Frequency domain

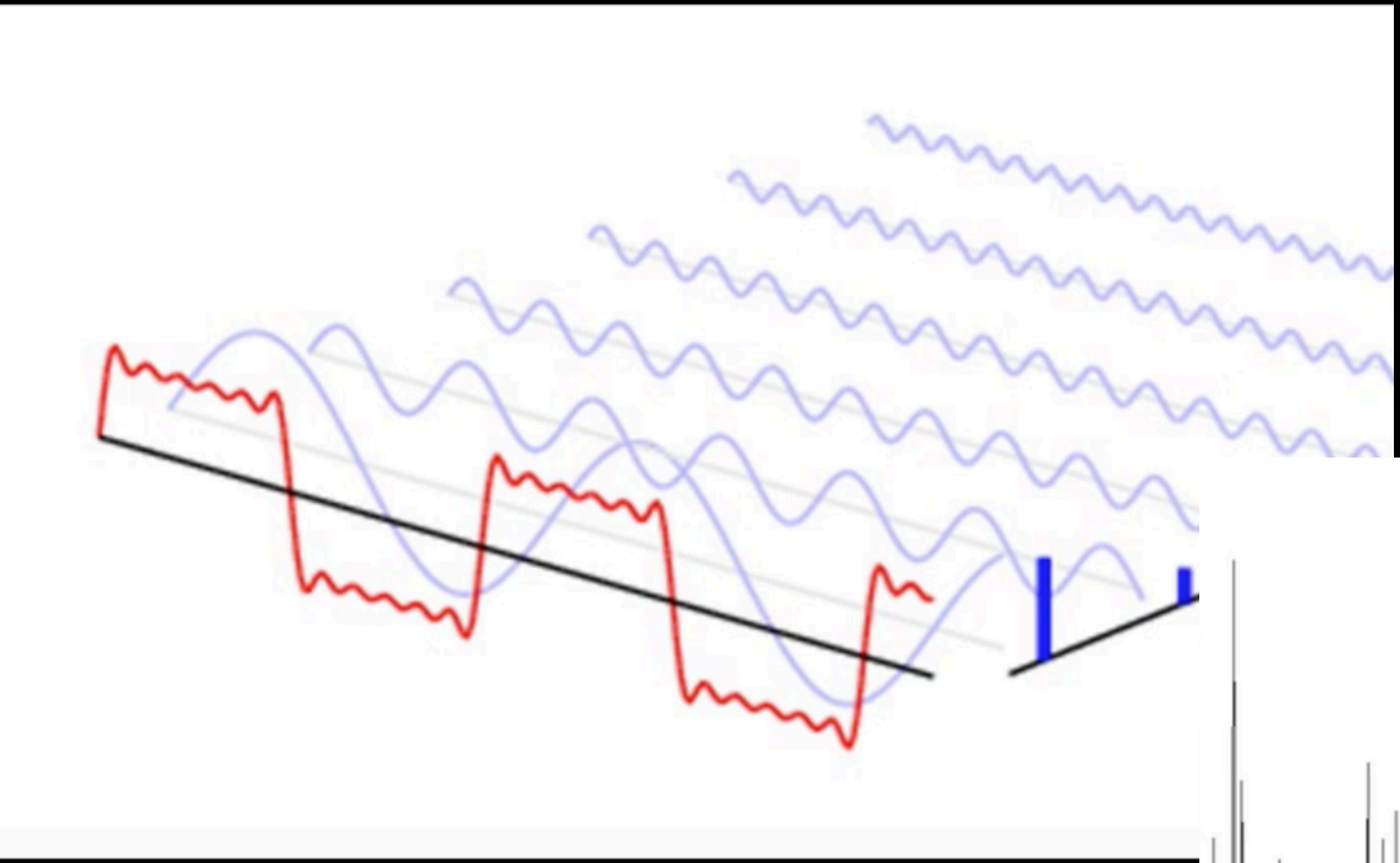
Computers vs. Humans

Recognize?

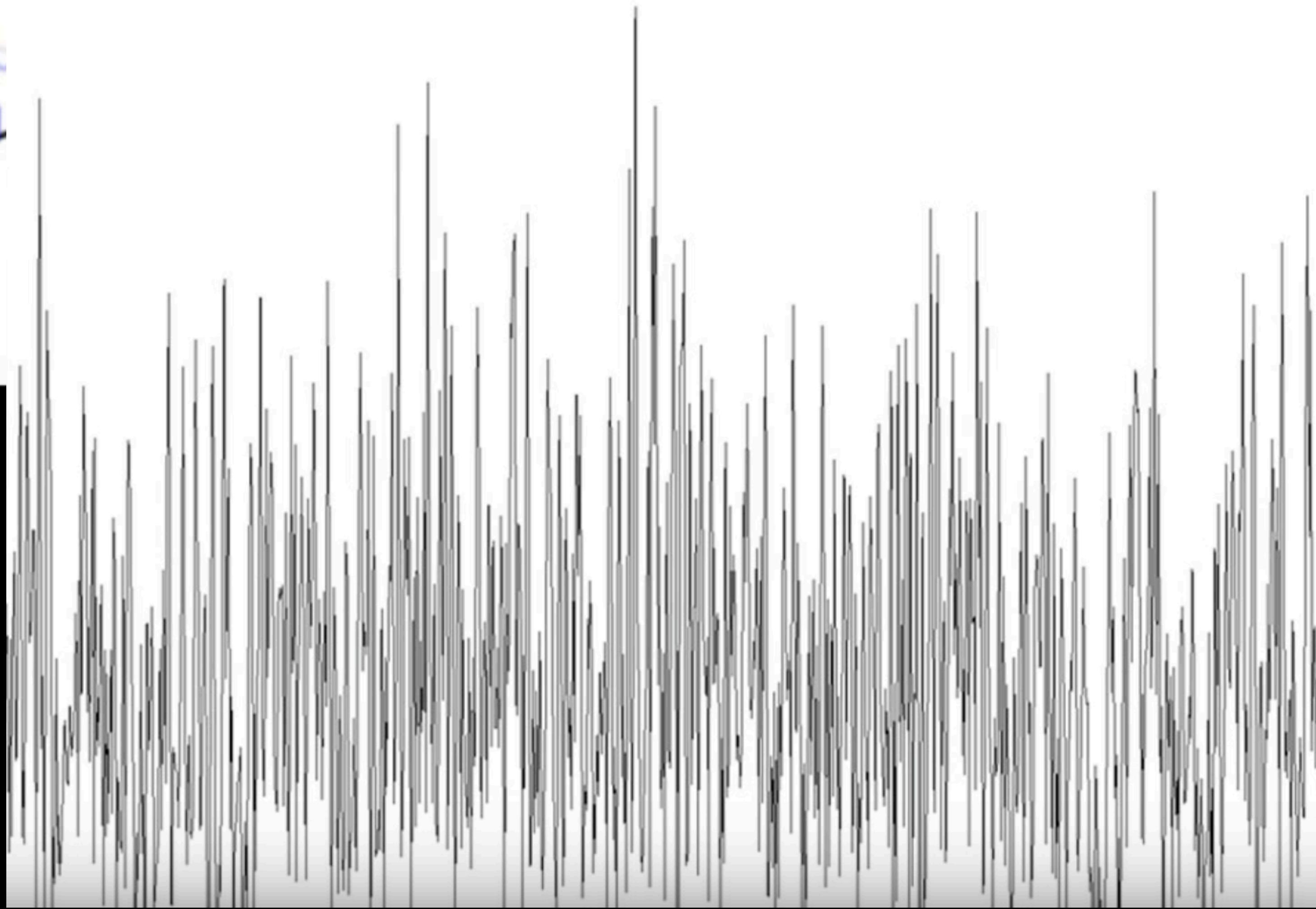
Differences?

Fourier Transforms

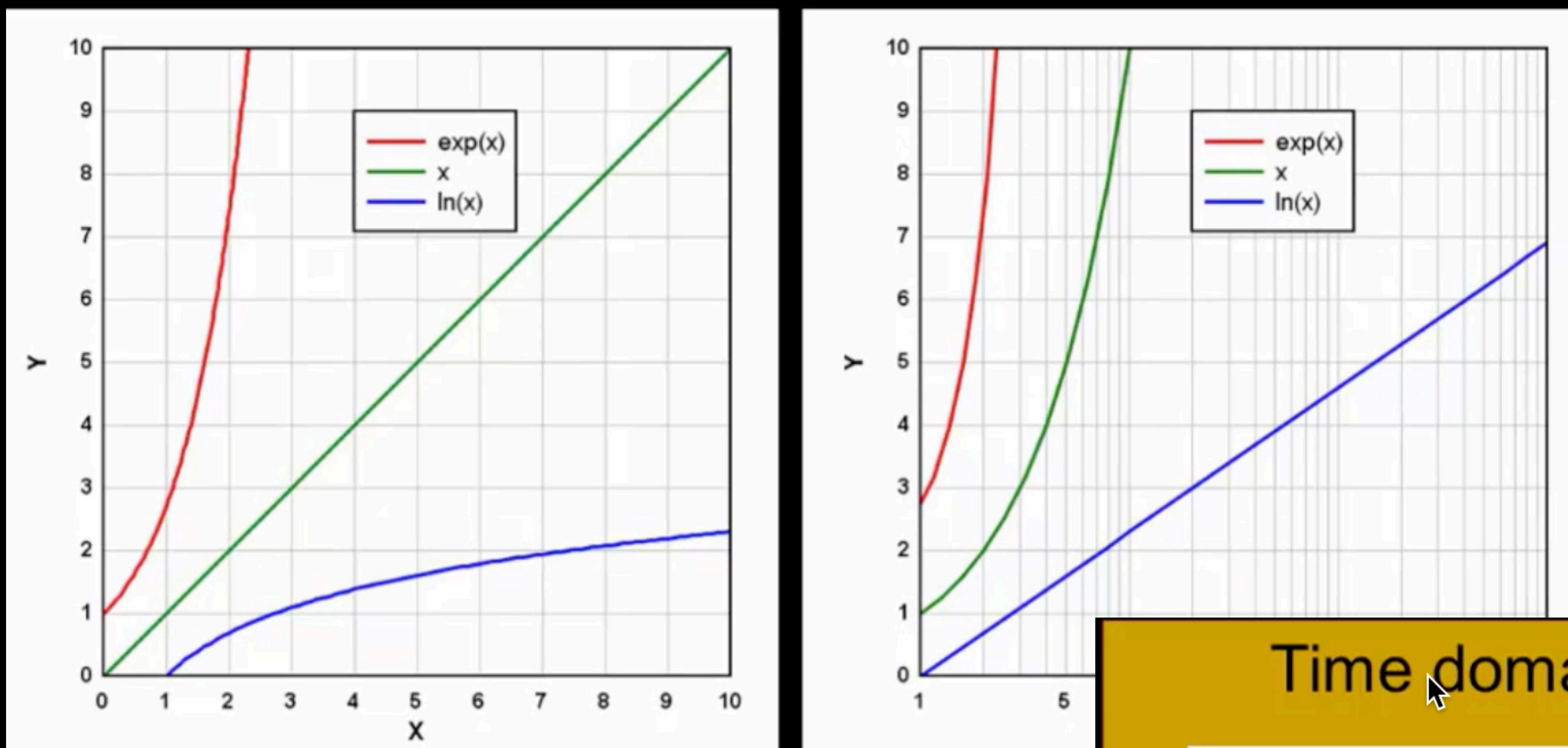
Fourier series: periodic continues signals



Fourier transform:
non-periodic signals (e.g. noise)

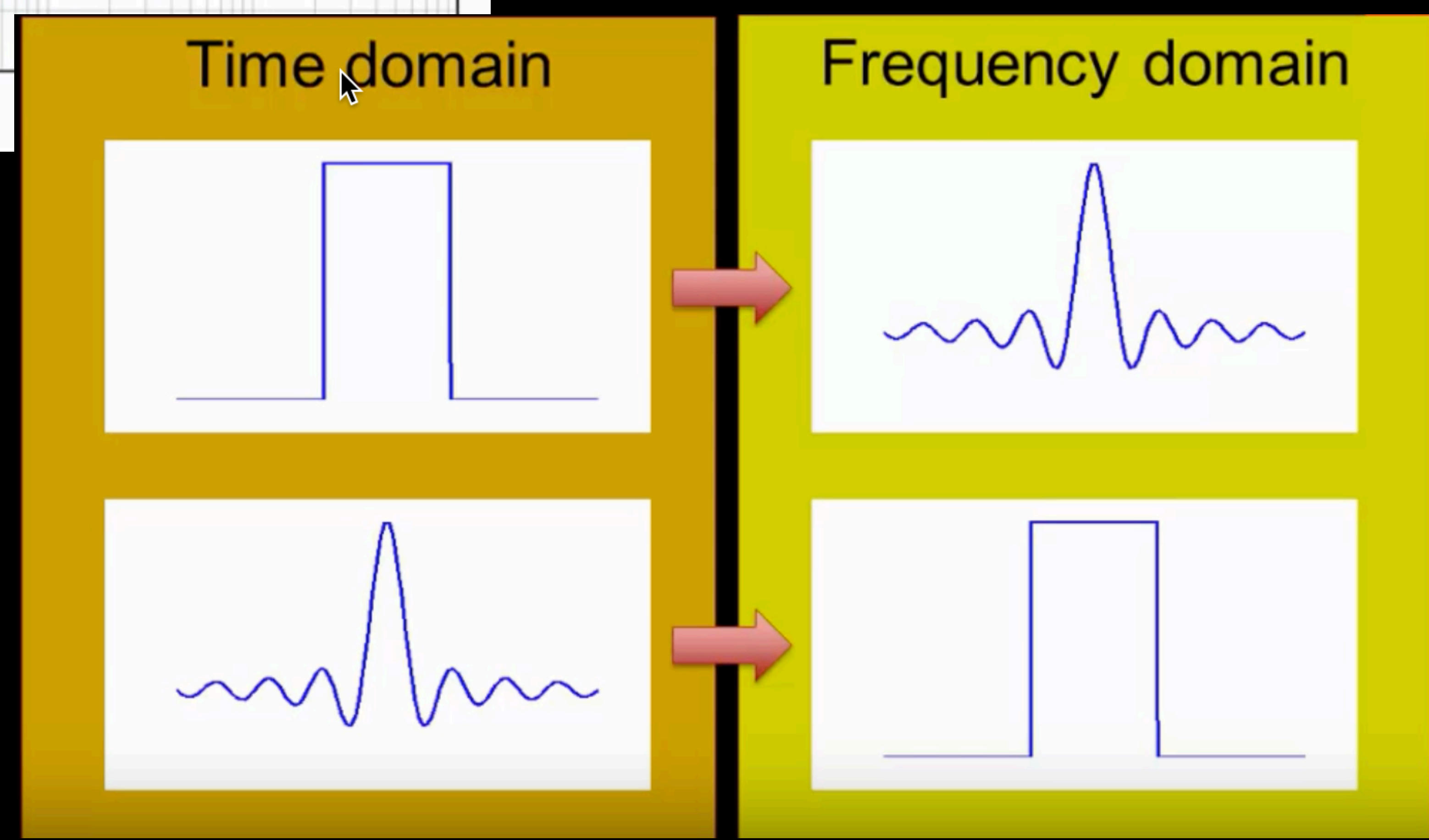


FT of signals: display data in a different form

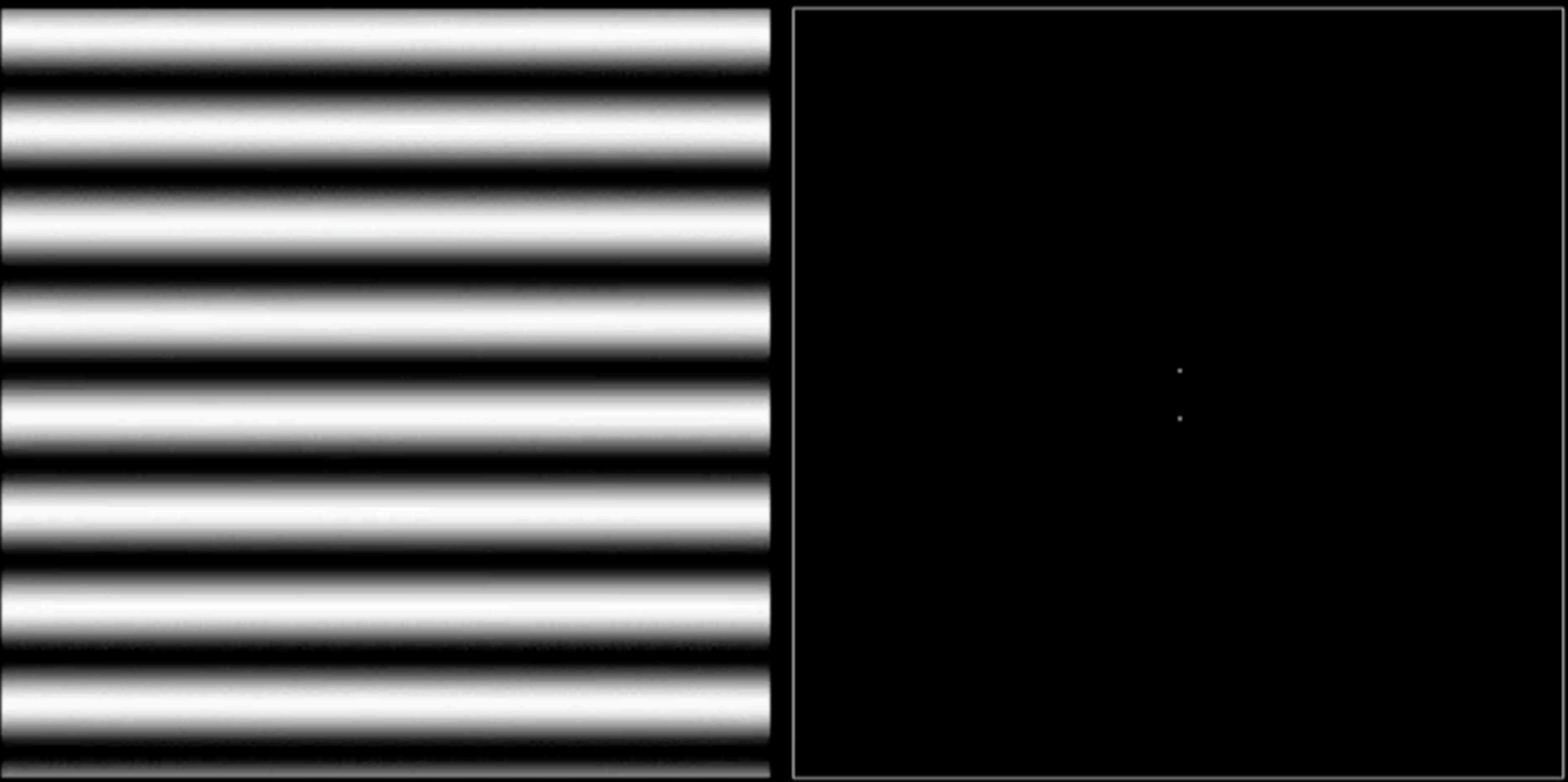


Cartesian vs. log scale

Box vs. Sink

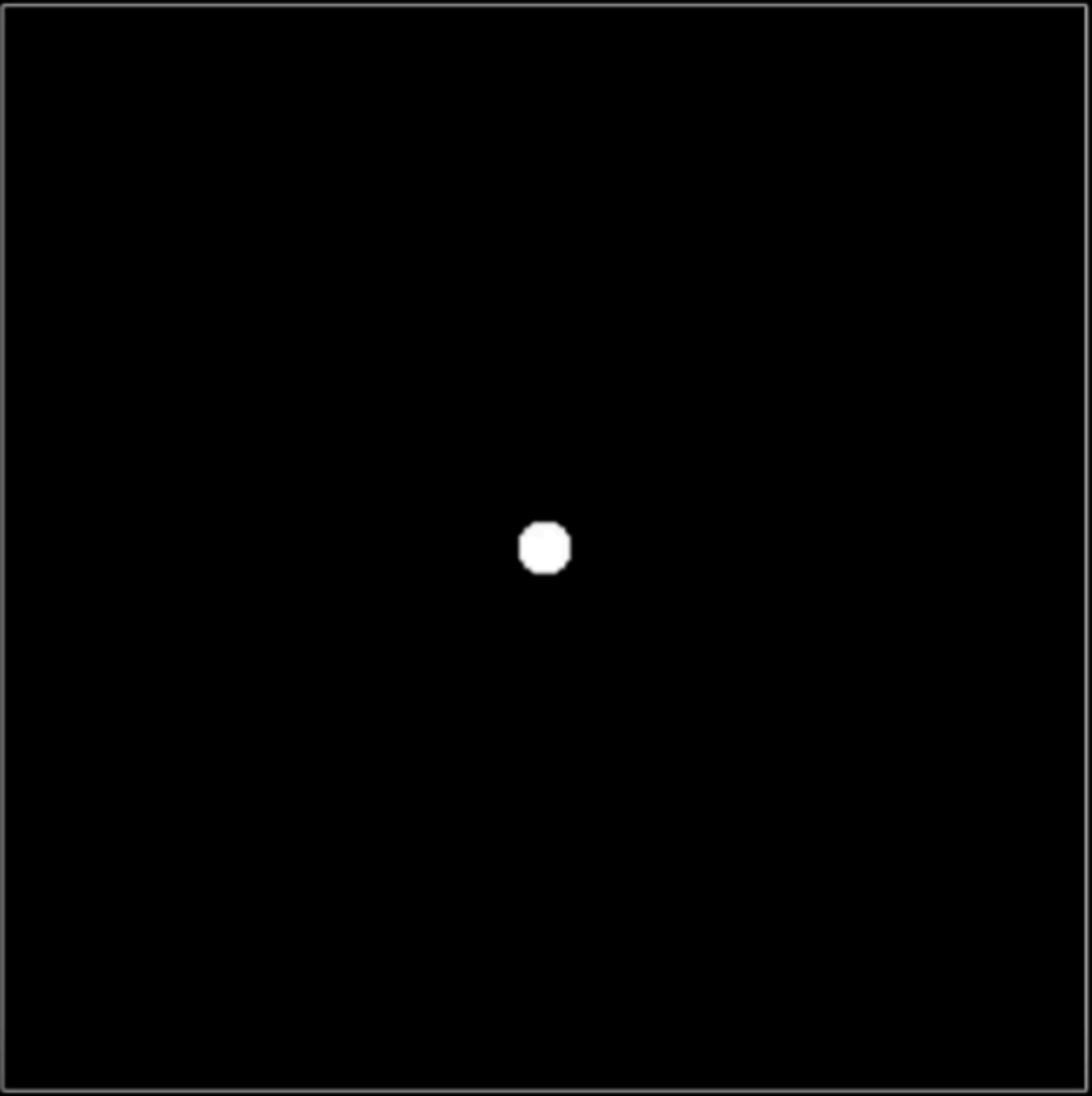


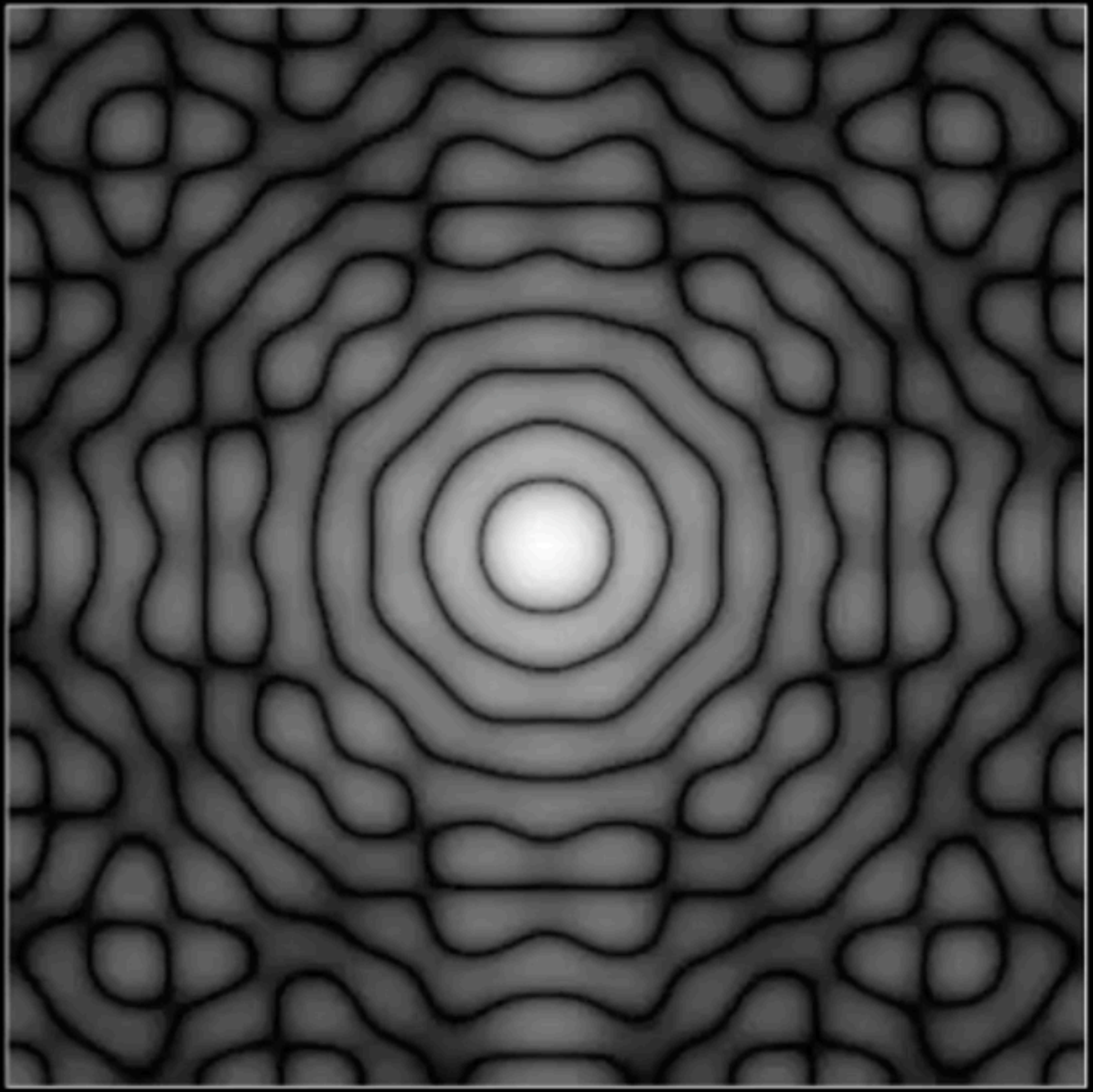
FT of images



Parallel lines vs. two white dots

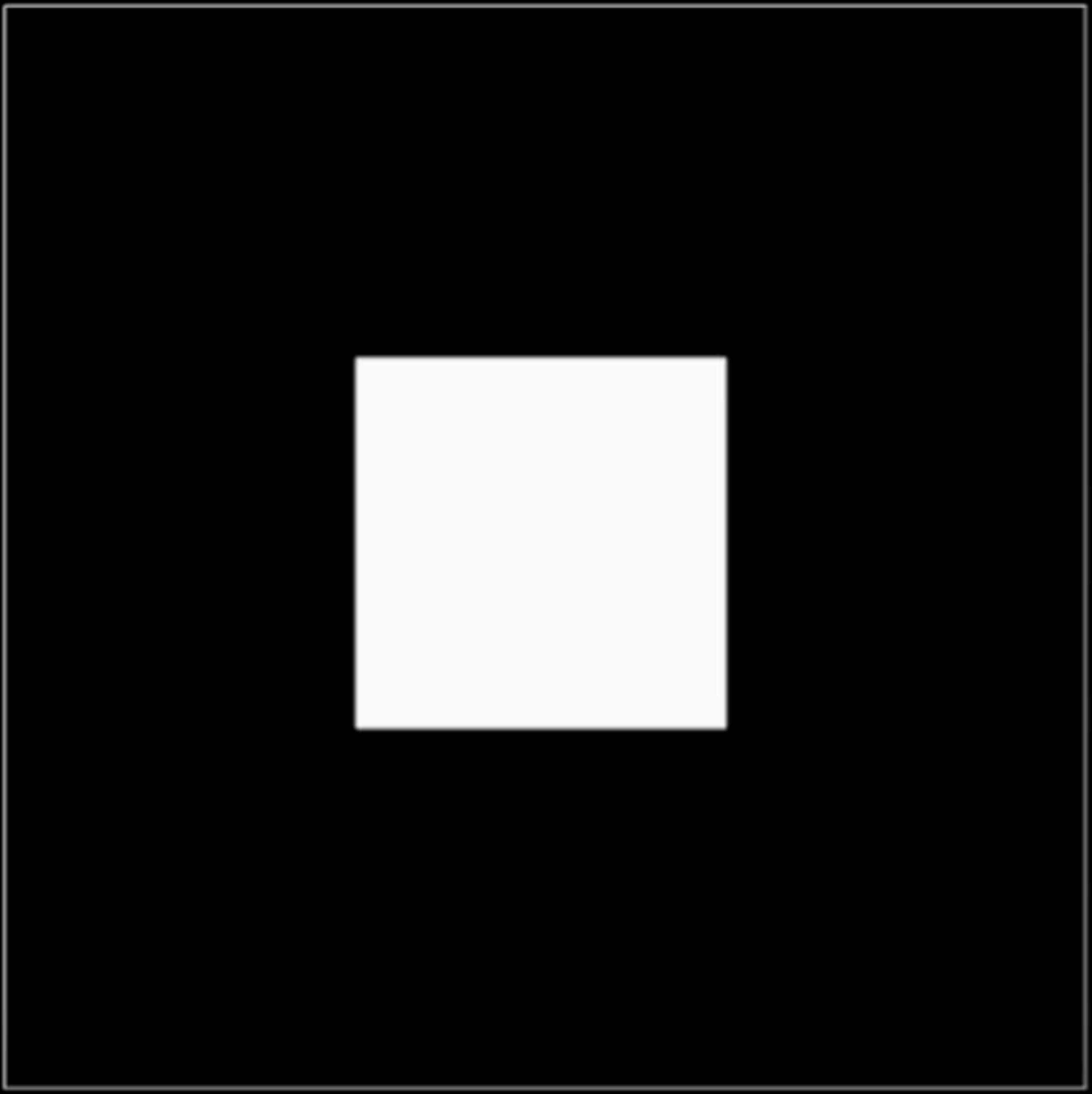
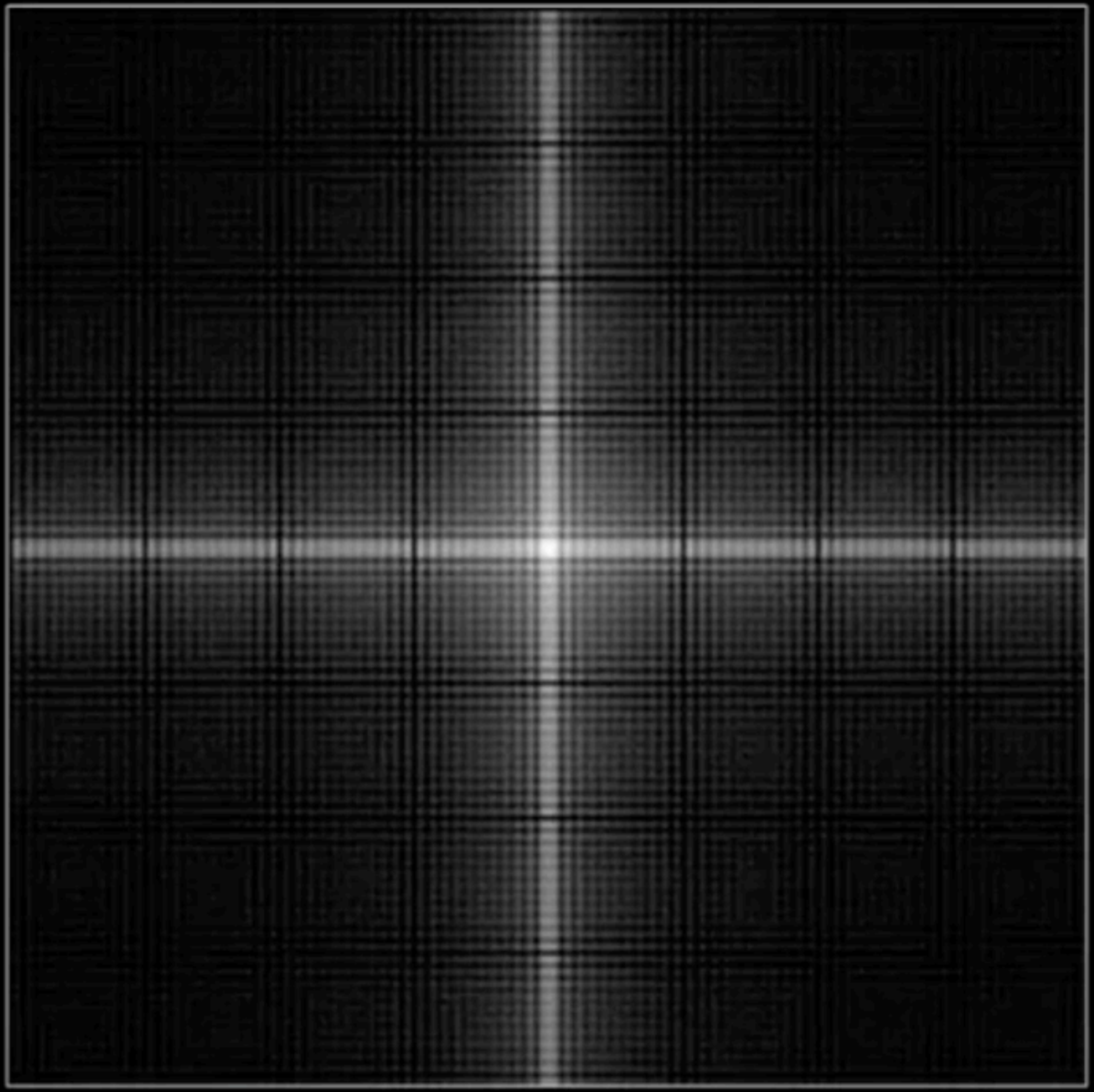
FT of images





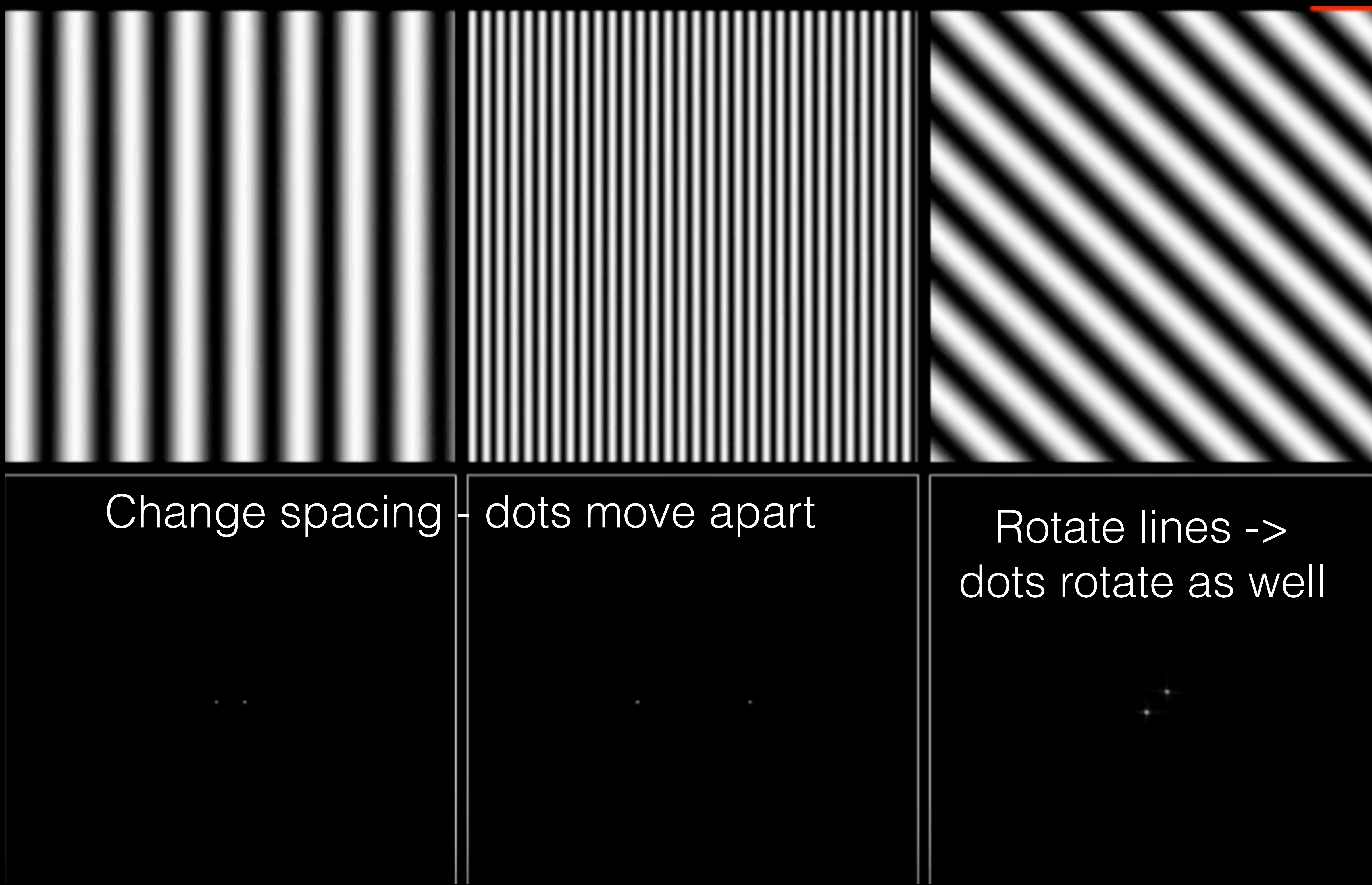
White circle vs. ripples from stone in water

