationship of $F_{odd}(u)$ and $F_{even}(u)$ can be derived from Eqs. (4-159)



4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

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

$$=\sum_{x=0}^{K-1} f(2x)\cdot W_{2K}^{u(2x)}+\sum_{x=0}^{K-1} f(2x+1)\cdot W_{2K}^{u(2x+1)} \qquad (4\text{-}163)$$

$$F(u)=\sum_{x=0}^{K-1} f(2x)\cdot W_{K}^{ux}+\sum_{x=0}^{K-1} f(2x+1)\cdot W_{K}^{ux}W_{2K}^{u} \qquad (4\text{-}164)$$

Define the even and odd series:

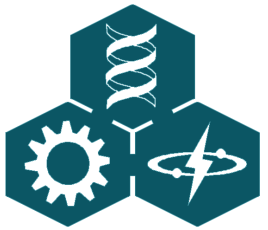
$$F_{even}(u)=\sum_{x=0}^{K-1} f(2x)\cdot W_{K}^{ux} \qquad (4\text{-}165)$$

$$F_{odd}(u)=\sum_{x=0}^{K-1} f(2x+1)\cdot W_{K}^{ux} \qquad (4\text{-}166)$$

then

$$F(u)=F_{even}(u)+F_{odd}(u)W_{2K}^{u} \qquad (4\text{-}167)$$

$$F(u+K)=F_{even}(u)-F_{odd}(u)W_{2K}^{u} \qquad (4\text{-}168)$$



4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

- Number of Operation

1. Recursive Expression

$$m(p) = 2m(p-1) + 2^{p-1} \qquad n \geq 1 \qquad (4\text{-}169)$$

$$a(p) = 2a(p-1) + 2^{p} \qquad n \geq 1 \qquad (4\text{-}170)$$

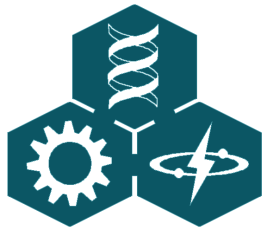
2. Logarithm Expression

$$m(p) = \frac{1}{2} M \log_2 M \qquad (4\text{-}171)$$

$$a(p) = M log_2 M \qquad (4\text{-}172)$$

$$C(M) = \frac{M^2}{M \log_2 M} = \frac{M}{\log_2 M} \qquad (4\text{-}173)$$

$$C(p) = \frac{2^p}{p} \qquad (4\text{-}174)$$



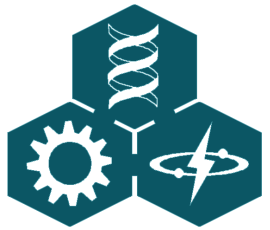
4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

- Computation Efficiency

A Comparison of N^2 versus $N \log_2 N$ for Various Values of N

N	N^2 (*Direct* FT)	$N \log_2 N$ (FFT)	*Computational Advantage* ($N/\log_2 N$)
2	4	2	2.00
4	16	8	2.00
8	64	24	2.67
16	256	64	4.00
32	1,024	160	6.40
64	4,096	384	10.67
128	16,384	896	18.29
256	65,536	2,048	32.00
512	262,144	4,608	56.89
1024	1,048,576	10,240	102.40
2048	4,194,304	22,528	186.18
4096	16,777,216	49,152	341.33
8192	67,108,864	106,496	630.15



