# Linear Filtering – Part II

Selim Aksoy
Department of Computer Engineering
Bilkent University
saksoy@cs.bilkent.edu.tr

# Fourier theory

- Jean Baptiste Joseph Fourier had a crazy idea: Any periodic function can be written as a weighted sum of sines and cosines of different frequencies (1807).

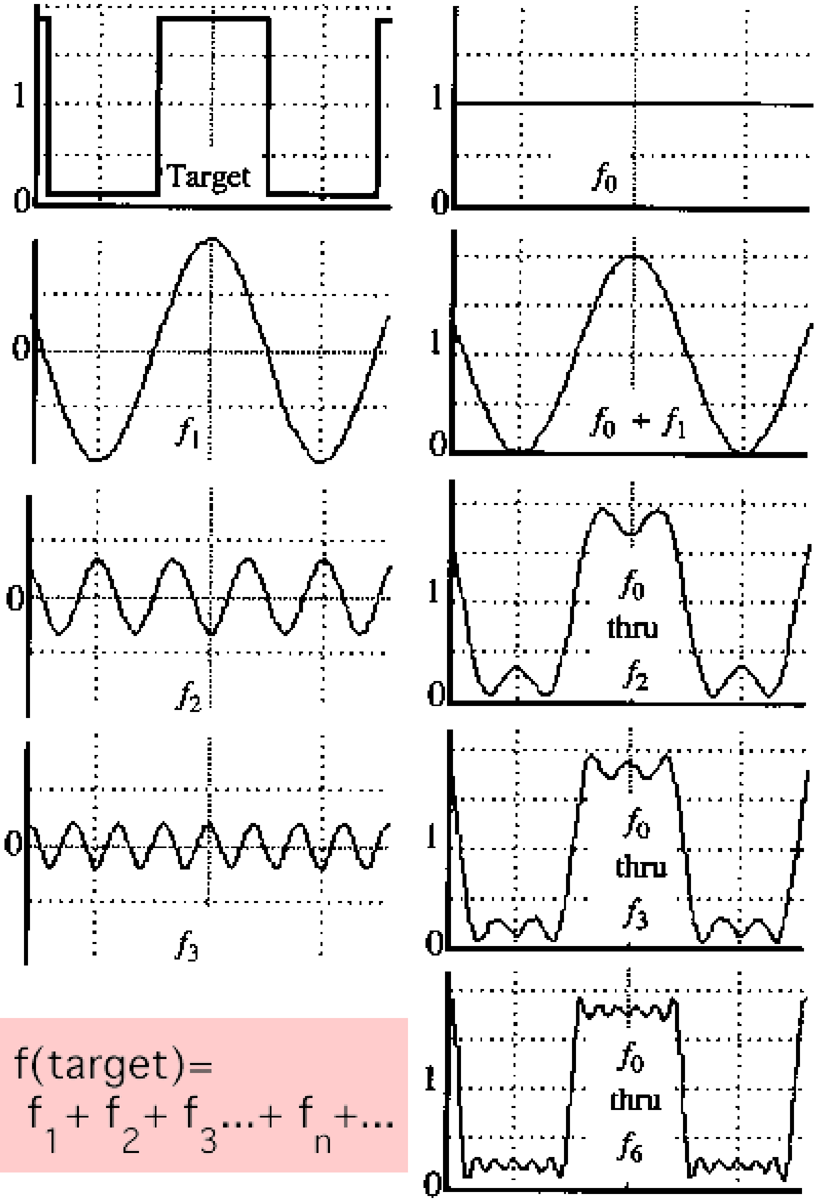
  → Fourier series

- Even functions that are not periodic (but whose area under the curve is finite) can be expressed as the integral of sines and cosines multiplied by a weighing function.

  → Fourier transform

# Fourier theory

- The Fourier theory shows how most real functions can be represented in terms of a basis of sinusoids.
- The building block:
  - A sin( ωx + Φ )
- Add enough of them to get any signal you want.



f(target)= $f_1 + f_2 + f_3 ... + f_n + ...$

Adapted from Alexei Efros, CMU

# Fourier transform

- The *Fourier transform*, $F(u)$, of a single variable, continuous function, $f(x)$, is defined by

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

- Given $F(u)$, we can obtain $f(x)$ using the *inverse Fourier transform*

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

# Fourier transform

- The *discrete Fourier transform (DFT)*, $F(u)$, of a discrete function of one variable, $f(x)$, $x = 0, 1, 2, \ldots, M-1$, is defined by

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

for $u = 0, 1, 2, \ldots, M-1$.

- Given $F(u)$, we can obtain the original function back using the *inverse DFT*

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

for $x = 0, 1, 2, \ldots, M-1$.

# Fourier transform

- These formulas can be extended for functions of two variables.
- Fourier transform:

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

- Inverse Fourier transform:

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

# Fourier transform

- Discrete Fourier transform:

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

for $u = 0, 1, 2, \ldots, M-1,\ v = 0, 1, 2, \ldots, N-1.$

- Inverse discrete Fourier transform:

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

for $x = 0, 1, 2, \ldots, M-1,\ y = 0, 1, 2, \ldots, N-1.$

# Fourier transform

- $F(u,v)$ can also be expressed in polar coordinates as

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

where

$$|F(u,v)| = \left(\Re^2\{F(u,v)\} + \Im^2\{F(u,v)\}\right)^{1/2}$$

is called the *magnitude* or *spectrum* of the Fourier transform, and

$$\phi(u,v) = \tan^{-1}\left(\frac{\Im\{F(u,v)\}}{\Re\{F(u,v)\}}\right)$$

is called the *phase angle* or *phase spectrum*.

- $\Re\{F(u,v)\}$ and $\Im\{F(u,v)\}$ are the real and imaginary parts of $F(u,v)$, respectively.