FT of images

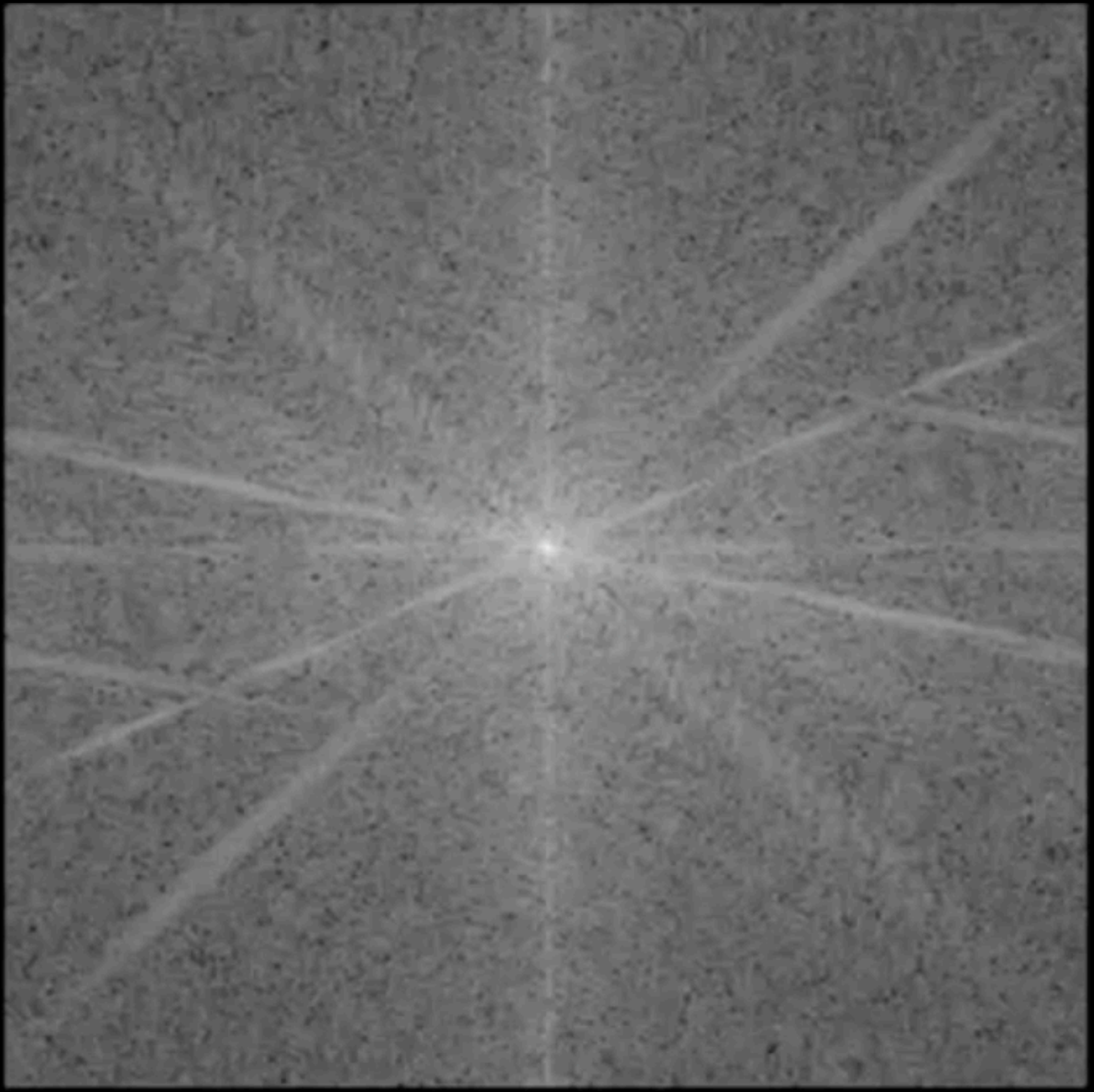



Square vs. cross

FT of images

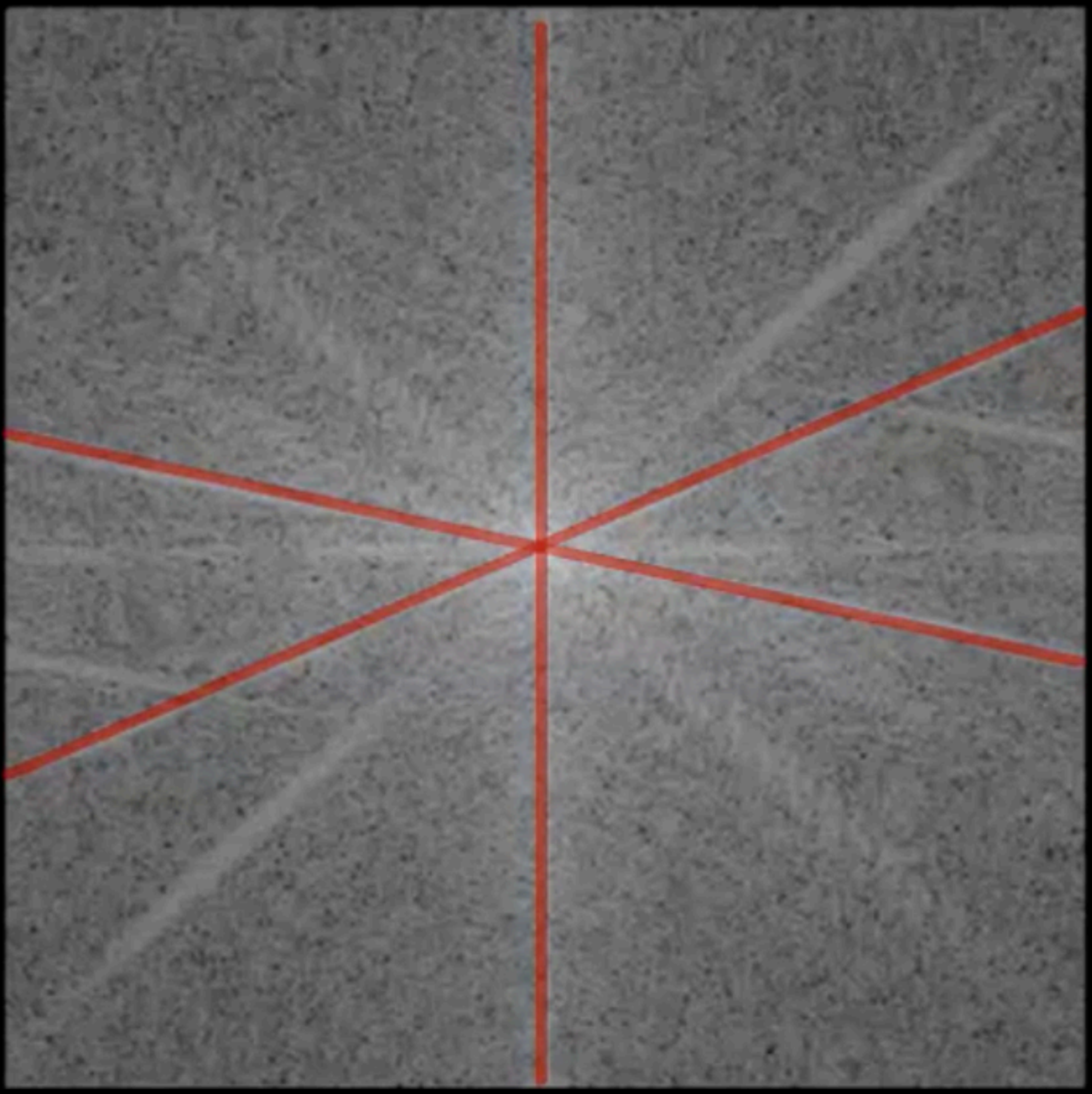

Change spacing - dots move apart
Rotate lines ->
dots rotate as well

FT of images



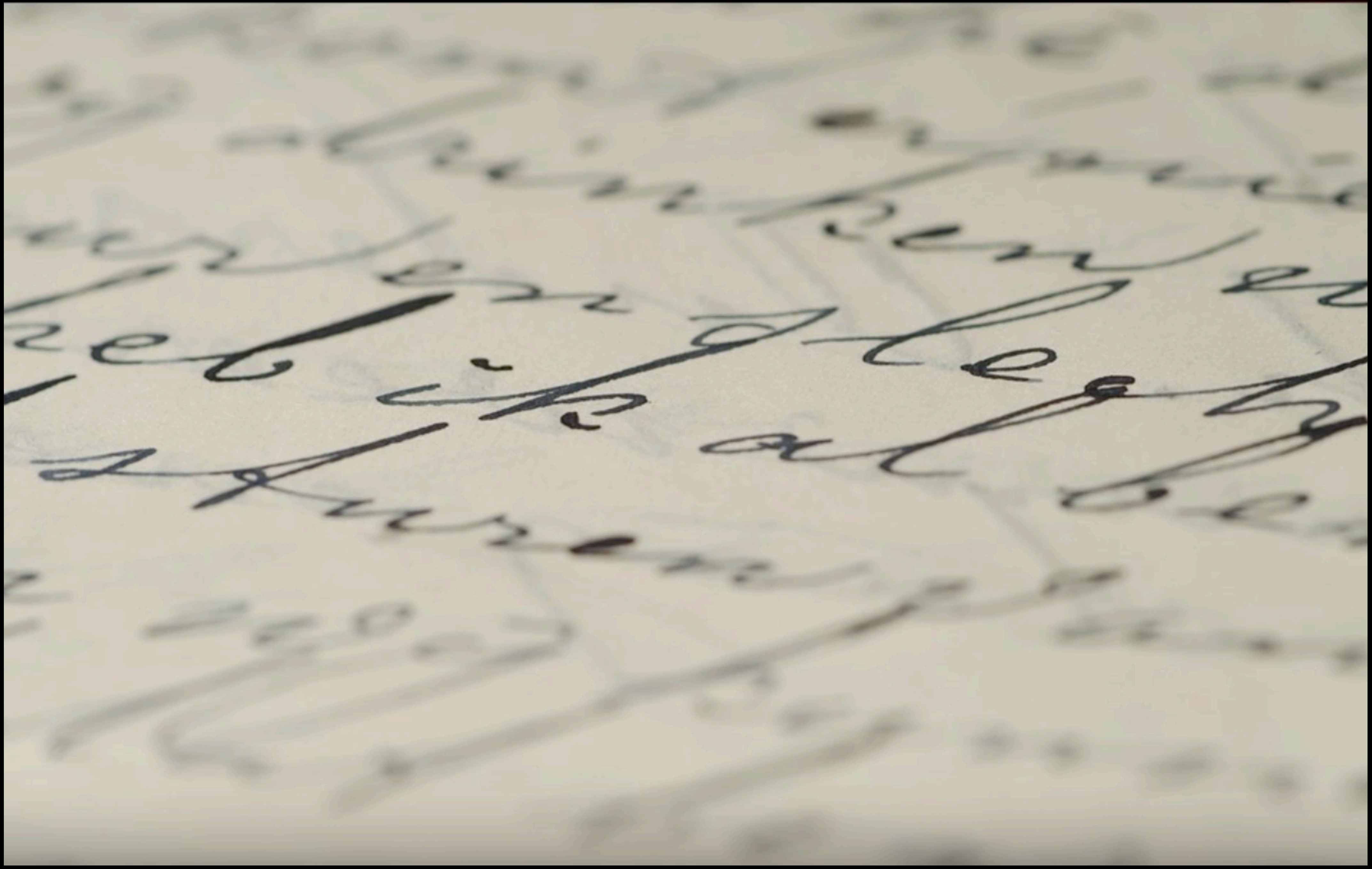
FT of a real image

FT of images



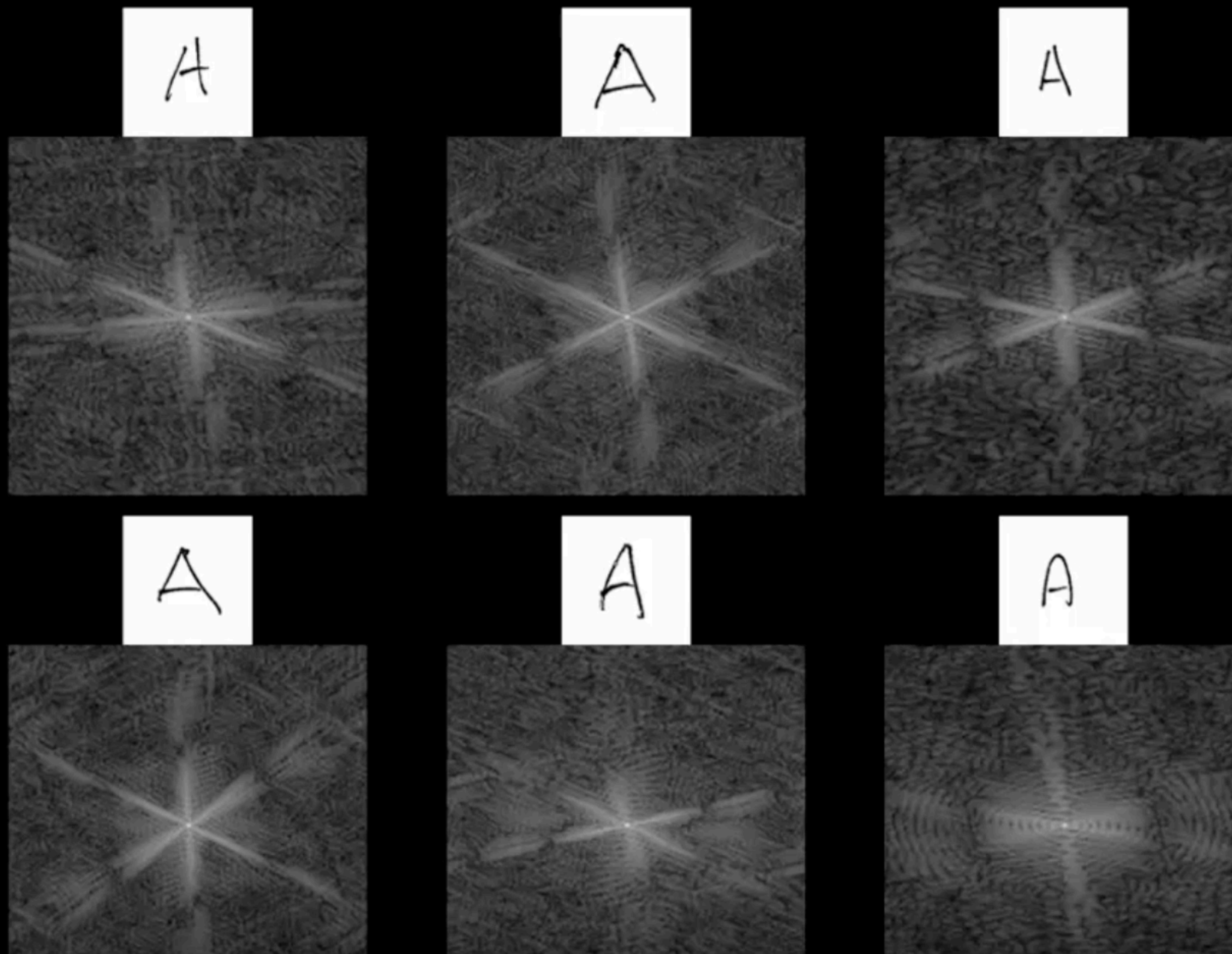
Example: Reading text

Difficult before DL, used FT



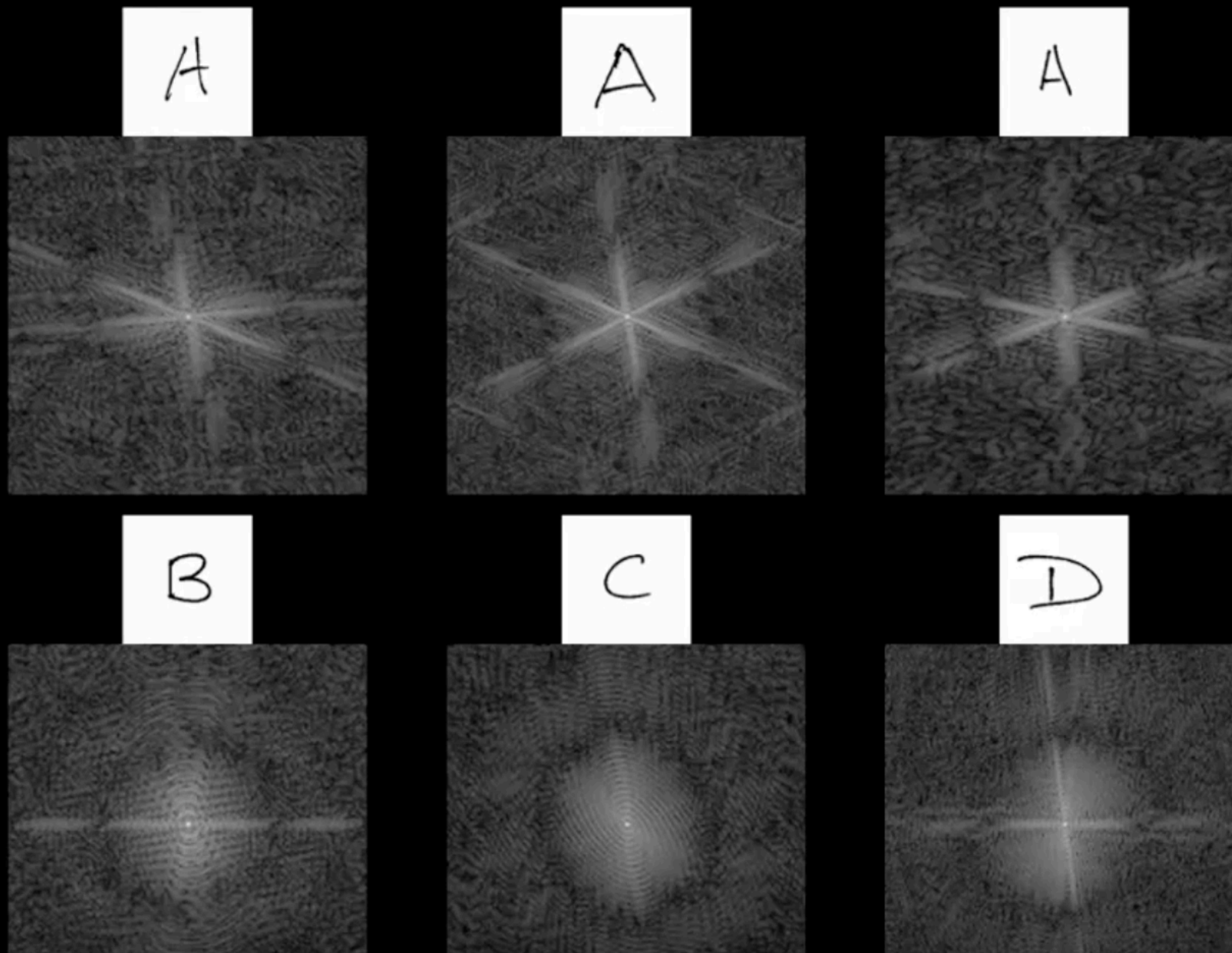
Recognize characters

Similar in frequency domain

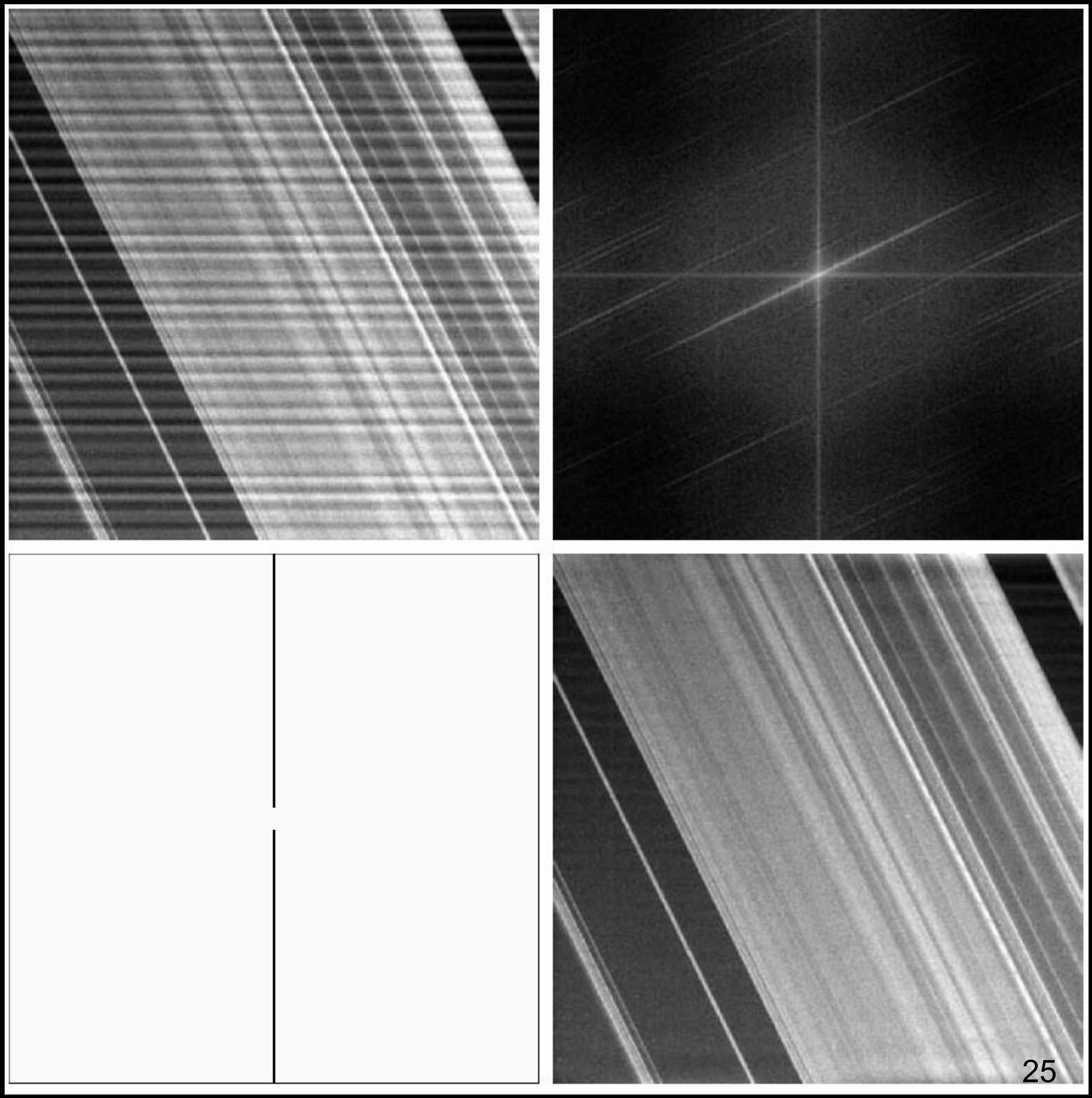


Recognize characters

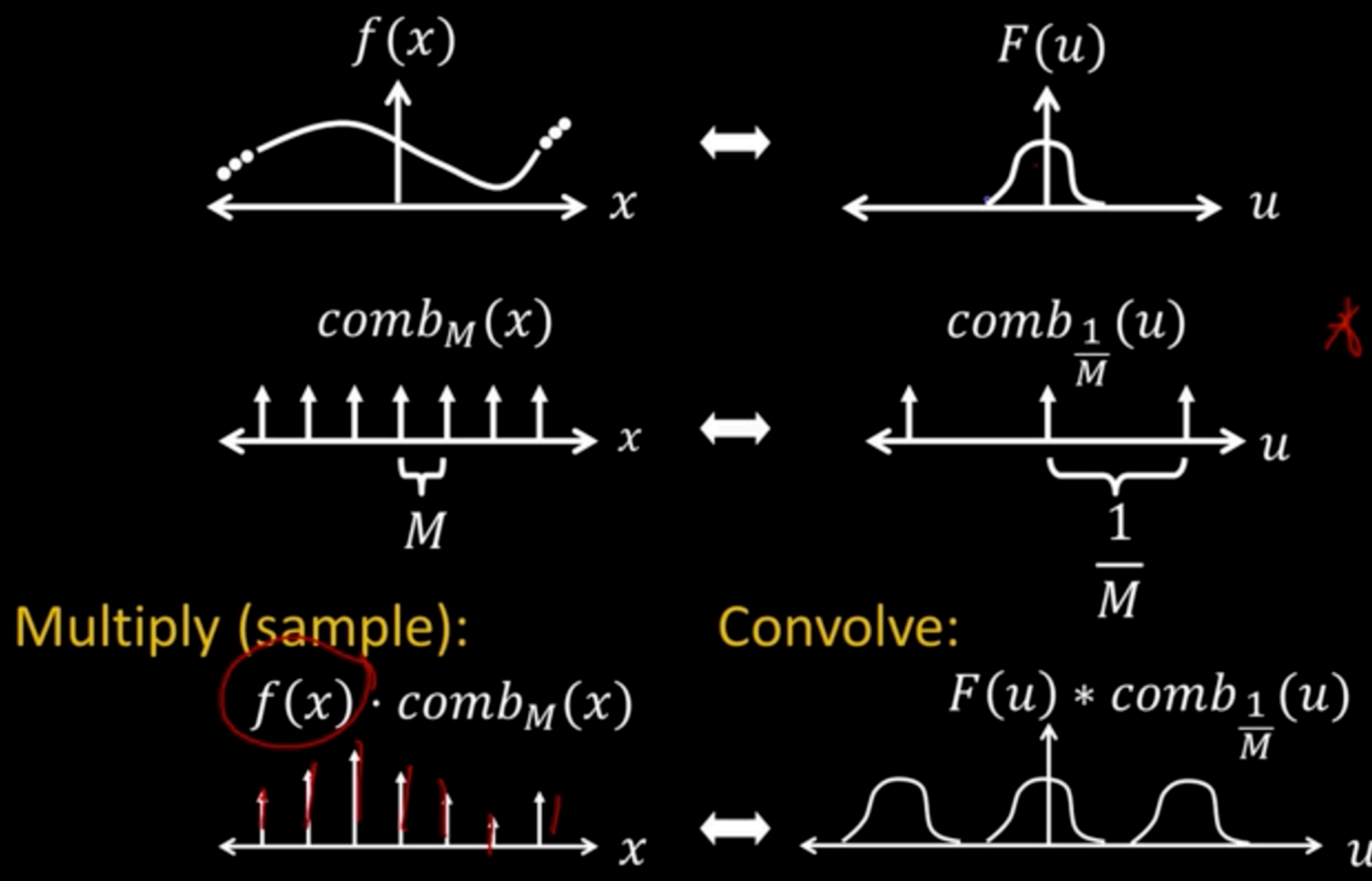
Different in frequency domain



Example: Filtering (Noise)



Example: Understanding



Overview: Fourier Domain

3Blue1Brown: What is the Fourier Transform?

Periodic function: weighted sum of sines & cosines

A sum of sines

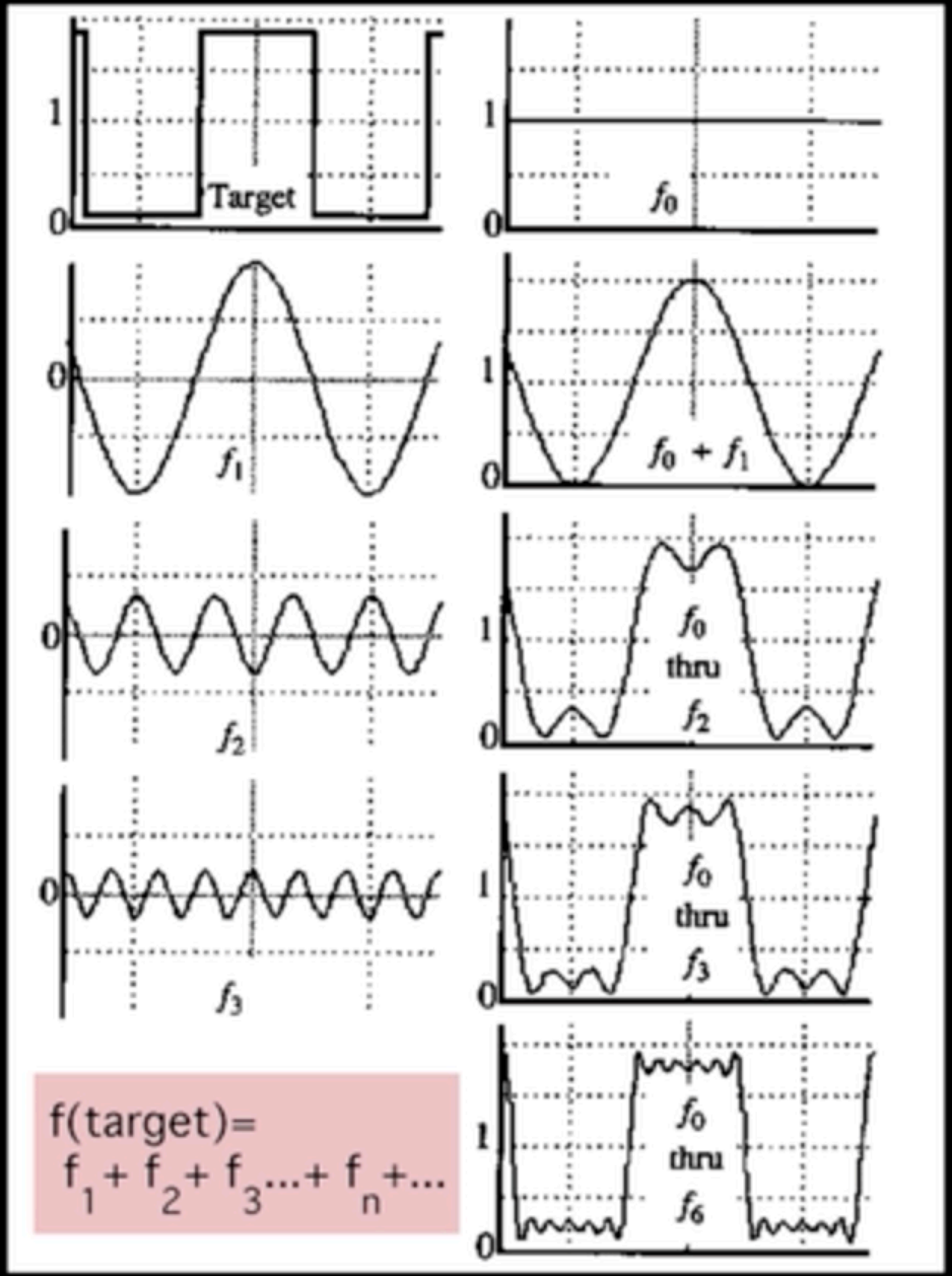
Our building block:

$$A\sin(\omega x+\varphi)$$

Add enough of them to get any signal *f(x)* you want!

How many degrees of freedom?

What does each control?



Fourier Series

Fourier *Transform*

We want to understand the frequency ω of our signal.
So, let's reparametrize the signal by ω instead of *x*:

$$f(x) \longrightarrow \boxed{\text{Fourier Transform}} \longrightarrow F(\omega)$$

$$F(\omega) = R(\omega) + iI(\omega)$$

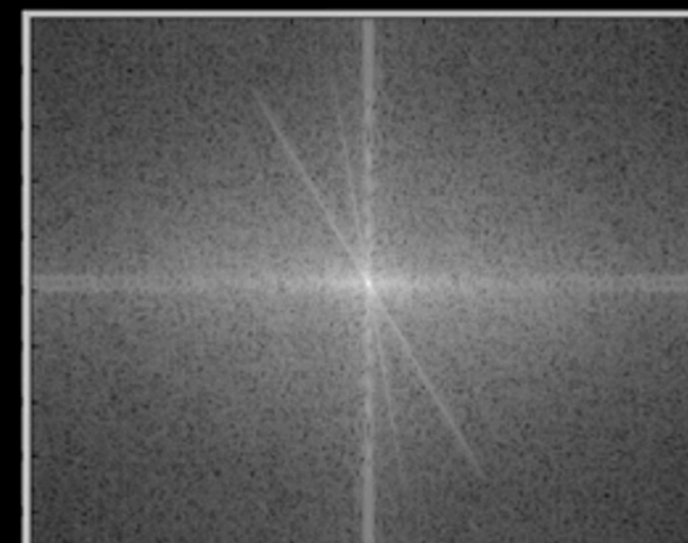
$$A = \pm\sqrt{R(\omega)^2 + I(\omega)^2}$$

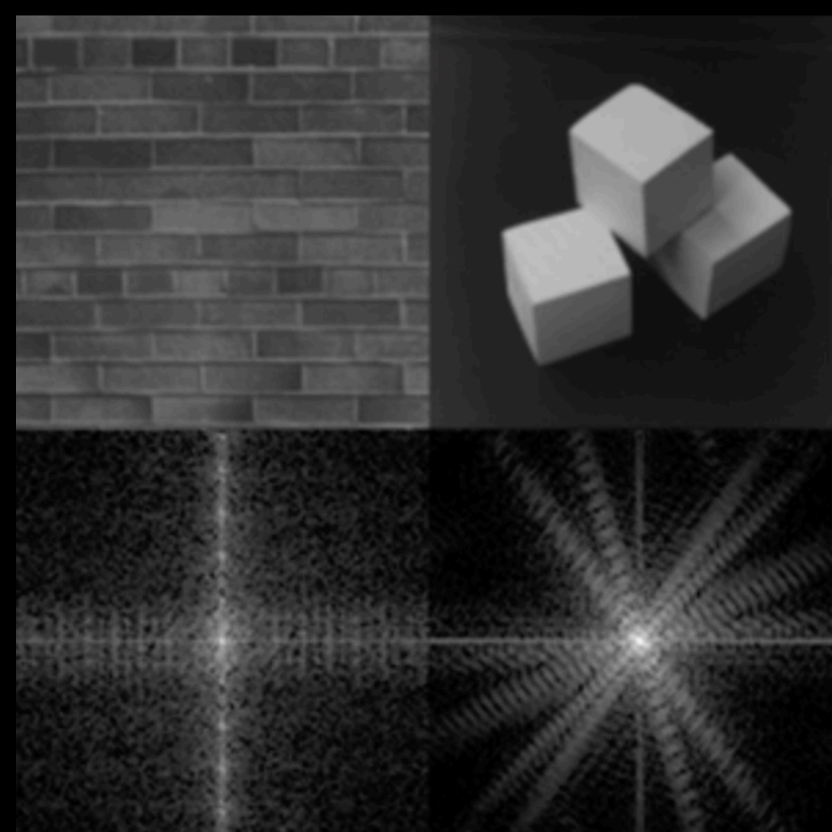
$$\varphi = \tan^{-1}\frac{I(\omega)}{R(\omega)}$$

Complex in general

Can also be expressed in polar form

Image & Fourier transform

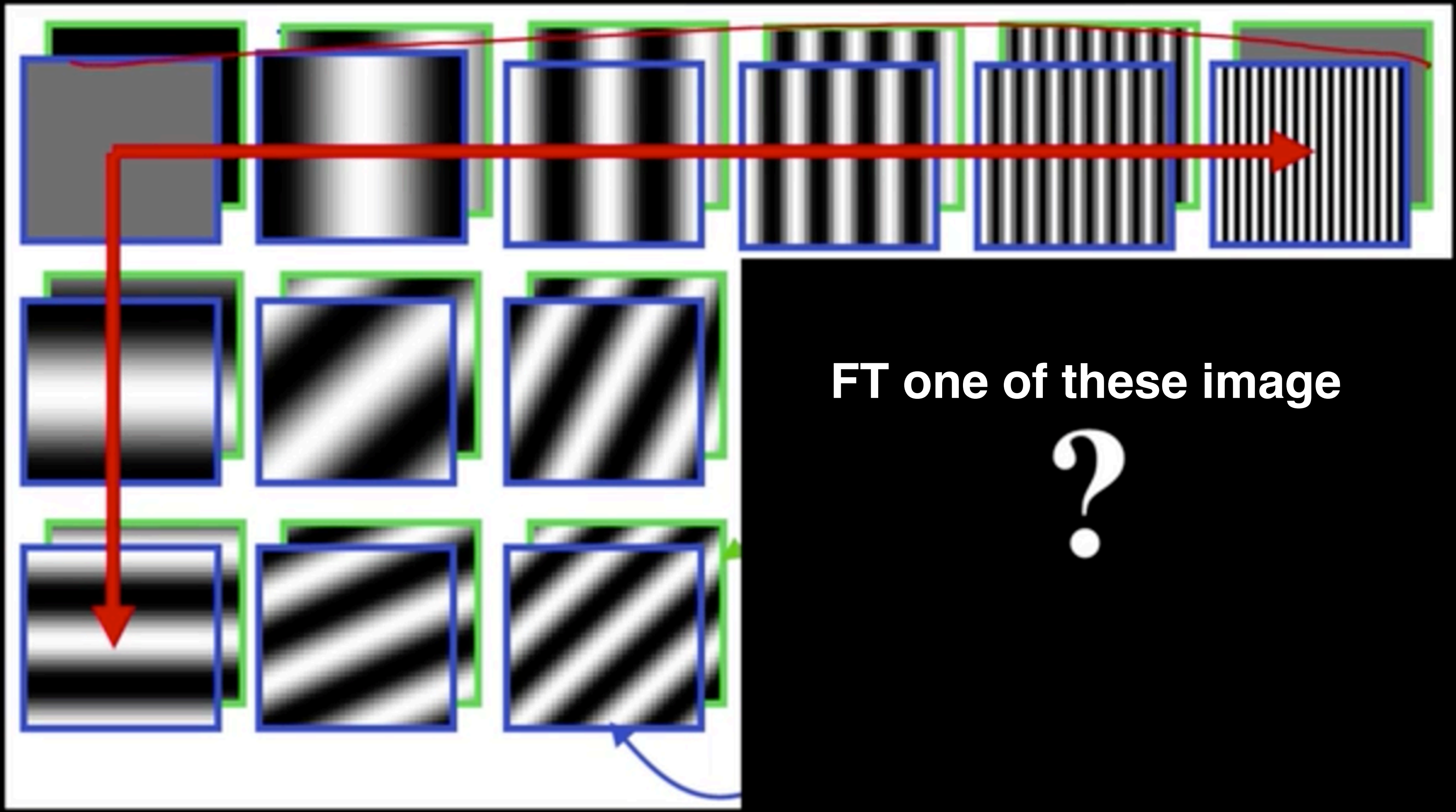




Centered

Log-transformed

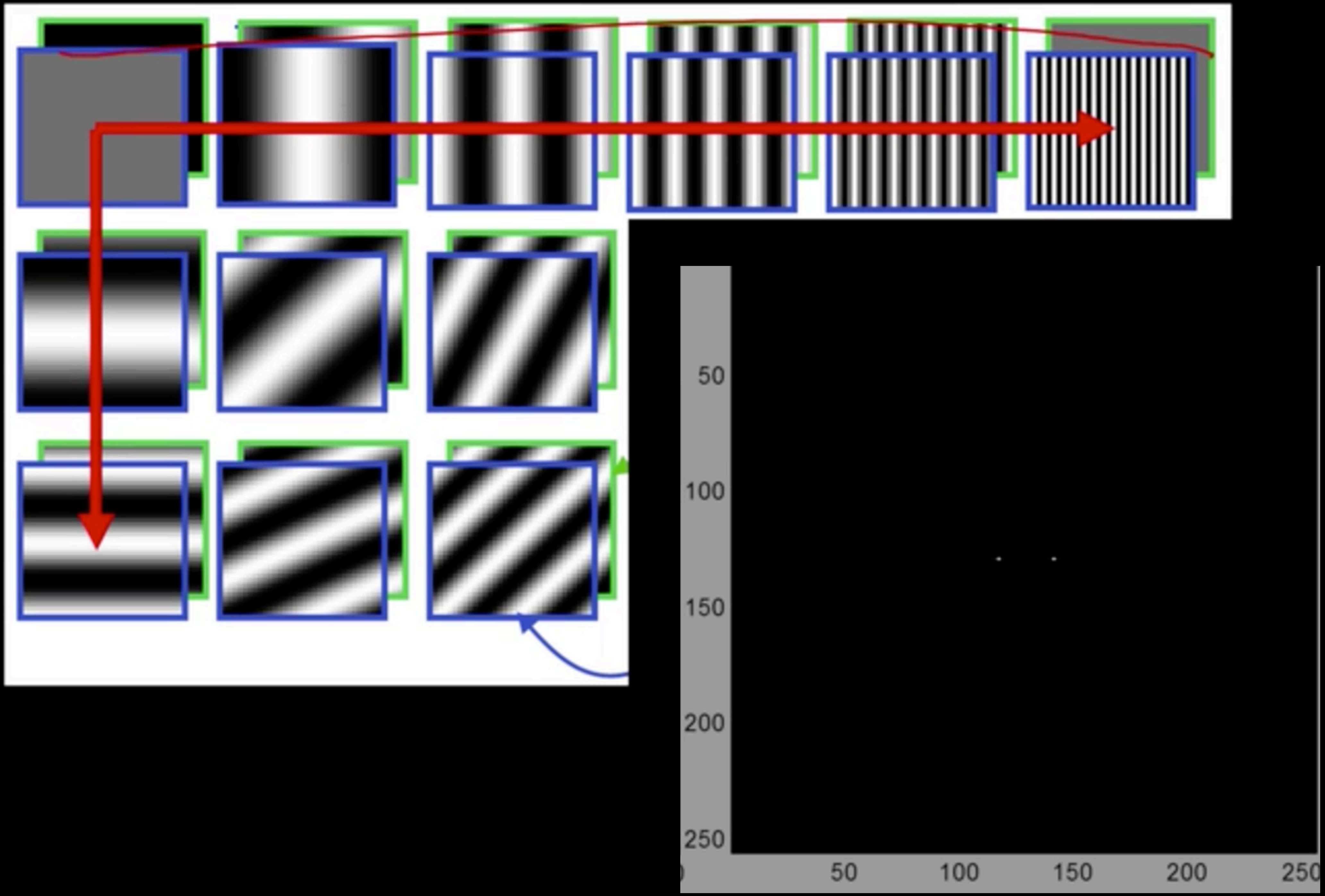
Decomposition & Basis



FT one of these image

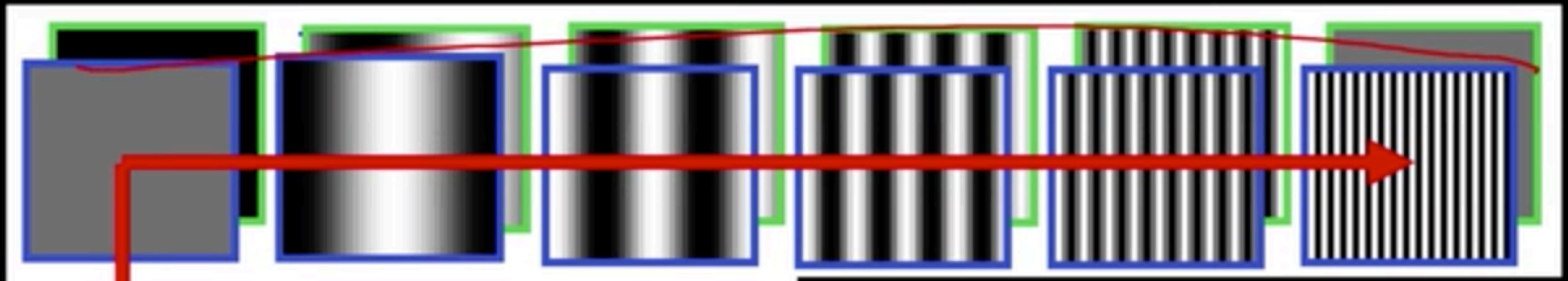
?