4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

- Number of Operation

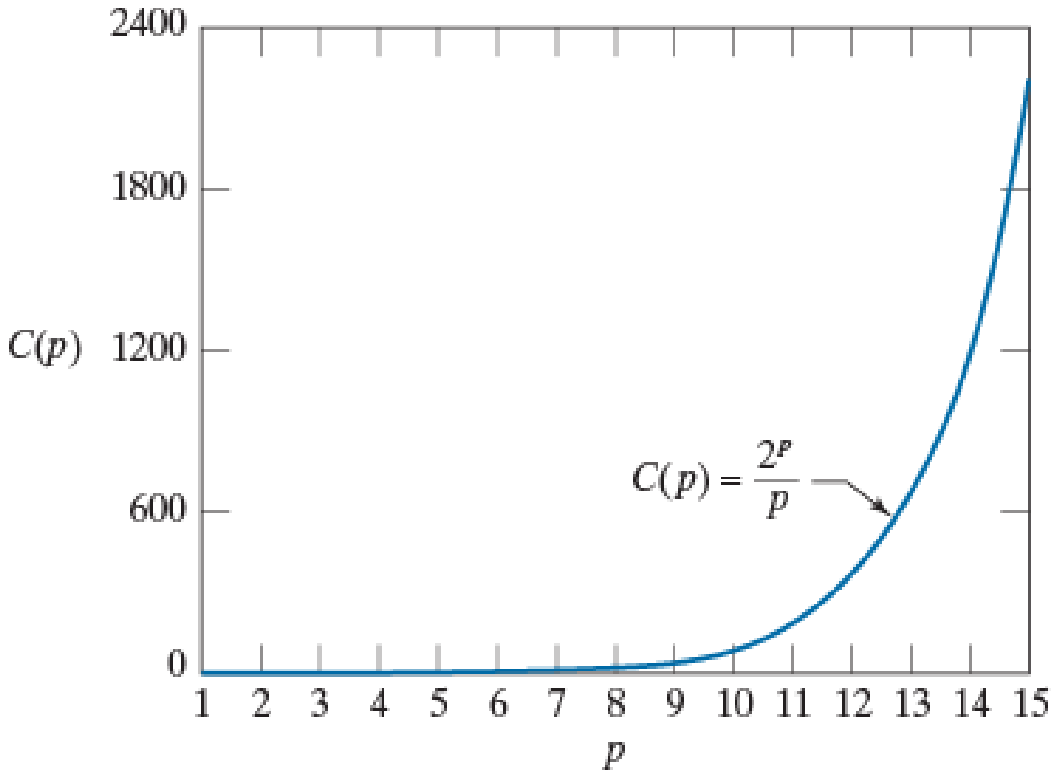


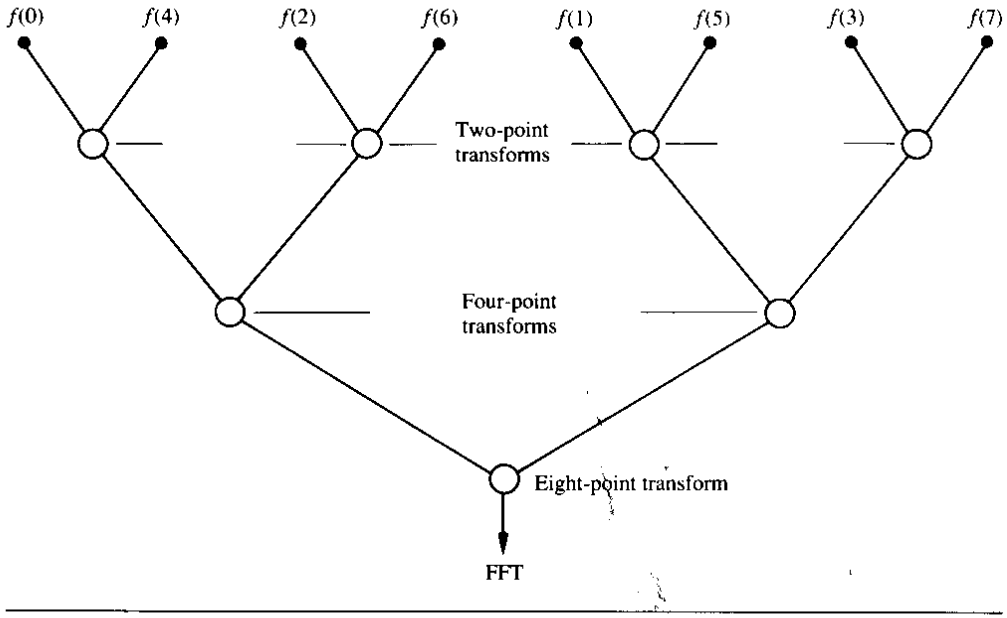
FIGURE 4.67 Computational advantage of the FFT over a direct implementation of the 1-D DFT. The number of samples is $M = 2^p$. The computational advantage increases rapidly as a function of p.

4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

- FFT Programs

Reversal and Reordering of Array for Input into FFT Algorithm



Ordered input array and its use in the successive-doubling method.