# Fourier transform

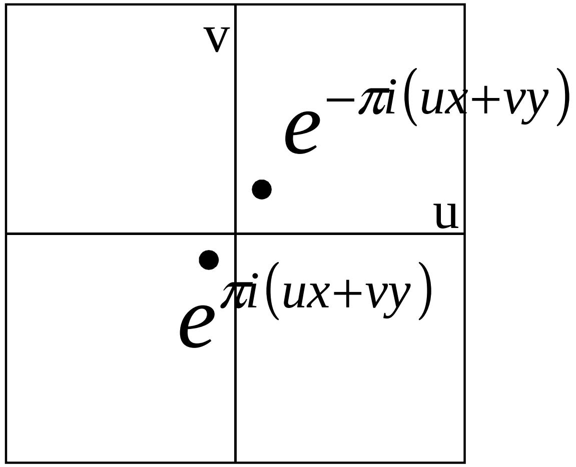
- The spectrum need not be interpreted as an image, but rather as a 2D display of the power in the original image versus the frequency components $u$ and $v$.
- The value $F(0,0)$ is the average of $f(x,y)$.
- Fourier transform is conjugate symmetric $(F(u,v) = F^*(-u,-v))$ and its spectrum is symmetric about the origin $(|F(u,v)| = |F(-u,-v)|)$ (when $f(x,y)$ is real).
- Usually the input image function is multiplied by $(-1)^{x+y}$ prior to computing the Fourier transform so that

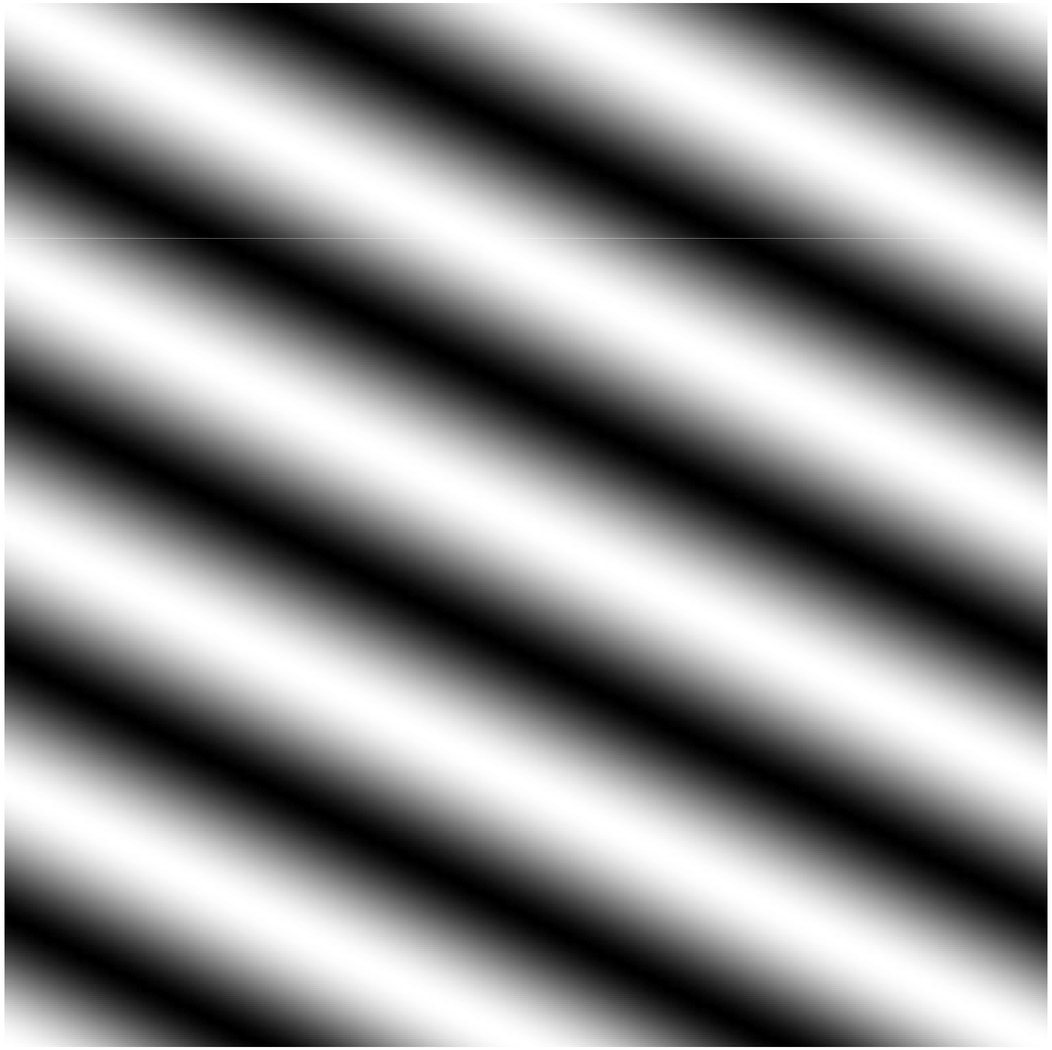
  $$\mathfrak{F}[f(x,y)\,(-1)^{x+y}] = F(u-M/2, v-N/2).$$

  The origin of the Fourier transform is located at $u = M/2$ and $v = N/2$.

# Fourier transform

To get some sense of what basis elements look like, we plot a basis element --- or rather, its real part --- as a function of x,y for some fixed u, v. We get a function that is constant when (ux+vy) is constant. The magnitude of the vector (u, v) gives a frequency, and its direction gives an orientation. The function is a sinusoid with this frequency along the direction, and constant perpendicular to the direction.