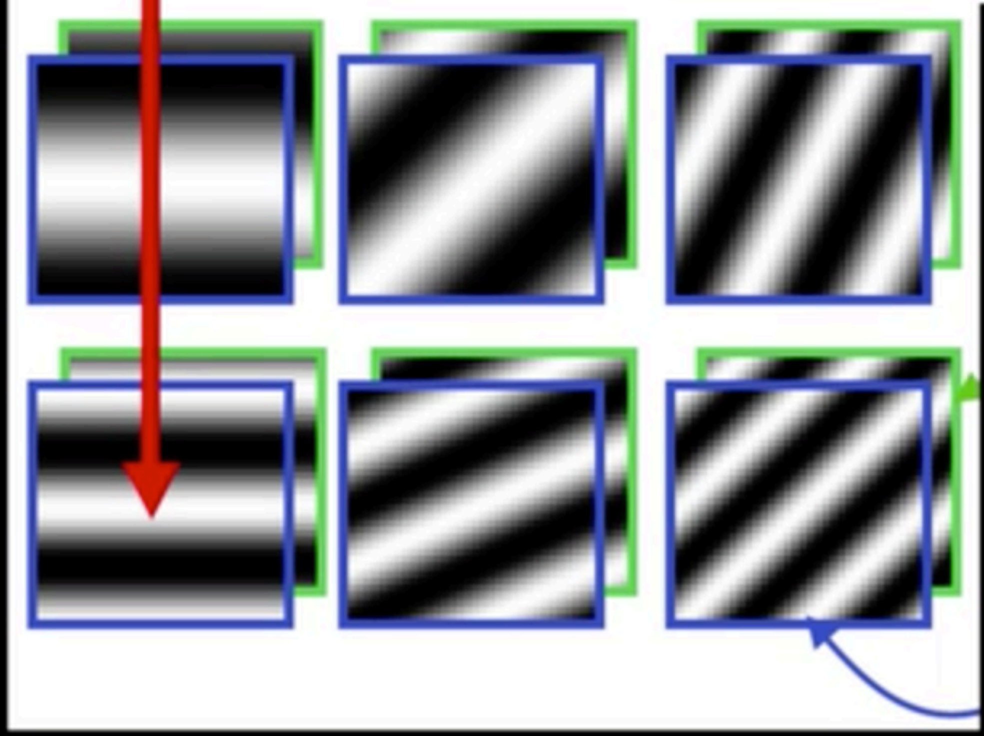
Decomposition & Basis

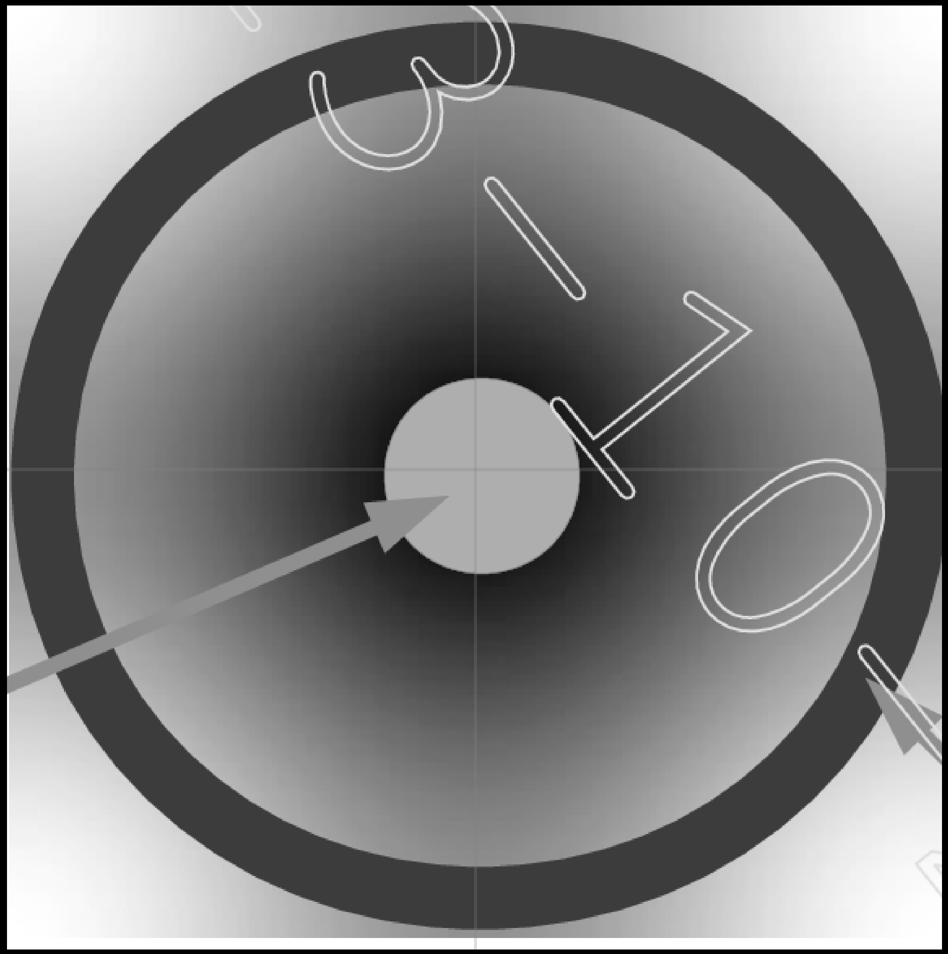


FT of one image

Decomposition & Basis



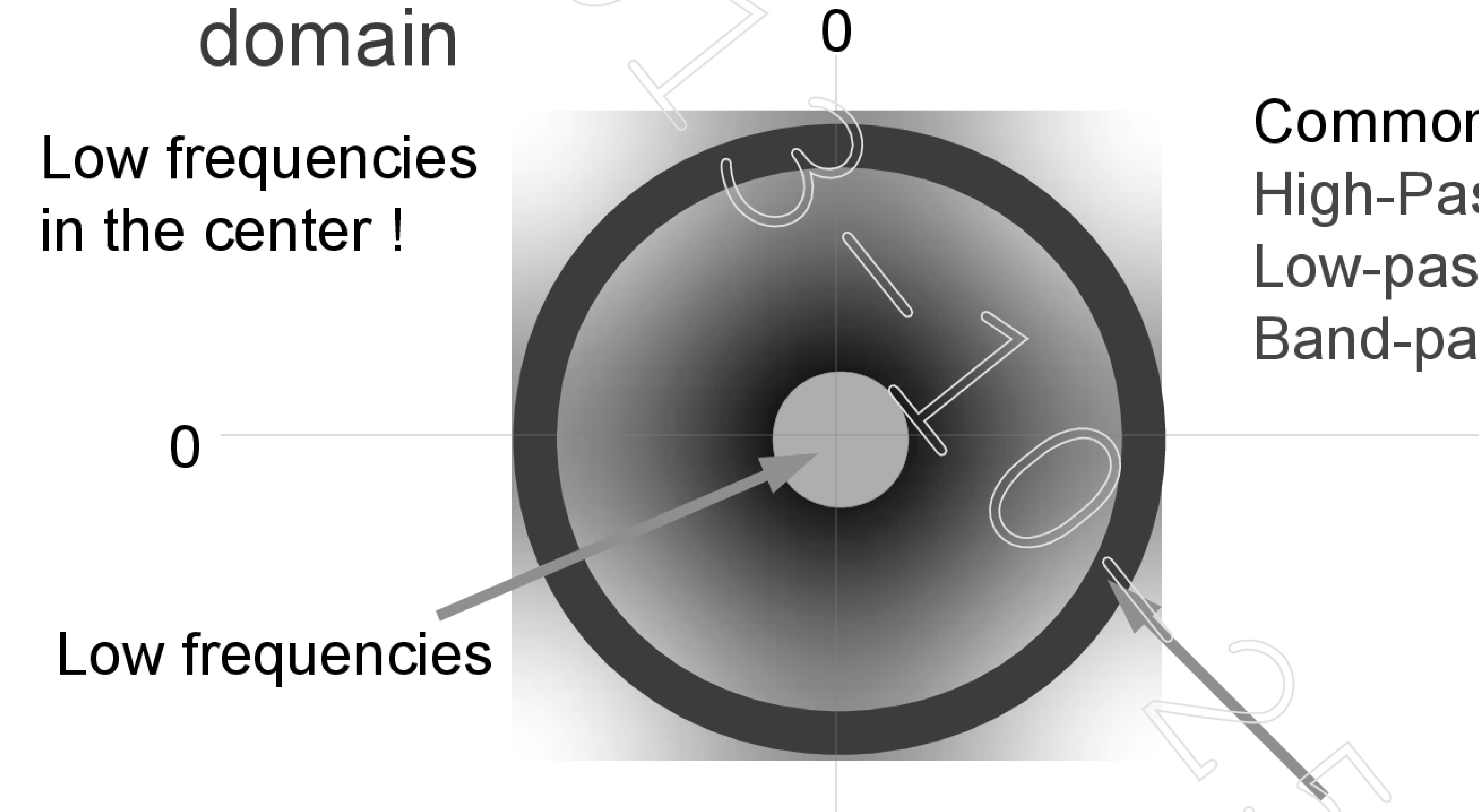




How much of each

Frequency Filter Interpretation

- Representation of the filter in the frequency domain



Common Filter types:
High-Pass
Low-pass
Band-pass

The frequency filter function is also known as **transfer function**.

Fourier Transform – more formally

Represent the signal as an infinite weighted sum of an infinite number of sinusoids:

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-i 2\pi u x} dx$$

Again: $e^{ik} = \cos k + i \sin k \qquad i = \sqrt{-1}$

Spatial Domain (x) $\longrightarrow$ Frequency Domain (ω or u or *even* s)
(Frequency Spectrum $F(u)$ or $F(\omega)$)

Fourier Transform – more formally

Inverse Fourier Transform (IFT) – add up all the sinusoids at x:

$$f(x) = \int_{-\infty}^{\infty} F(u) e^{i2\pi ux} du$$

The 1D Fourier transform (continuous)

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

Inverse Fourier transform

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

2D continuous

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

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

Fourier Transform ⇔ Fourier Series

- The ***Discrete FT***:

$$F(k) = \sum_{x=0}^{x=N-1} f(x) e^{-i\frac{2\pi kx}{N}}$$

… where x is discrete and goes from the start of the signal to the end *(N-1)*

… and *k* is the number "cycles per period of the signal" or "cycles per image.

  - Only makes sense k = $-N/2$ to $N/2$. Why? What's the highest frequency you can unambiguously have in a discrete image?

2D Fourier Transforms

- The two dimensional version: .

$$F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\, e^{-i2\pi(ux+vy)}\, dx\, dy$$

- And the 2D ***Discrete FT***:

$$F(k_x,k_y) = \sum_{x=0}^{x=N-1}\sum_{y=0}^{y=N-1} f(x,y)\, e^{-i\frac{2\pi(k_x x+k_y y)}{N}}$$

- Works best when you put the origin of *k* in the middle….

Discrete Fourier transform and inverse

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

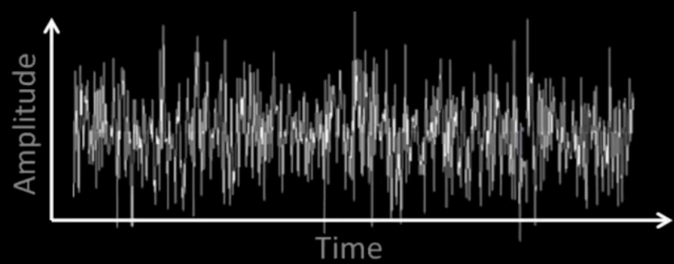
$$f(x) = \frac{1}{M}\sum_{u=0}^{M-1} F(u) e^{j2\pi ux/M} \quad \text{for } x = 0,1,\ldots,M-1$$

2D discrete

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

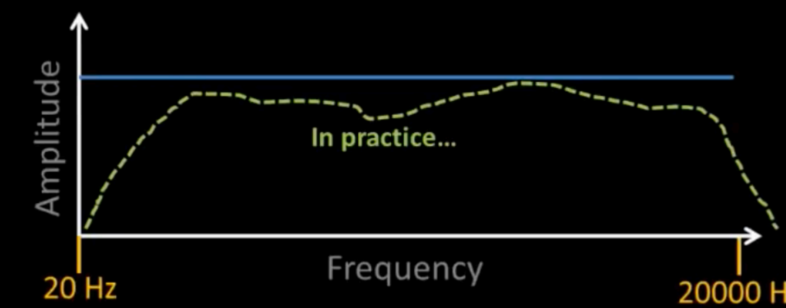
$$f(x,y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)\, e^{j2\pi(ux/M+vy/N)} \quad \text{for } x = 0,1,\ldots,M-1 \text{ and } y = 0,1,\ldots,N-1$$

Fourier Series & Transform



coefficients

$$f(t) = \frac{1}{2}a_0 + \sum_{k=1}^{\infty}(\mathbf{a_k} cos2\pi kt + \mathbf{b_k} sin2\pi kt)$$



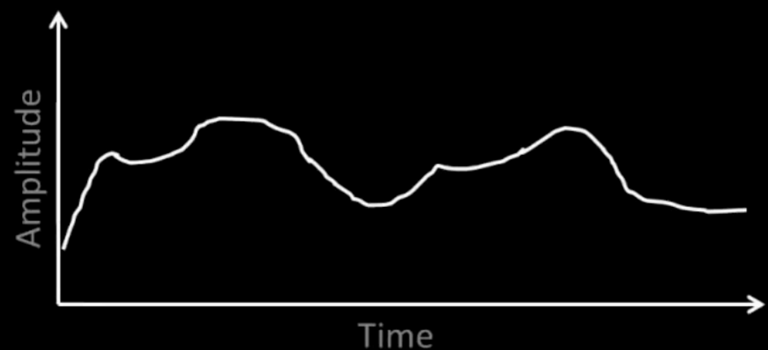
$$\mathbf{X(F)} = \int_{-\infty}^{\infty} \overbrace{x(t)}^{\text{function}} \underbrace{e^{-\mathbf{j}2\pi Ft}dt}_{\text{analyzing function: sinusoids}}$$

Result: One complex coefficient per frequency

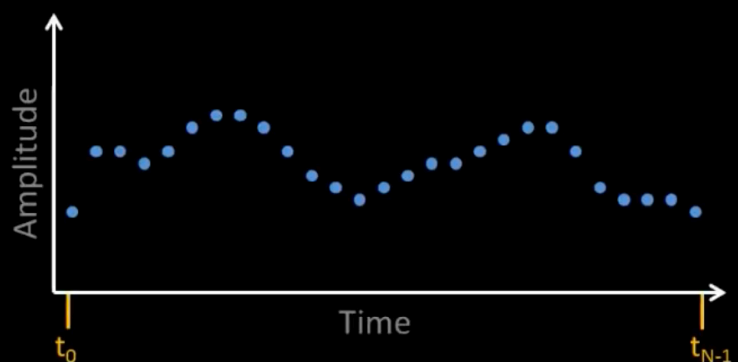
$$\mathbf{X_a(F)} = \int_{-\infty}^{\infty} x(t)cos\ 2\pi Ft\ dt\,, \qquad \mathbf{X_b(F)} = \int_{-\infty}^{\infty} x(t)sin\ 2\pi Ft\ dt$$

Result: Two real coefficients per frequency

Fourier Series & Transform (2)



$$X(F) = \int_{-\infty}^{\infty} x(t)\ e^{-j2\pi F t} dt$$



$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{j2\pi kn}{N}} \qquad b_n \qquad \frac{k}{N} \hat{=} F \qquad n \hat{=} t$$

"kth" frequency bin

$$X_k = x_0 e^{-b_0 j} + x_1 e^{-b_1 j} + \ \ldots + x_n e^{-b_{N-1} j}$$

"nth" sample value

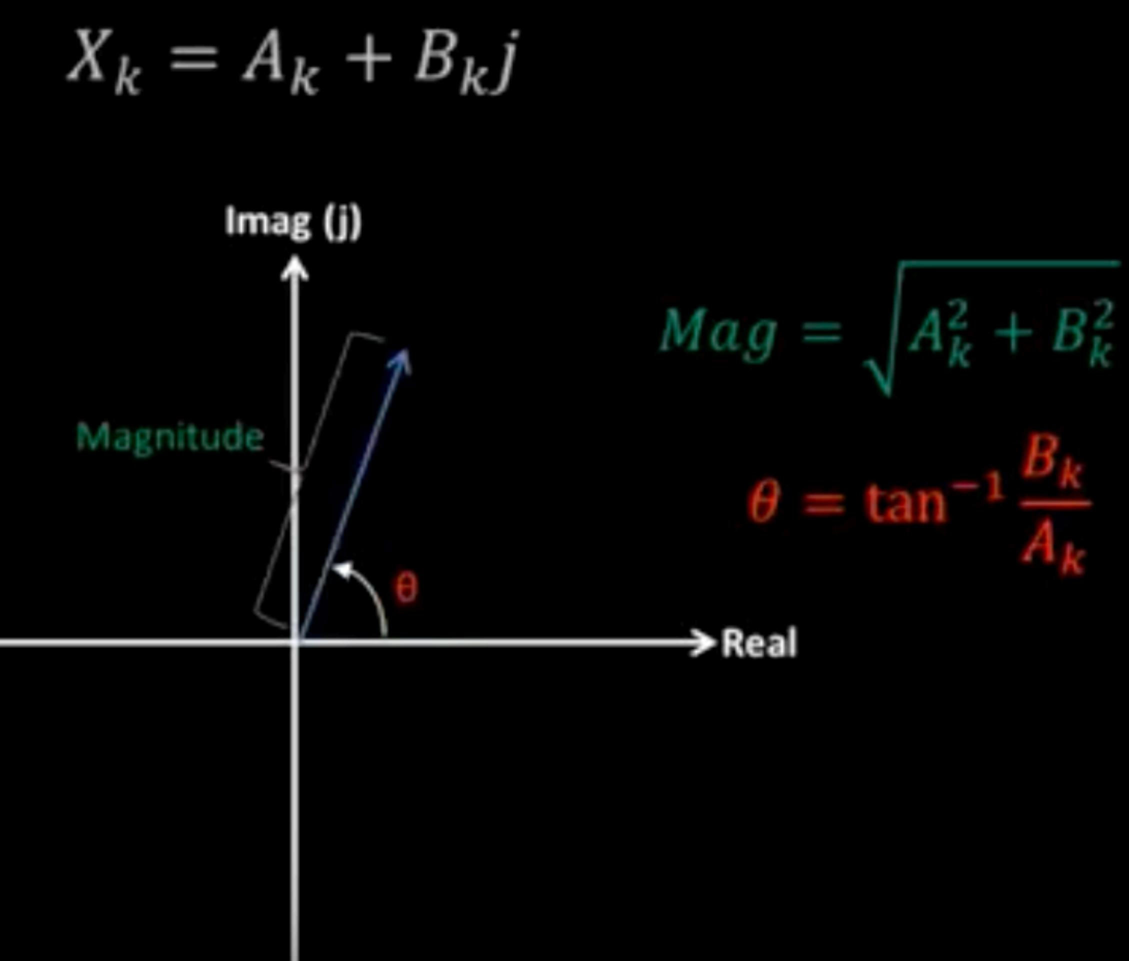
Euler's Formula:

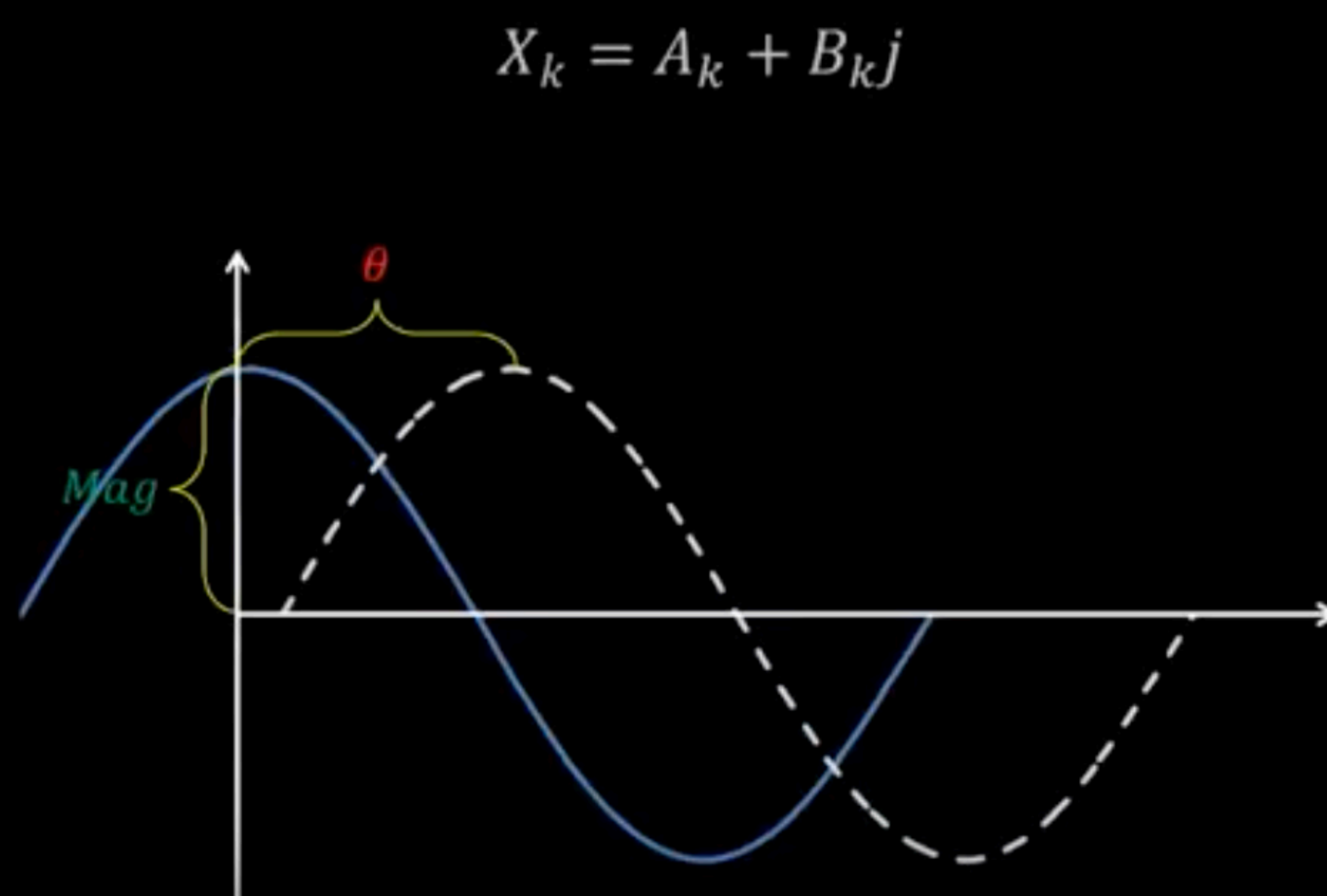
$$e^{jx} = cos\ x + j\ sin\ x$$

$$X_k = x_0[\cos(-b_0) + jsin(-b_0)] + \cdots$$

$$X_k = A_k + B_k j$$

Fourier Series & Transform (3)

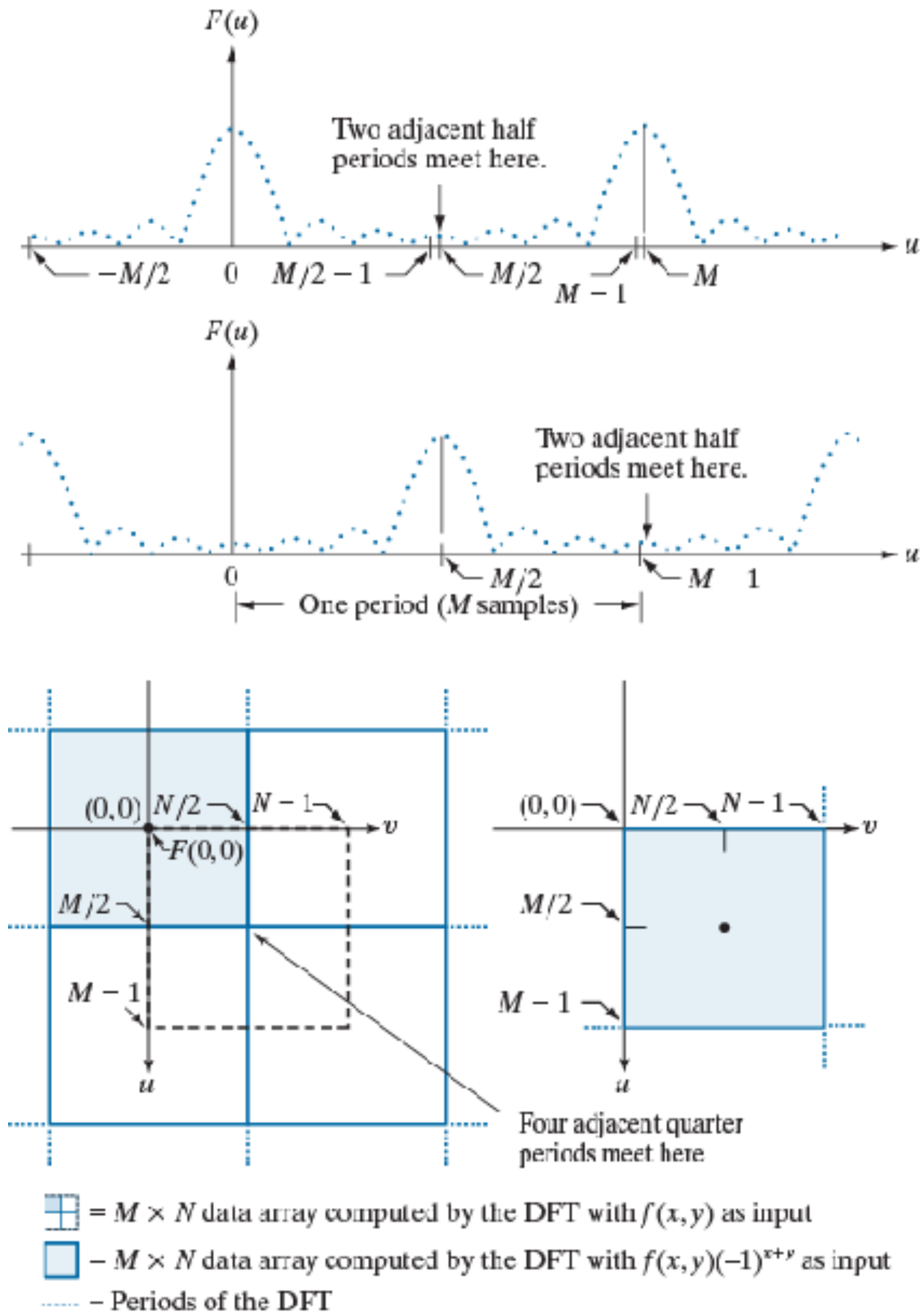




Discrete Fourier transform and inverse

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

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).

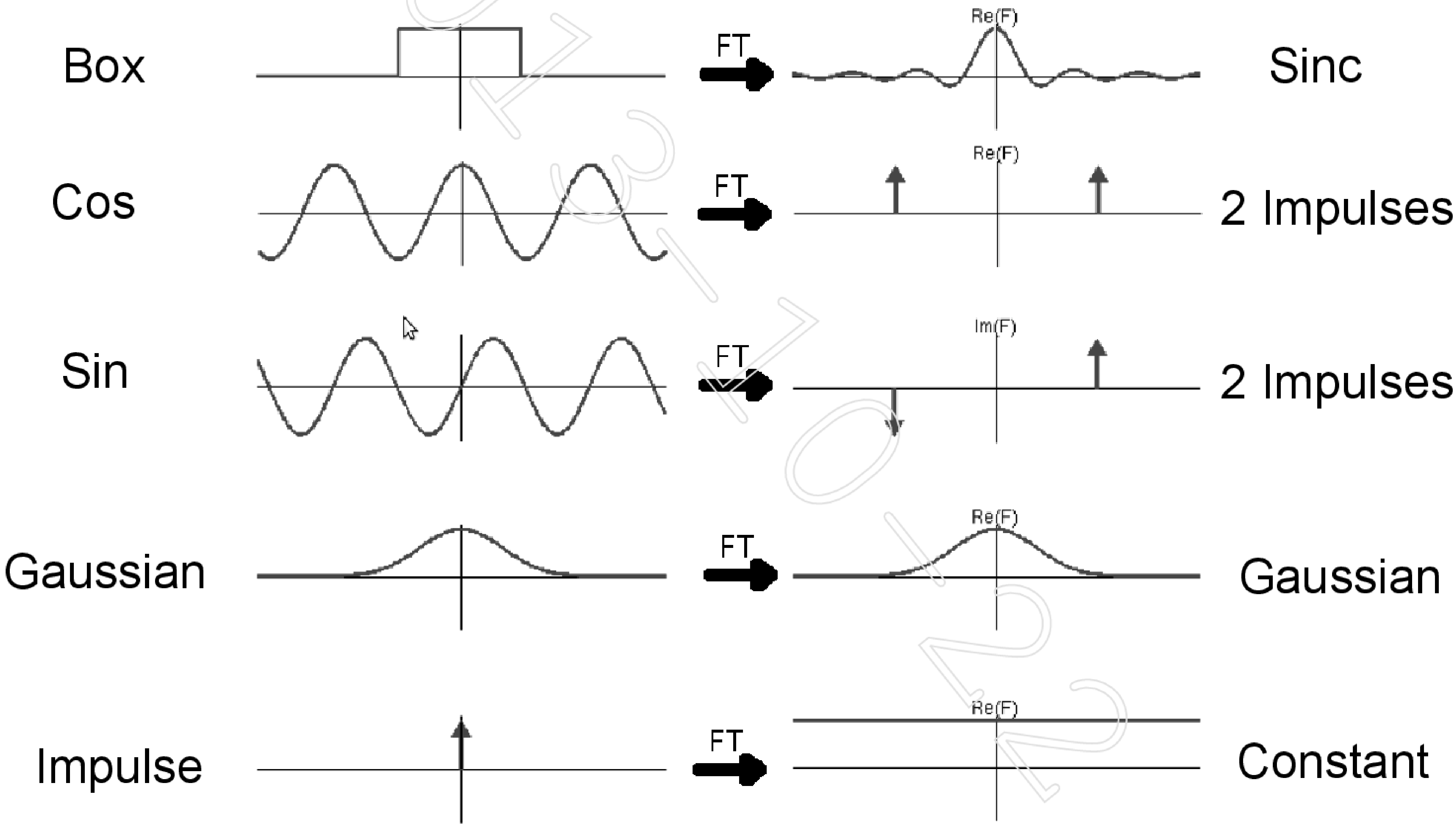
Forward and Inverse FT: Infinitely periodic

Centering / Shifting:

$$f(x)(-1)^x <=> F(u - M/2)$$

$$f(x,y)(-1)^{x+y} <=> F(u - M/2, v - N/2)$$

Some 1D Fourier Transforms



Fourier transform pairs

Convolution theorem

Fourier Transform and Convolution

Then $G(u) = \int_{-\infty}^{\infty} g(x) e^{-i2\pi ux} dx$ Let $g = f * h$

$$= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(\tau) h(x-\tau) e^{-i2\pi ux} d\tau dx$$

$$= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \left[f(\tau) e^{-i2\pi u\tau} d\tau \right] \left[h(x-\tau) e^{-i2\pi u(x-\tau)} dx \right]$$

$$= \int_{-\infty}^{\infty} \left[f(\tau) e^{-i2\pi u\tau} d\tau \right] \int_{-\infty}^{\infty} \left[h(x') e^{-i2\pi ux'} dx' \right]$$

$$= F(u) H(u)$$

Convolution in spatial domain

$\Leftrightarrow$ *Multiplication in frequency domain*

Fourier Transform and Convolution

Spatial Domain *(x)*		Frequency Domain *(u)*	
$g = f * h$	$\longleftrightarrow$	$G = F \cdot H$	**Filtering**
$g = f \cdot h$	$\longleftrightarrow$	$G = F * H$	**Sampling**

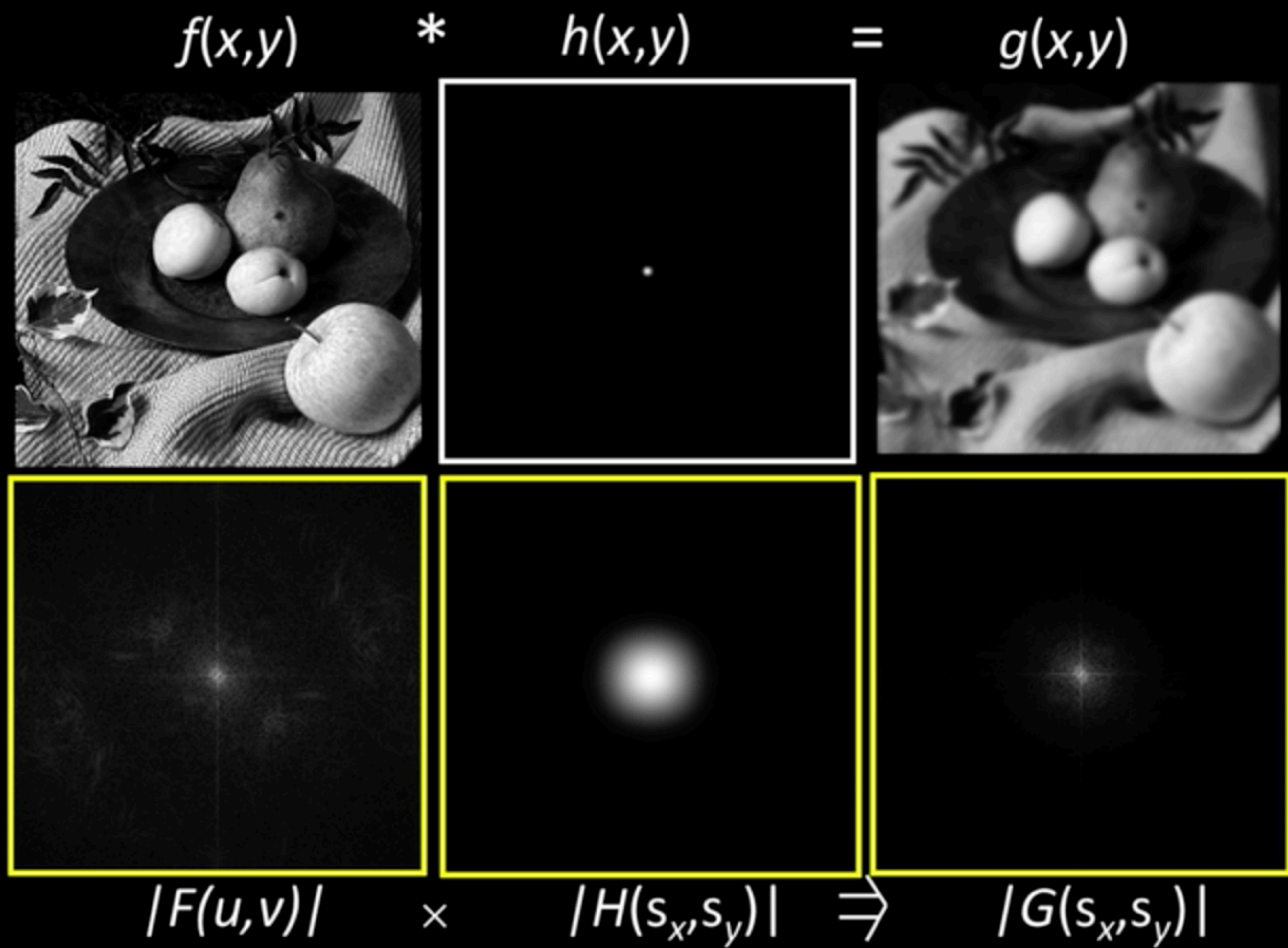
Filtering:

Fourier Transform and Convolution

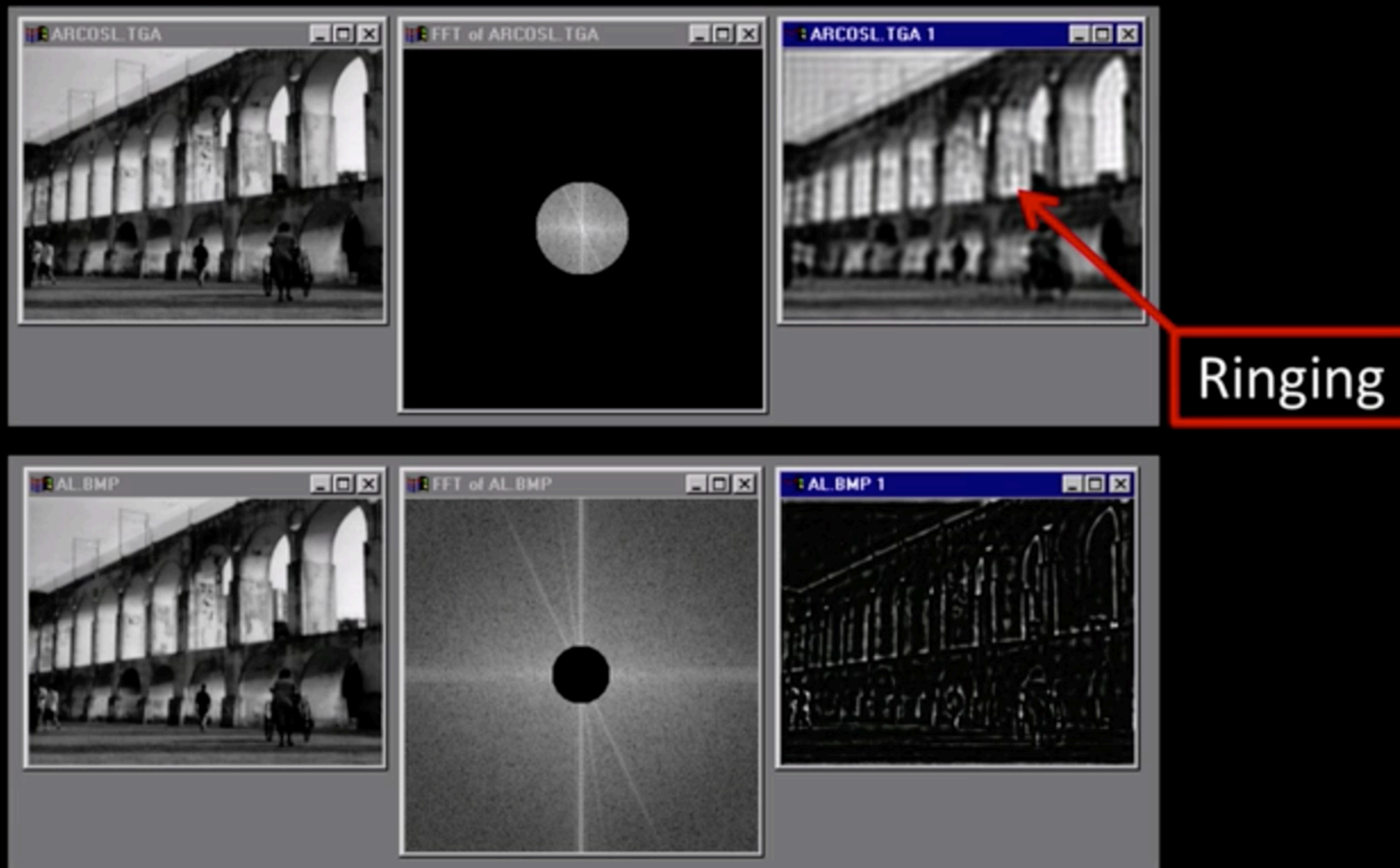
So, we can find *g(x)* by Fourier transform

$$
\begin{tikzcd}
g & = & f \arrow[d, "\mathrm{FT}"] & * & h \arrow[d, "\mathrm{FT}"] \\
G \arrow[u, "\mathrm{IFT}"] & = & F & \times & H
\end{tikzcd}
$$

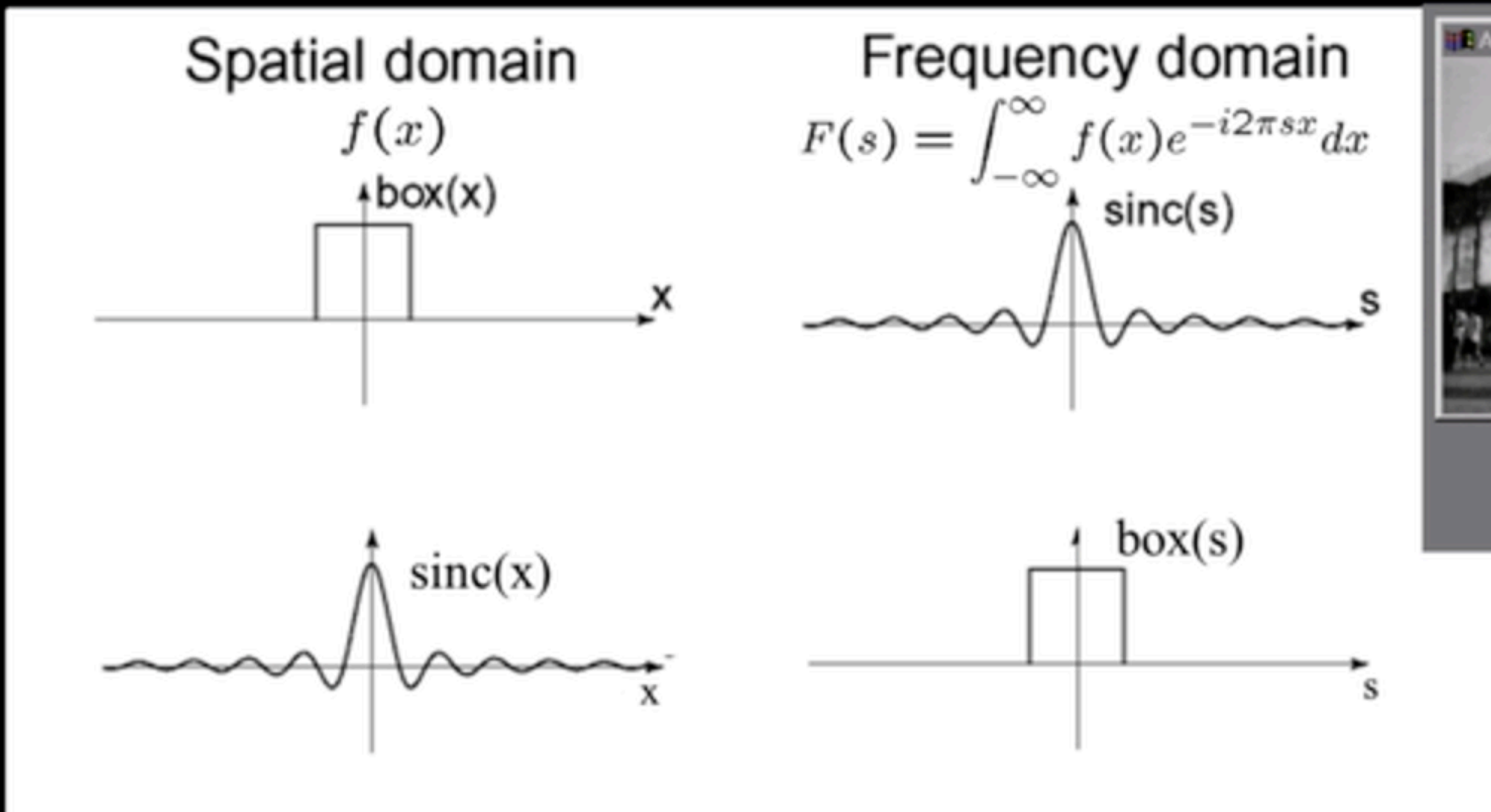
Or design H in freq. domain


f(x,y) * h(x,y) = g(x,y)
|F(u,v)| × |H(s_x,s_y)| ⇒ |G(s_x,s_y)|

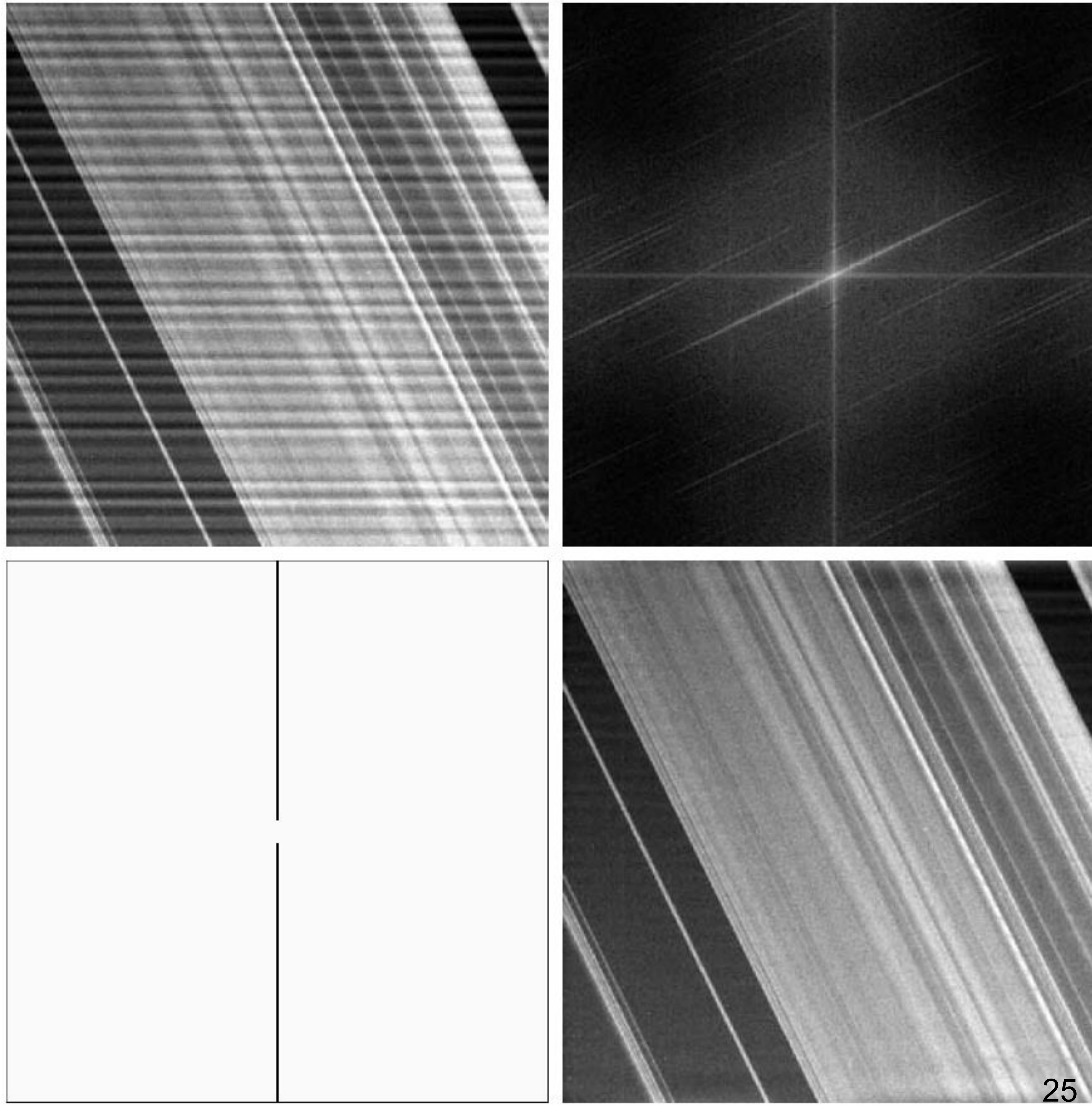
Low and High Pass filtering



Fourier Transform smoothing pairs

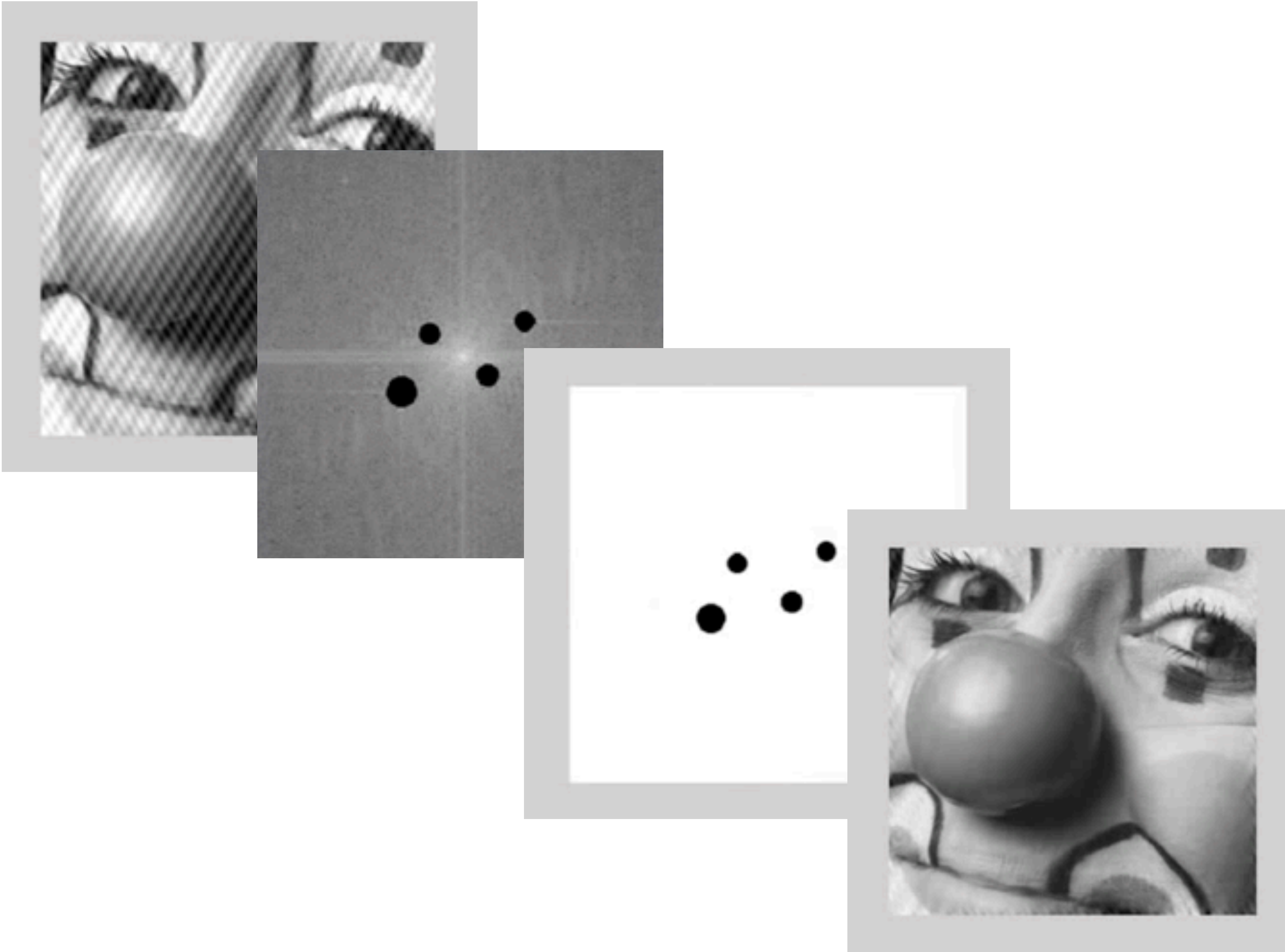


- Example of horizontal scan lines
- Create a notch of vertical lines in frequency domain

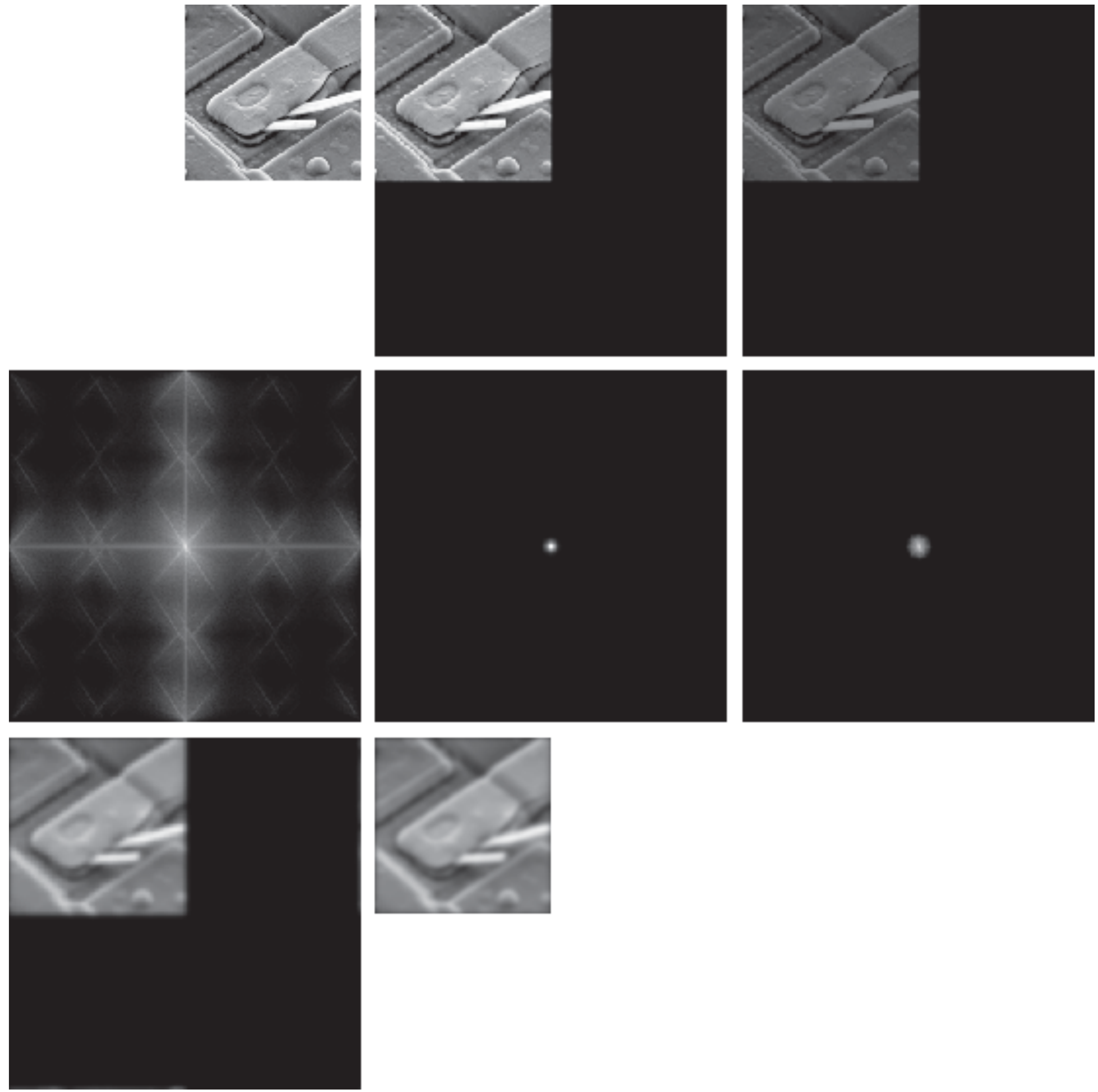


a	b
c	d

FIGURE 4.65 (a) 674 × 674 image of the 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. (d) Result of filtering. The thin black border in (c) was added for clarity; it is not part of the data. (Original image courtesy of Dr. Robert A. West,



Practical
(FT of image)



a	b	c
d	e	f
g	h	

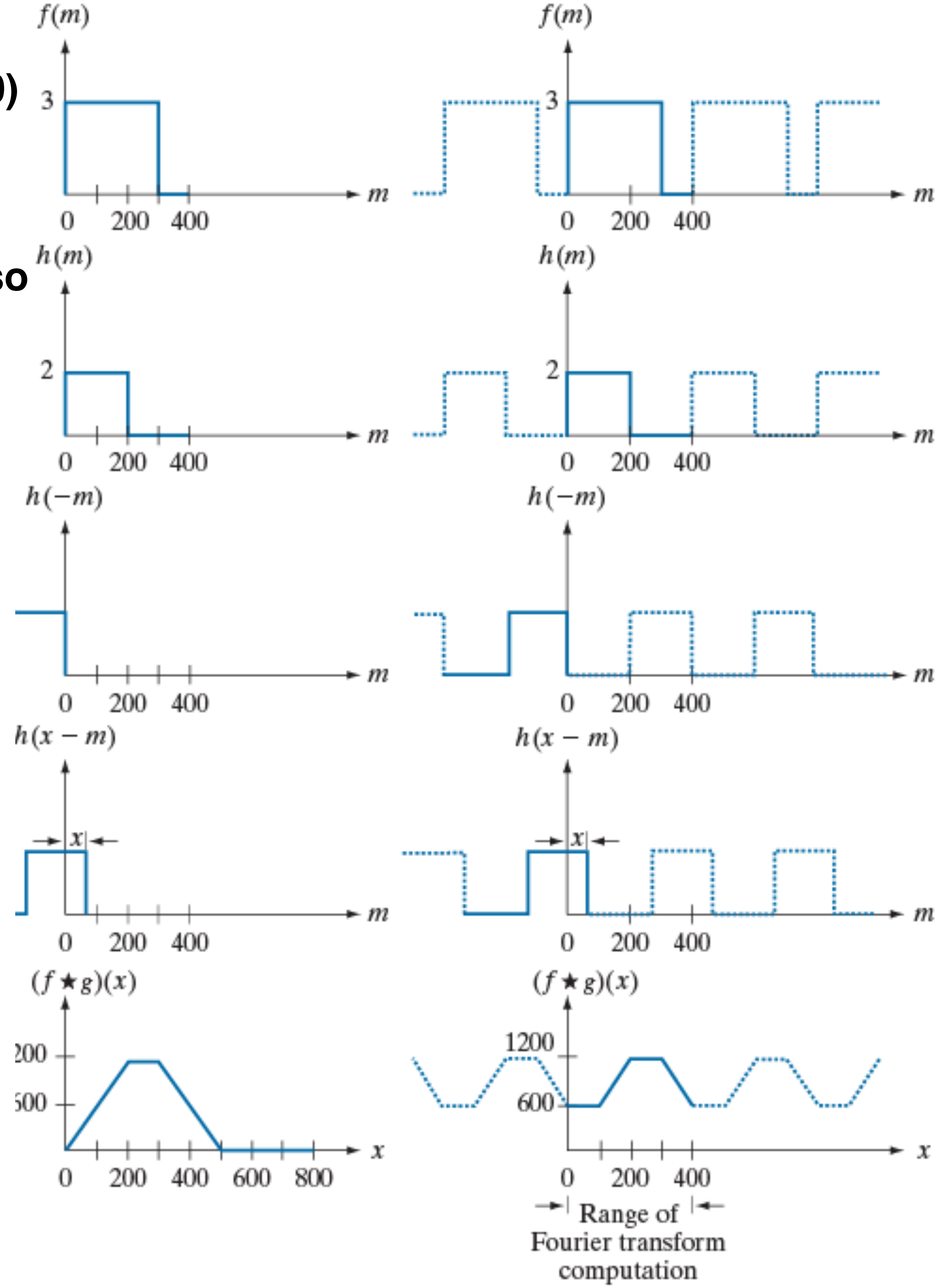
FIGURE 4.35
(a) An $M \times N$ image, f. (b) Padded image, f_p of size $P \times Q$. (c) Result of multiplying f_p by $(-1)^{x+y}$. (d) Spectrum of F. (e) Centered Gaussian lowpass filter transfer function, H, of size $P \times Q$. (f) Spectrum of the product HF. (g) Image g_p, the real part of the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

Circular / periodic convolution
Spatial and IFT approach same?
Flip, displacement, complete 0..799 (f/h: 400)

Functions and transforms implicit periodic
Same approach -> incorrect result (j vs. e)
Convolving periodic functions -> results also periodic.
Interfere: wraparound error

- **Erroneous 400 segment of periodic conv.**
- **Or one period of circular conv.**

Solution: Padding
P=2M & Q=2N



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.

Filtering in the Frequency domain

1. Given an input image f(x,y) of size M*N, obtain the padding parameters P = 2M, Q = 2N (f and h same size, avoid wraparound effects due to the periodicity implied by the DFT)

2. Form a padded f_p(x,y) of size P*Q by appending the necessary number of zeros to f(x,y)

3. Multiply f_p(x,y) 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*Q with cents at (P/2,Q/2). From product G(u,v) = H(u,v)F(u,v)

6. Obtain the processed image: g_p(x,y) = (real(IDFT(G)))(-1)^{x+y}

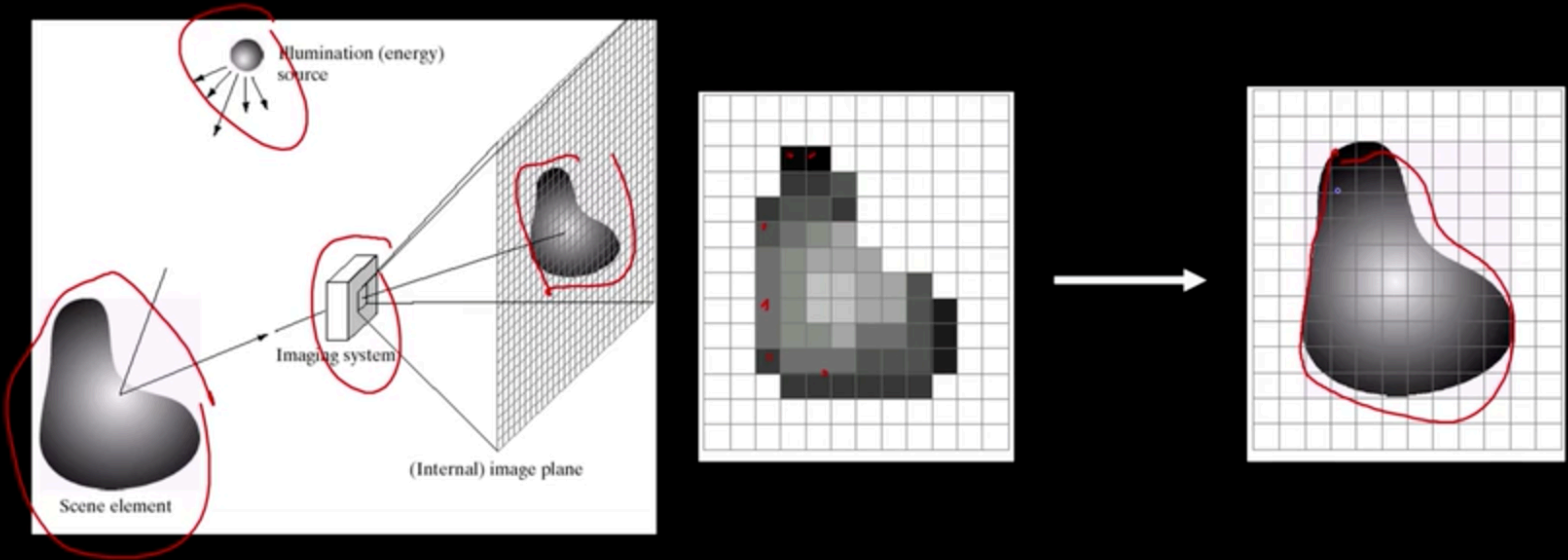
7. Obtain g(x,y) by extracting the M*N region from the top, left quadrant of g_p(x,y)

Sampling & Aliasing

Fourier Transform and Convolution

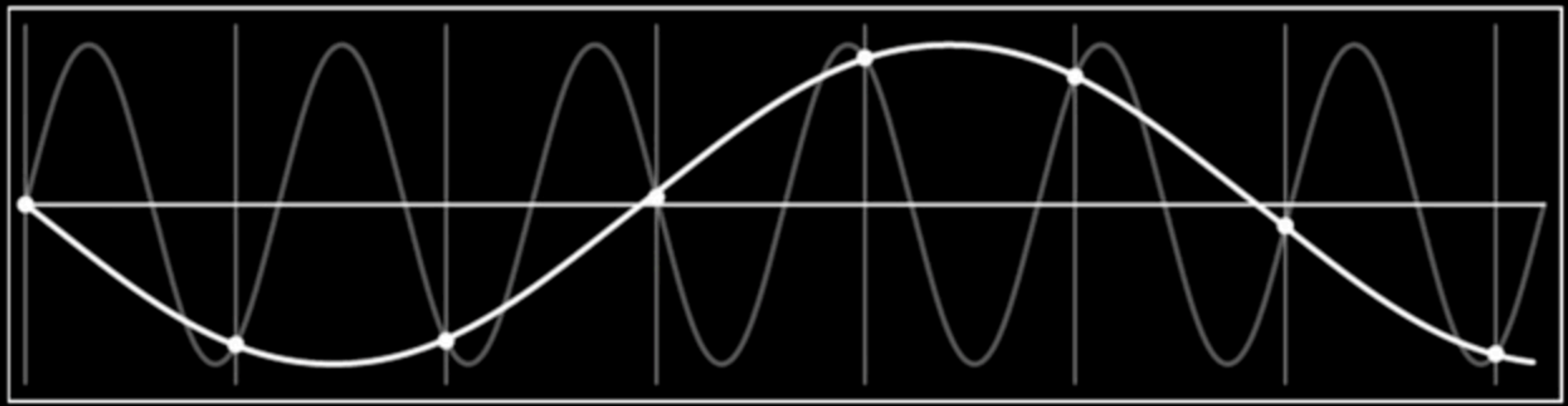
Spatial Domain *(x)*		Frequency Domain *(u)*
$g = f * h$	$\longleftrightarrow$	$G = F \cdot H$
$g = f \cdot h$	$\longleftrightarrow$	$G = F * H$

Sampling and Reconstruction



Undersampling

- Simple example: undersampling a sine wave
  - unsurprising result: information is lost
  - surprising result: indistinguishable from lower frequency

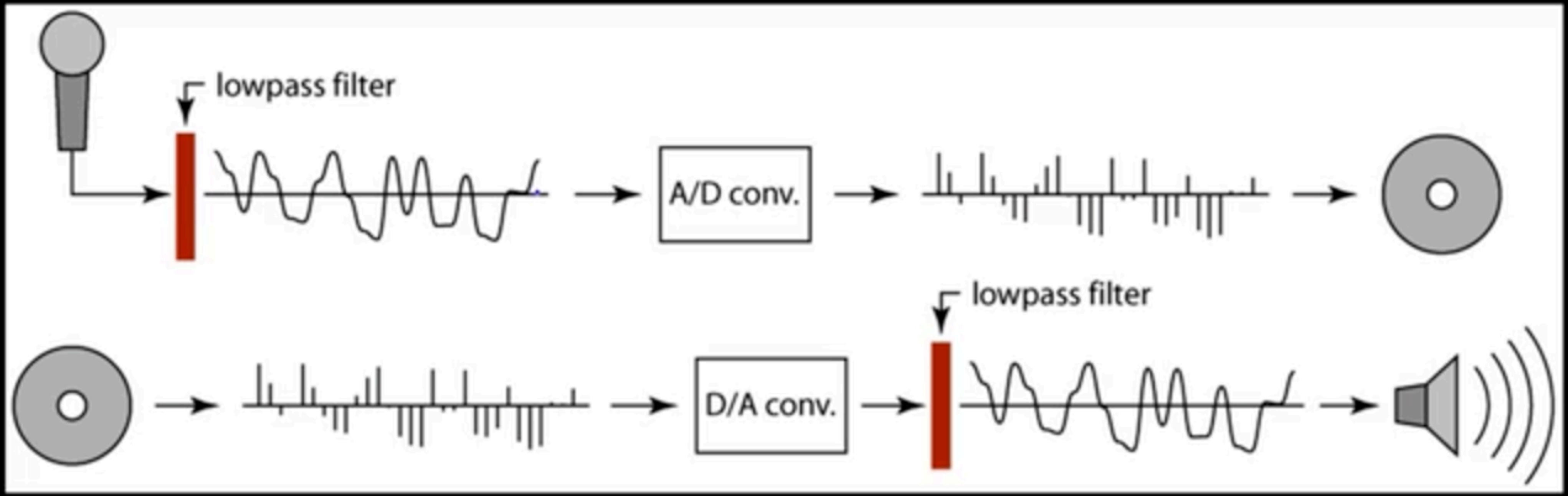


S. Marschner

Aliasing: Not able to reconstruct the true signal

Preventing aliasing

- Introduce *lowpass* filters:
  - remove high frequencies leaving only safe, low frequencies to be reconstructed