4.11 The Fast Fourier Transform

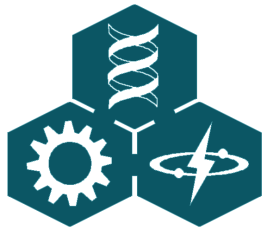
4.11.3 The Fast Fourier Transform (FFT)

- FFT Programs

Reversal and Reordering of Array for Input into FFT Algorithm

Example of Bit Reversal and Reordering of Array for Input into FFT Algorithm

Original Argument			*Original Array*	*Bit-Reversed Argument*			*Reordered Array*
0	0	0	$f(0)$	0	0	0	$f(0)$
0	0	1	$f(1)$	1	0	0	$f(4)$
0	1	0	$f(2)$	0	1	0	$f(2)$
0	1	1	$f(3)$	1	1	0	$f(6)$
1	0	0	$f(4)$	0	0	1	$f(1)$
1	0	1	$f(5)$	1	0	1	$f(5)$
1	1	0	$f(6)$	0	1	1	$f(3)$
1	1	1	$f(7)$	1	1	1	$f(7)$



4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

- FFT Programs

```
      SUBROUTINE FFT(F,LN)
      COMPLEX F(1024),U,W,T,CMPLX
      PI=3.141593
      N=2**LN
      NV2=N/2
      NM1=N-1
      J=1
      DO 3 I=1,NM1
         IF(I.GE.J) GO TO 1
         T=F(J)
         F(J)=F(I)
         F(I)=T
1     K=NV2
2     IF(K.GE.J) GO TO 3
         J=J-K
         K=K/2
         GO TO 2
3     J=J+K
      DO 5 L=1,LN
         LE=2**L
         LE1=LE/2
         U=(1.0,0.0)
         W=CMPLX(COS(PI/LE1),-SIN(PI/LE1))
         DO 5 J=1,LE1
            DO 4 I=J,N,LE
               IP=I+LE1
               T=F(IP)*U
               F(IP)=F(I)-T
4           F(I)=F(I)+T
5     U=U*W
      DO 6 I=1,N
6     F(I)=F(I)/FLOAT(N)
      RETURN
      END
```

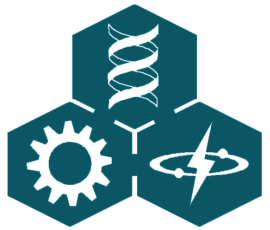
- Reversal and Reordering
- SDM complex operation
- Divide by constant N

4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

- FFT Programs - C language (P1)

```
/* dft.h - function prototypes and structures for dft and fft
functions */
/* COMPLEX STRUCTURE */
typedef struct {
    float real, imag;
} COMPLEX;

/* function prototypes for dft and inverse dft functions */
    extern void fft(COMPLEX *,int);
    extern void ifft(COMPLEX *,int);
    extern void dft(COMPLEX *,COMPLEX *,int);
    extern void idft(COMPLEX *,COMPLEX *,int);
    extern void rfft(float *,COMPLEX *,int);
    extern void ham(COMPLEX *,int);
    extern void han(COMPLEX *,int);
    extern void triang(COMPLEX *,int);
    extern void black(COMPLEX *,int);
    extern void harris(COMPLEX *,int);
    extern int log2(unsigned int);
```

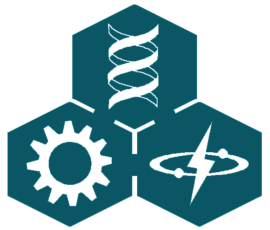


4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

- FFT Programs - C language (P2)

```
/*****************************************************************************/
/*                                                                           */
/*  fft - In-place radix 2 decimation in time FFT                            */
/*                                                                           */
/*  Requires pointer to complex array, x and power of 2 size of FFT, m       */
/*  (size of FFT = 2**m).  Places FFT output on top of input COMPLEX array   */
/*                                                                           */
/*****************************************************************************/

void fft(COMPLEX *x, int m)
{
    static COMPLEX *w;          /* used to store the w complex array */
    static int mstore = 0;      /* stores m for future reference */
    static int n = 1;           /* length of fft stored for future */
    COMPLEX u,temp,tm;
    COMPLEX *xi,*xip,*xj,*wptr;

    int i,j,k,l,le,windex;
    double arg,w_real,w_imag,wrecur_real,wrecur_imag,wtemp_real;
```

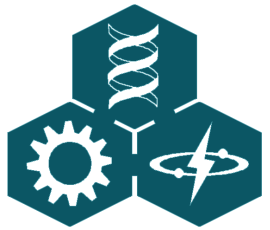



4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

- FFT Programs - C language (P3)

```
 if(m != mstore) {

/* free previously allocated storage and set new m */
        if(mstore != 0) free(w);
        mstore = m;
        if(m == 0) return;       /* if m=0 then done */

/* n = 2**m = fft length */
        n = 1 << m;
        le = n/2;

/* allocate the storage for w */
        w = (COMPLEX *) calloc(le-1,sizeof(COMPLEX));
        if(!w) {
            printf("\nUnable to allocate complex W array\n");
            exit(1);
        }
```