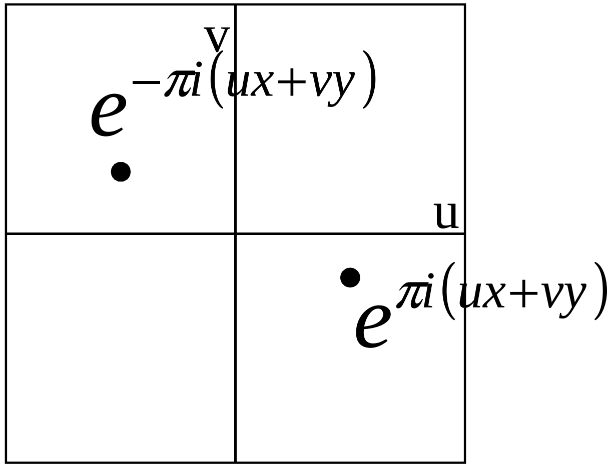


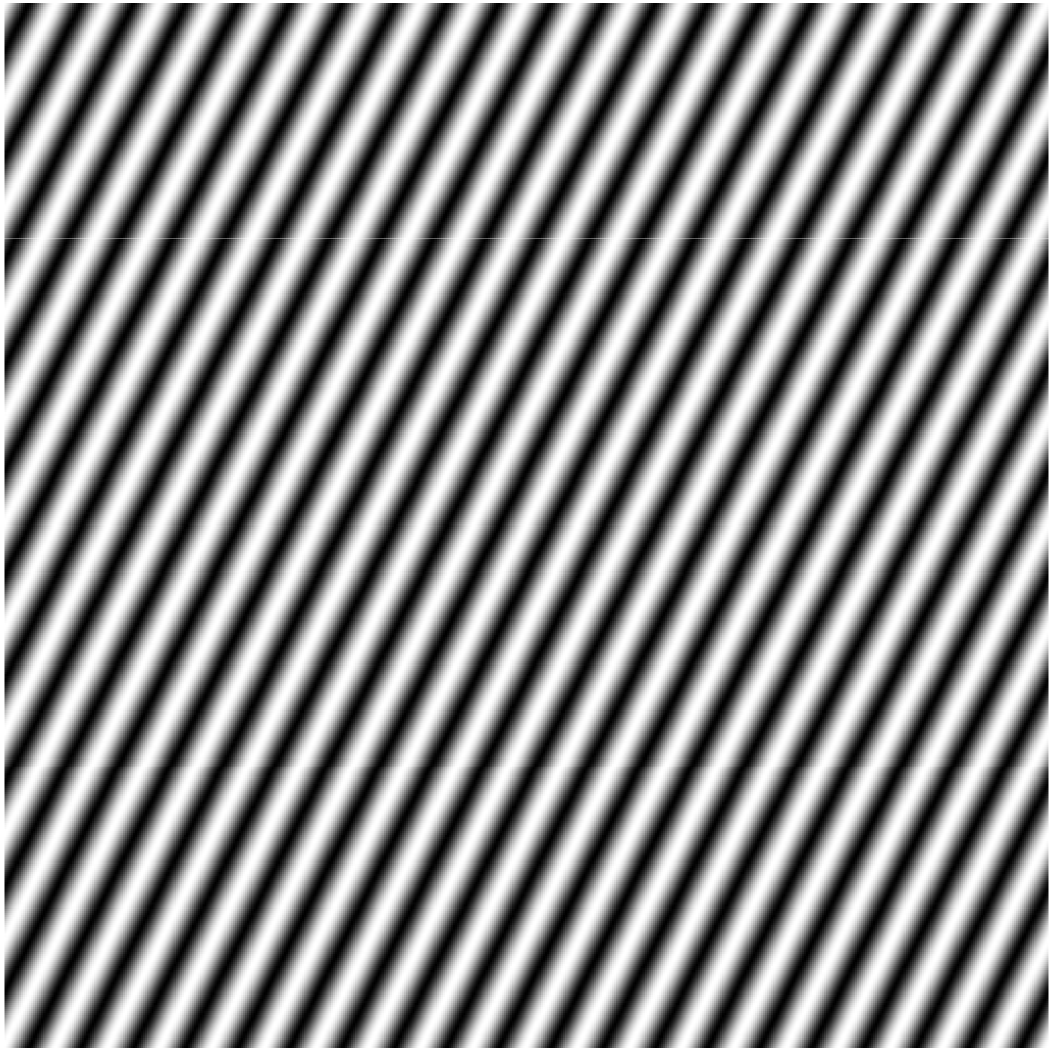


Adapted from Antonio Torralba

# Fourier transform

Here u and v are larger than in the previous slide.