S. Marschner

Impulse Train

Define a *comb* function (impulse train) in 1D as follows

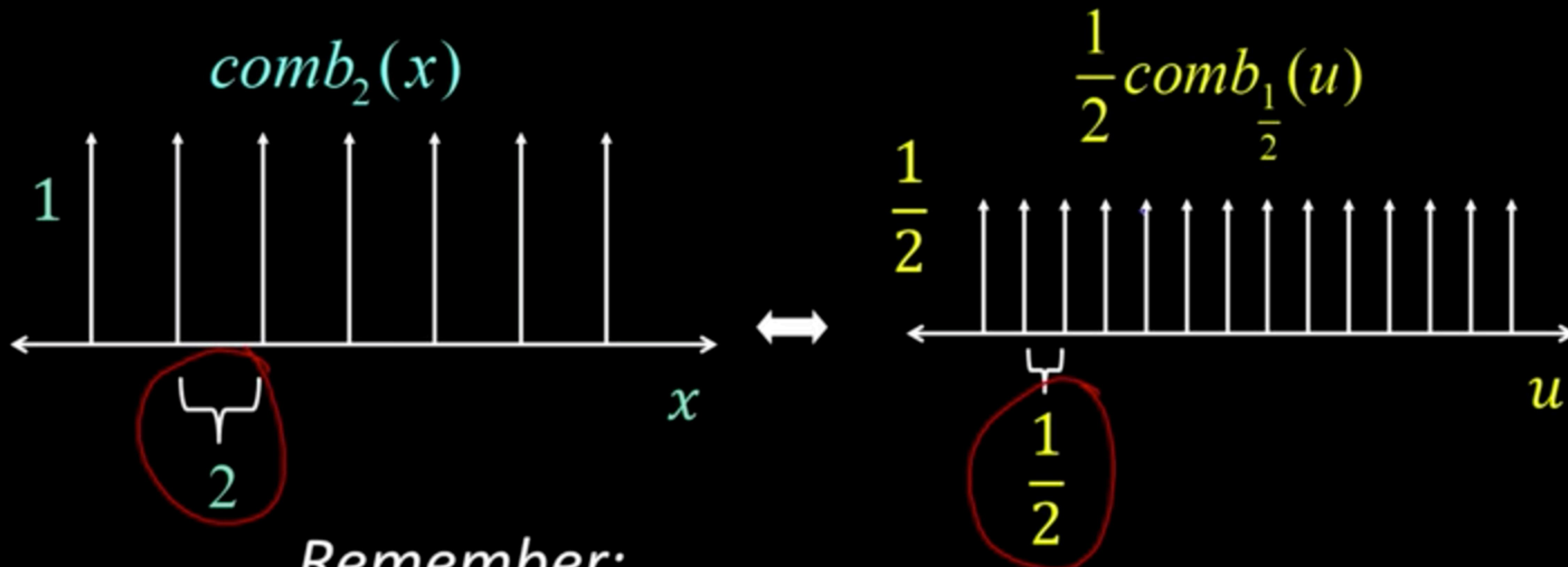
$$comb_M[x] = \sum_{k=-\infty}^{\infty} \delta[x - kM]$$

where *M* is an integer

$comb_2[x]$

B.K. Gunturk

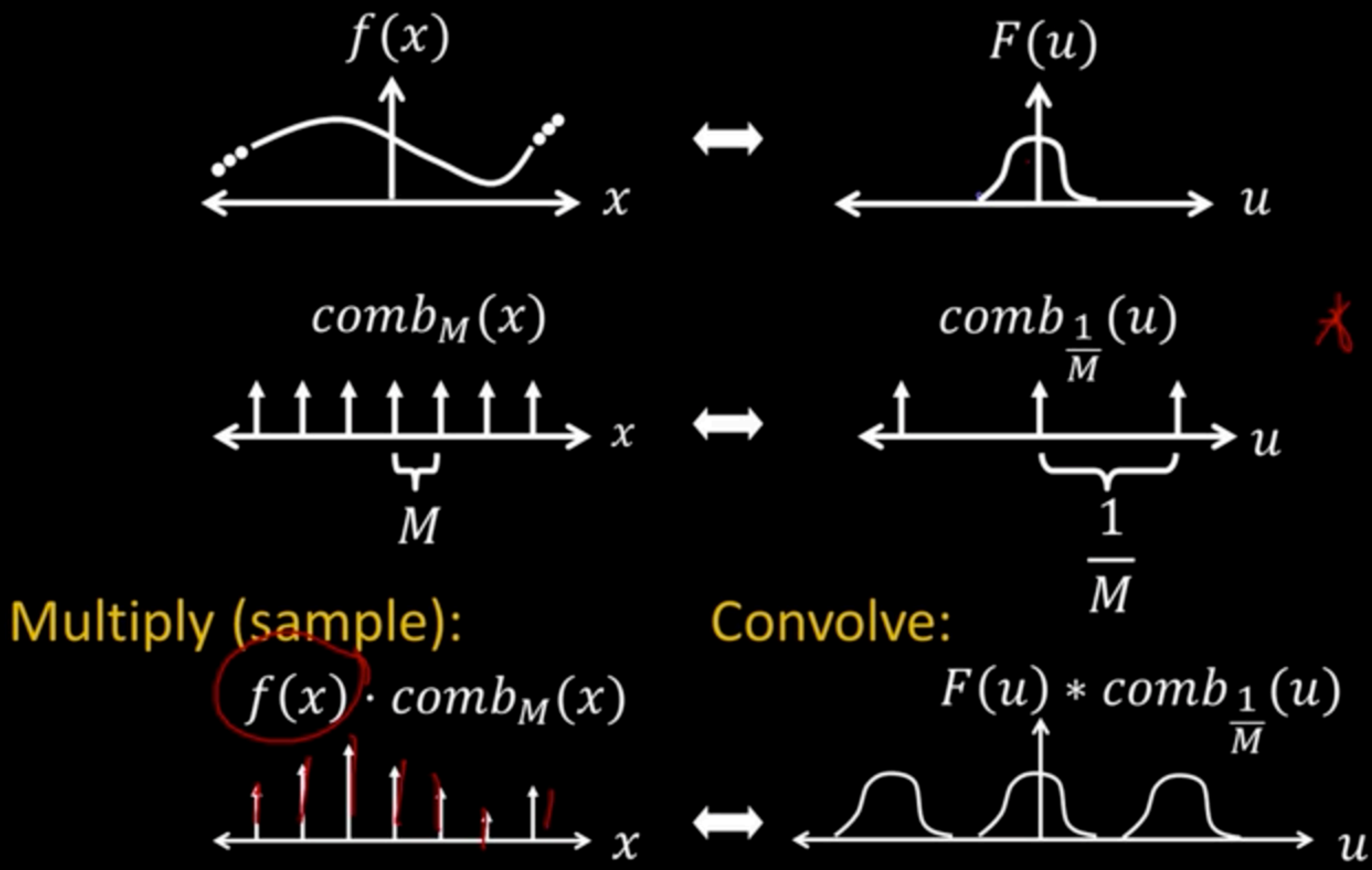
FT of Impulse Train in 1D



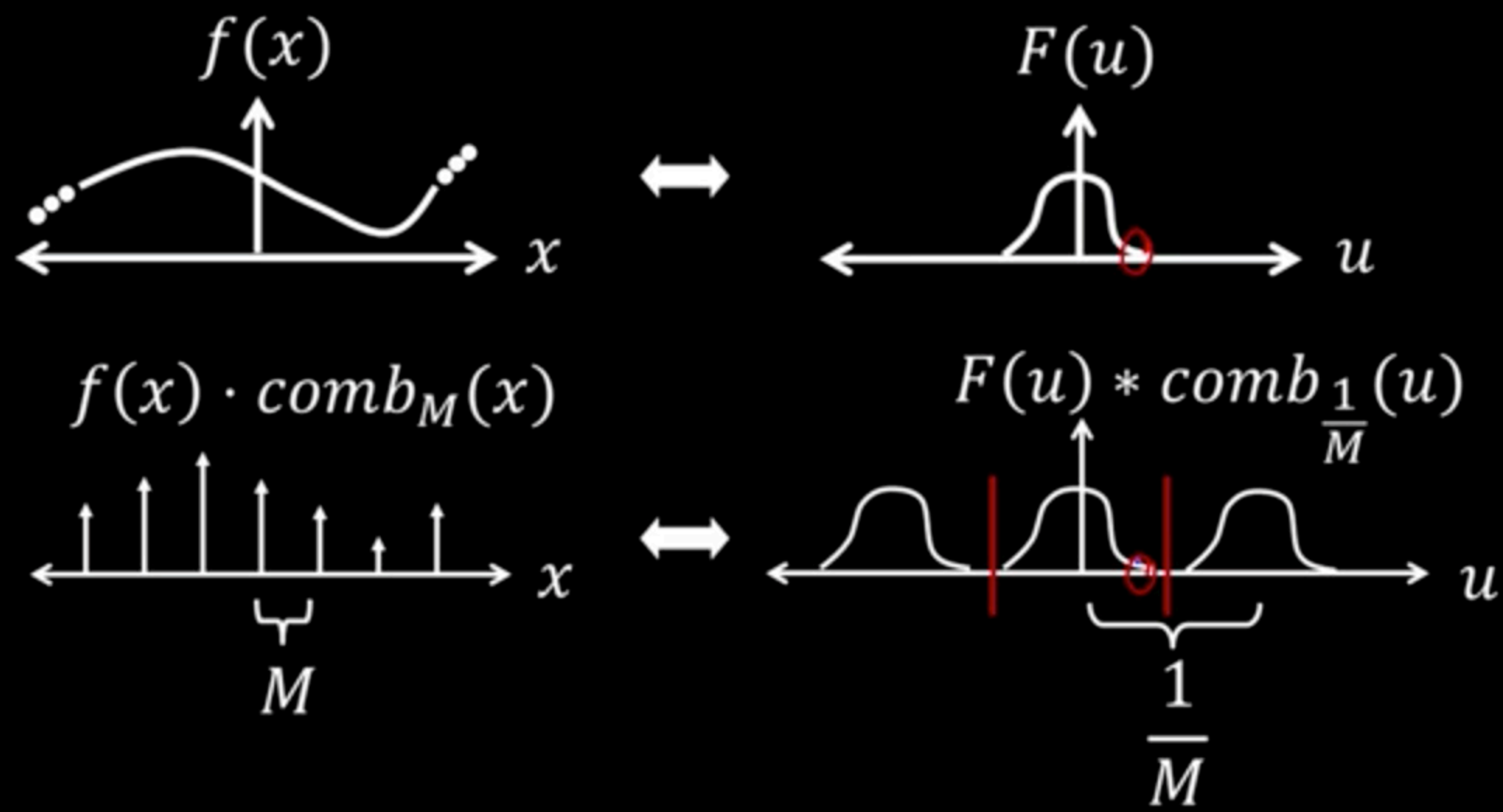
Remember:

Scaling $f(ax)$ $\quad \frac{1}{|a|} F\left(\frac{u}{a}\right)$

B.K. Gunturk


$f(x)$
$F(u)$
x
u
$comb_M(x)$
$comb_{\frac{1}{M}}(u)$
M
$\frac{1}{M}$
Multiply (sample):
$f(x) \cdot comb_M(x)$
Convolve:
$F(u) * comb_{\frac{1}{M}}(u)$

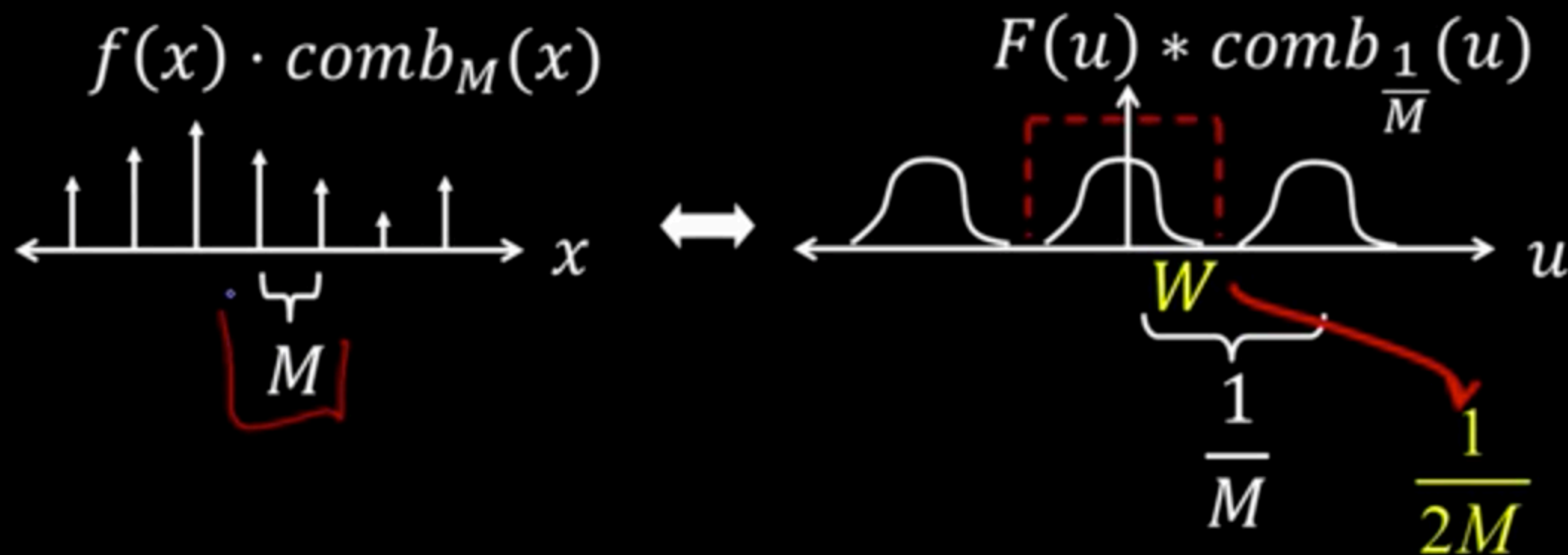
B.K. Gunturk



No "problem" if the maximum frequency of the signal is "small enough"

B.K. Gunturk

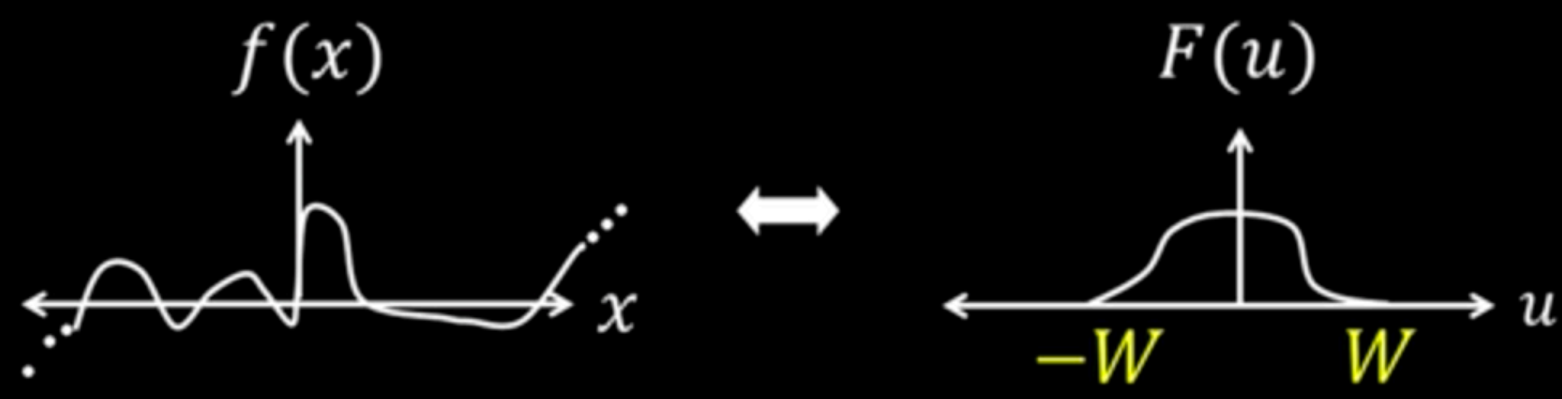
Sampling low frequency signal



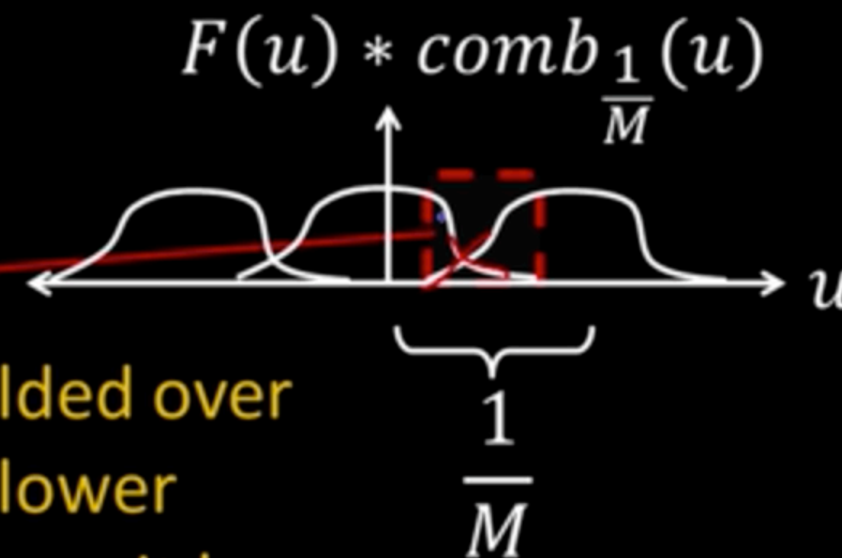
If there is no overlap, $W < \frac{1}{2M}$, the original signal can be recovered from its samples by low-pass filtering.

B.K. Gunturk

Sampling high frequency signal

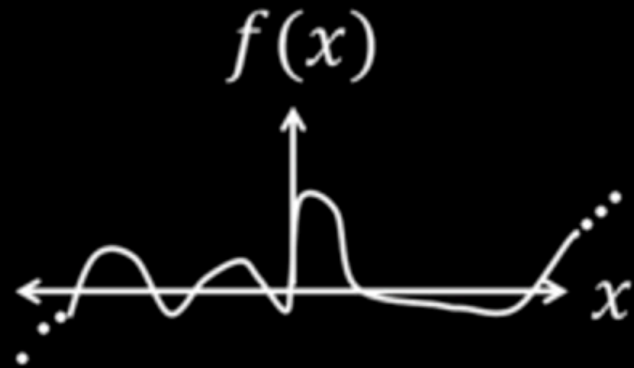


$< f(x) \cdot comb_M(x) >$ ⟺

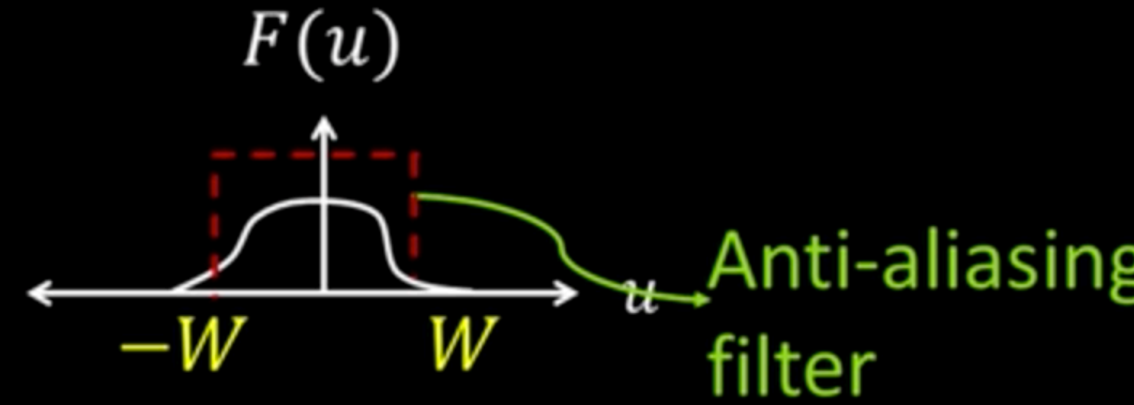


Overlap: The high frequency energy is folded over into low frequency. It is “aliasing” as lower frequency energy. And you cannot fix it once it has happened.

Sampling high frequency signal



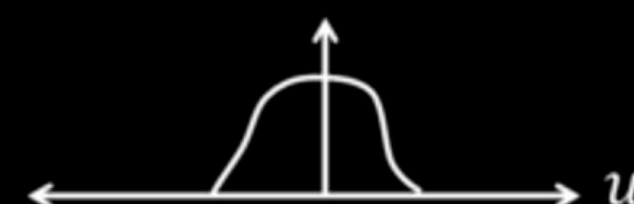
$\Leftrightarrow$



$$f(x) * h(x)$$

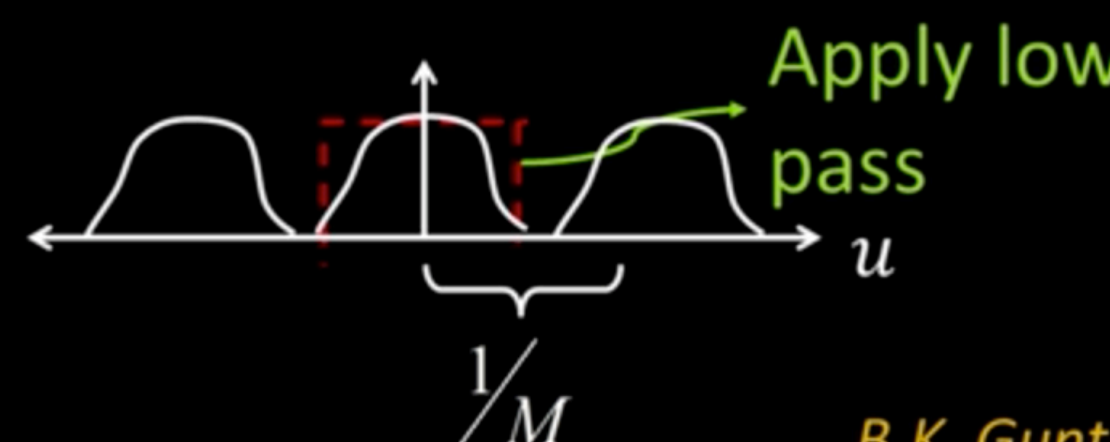
Anti-aliasing filter

$\Leftrightarrow$



$$[f(x) * h(x)]comb_M(x)$$

$\Leftrightarrow$



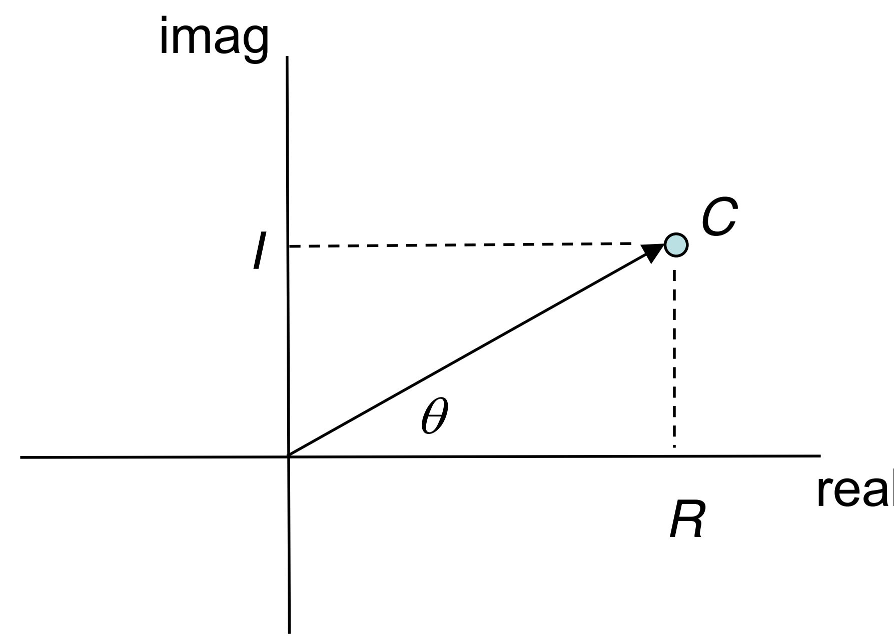
B.K. Gunturk

Some more details

(repetitions, alternative views, more details, additional info)

Part 1: 1D

Complex Numbers



- A Fourier series is a sum of sines and cosines
- We can simplify its representation by using complex numbers

- Let *j* be the imaginary number

$$j = \sqrt{-1}$$

- A complex number is defined as

$$C = R + jI$$

- In polar coordinates

$$C = |C|(\cos\theta + j\sin\theta)$$

- Euler's formula

$$e^{j\theta} = \cos\theta + j\sin\theta, \text{ so } C = |C|e^{j\theta}$$

- Complex conjugate

$$C^* = R - jI$$

- Matlab functions
  - `complex, conj, abs, angle`
- Fourier series

$$f(t) = \sum_{n=-\infty}^{\infty} c_n e^{j2\pi n/T}$$

Complex Numbers / Complex Functions

Impulse function

- The "impulse" function is helpful to understanding Fourier transforms
- It is defined as

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

  – where we constrain it such that

$$\int_{-\infty}^{\infty} \delta(t)\,dt = 1$$

Discrete equivalents:

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

$$\sum_{x=-\infty}^{\infty} \delta(x) = 1$$

$$\sum_{x=-\infty}^{\infty} f(x)\delta(x - x_0) = f(x_0)$$

- Sifting property

$$\int_{-\infty}^{\infty} f(t)\delta(t)\,dt = f(0), \quad \text{also} \int_{-\infty}^{\infty} f(t)\delta(t - t_0)\,dt = f(t_0)$$

Fourier Transform

- The 1D Fourier transform (continuous)

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

x: spatial domain

- Inverse Fourier transform

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

u: frequency domain

- We often write

$$\mathfrak{F}\left\{ f(x) \right\} = F(u)$$

$$\mathfrak{F}^{-1}\left\{ F(u) \right\} = f(x)$$

- *f(x)* and *F(u)* are called Fourier transform pairs

$$f(x) \leftrightarrow F(u)$$

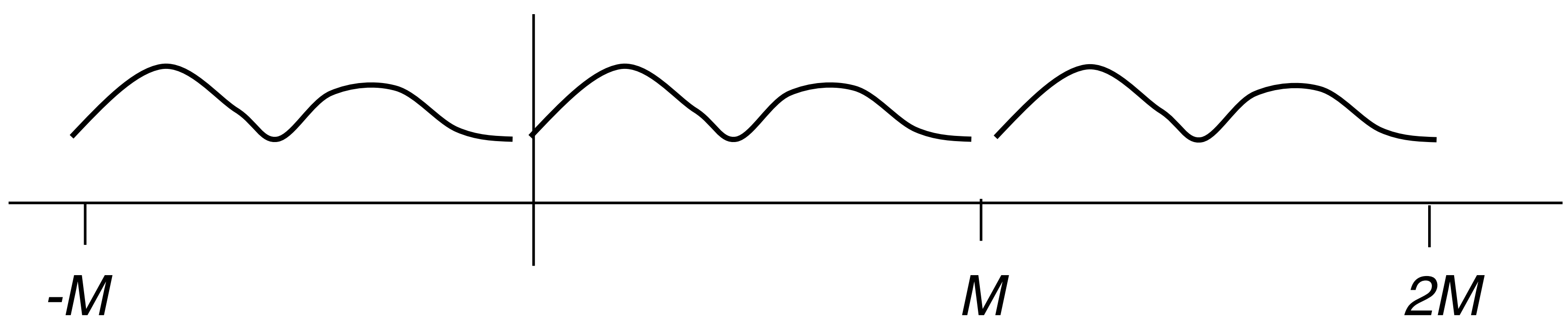
Discrete Fourier Transform

- Discrete Fourier transform and inverse

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

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

- We consider f(x) and F(u) to be periodic in the discrete domain



What is F(0)?

A Note on Units

- Units
  - Spatial units (x): length (e.g., x has units of meters)
  - Frequency (u): 1 / length (e.g., u has units of meters^{-1})

- We have M samples of *f(x)* taken at integer spacing:

  $0, 1, \ldots, M\text{-}1$

- We can consider the samples to be taken at intervals of Δx:

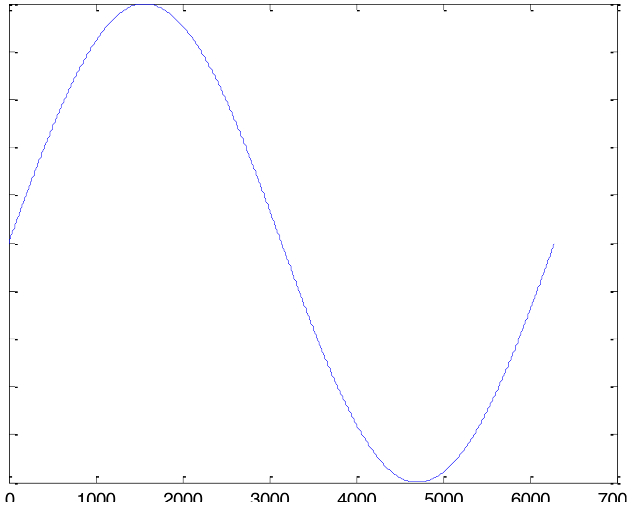
  $f(0), f(\Delta x), f(2\Delta x), \ldots, f((M\text{-}1)\Delta x)$

- Similarly *F(u)* is sampled at:

  $F(0), F(\Delta u), F(2\Delta u), \ldots, F((M\text{-}1)\, \Delta u)$

- The longest possible wavelength is $M\Delta x$

  => corresponds to min freq of $\Delta u = 1/(M\Delta x)$



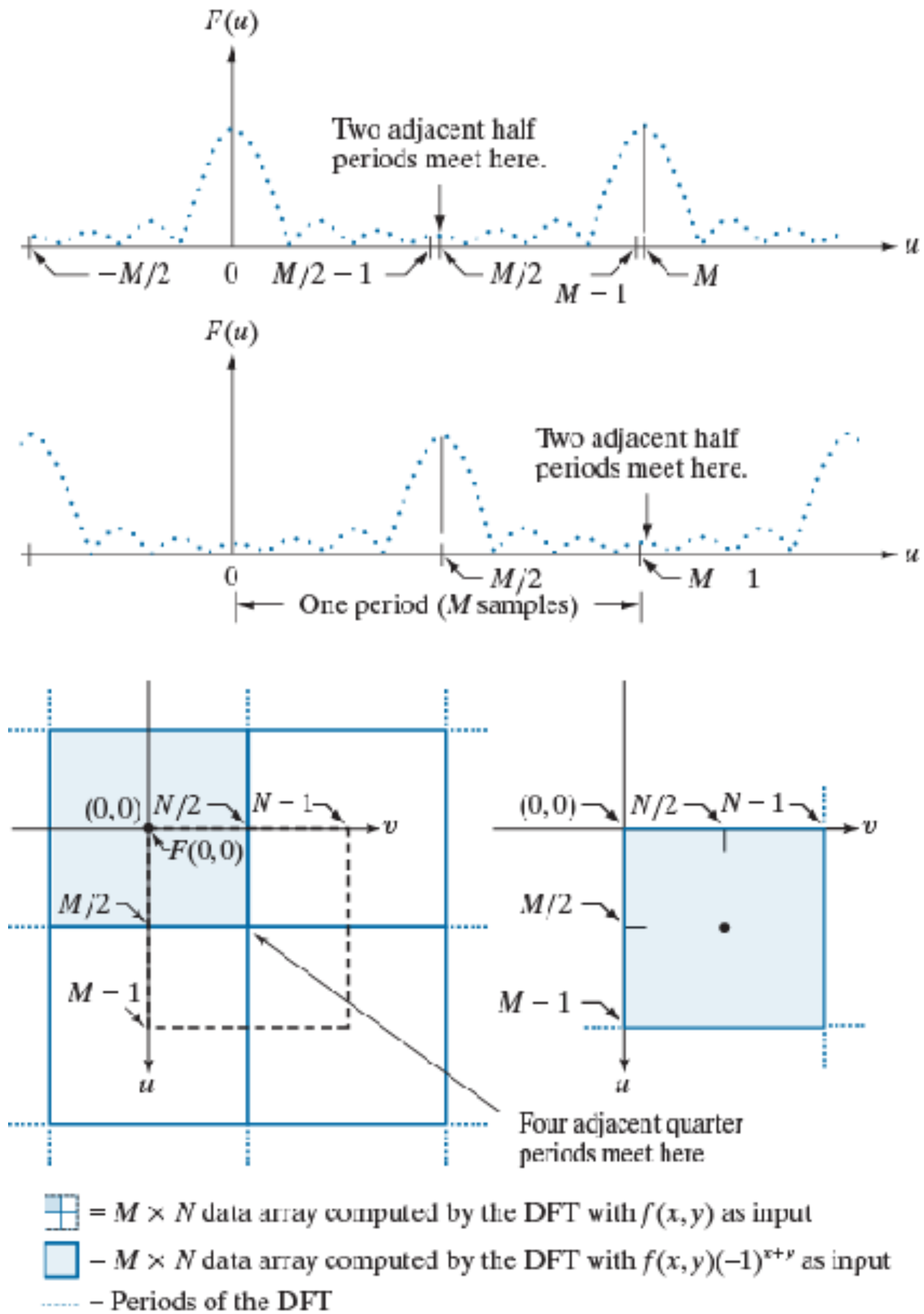
Discrete Fourier transform and inverse

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

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).



Forward and Inverse FT: Infinitely periodic

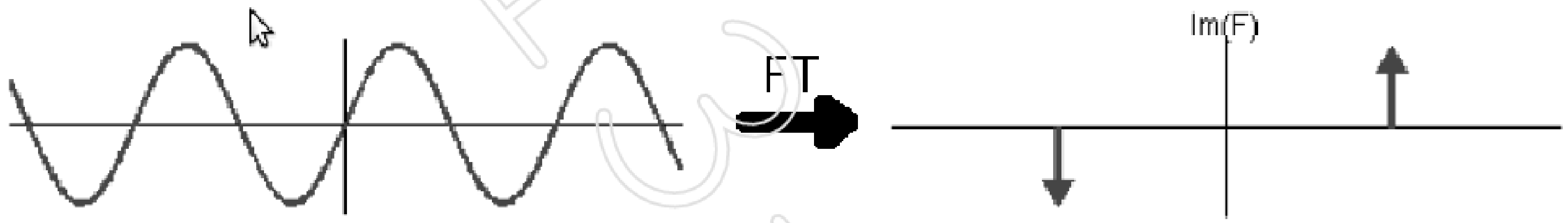
Centering / Shifting:

$$f(x)(-1)^x <=> F(u - M/2)$$

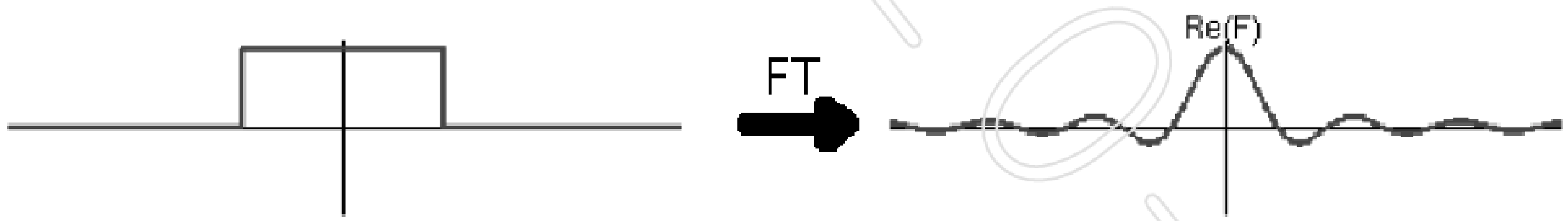
$$f(x, y)(-1)^{x+y} <=> F(u - M/2, v - N/2)$$

Symmetry-Relations

- An odd function $\rightarrow$ odd symmetry, purely imaginary Fourier Transform



- An even function $\rightarrow$ even symmetry, purely real Fourier Transform



- A mixed function $\rightarrow$ conjugate symmetric, complex valued Fourier Transform

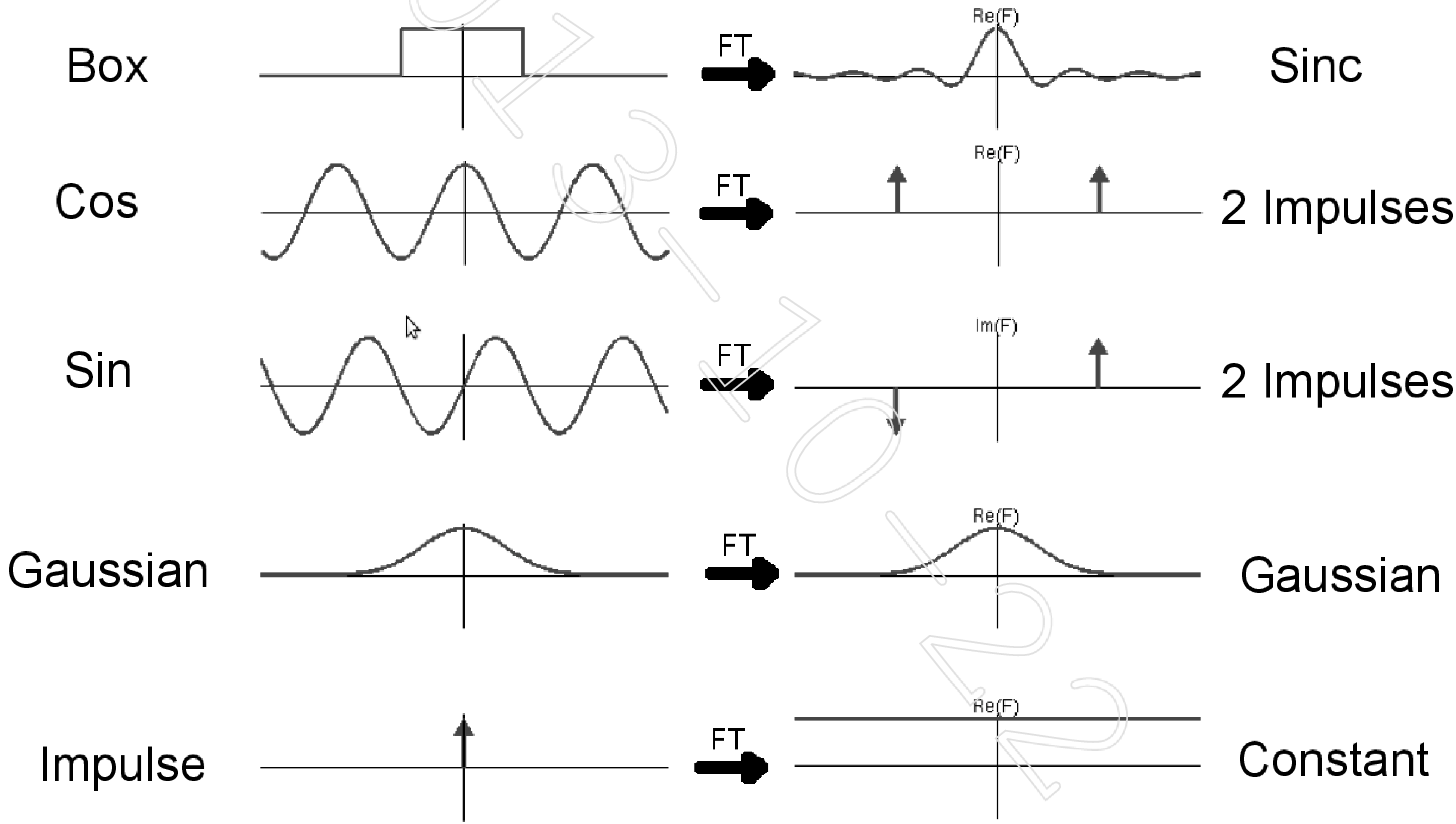
$$F(u) = F^*(-u)$$

"Complex conjugated"

$$c = a + i\,b$$

$$c^* = a - i\,b$$

Some 1D Fourier Transforms



Convolution Theorem

- Convolution in one domain is equivalent to multiplication in the other domain

$$h(x) * f(x) \Leftrightarrow H(u)F(u)$$

convolution *point-by-point multiplication*

$$h(x)f(x) \Leftrightarrow H(u) * F(u)$$

point-by-point multiplication *convolution*

- Similarly for correlation, except that we have complex conjugate

$$h(x) \otimes f(x) \Leftrightarrow H^*(u)F(u)$$

$$h^*(x)f(x) \Leftrightarrow H(u) \otimes F(u)$$

Complex conjugate

$$C^* = R - jI$$

- As we will see a little later, this is very useful for implementing large filters

Understanding Sampling and Aliasing

- We often subsample a signal
  - When we originally digitize it
  - When we shrink it
- We can reconstruct the signal exactly from the samples if the samples are “dense” enough
- The sampling theorem says that the sampling rate must be more than twice the maximum frequency of the input signal (this is the “Nyquist rate”)
- If the sampling rate is lower, we can get errors in the reconstructed signal (called “aliasing”)