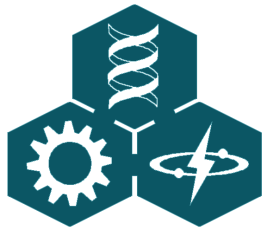



4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

- FFT Programs - C language (P4)

```
/* calculate the w values recursively */

    arg = 4.0*atan(1.0)/le;        /* PI/le calculation */
    wrecur_real = w_real = cos(arg);
    wrecur_imag = w_imag = -sin(arg);
    xj = w;
    for (j = 1 ; j < le ; j++) {
        xj->real = (float)wrecur_real;
        xj->imag = (float)wrecur_imag;
        xj++;
        wtemp_real = wrecur_real*w_real - wrecur_imag*w_imag;
        wrecur_imag = wrecur_real*w_imag + wrecur_imag*w_real;
        wrecur_real = wtemp_real;
    }
}
```

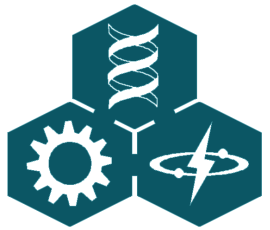



4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

- FFT Programs - C language (P5)

```
/* start fft */

   le = n;
   windex = 1;
   for (l = 0 ; l < m ; l++) {
      le = le/2;

/* first iteration with no multiplies */

      for(i = 0 ; i < n ; i = i + 2*le) {
         xi = x + i;
         xip = xi + le;
         temp.real = xi->real + xip->real;
         temp.imag = xi->imag + xip->imag;
         xip->real = xi->real - xip->real;
         xip->imag = xi->imag - xip->imag;
         *xi = temp;
      }
```

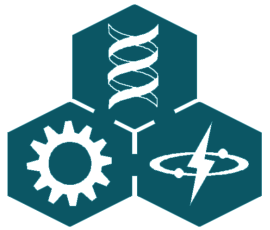



4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

- FFT Programs - C language (P6)

```
/* remaining iterations use stored w */
      wptr = w + windex - 1;
      for (j = 1 ; j < le ; j++) {
         u = *wptr;
         for (i = j ; i < n ; i = i + 2*le) {
            xi = x + i;
            xip = xi + le;
            temp.real = xi->real + xip->real;
            temp.imag = xi->imag + xip->imag;
            tm.real = xi->real - xip->real;
            tm.imag = xi->imag - xip->imag;
            xip->real = tm.real*u.real - tm.imag*u.imag;
            xip->imag = tm.real*u.imag + tm.imag*u.real;
            *xi = temp;
         }
         wptr = wptr + windex;
      }
      windex = 2*windex;
   }
```

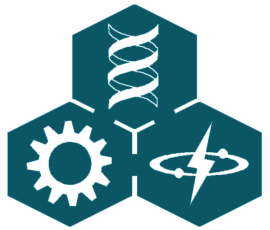



4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

- FFT Programs - C language (P7)

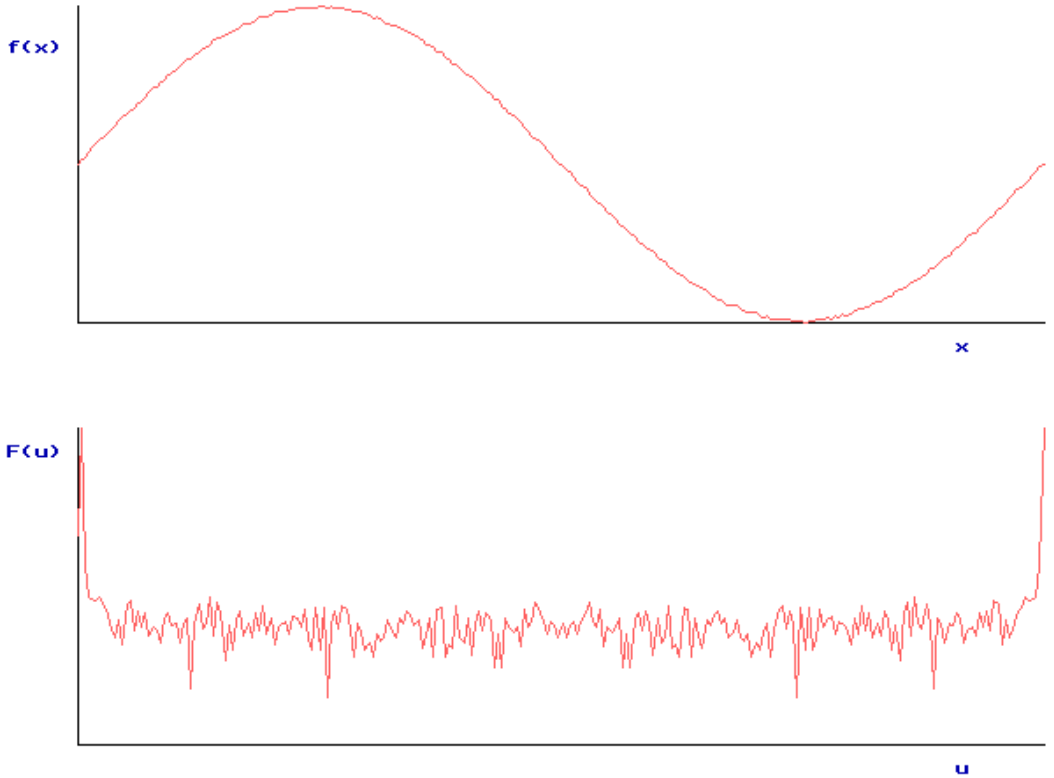
```
/* rearrange data by bit reversing */

   j = 0;
   for (i = 1 ; i < (n-1) ; i++) {
      k = n/2;
      while(k <= j) {
         j = j - k;
         k = k/2;
      }
      j = j + k;
      if (i < j) {
         xi = x + i;
         xj = x + j;
         temp = *xj;
         *xj = *xi;
         *xi = temp;
      }
   }
}
```