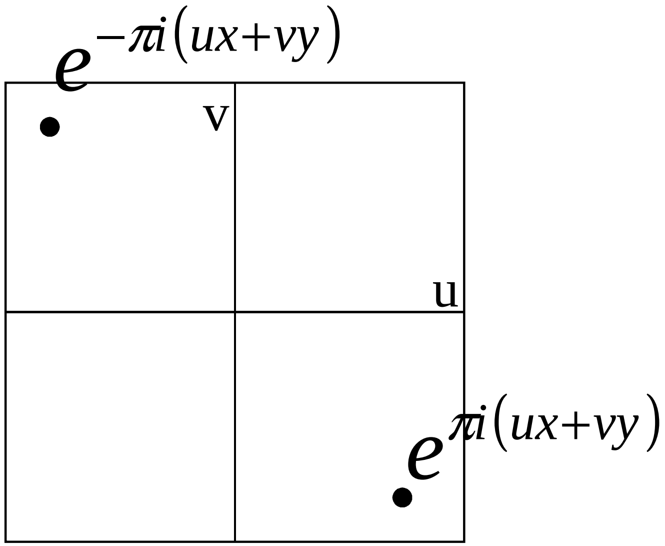


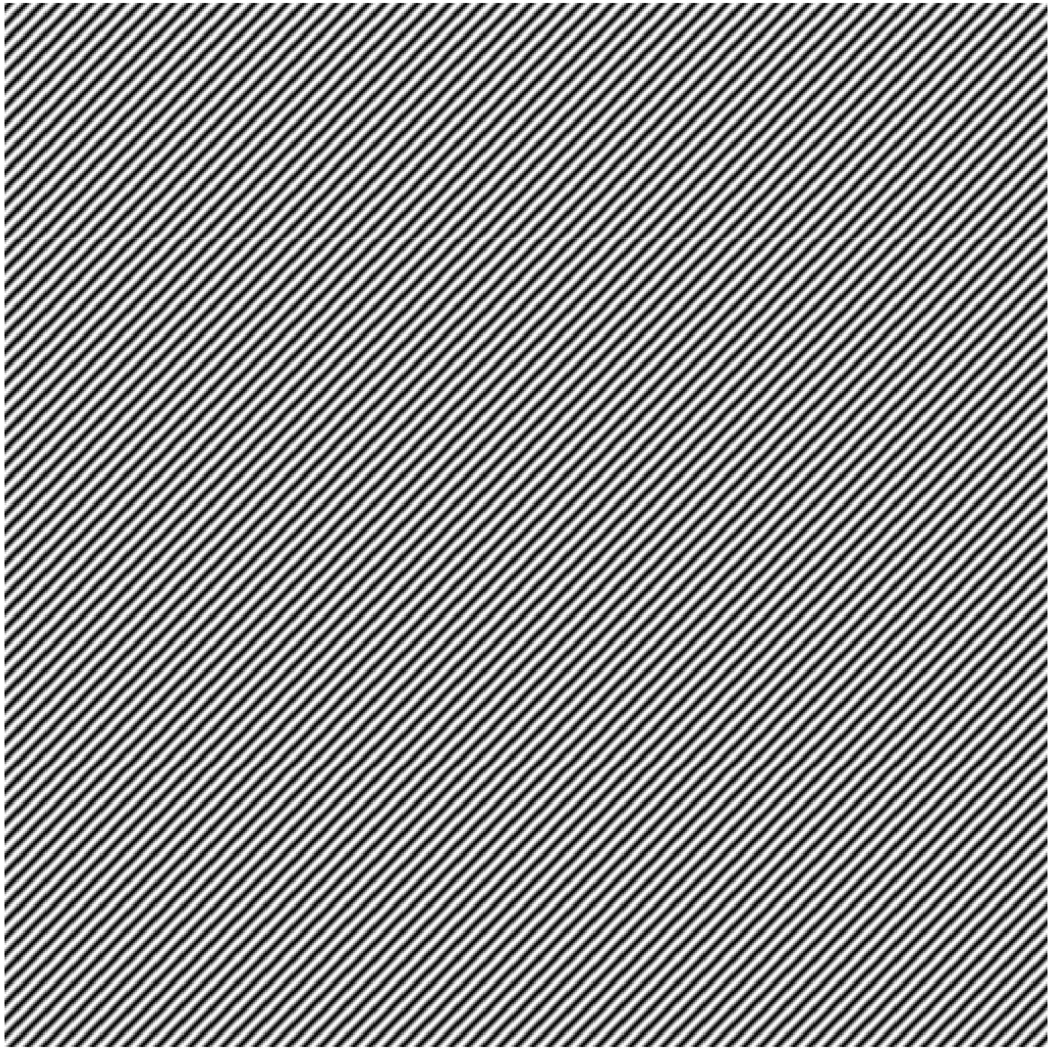


Adapted from Antonio Torralba

# Fourier transform

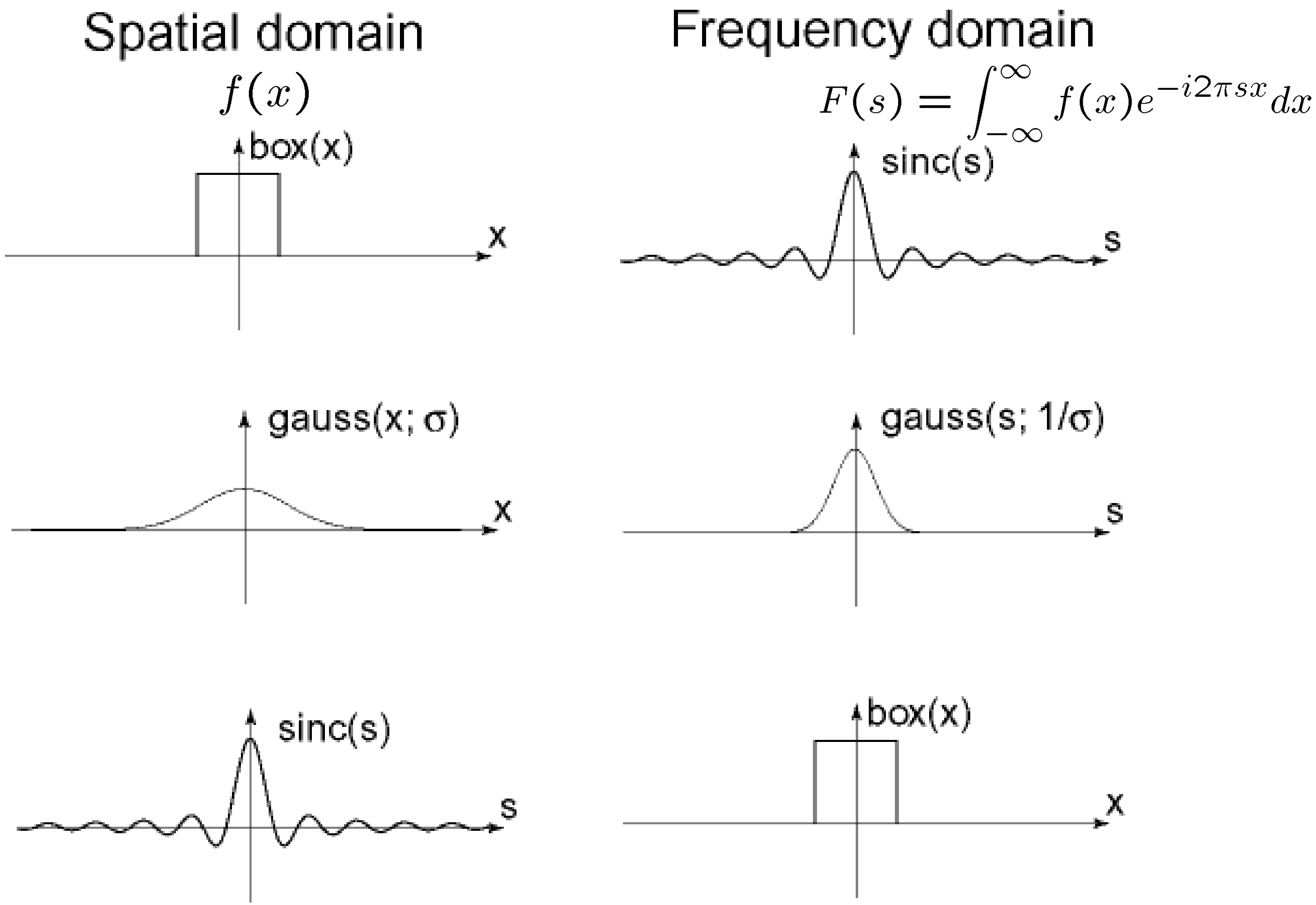
And larger still…





Adapted from Antonio Torralba

# Fourier transform



Spatial domain

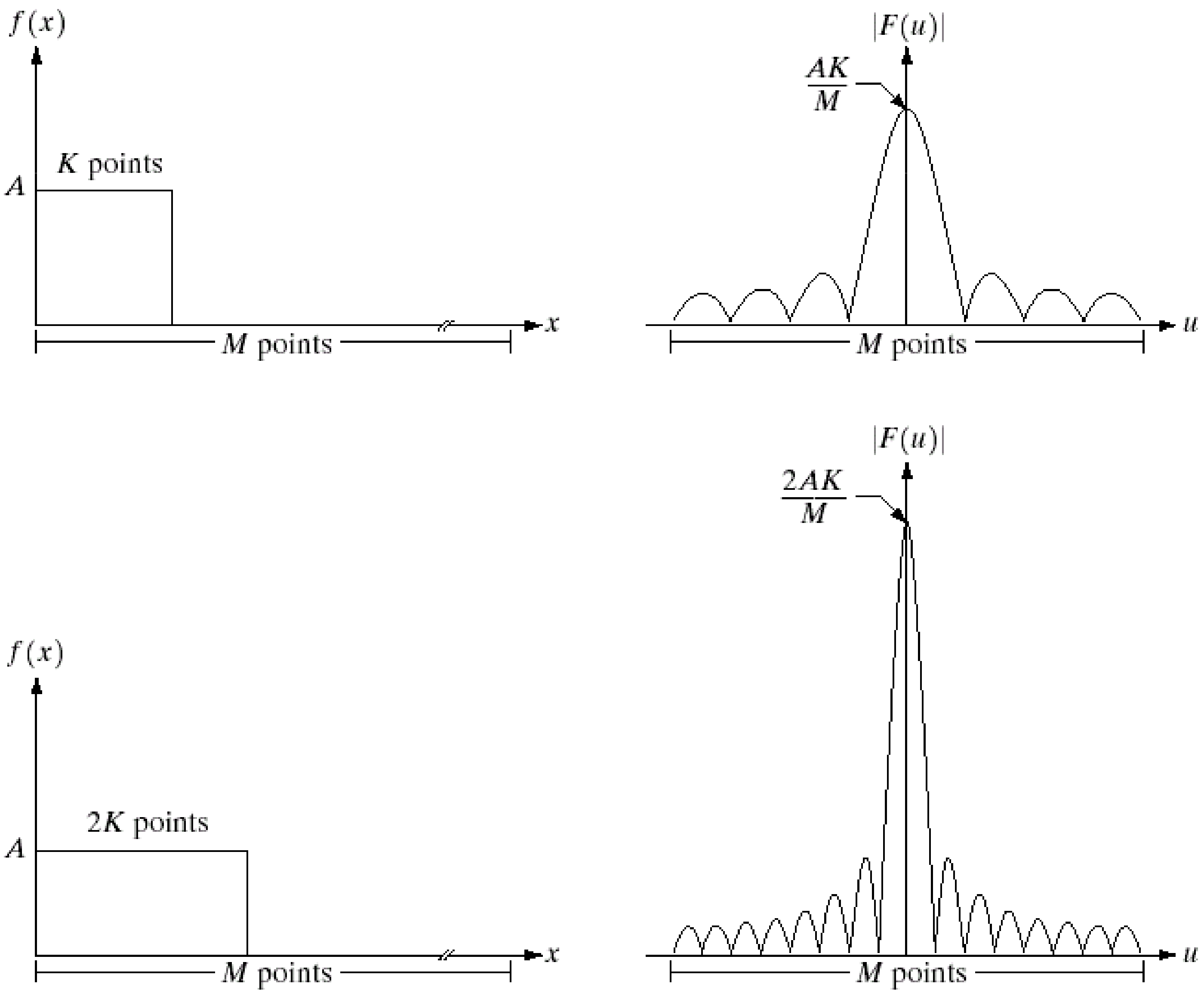
$$f(x)$$

Frequency domain

$$F(s) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi sx} dx$$

Adapted from Alexei Efros, CMU

# Fourier transform



a b
c d