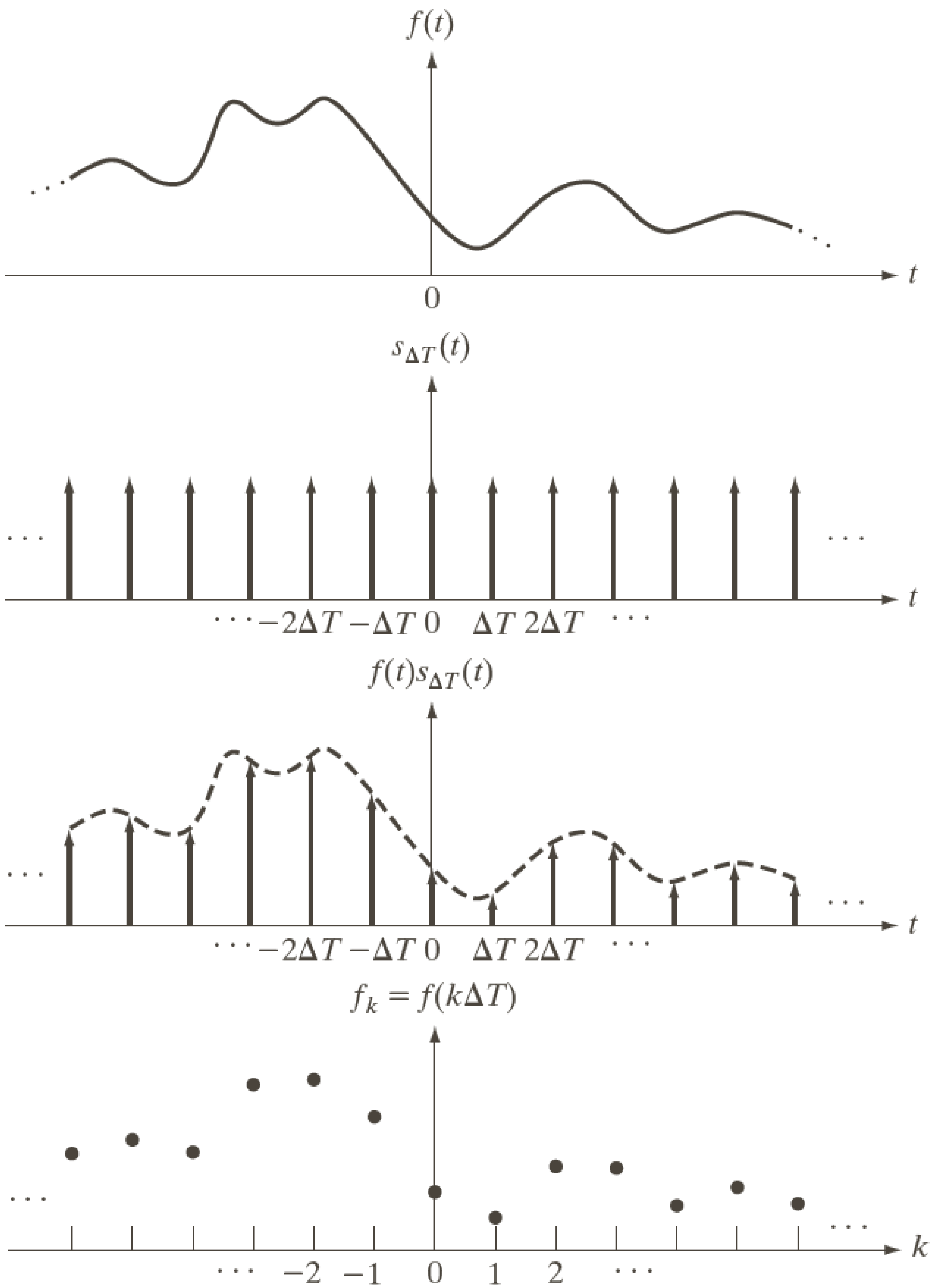
Sampling

- Sampling of a continuous function f(t) can be modeled by multiplying it with an impulse train

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

$$\tilde{f}(t) = f(t)\, s_{\Delta T}(t)$$

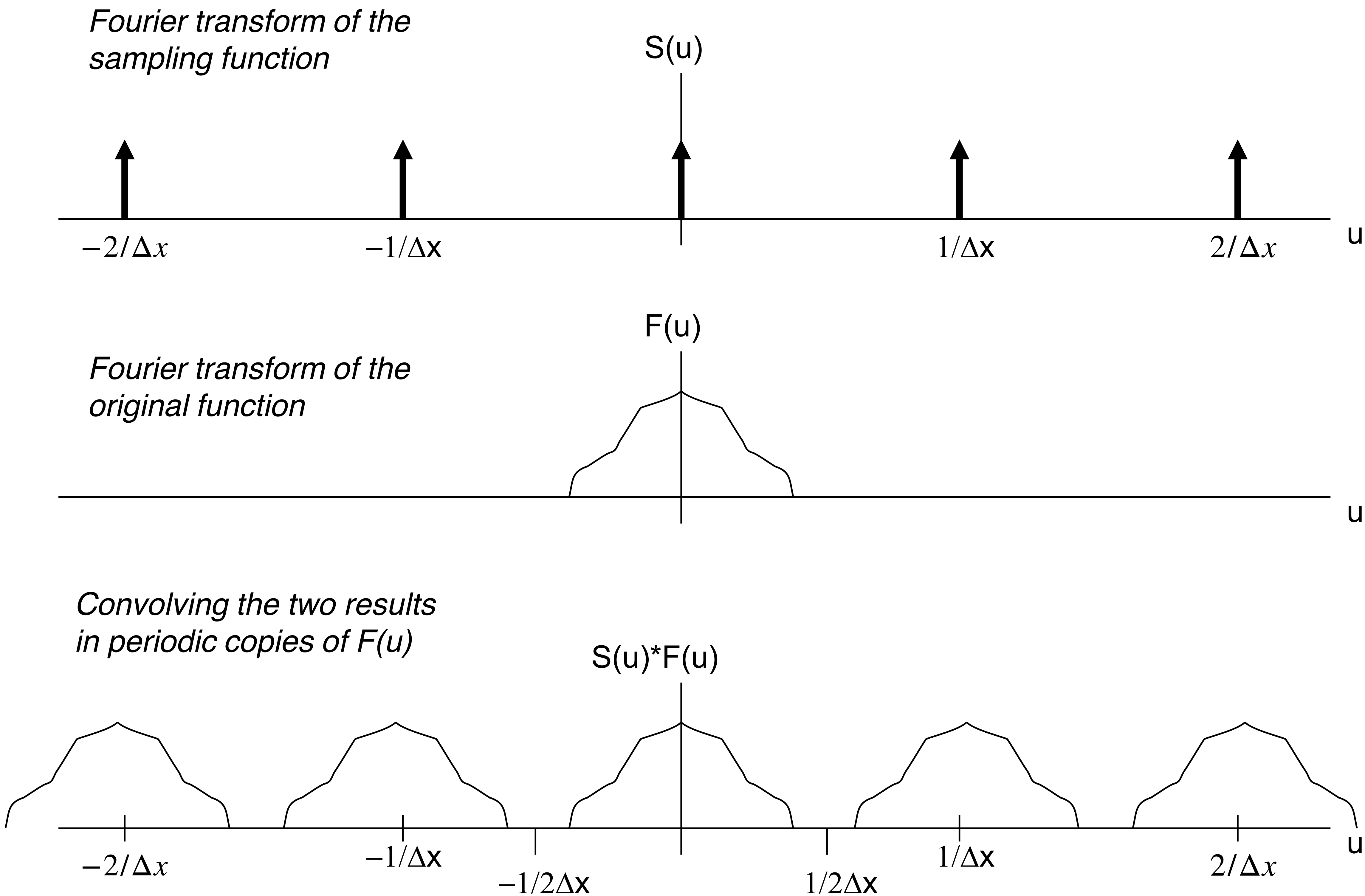
$$= \sum_{n=-\infty}^{\infty} f(t)\delta(t - n\Delta T)$$



a
b
c
d

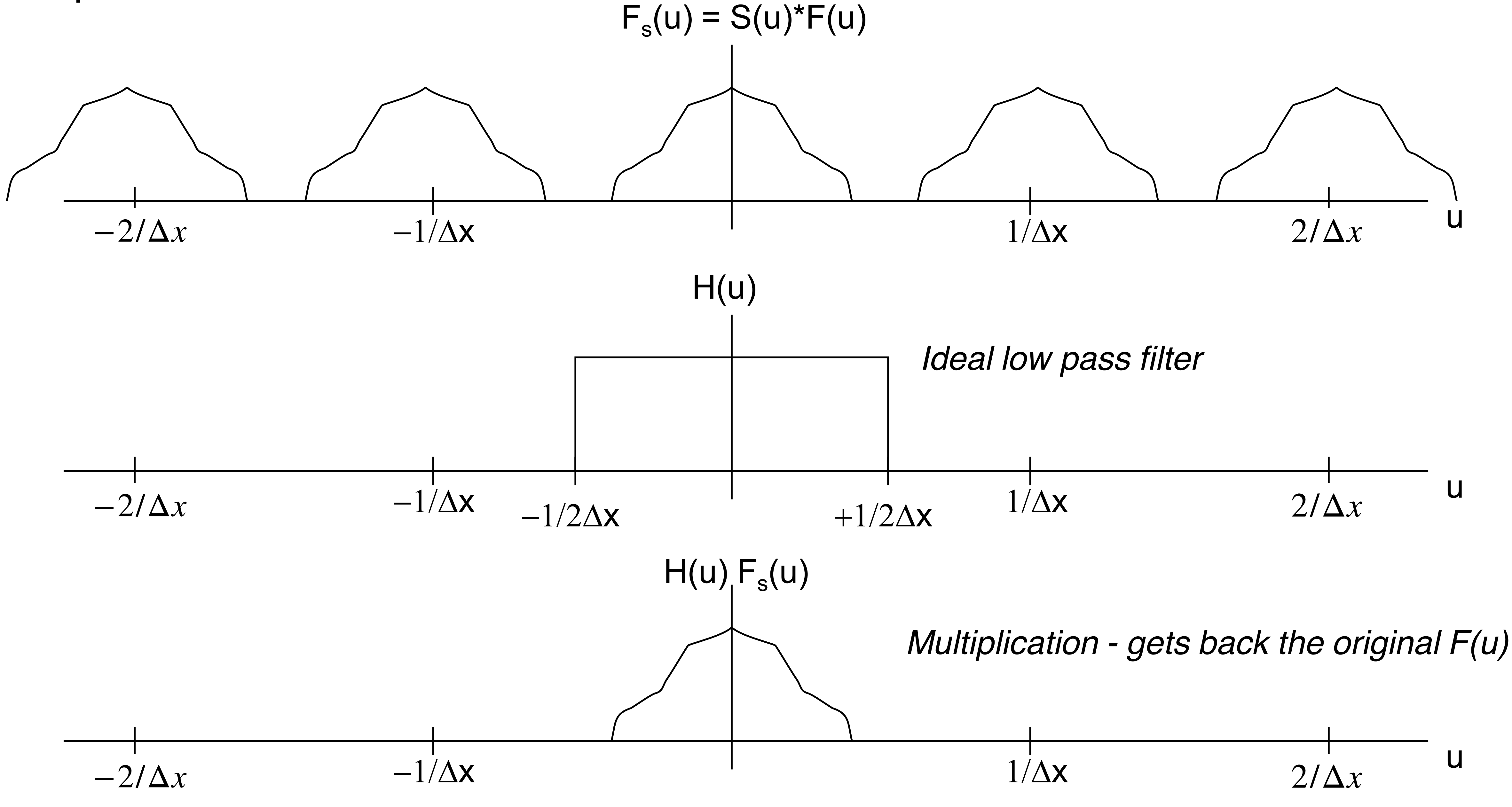
FIGURE 4.5
(a) A continuous function. (b) Train of impulses used to model the sampling process. (c) Sampled function formed as the product of (a) and (b). (d) Sample values obtained by integration and using the sifting property of the impulse. (The dashed line in (c) is shown for reference. It is not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

Sampling


Fourier transform of the sampling function
S(u)
u
$-2/\Delta x$
$-1/\Delta x$
$1/\Delta x$
$2/\Delta x$
F(u)
Fourier transform of the original function
u
Convolving the two results in periodic copies of F(u)
S(u)*F(u)
u
$-2/\Delta x$
$-1/\Delta x$
$-1/2\Delta x$
$1/2\Delta x$
$1/\Delta x$
$2/\Delta x$

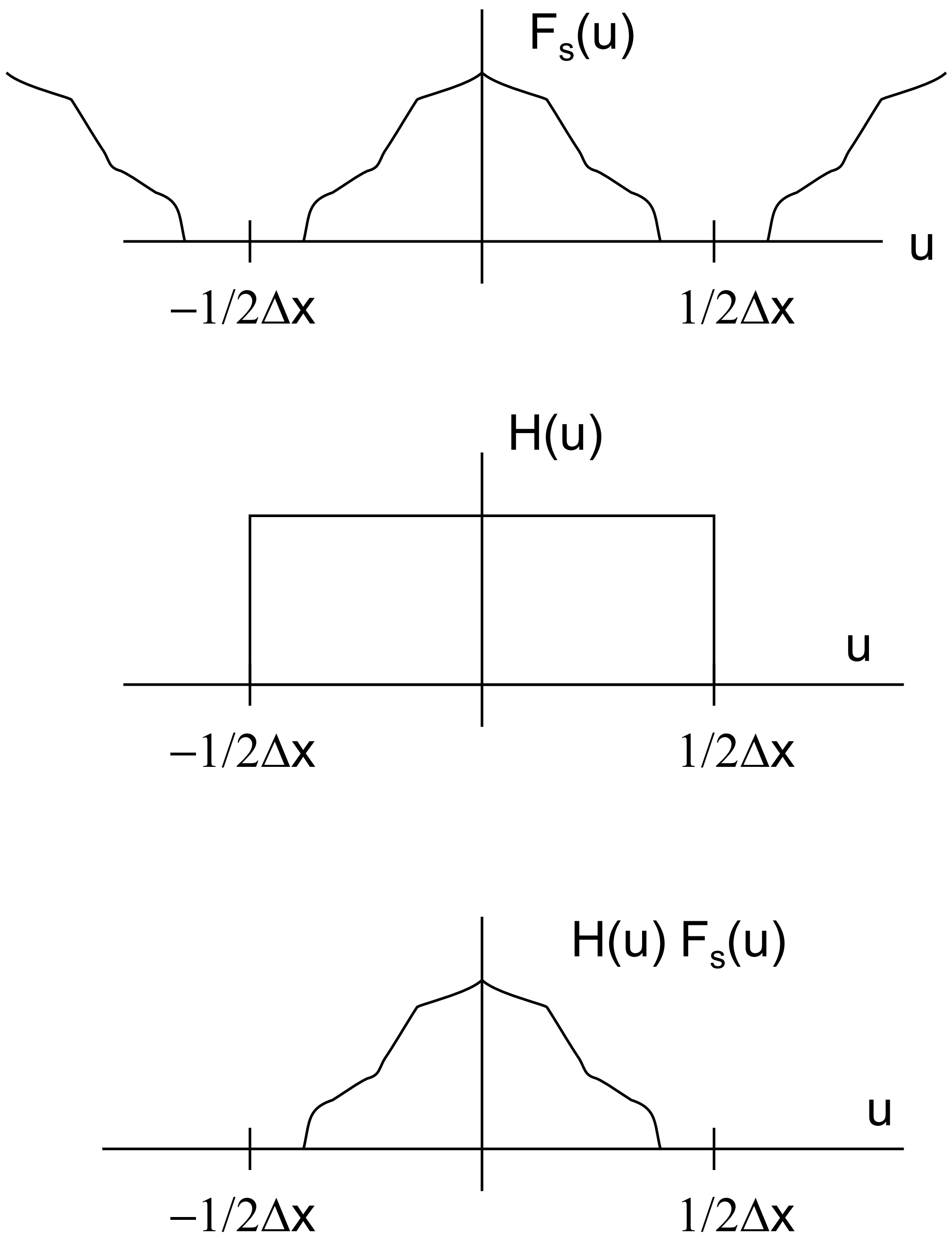
Reconstruction

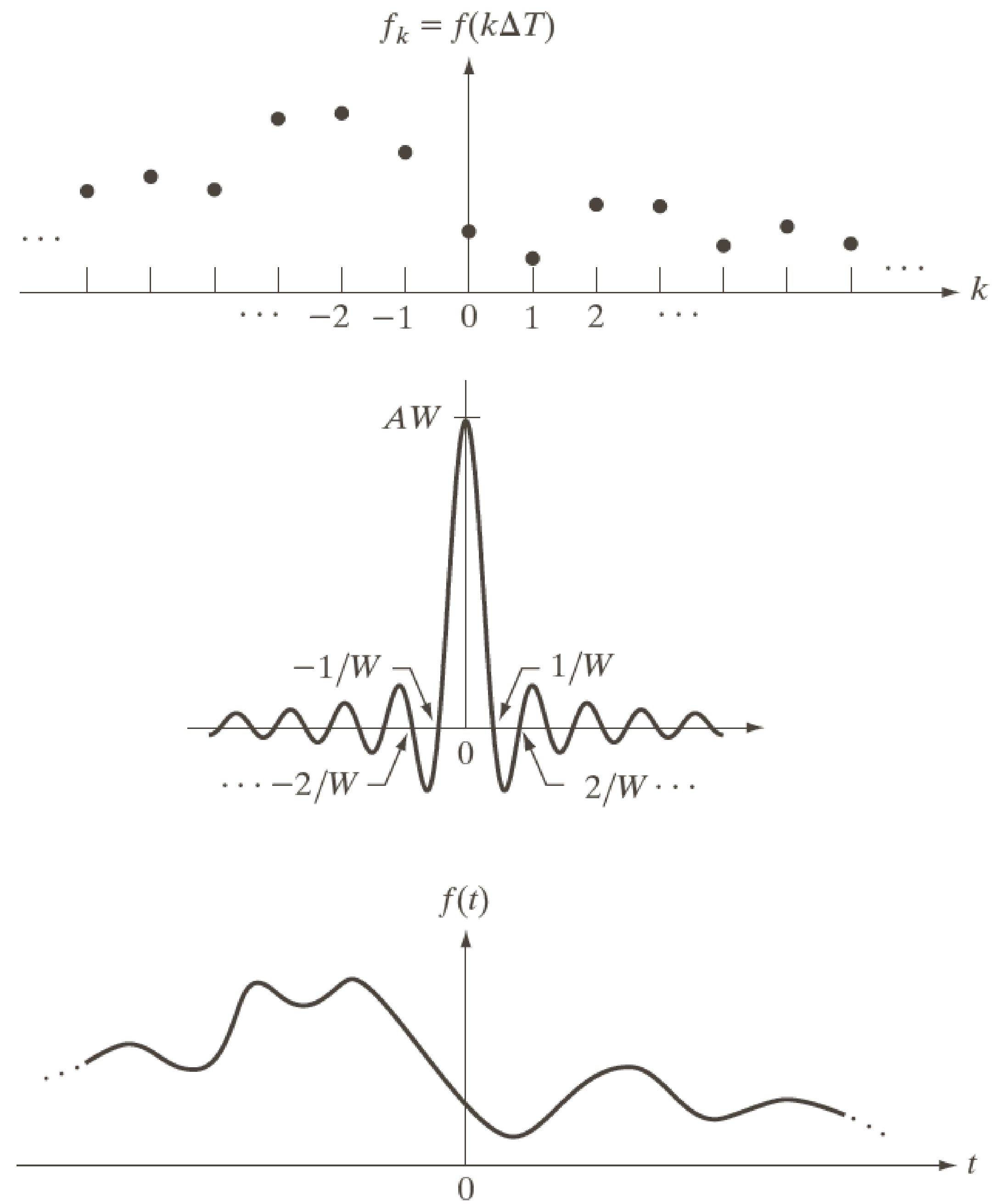
- We can reconstruct the original continuous function from the samples
- We just need to eliminate the extra copies by multiplying by an ideal low pass filter



Reconstruction

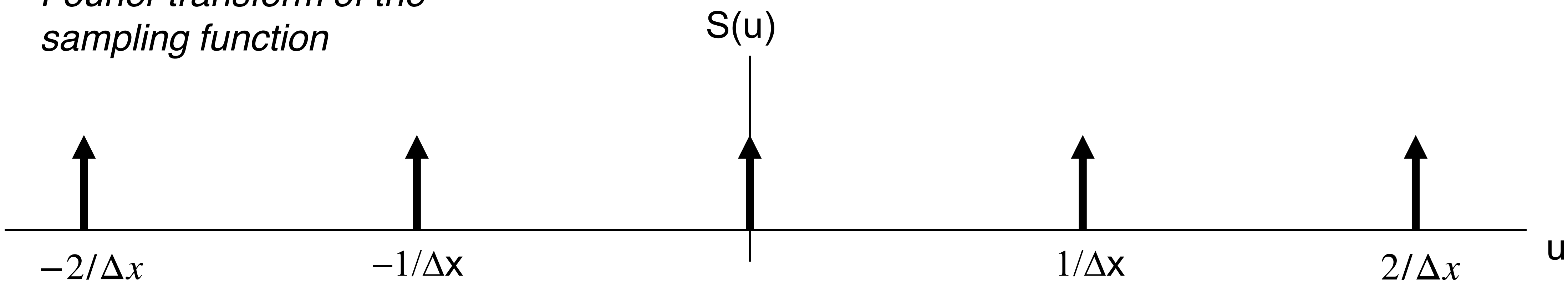
- Recall that multiplication in frequency domain is equivalent to convolution in the spatial domain



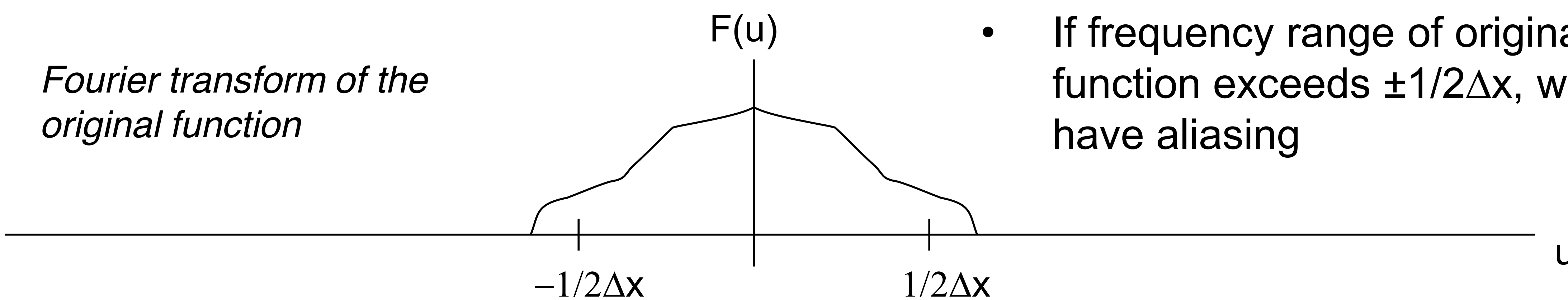


Under Sampling

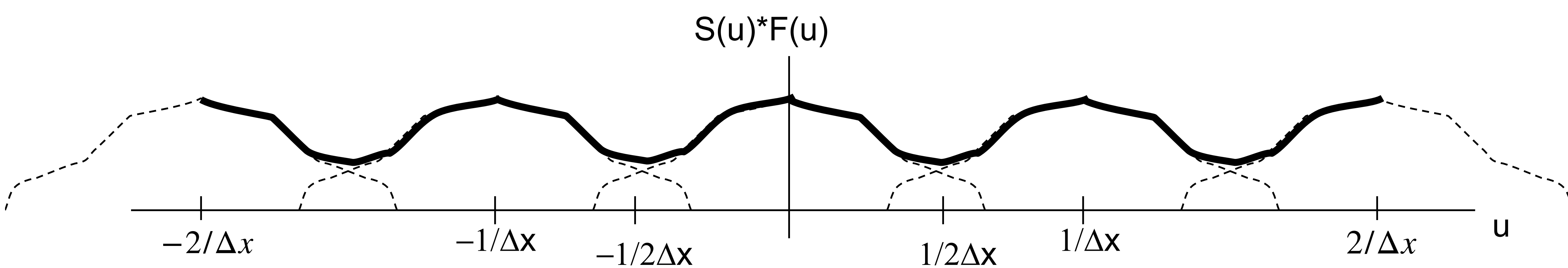
Fourier transform of the sampling function



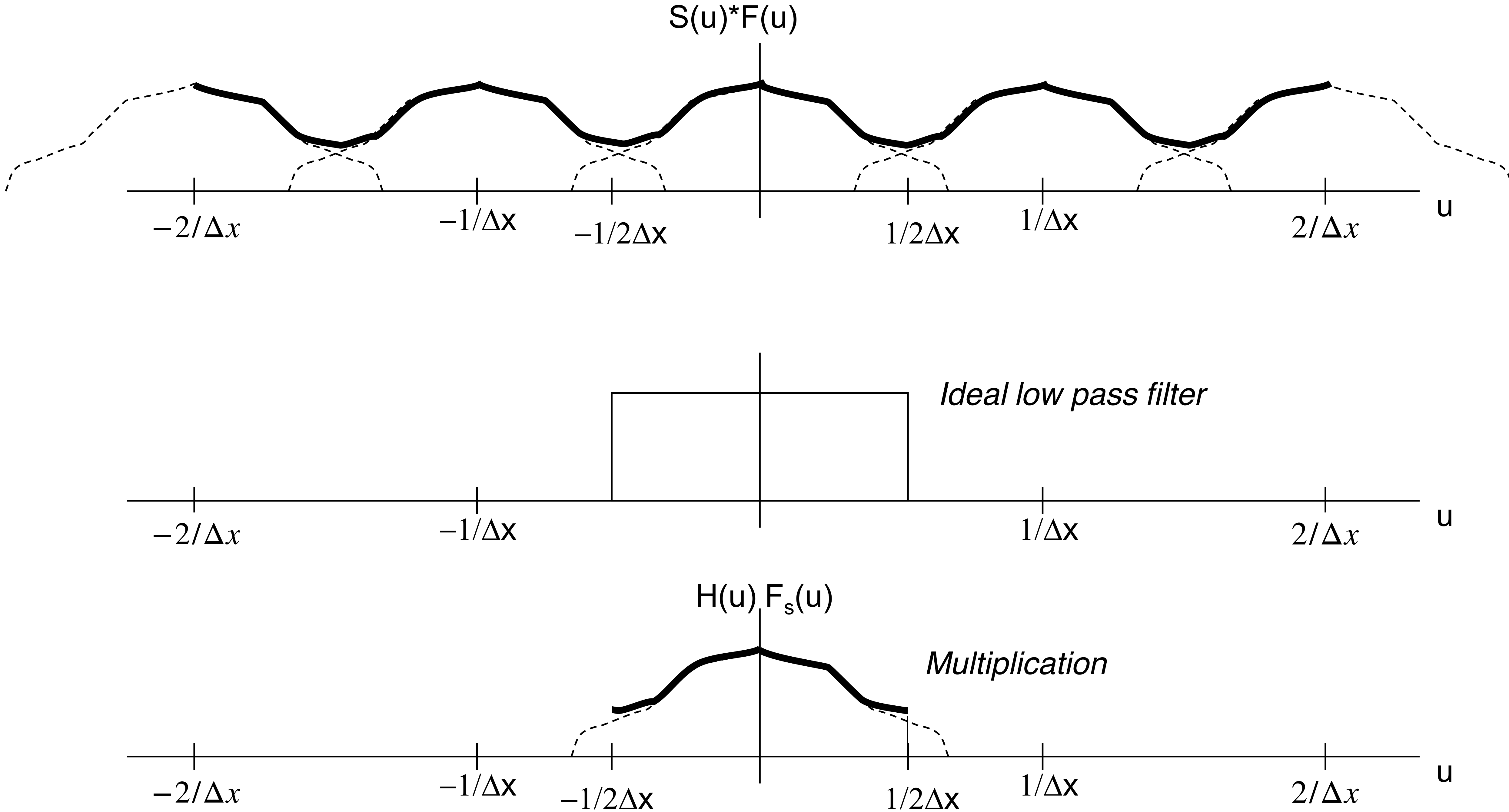
Fourier transform of the original function



- If frequency range of original function exceeds $\pm 1/2\Delta x$, we have aliasing



Under Sampling



Example of aliasing

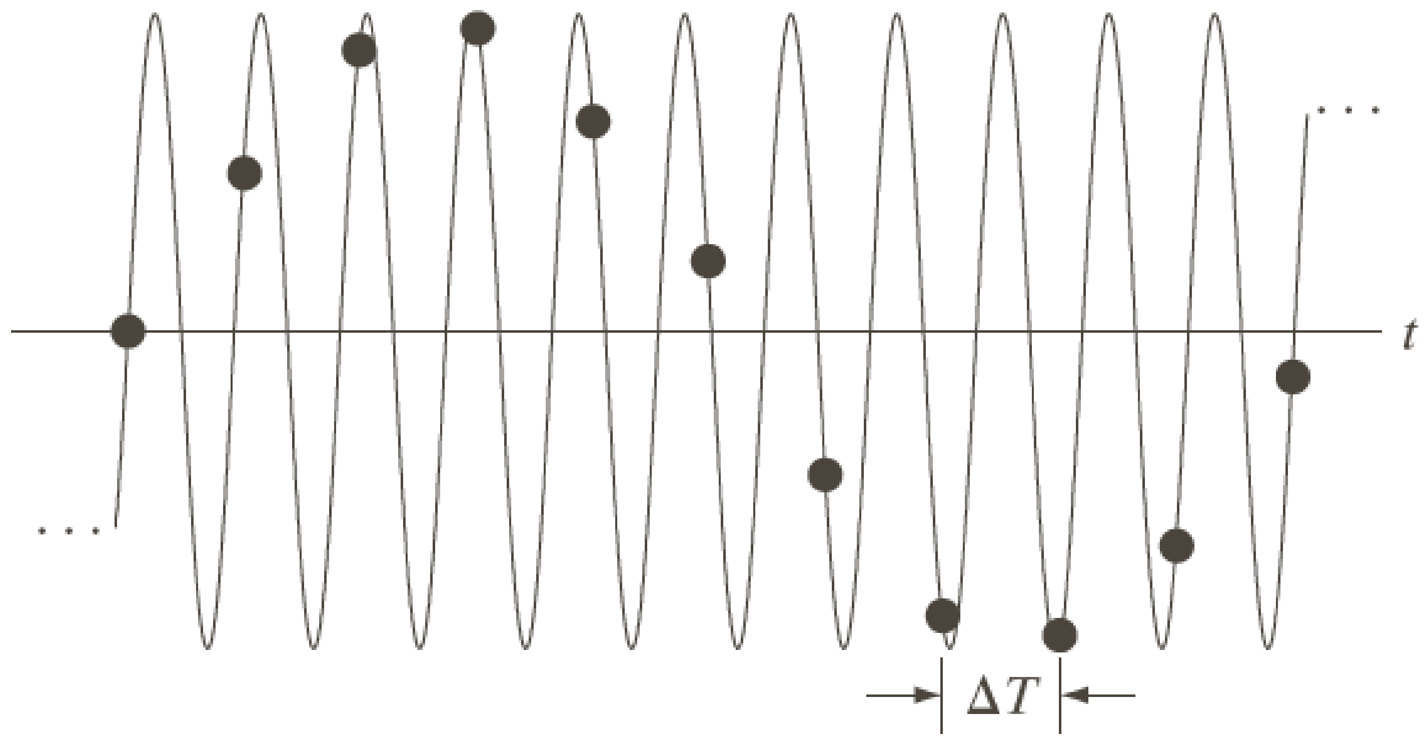


FIGURE 4.10 Illustration of aliasing. The under-sampled function (black dots) looks like a sine wave having a frequency much lower than the frequency of the continuous signal. The period of the sine wave is 2 s, so the zero crossings of the horizontal axis occur every second. ΔT is the separation between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

Part 2: 2D

Fourier Transforms – 2D

- 2D continuous

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

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

$$\boxed{\begin{array}{l} \mathcal{F}\left\{f(x,y)\right\} = F(u,v) \\ \mathcal{F}^{-1}\left\{F(u,v)\right\} = f(x,y) \end{array}}$$

- 2D discrete

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

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

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

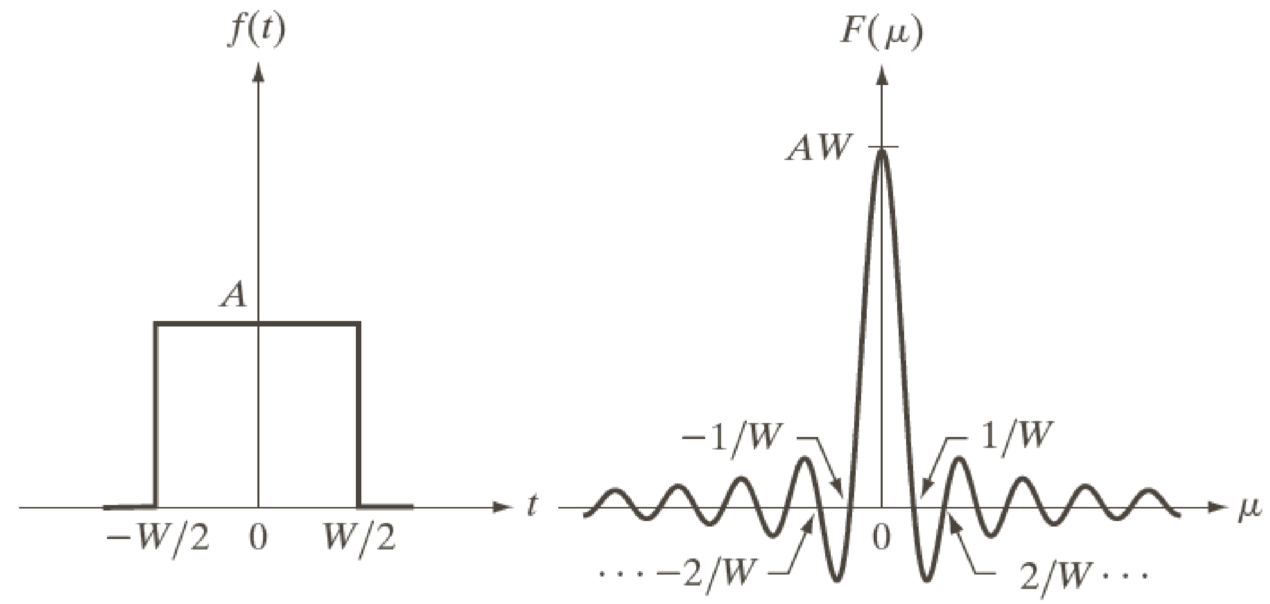
for $x = 0,1,\ldots,M-1$
and $y = 0,1,\ldots,N-1$

- Notes
  - **u** = (u,v) is spatial frequency *and* direction
  - *F*(0,0) is the “dc” component (ie sum of values)

The Fourier Transform Pair of the 2D Rectangle

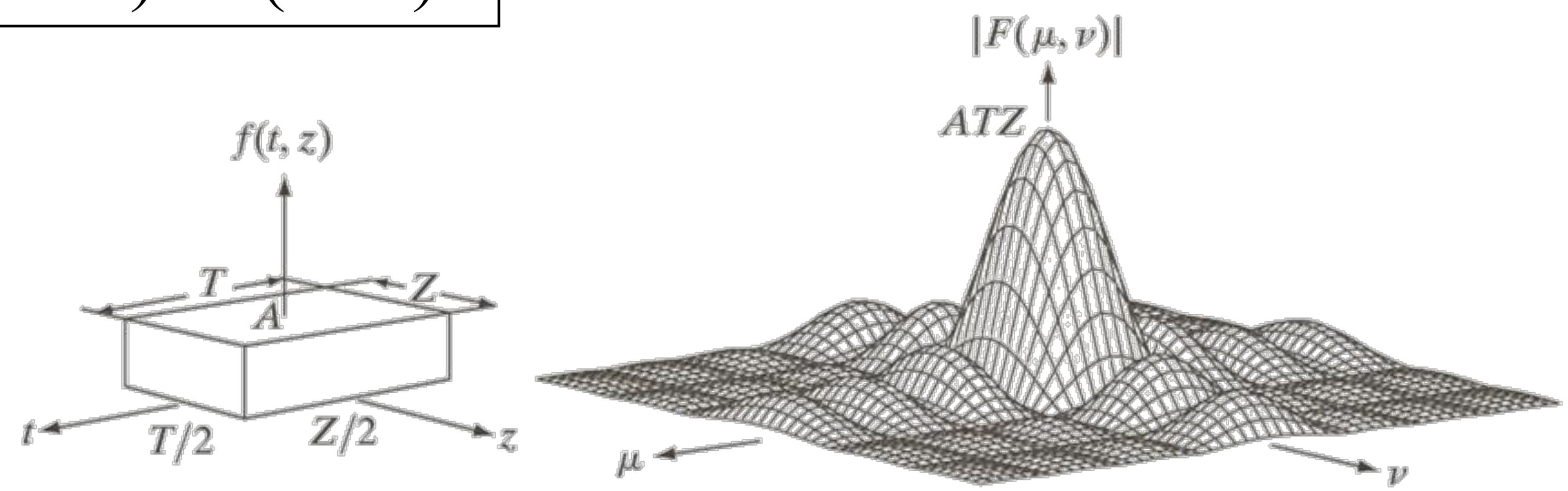
- 1D Rectangle

$$\boxed{\mathrm{rect}(W) \Leftrightarrow W\,\frac{\sin(\pi u W)}{(\pi u W)}}$$



- 2D rectangle

$$\boxed{\mathrm{rect}(T, Z) \Leftrightarrow TZ\,\frac{\sin(\pi u T)}{(\pi u T)}\,\frac{\sin(\pi v Z)}{(\pi v Z)}}$$