4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

- FFT Programs - example

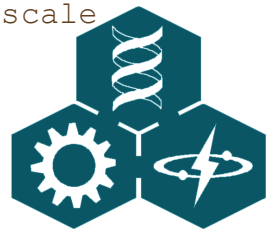


4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

- FFT Programs – OpenCV code

```
int main(int argc, char ** argv)
{
    const char* filename = argc >=2 ? argv[1] : "../data/lena.jpg";
    Mat I = imread(filename, IMREAD_GRAYSCALE);
    if( I.empty()){
        cout << "Error opening image" << endl;
        return -1;
    }
    Mat padded;                            //expand input image to optimal size
    int m = getOptimalDFTSize( I.rows );
    int n = getOptimalDFTSize( I.cols ); // on the border add zero values
    copyMakeBorder(I, padded, 0, m - I.rows, 0, n - I.cols, BORDER_CONSTANT,
         Scalar::all(0));
    Mat planes[] = {Mat_<float>(padded), Mat::zeros(padded.size(), CV_32F)};
    Mat complexI;
    merge(planes, 2, complexI); // Add to the expanded another plane with zeros
    dft(complexI, complexI);    // this way the result may fit in the source matrix
    // compute the magnitude and switch to logarithmic scale
    // => log(1 + sqrt(Re(DFT(I))^2 + Im(DFT(I))^2))
    split(complexI, planes);    // planes[0] = Re(DFT(I), planes[1] = Im(DFT(I))
    magnitude(planes[0], planes[1], planes[0]);// planes[0] = magnitude
    Mat magI = planes[0];
    magI += Scalar::all(1);                    // switch to logarithmic scale
    log(magI, magI);
```



4.11 The Fast Fourier Transform

4.11.3 The Fast Fourier Transform (FFT)

```
    // crop the spectrum, if it has an odd number of rows or columns
    magI = magI(Rect(0, 0, magI.cols & -2, magI.rows & -2));
    // rearrange the quadrants of Fourier image so the origin is at image center
    int cx = magI.cols/2;
    int cy = magI.rows/2;
    Mat q0(magI, Rect(0, 0, cx, cy));   // Top-Left - Create a ROI per quadrant
    Mat q1(magI, Rect(cx, 0, cx, cy));  // Top-Right
    Mat q2(magI, Rect(0, cy, cx, cy));  // Bottom-Left
    Mat q3(magI, Rect(cx, cy, cx, cy)); // Bottom-Right
    Mat tmp;                           // swap quadrants (Top-Left with Bottom-Right)
    q0.copyTo(tmp);
    q3.copyTo(q0);
    tmp.copyTo(q3);
    q1.copyTo(tmp);                    // swap quadrant (Top-Right with Bottom-Left)
    q2.copyTo(q1);
    tmp.copyTo(q2);
    normalize(magI, magI, 0, 1, NORM_MINMAX);
    // Transform the matrix with float values into a viewable image form
    // (float between values 0 and 1).
    imshow("Input Image"       , I   );    // Show the result
    imshow("spectrum magnitude", magI);
    waitKey();
    return 0;
}
```