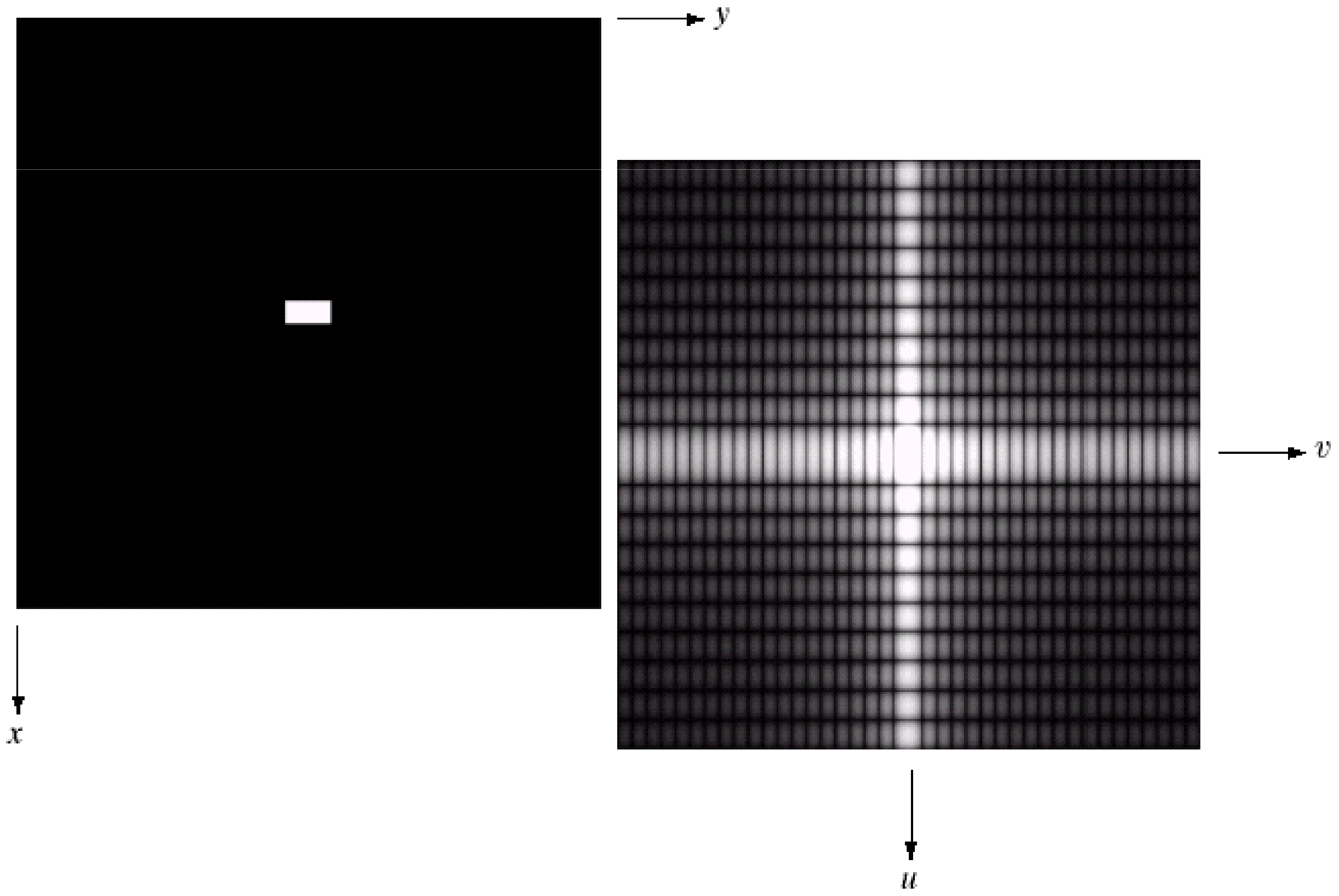
**FIGURE 4.2** (a) A discrete function of *M* points, and (b) its Fourier spectrum. (c) A discrete function with twice the number of nonzero points, and (d) its Fourier spectrum.

Adapted from Gonzales and Woods

# Fourier transform

a b

**FIGURE 4.3** (a) Image of a 20 × 40 white rectangle on a black background of size 512 × 512 pixels. (b) Centered Fourier spectrum shown after application of the log transformation given in Eq. (3.2-2). Compare with Fig. 4.2.