In 2D

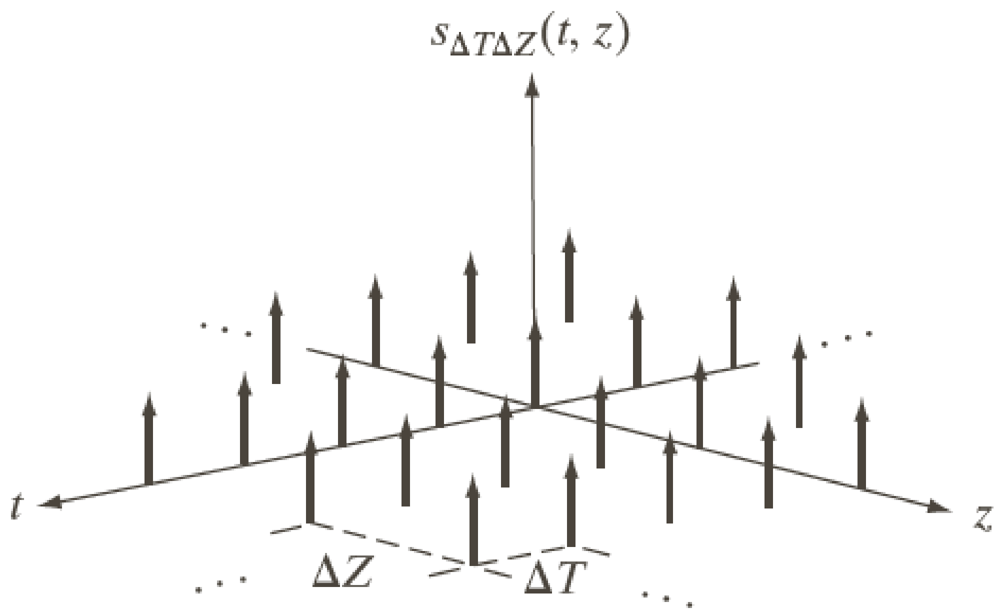
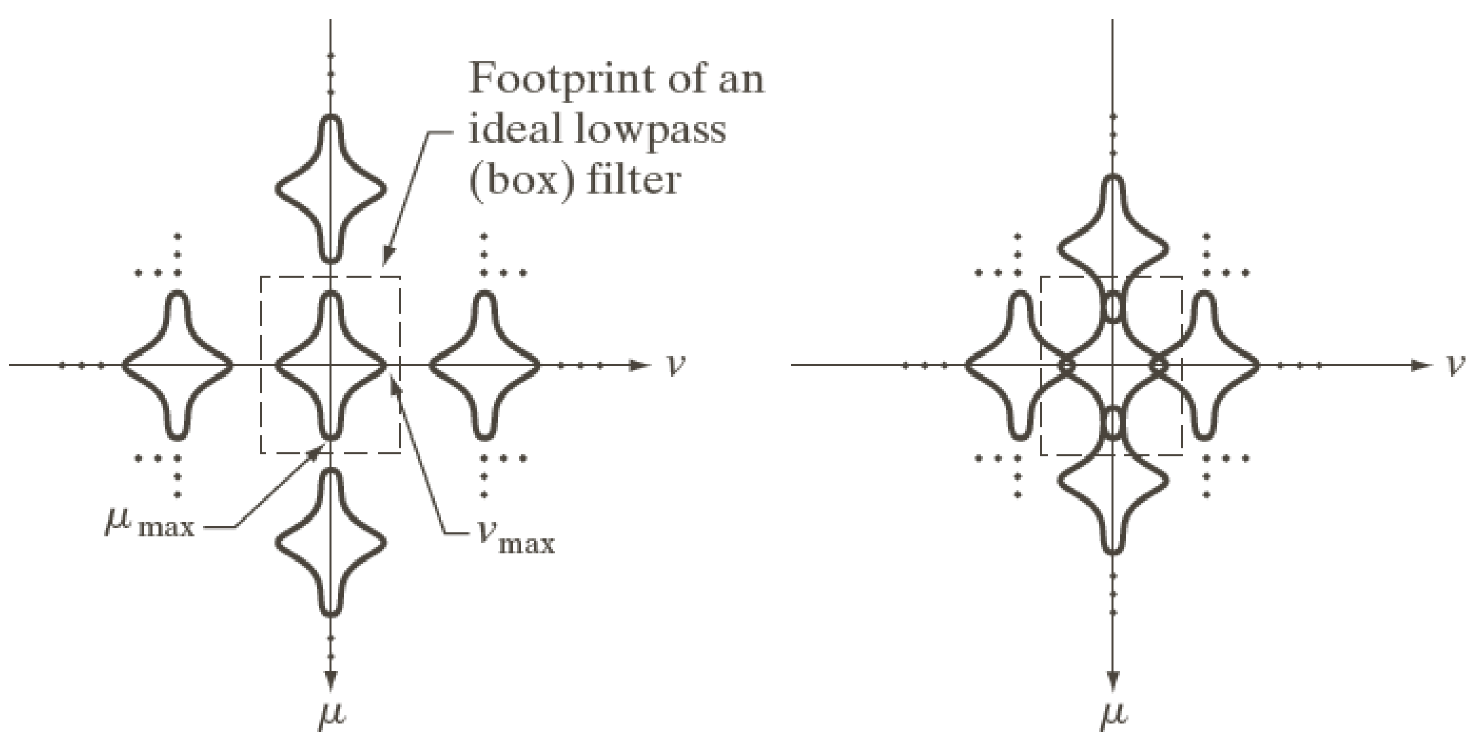


FIGURE 4.14 Two-dimensional impulse train.



a b

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

Part 3: Filtering

Filtering in Frequency Domain

- If we want to do a spatial filtering operation

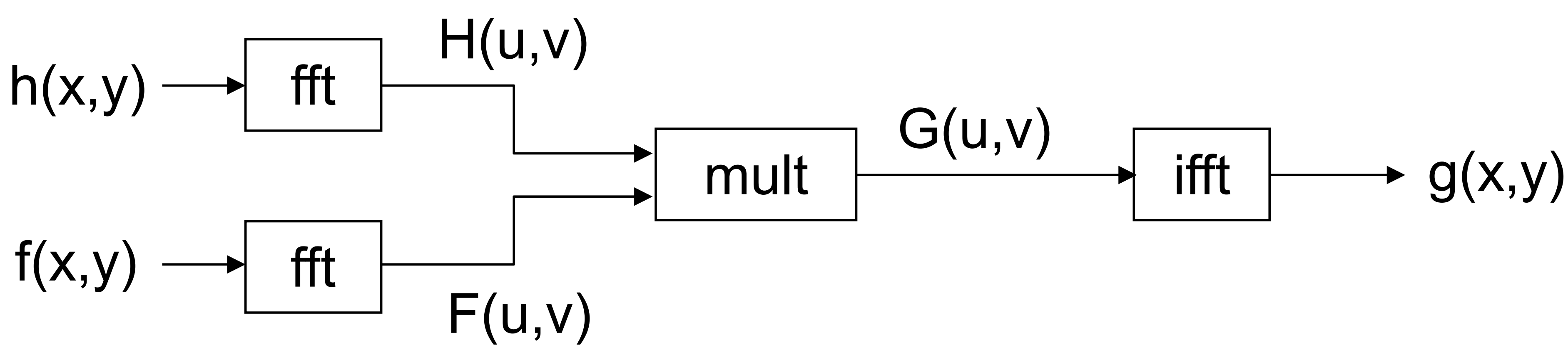
$$g(x,y) = h(x,y) * f(x,y)$$

- By the convolution theorem, we can transform the mask and the image to the frequency domain and do the operation there

$$\mathcal{F}\left\{h(x,y) * f(x,y)\right\} = H(u,v)F(u,v)$$

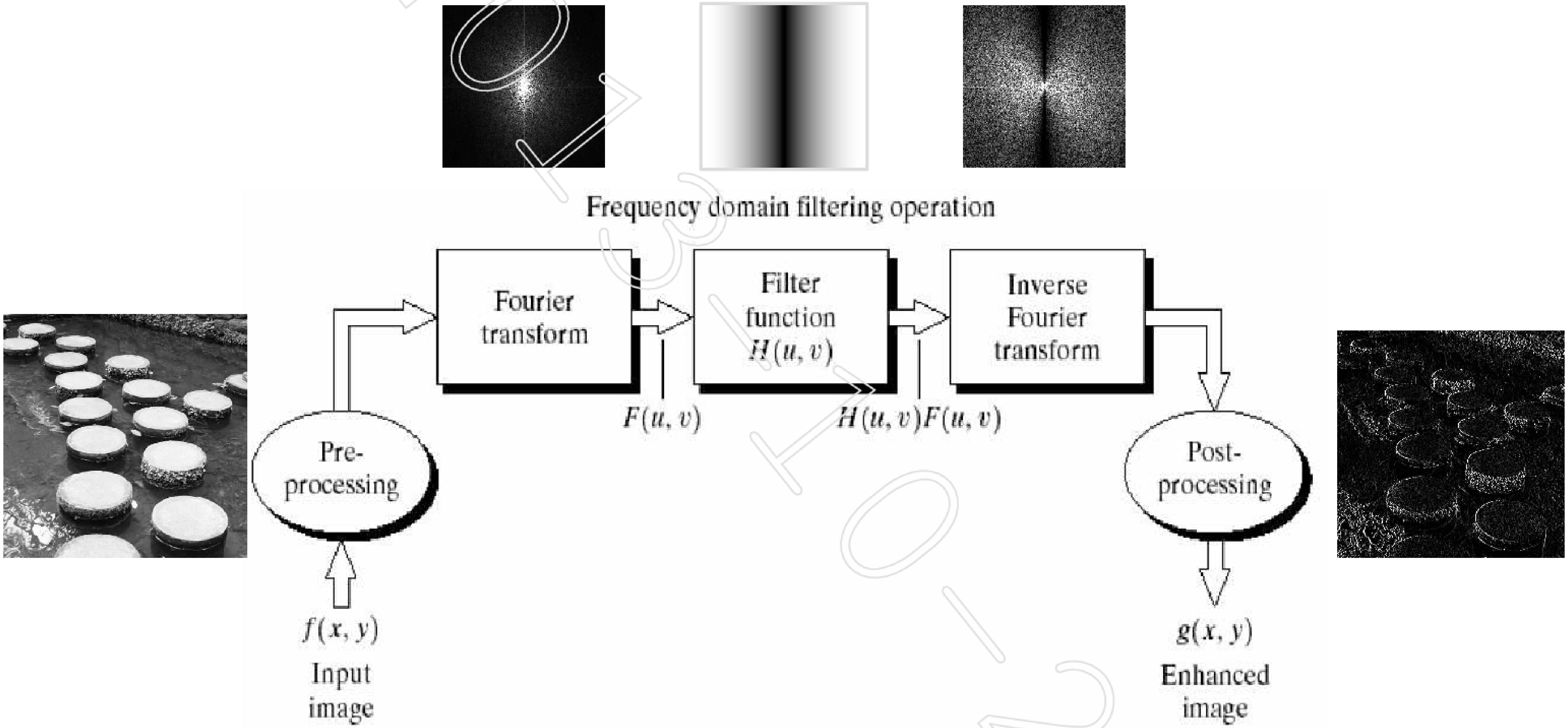
  – then

$$g(x,y) = \mathcal{F}^{-1}\left\{H(u,v)F(u,v)\right\}$$



- Notes:
  – The convolution kernel is the same size as the image (you have to pad the kernel with zeros if necessary)
  – Multiplication is point-by-point, of complex numbers

Frequency Domain Filtering



We transform a spatial filter into the frequency domain and can analyze how the filter works.

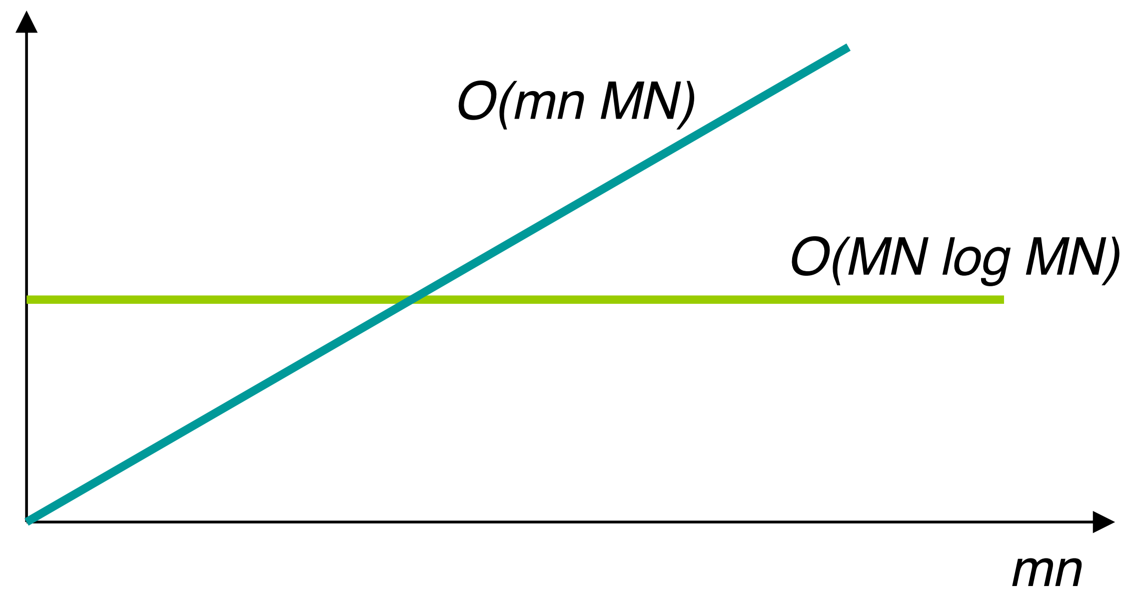
Advantages of Filtering in Frequency Domain

- Cost (number of operations) of the computation of Fast Fourier Transform is

  $O(MN \log MN)$

  where MN = number of points in image
- The total cost of filtering in the frequency domain is dominated by FFT
- Compare this to convolution in spatial domain - it is $O((mn)(MN))$

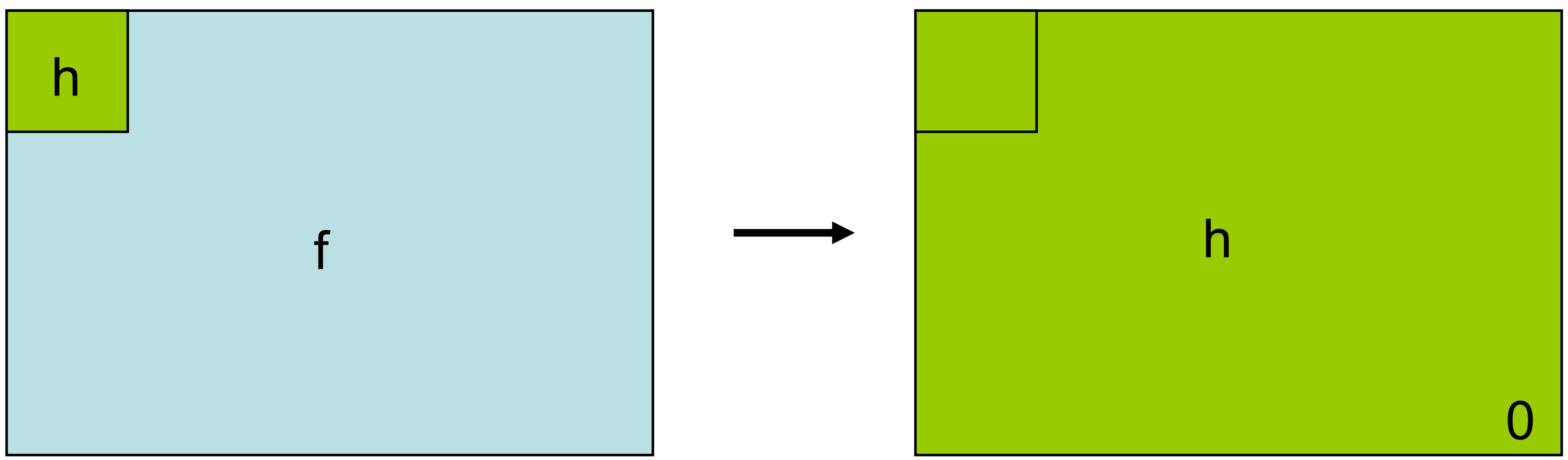
Plot of cost vs mn, with image size MN fixed



Convolution in frequency domain faster for large kernels (when mn gets much larger than log(MN))

Fourier-Domain Filtering in Matlab

- Need to pad filter to be same size as image
  - Can do this by setting the point in the lower right corner
    `h(size(f,1), size(f,2)) = 0;`
  - where `size(f)` is the size (#rows, #cols) of the image
  - Matlab expands the filter and fills new values to zero

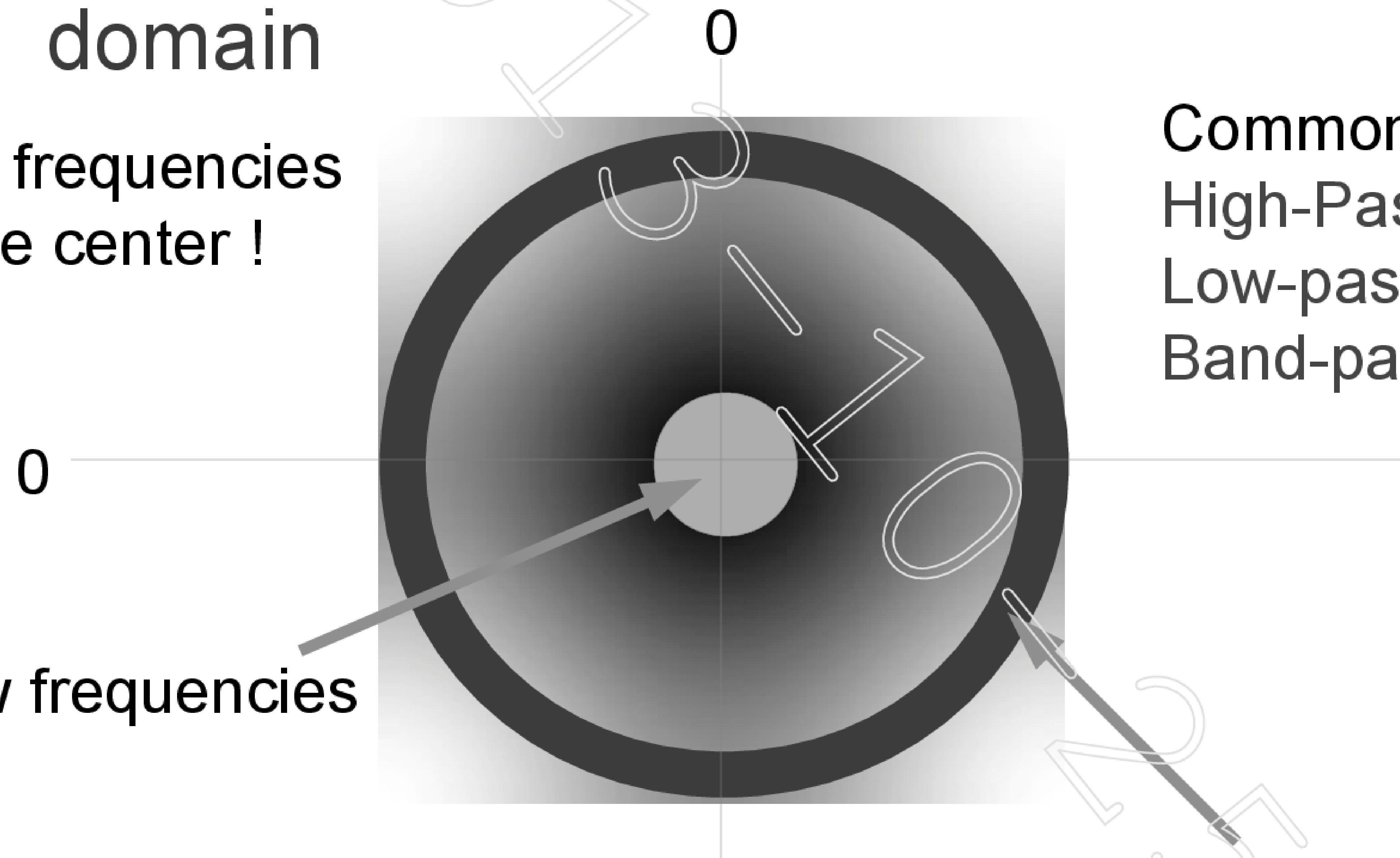


- The inverse Fourier Transform (`ifft2)` should yield a real image
  - But take `real` of final result (to get rid of tiny imaginary values)

Frequency Filter Interpretation

- Representation of the filter in the frequency domain

Low frequencies in the center !



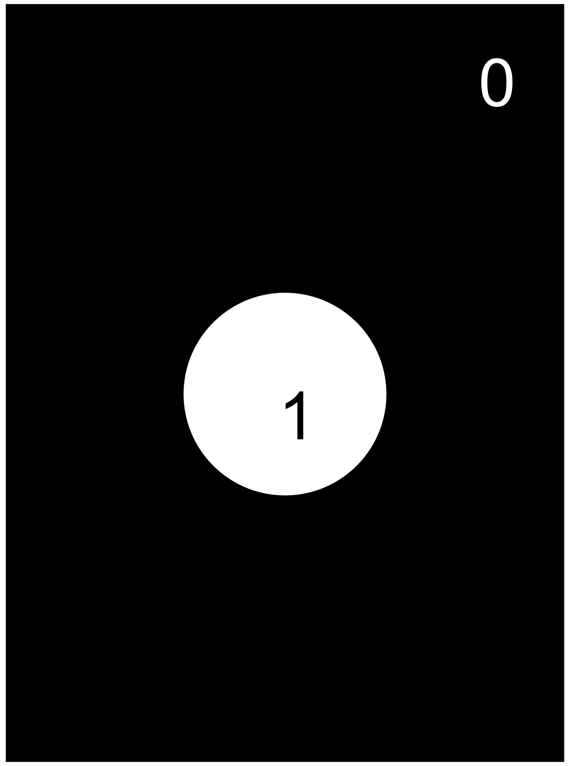
Common Filter types:
High-Pass
Low-pass
Band-pass

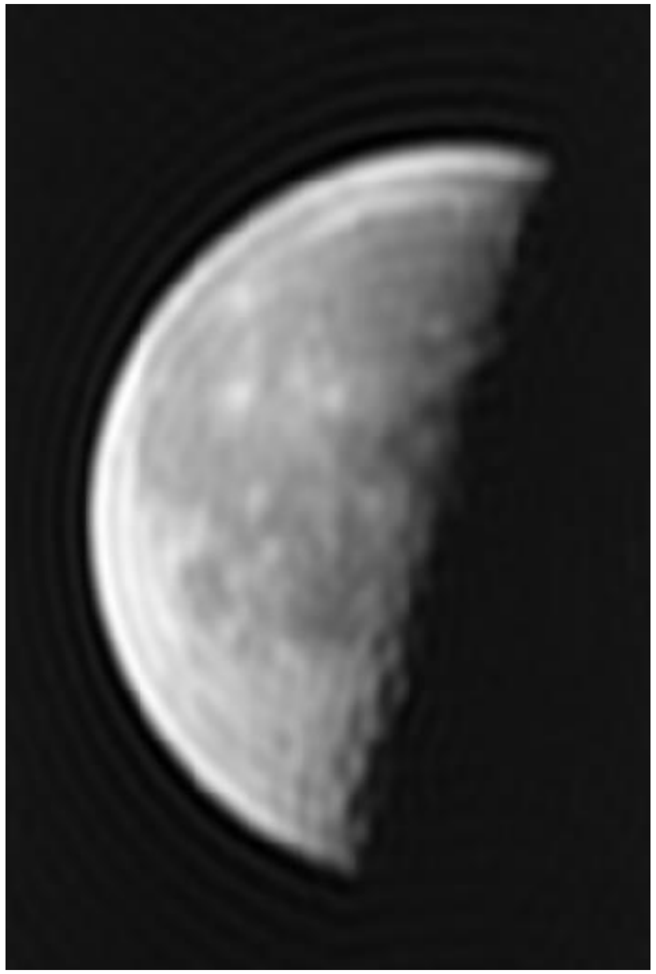
The frequency filter function is also known as **transfer function**.

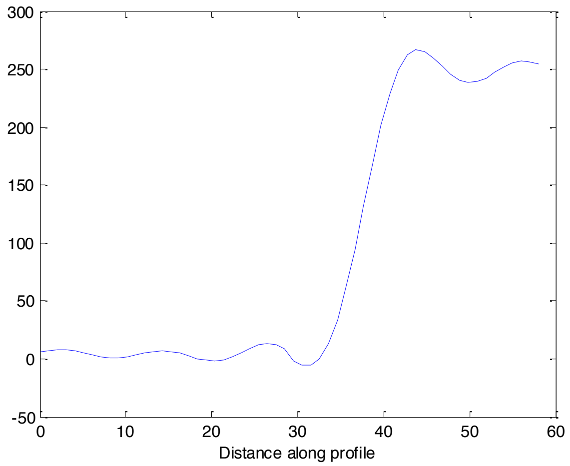
Matlab Example

- Create image `H` of a disk in center
- Multiplication in freq domain

  `G = H .* F`
- `ifft2(G)`
- Note ringing (do improfile)



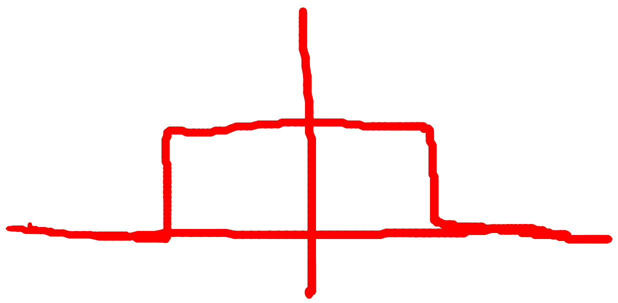




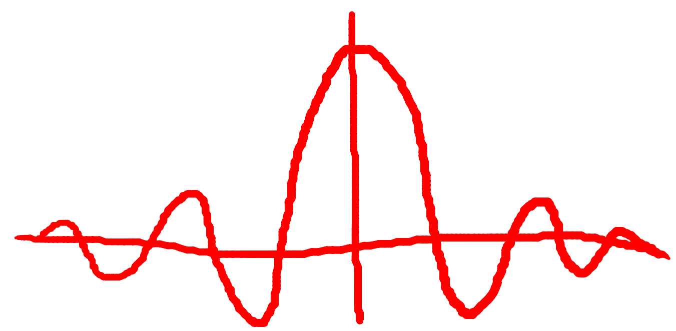
Low Pass Filters

- Box filter

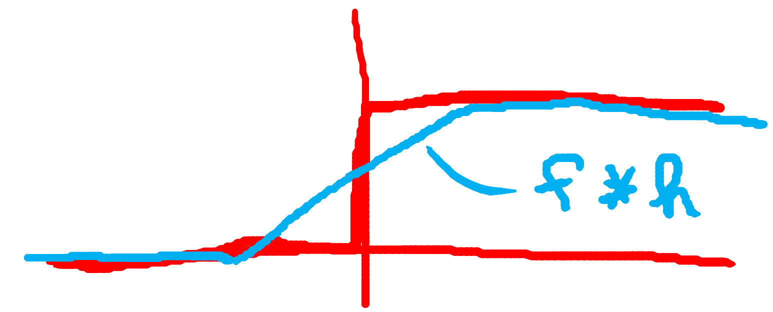
h(x)



H(u)



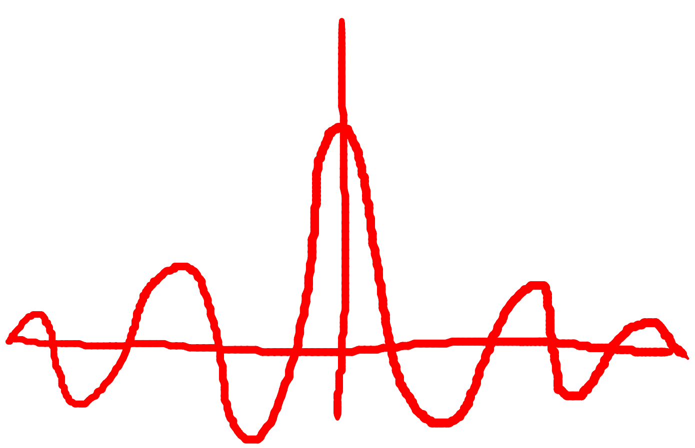
f∗h



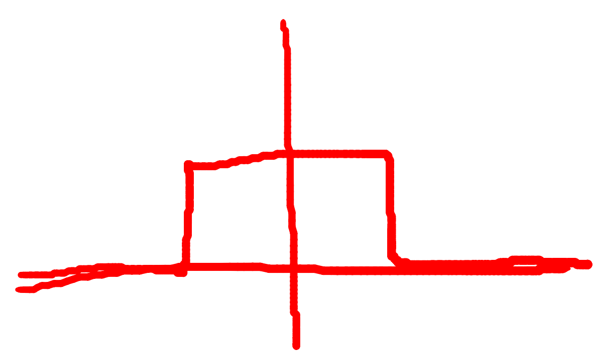
Convolution w/ step edge

- Ideal low pass filter

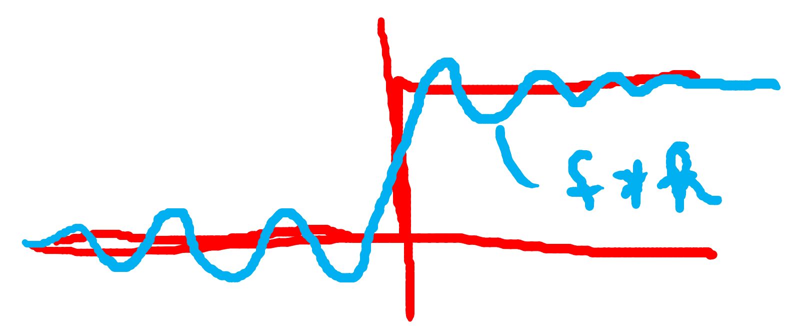
h(x)



H(u)



f∗h



Convolution w/ step edge

Ideal Low Pass Filter in 2D

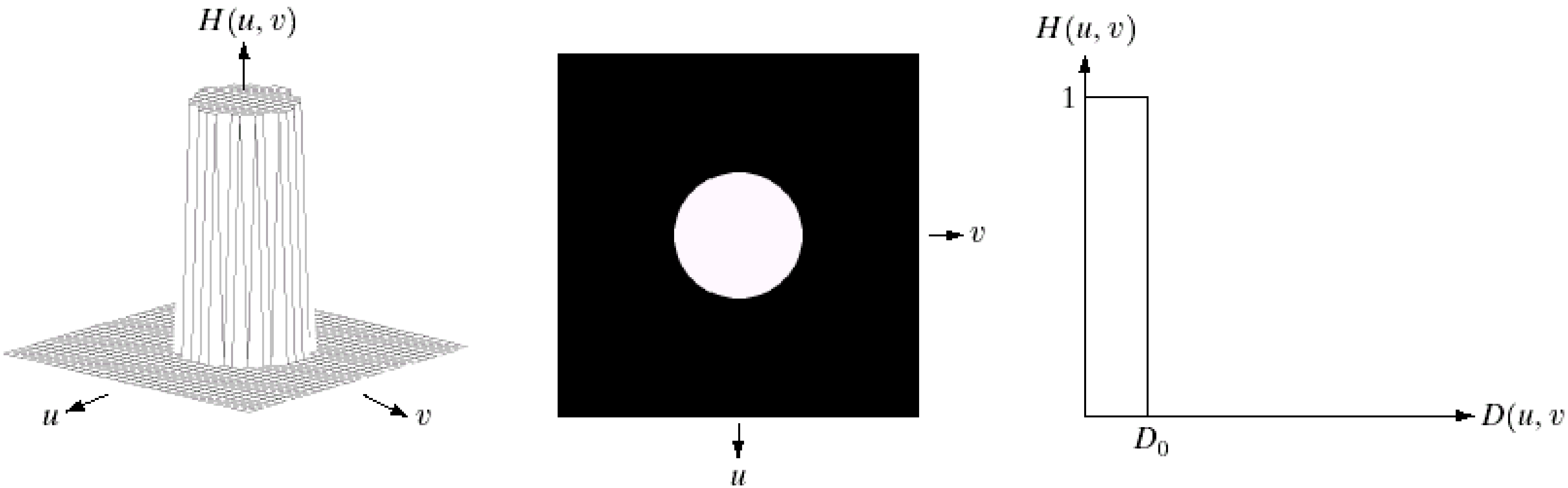


FIGURE 4.10 (a) Perspective plot of an ideal lowpass filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross section.

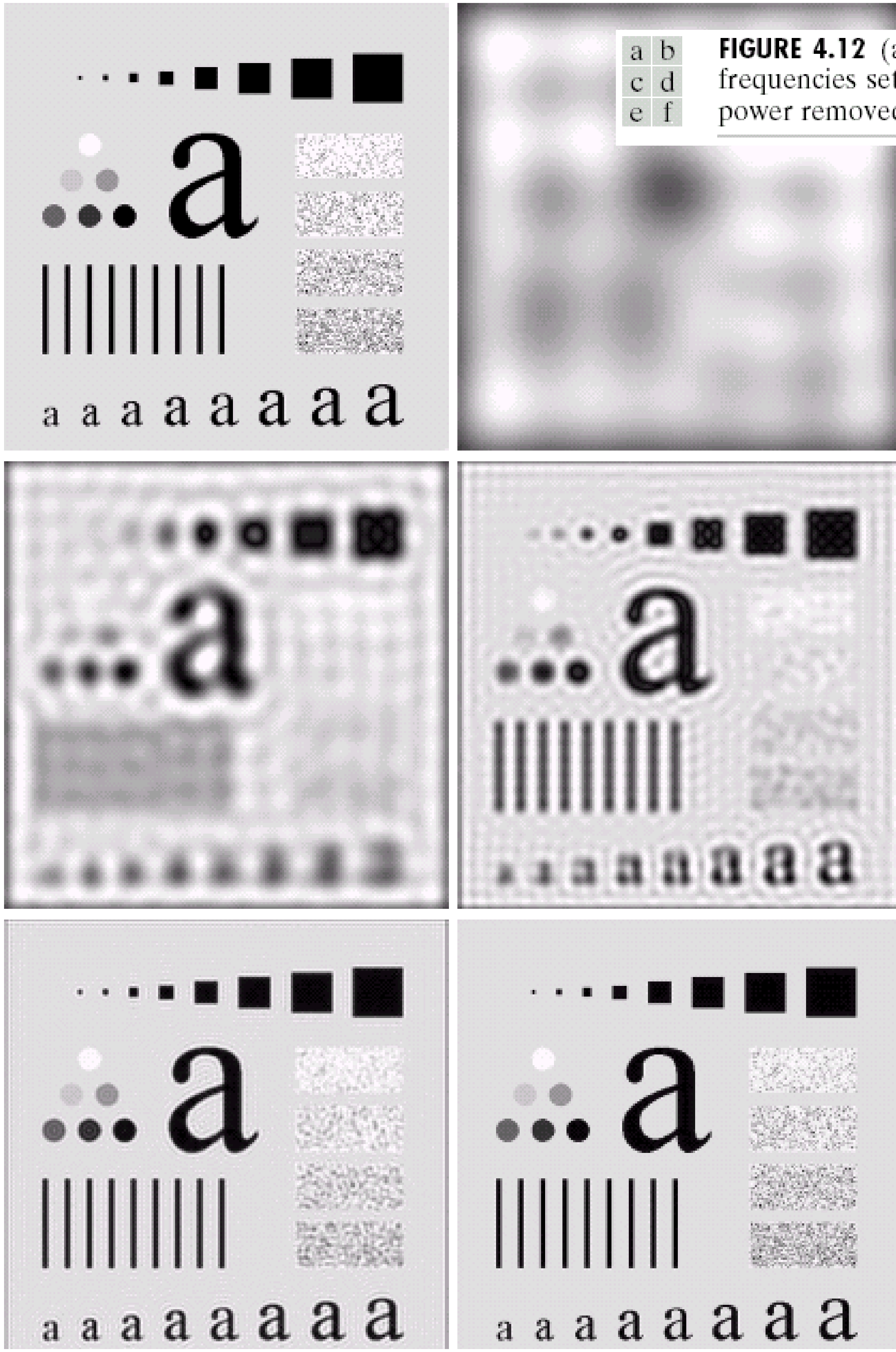
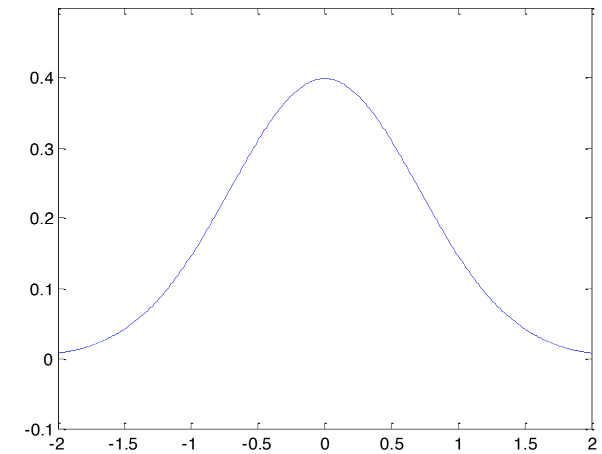


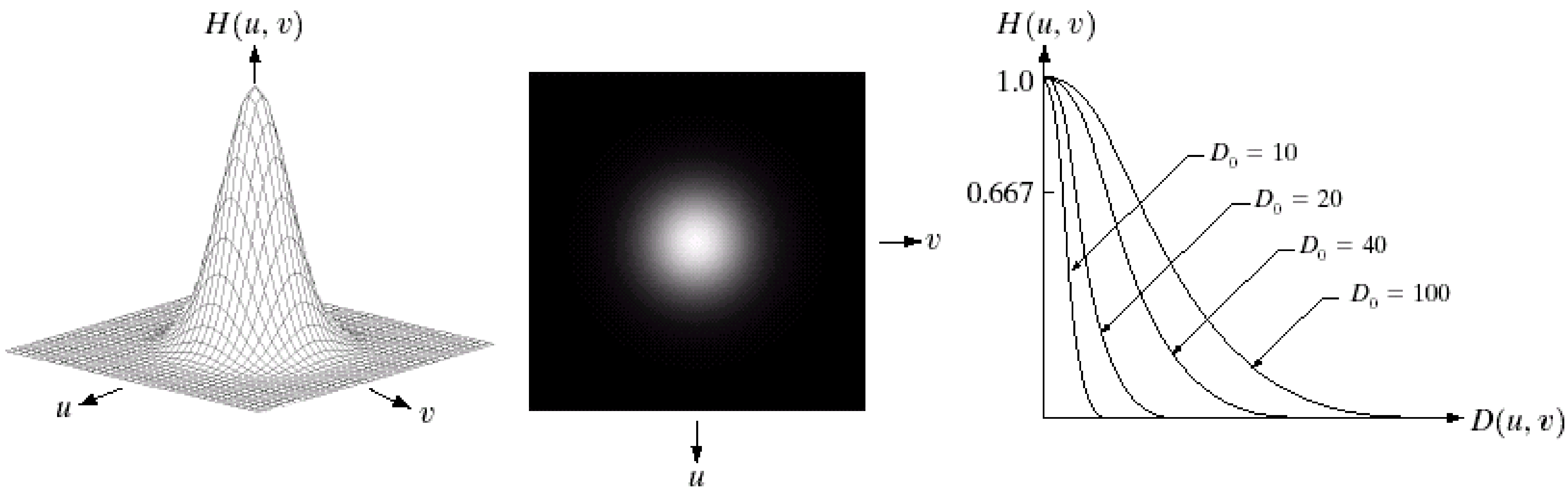
FIGURE 4.12 (a) Original image. (b)–(f) Results of ideal lowpass filtering with cutoff frequencies set at radii values of 5, 15, 30, 80, and 230, as shown in Fig. 4.11(b). The power removed by these filters was 8, 5.4, 3.6, 2, and 0.5% of the total, respectively.