Adapted from Gonzales and Woods

# Fourier transform

- The *power spectrum* is defined as the square of the Fourier spectrum:

$$\begin{aligned} P(u,v) &= |F(u,v)|^2 \\ &= \Re^2\{F(u,v)\} + \Im^2\{F(u,v)\}. \end{aligned}$$

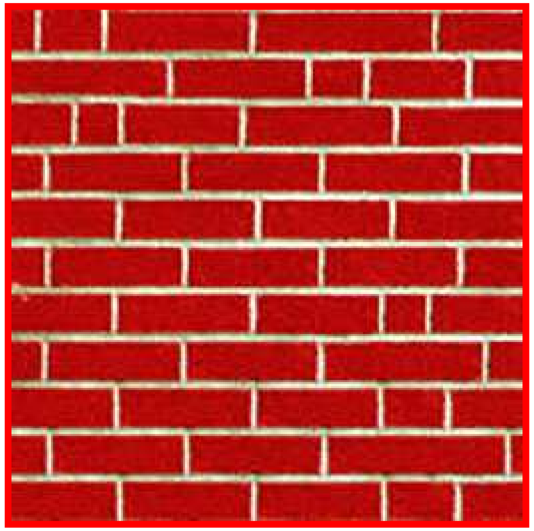
- The radial distribution of values in the Fourier spectrum of an image is sensitive to texture coarseness in that image.
  - A coarse texture will have high values concentrated near the origin of the spectrum.
  - A fine texture will cause the values to be spread out.

# Fourier transform

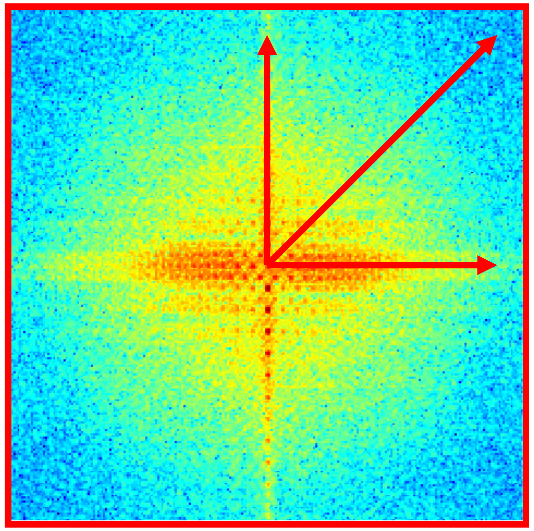
- The angular distribution of values in the spectrum is sensitive to the directionality of the texture in the image.
  - A texture with many edges in a given direction $\theta$ will have high values of the spectrum concentrated around the perpendicular direction $\theta + \pi/2$.
  - For a non-directional texture, the spectrum is also non-directional.
- We will come back to this when we talk about texture.

# Fourier transform

## How to interpret a Fourier spectrum:



Vertical orientation



45 deg.