Note “ringing” near sharp edges

Gaussian Lowpass Filter

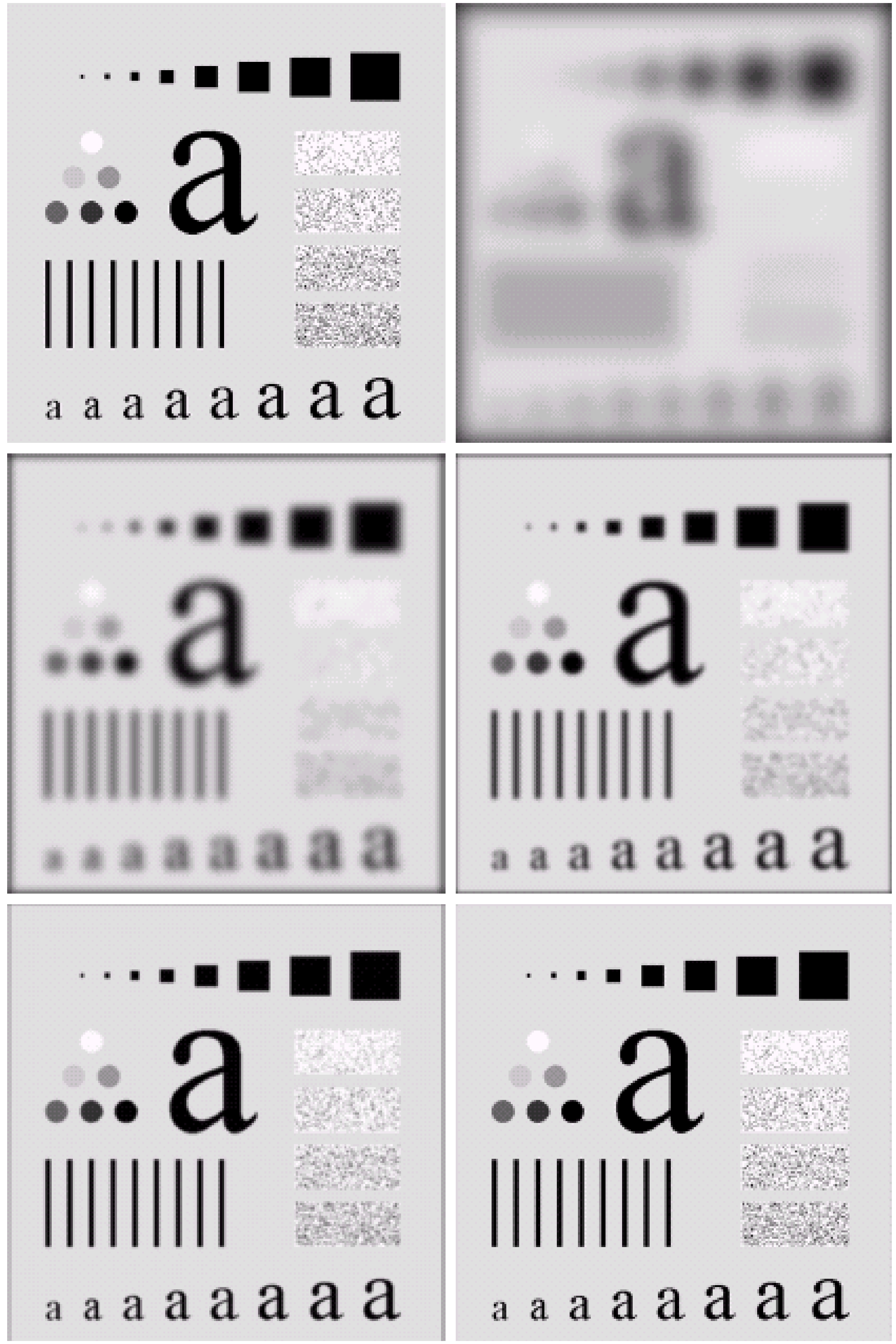
- A Gaussian in the spatial domain also has the form of a Gaussian in the frequency domain
- No ringing, but allows high frequencies to pass





a b c

FIGURE 4.17 (a) Perspective plot of a GLPF transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections for various values of D_0.

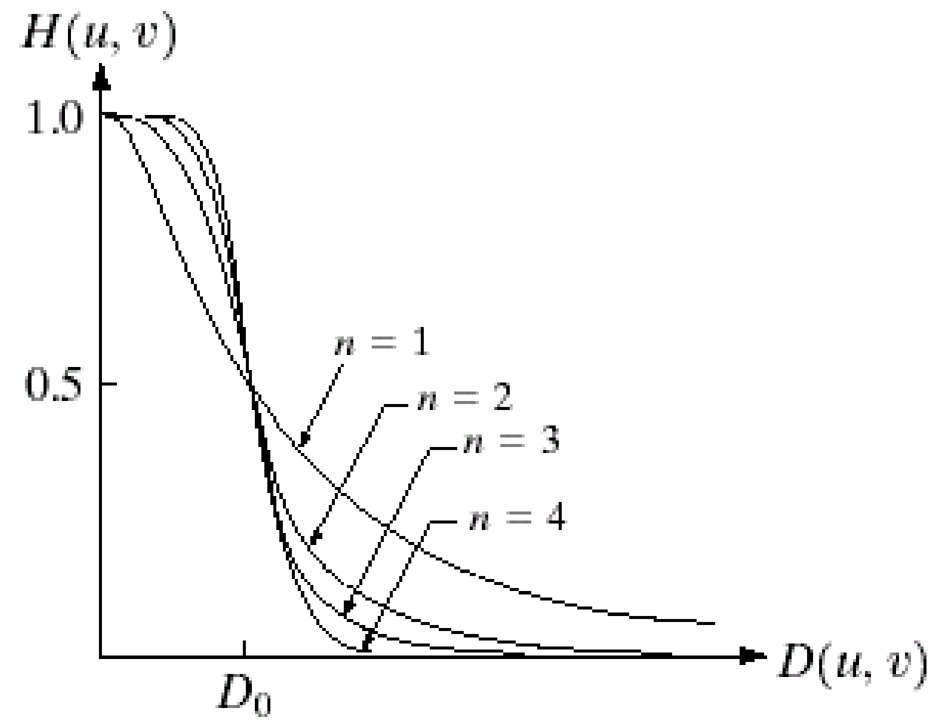


Butterworth Lowpass Filter

- Definition

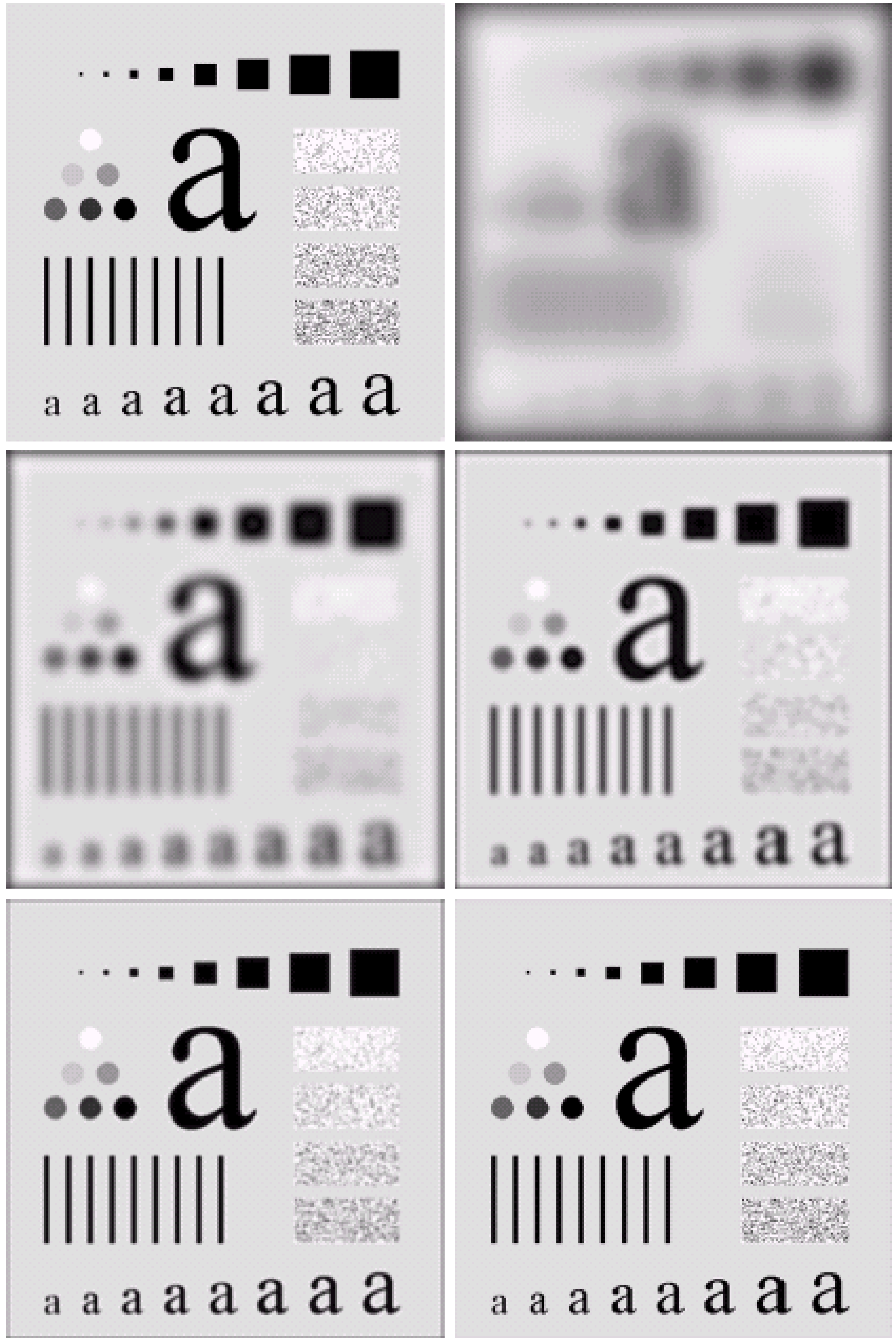
$$H(u,v) = \frac{1}{1+\left[D(u,v)/D_0\right]^{2n}}$$

  - $D(u,v)$ is distance from (0,0) to (u,v)
  - D_0 is cutoff frequency
  - n is the “order” of the filter

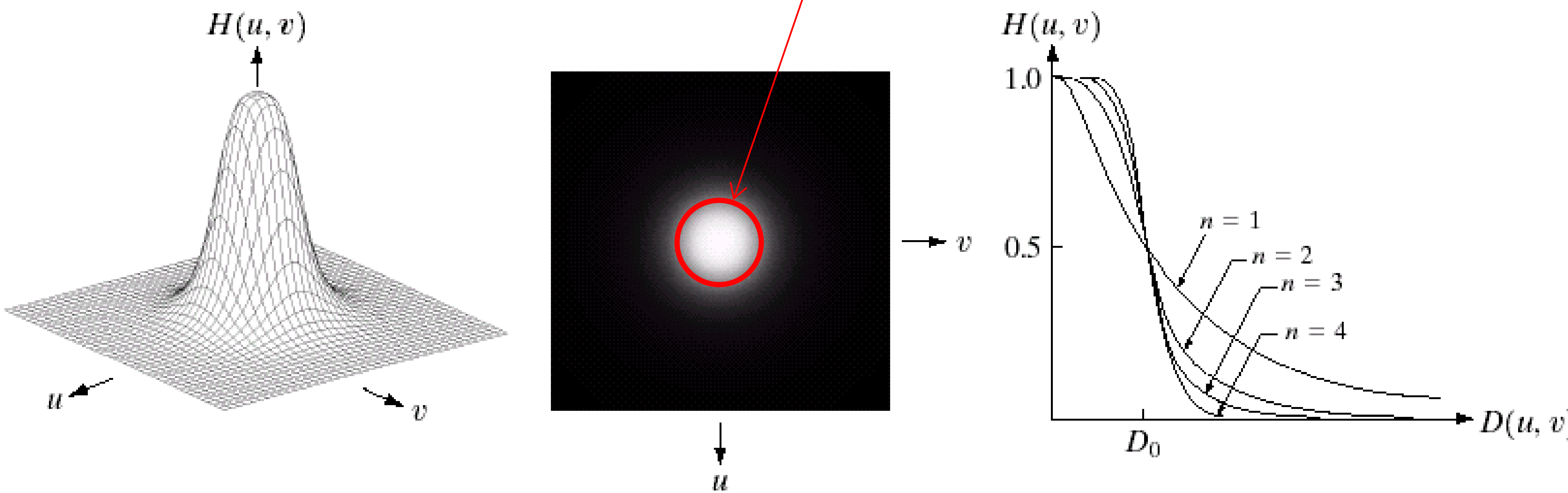


- Properties
  - For $D(u,v) << D_0$, $H \approx 1$
  - For $D(u,v) >> D_0$, $H \approx 0$
  - At $D(u,v) = D_0$, $H = 1/2$

- Advantages
  - Reduces “ringing” while keeping clear cutoff
  - Tradeoff between amount of ringing and sharpness of cutoff



D_0 is the radius where the magnitude drops to 0.5



a b c

FIGURE 4.14 (a) Perspective plot of a Butterworth lowpass filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections of orders 1 through 4.

For large n, H(u,v) approaches the ideal low pass filter

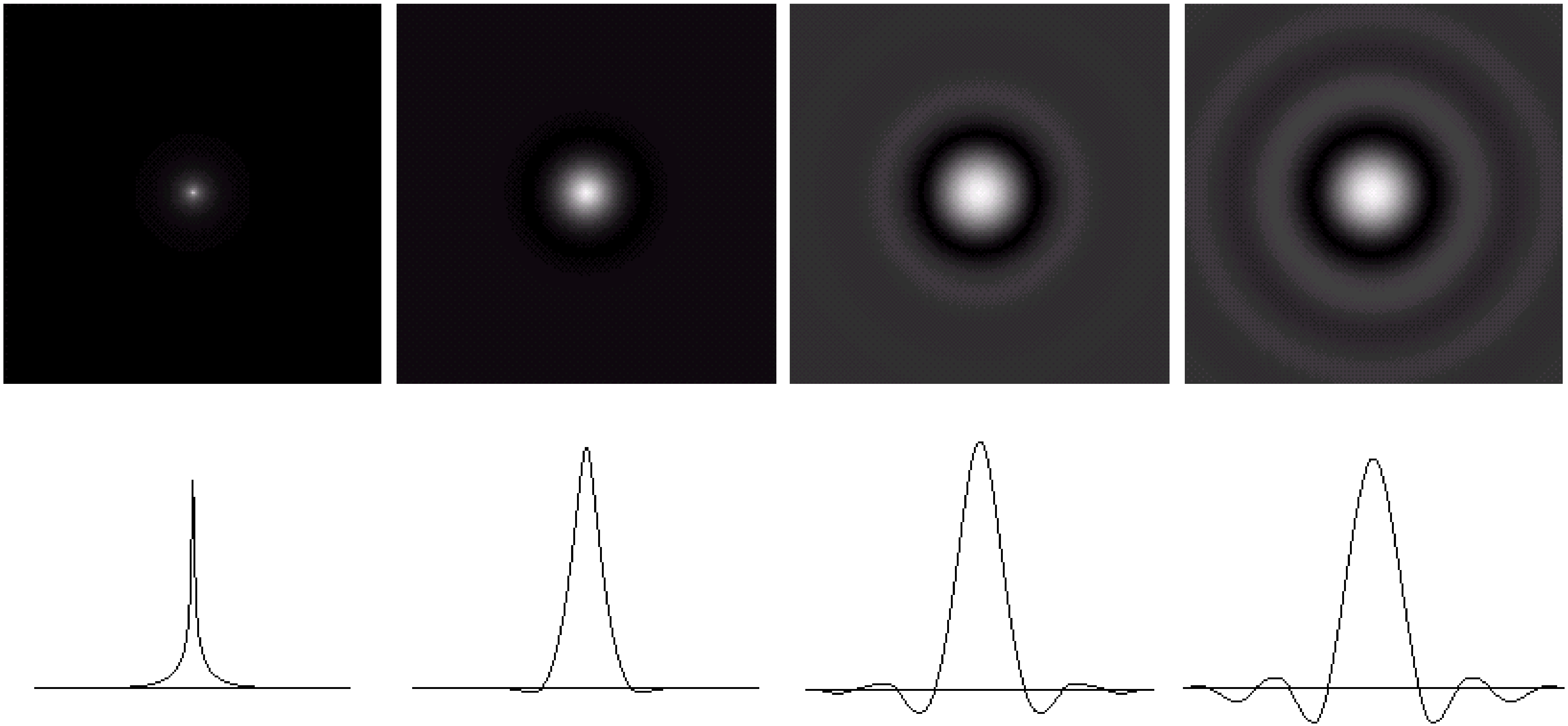


FIGURE 4.16 (a)–(d) Spatial representation of BLPFs of order 1, 2, 5, and 20, and corresponding gray-level profiles through the center of the filters (all filters have a cutoff frequency of 5). Note that ringing increases as a function of filter order.

Sharpening Filters

- Can obtain by

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

- Types
  - Ideal high pass
  - Butterworth high pass
  - Gaussian high pass

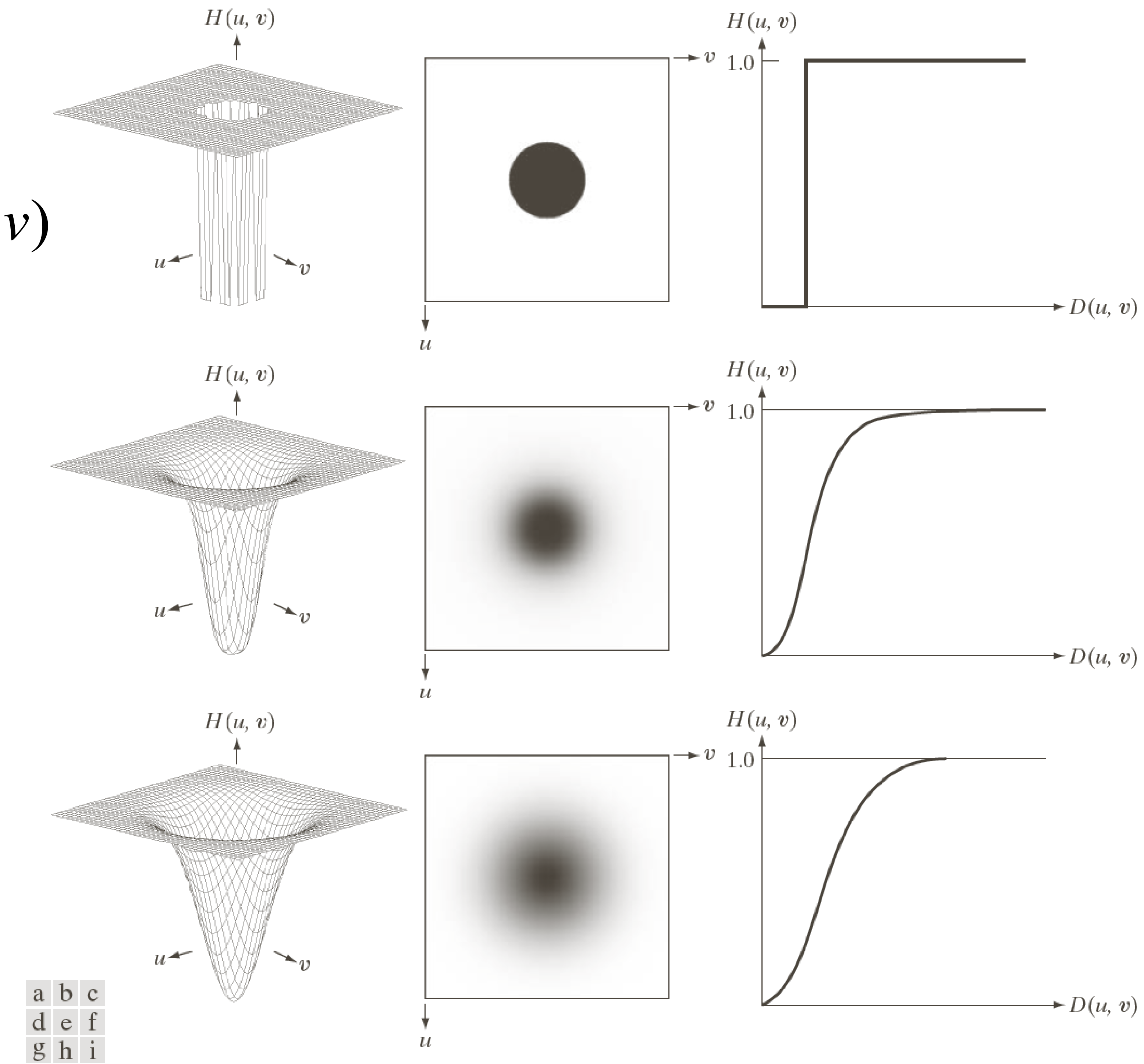
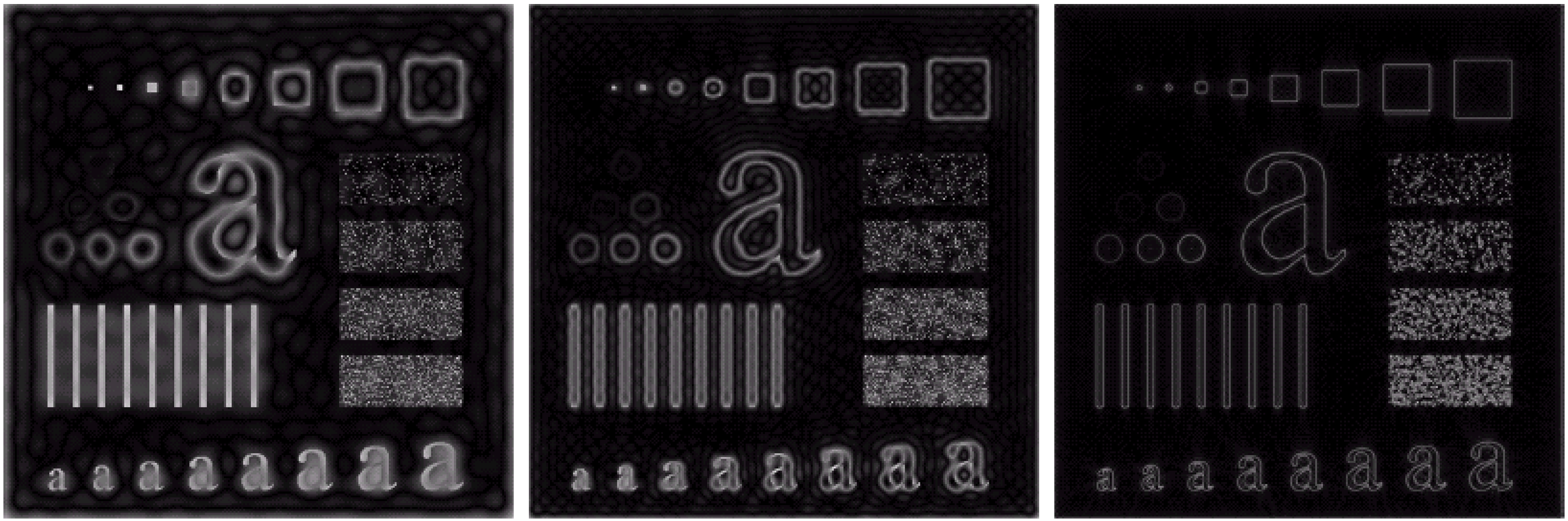


FIGURE 4.52 Top row: Perspective plot, image representation, and cross section of a typical ideal highpass filter. Middle and bottom rows: The same sequence for typical Butterworth and Gaussian highpass filters.



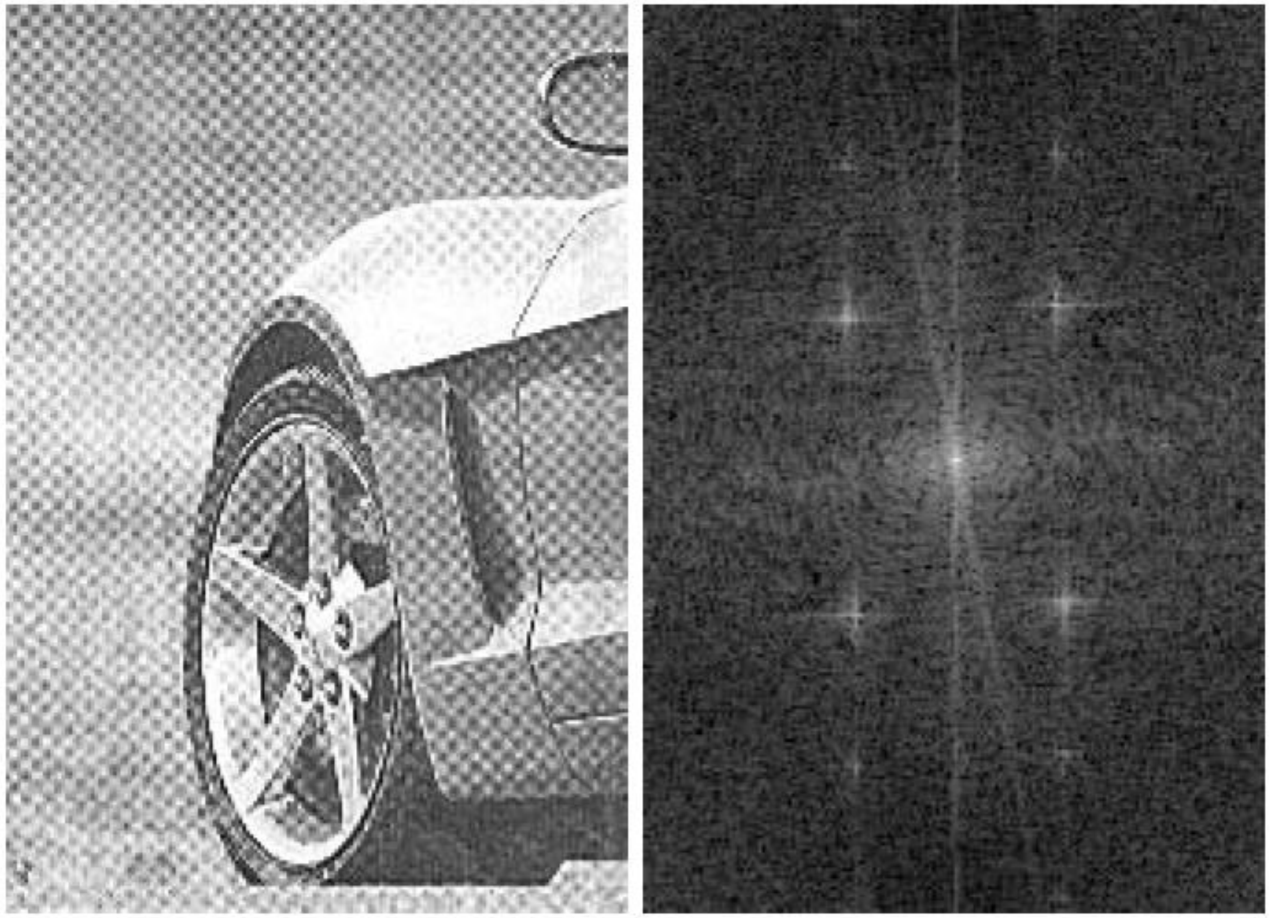
a b c

FIGURE 4.24 Results of ideal highpass filtering the image in Fig. 4.11(a) with $D_0 = 15$, 30, and 80, respectively. Problems with ringing are quite evident in (a) and (b).

Notch Filters

- A filter that rejects (or passes) specific frequencies
- Example: periodic noise corresponds to spikes or lines in the Fourier domain
- Can design a filter with zeros at those frequencies … this will remove the noise
- Examples:
  - Image mosaics
  - Scan line noise
  - Halftoning noise (moire patterns)

FIGURE 4.21 A newspaper image of size 246 × 168 pixels sampled at 75 dpi showing a moiré pattern. The moiré pattern in this image is the interference pattern created between the ±45° orientation of the halftone dots and the north–south orientation of the sampling grid used to digitize the image.



Steps in Notch Filtering

- Look at spectrum $|F(u,v)|$ of noisy image $f(x,y)$, find frequencies corresponding to the noise

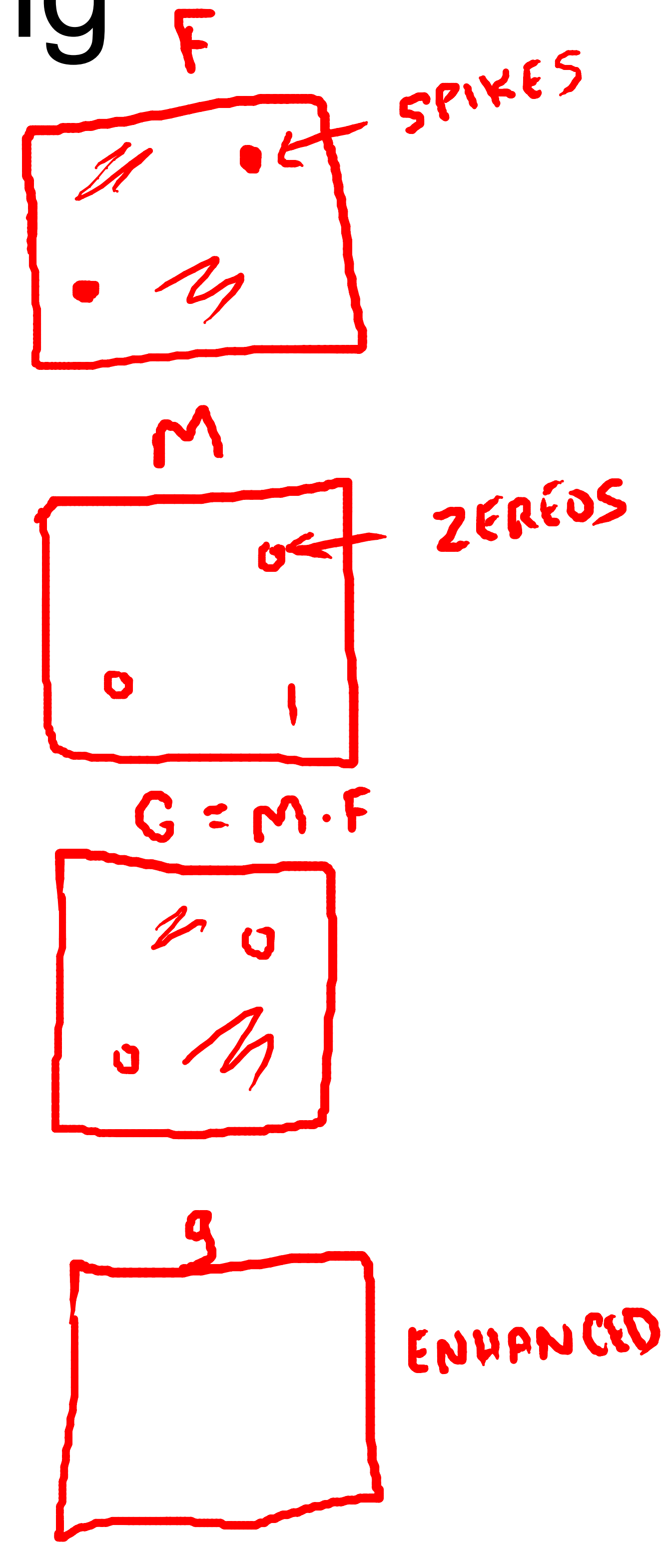
- Create a mask image $M(u,v)$ with notches (zeros) at those places, 1's elsewhere

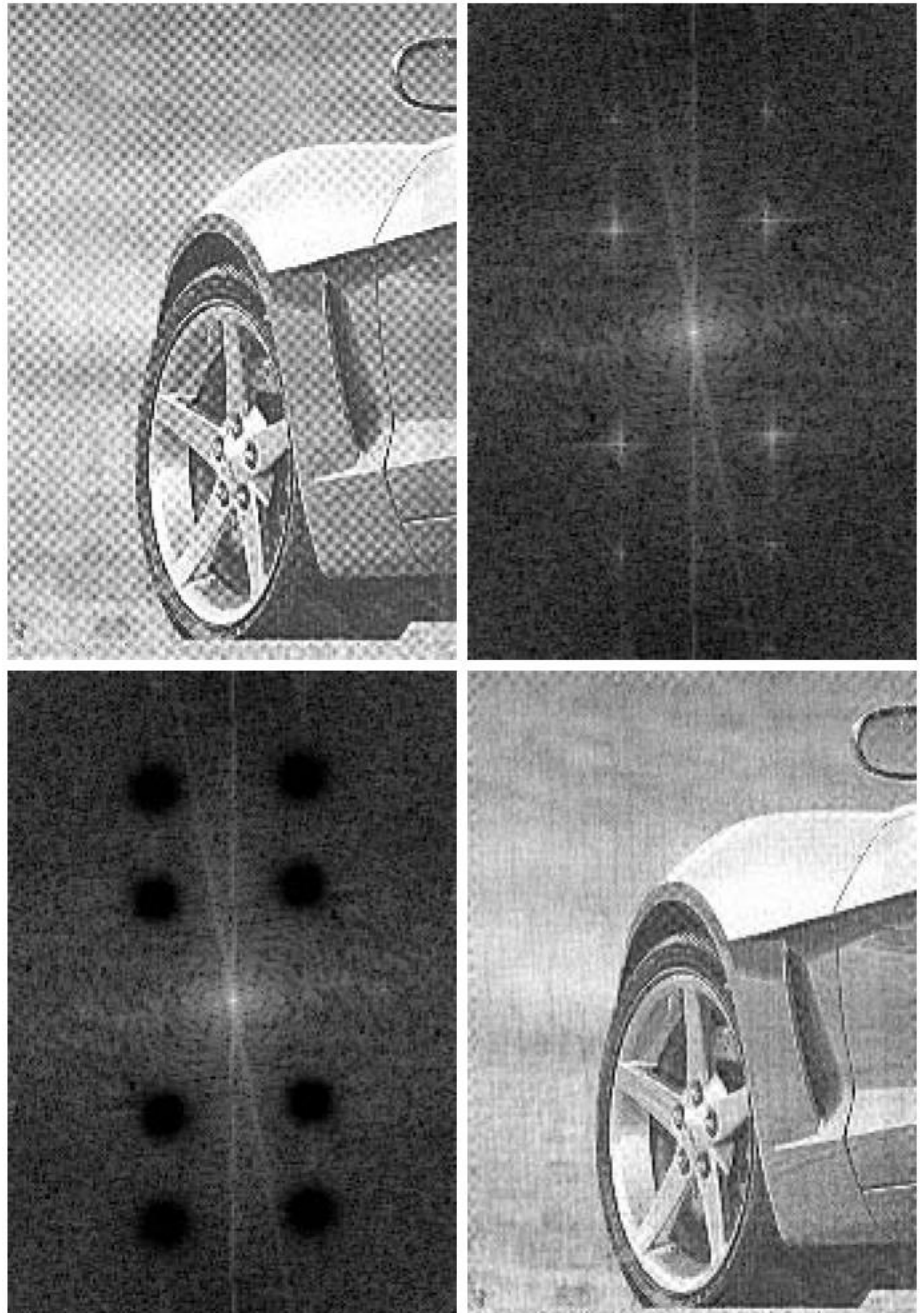
- Multiply mask with original image transform; this zeros out noise frequencies

  $$G(u,v) = M(u,v)\, F(u,v)$$

- Take inverse Fourier transform to get restored image

  $$g(x,y) = \mathcal{F}^{-1}(G(u,v))$$

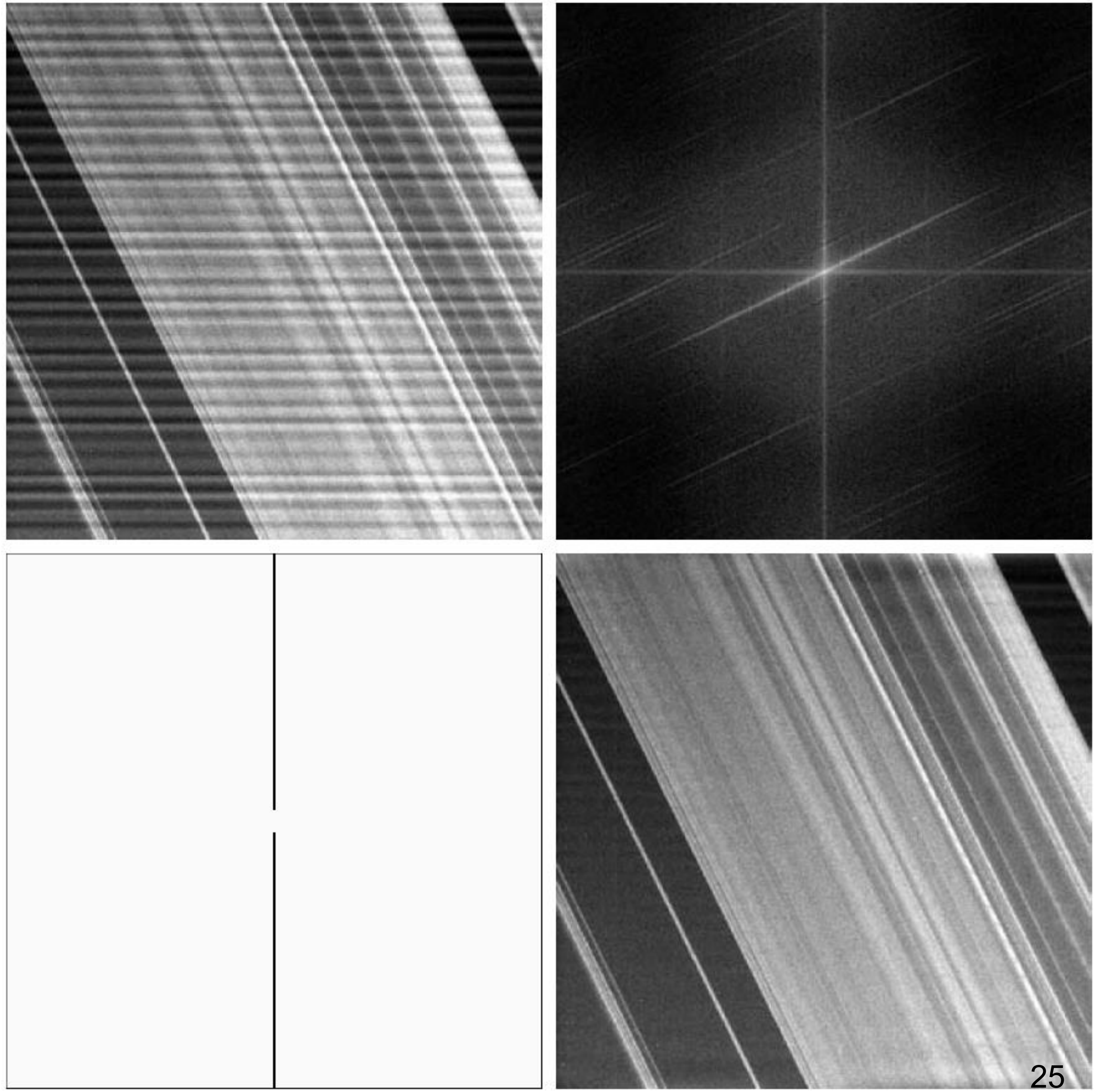




a b
c d

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

- Example of horizontal scan lines
- Create a notch of vertical lines in frequency domain



a b
c d

FIGURE 4.65 (a) 674 × 674 image of the 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. (d) Result of filtering. The thin black border in (c) was added for clarity; it is not part of the data. (Original image courtesy of Dr. Robert A. West,