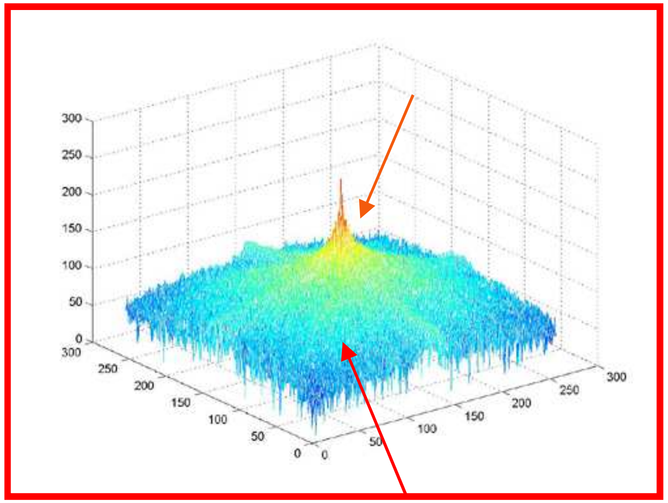
Horizontal orientation

0

0 fmax

fx in cycles/image

**Low spatial frequencies**

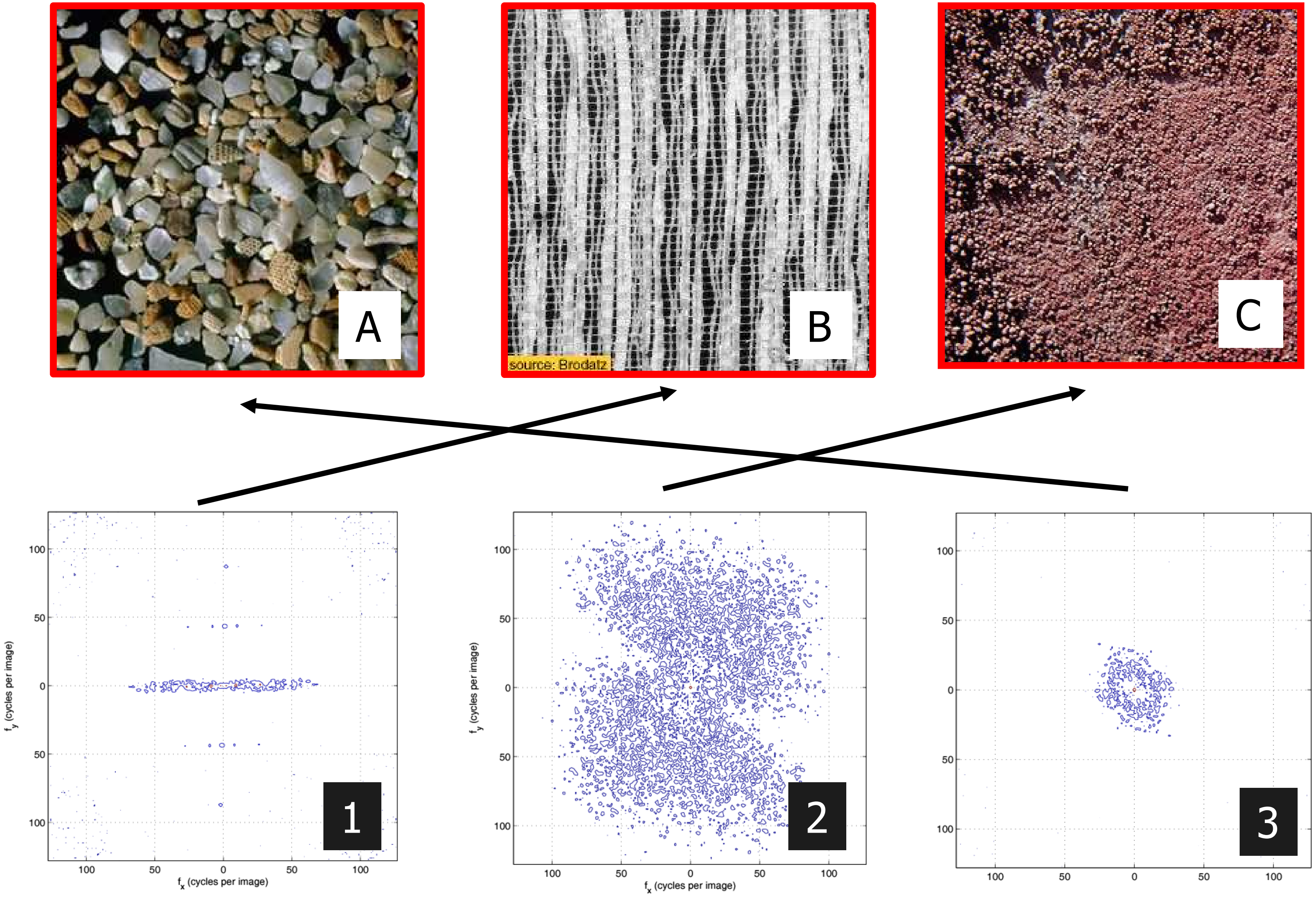


**High spatial frequencies**

Log power spectrum

Adapted from Antonio Torralba

# Fourier transform



Adapted from Antonio Torralba

# Fourier transform

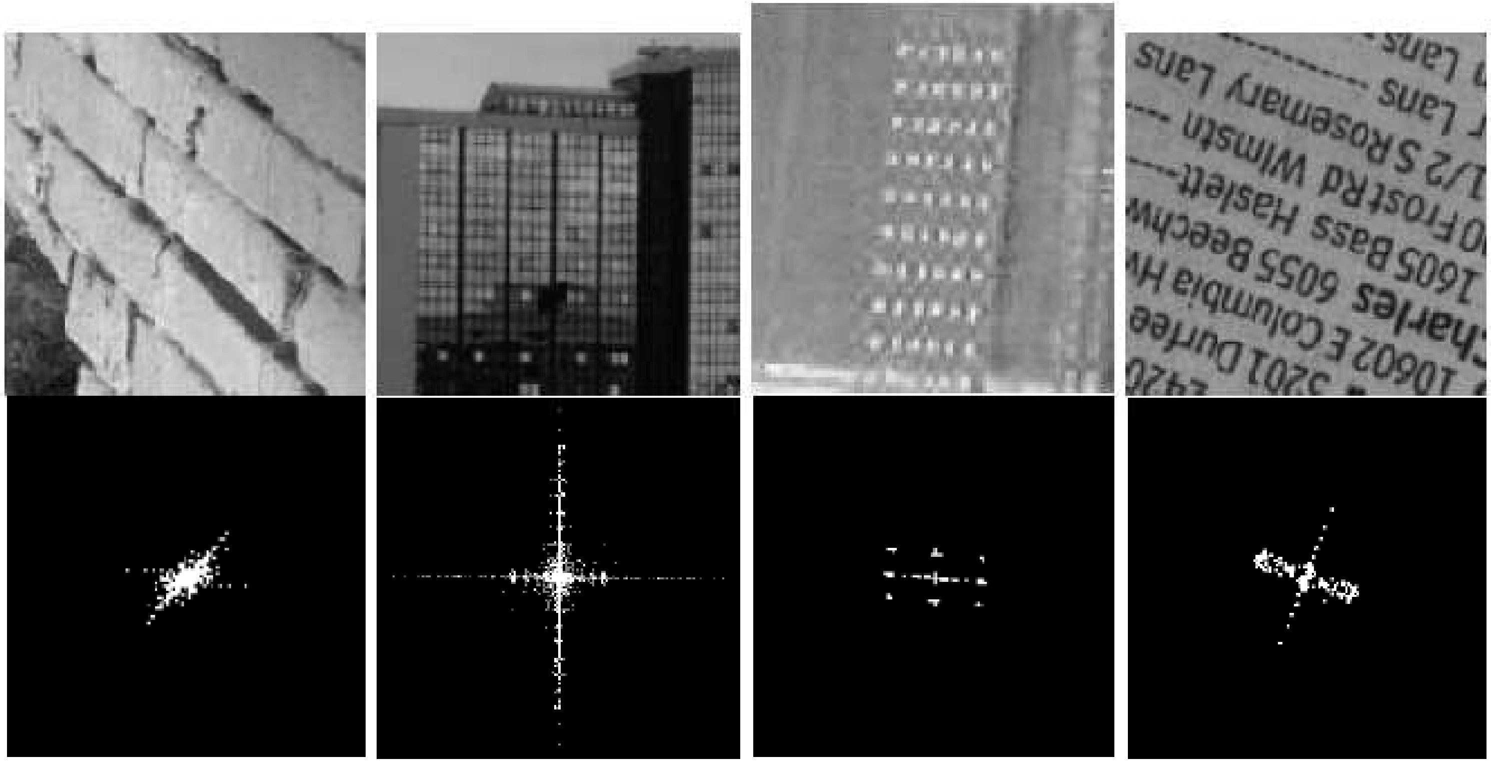
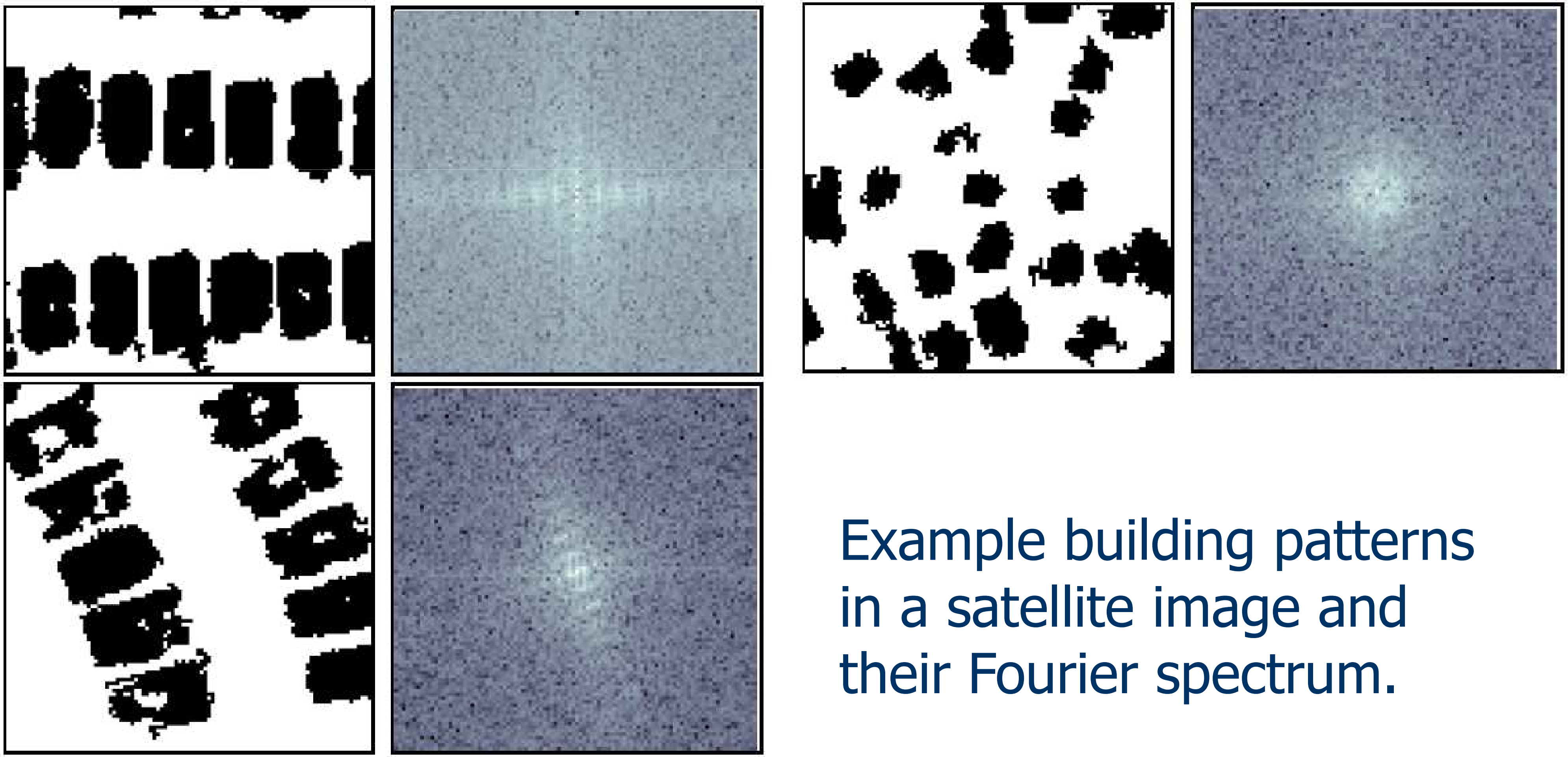


Figure 5.42: Four images (above) and their power spectrums (below). The power spectrum of the brick texture shows energy in many sinusoids of many frequencies, but the dominant direction is perpendicular to the 6 dark seams running about 45 degrees with the $X$-axis. There is noticable energy at 0 degrees with the $X$ axis, due to the several short vertical seams. The power spectrum of the building shows high frequency energy in waves along the $X$-direction and the $Y$-direction. The third image is an aerial image of an orchard: the power spectrum shows the rows and columns of the orchard and also the "diagnonal rows". The far right image, taken from a phone book, shows high frequency power at about $60^o$ with the $X$-axis, which represents the texture in the lines of text. Energy is spread more broadly in the perpendicular direction also in order to model the characters and their spacing.

Adapted from Shapiro and Stockman

# Fourier transform



Example building patterns in a satellite image and their Fourier spectrum.

# Convolution theorem

- The discrete *convolution* of two functions $f(x,y)$ and $h(x,y)$ of size $M \times N$ is defined as

$$f(x,y) \star h(x,y) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n)\, h(x-m, y-n).$$

- This is equivalent to the *correlation* of $f(x,y)$ with $h(x,y)$ flipped about the origin.

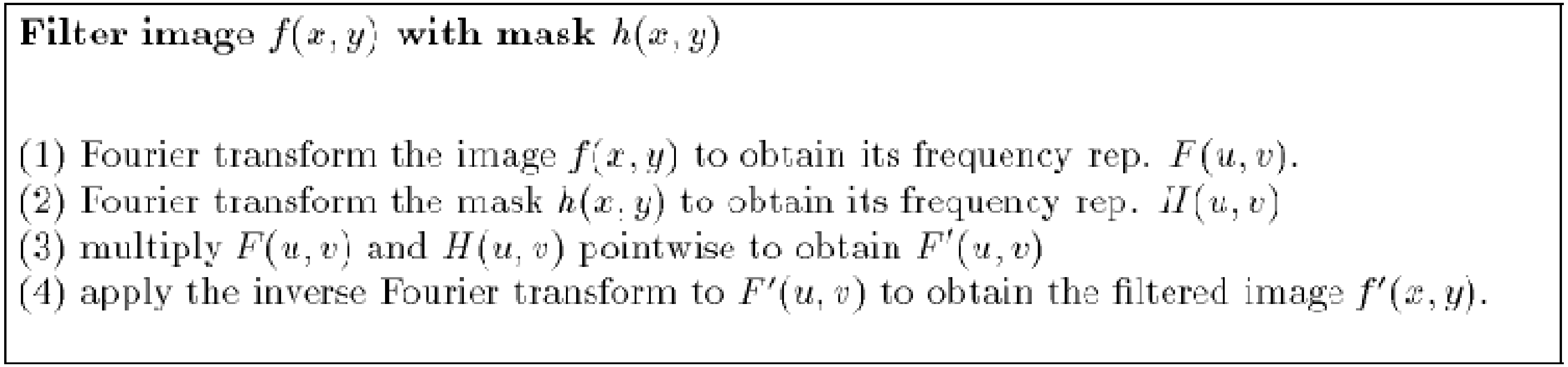
- Convolution theorem:

$$f(x,y) \star h(x,y) \Leftrightarrow F(u,v)\, H(u,v)$$

$$f(x,y)\, h(x,y) \Leftrightarrow F(u,v) \star H(u,v)$$

where "$\Leftrightarrow$" indicates a Fourier transform pair.

# Frequency domain filtering



**Filter image $f(x,y)$ with mask $h(x,y)$**

(1) Fourier transform the image $f(x,y)$ to obtain its frequency rep. $F(u,v)$.
(2) Fourier transform the mask $h(x,y)$ to obtain its frequency rep. $H(u,v)$
(3) multiply $F(u,v)$ and $H(u,v)$ pointwise to obtain $F'(u,v)$
(4) apply the inverse Fourier transform to $F'(u,v)$ to obtain the filtered image $f'(x,y)$.

**Algorithm 3:** Filtering image $f(x,y)$ with mask $h(x,y)$ using the Fourier transform

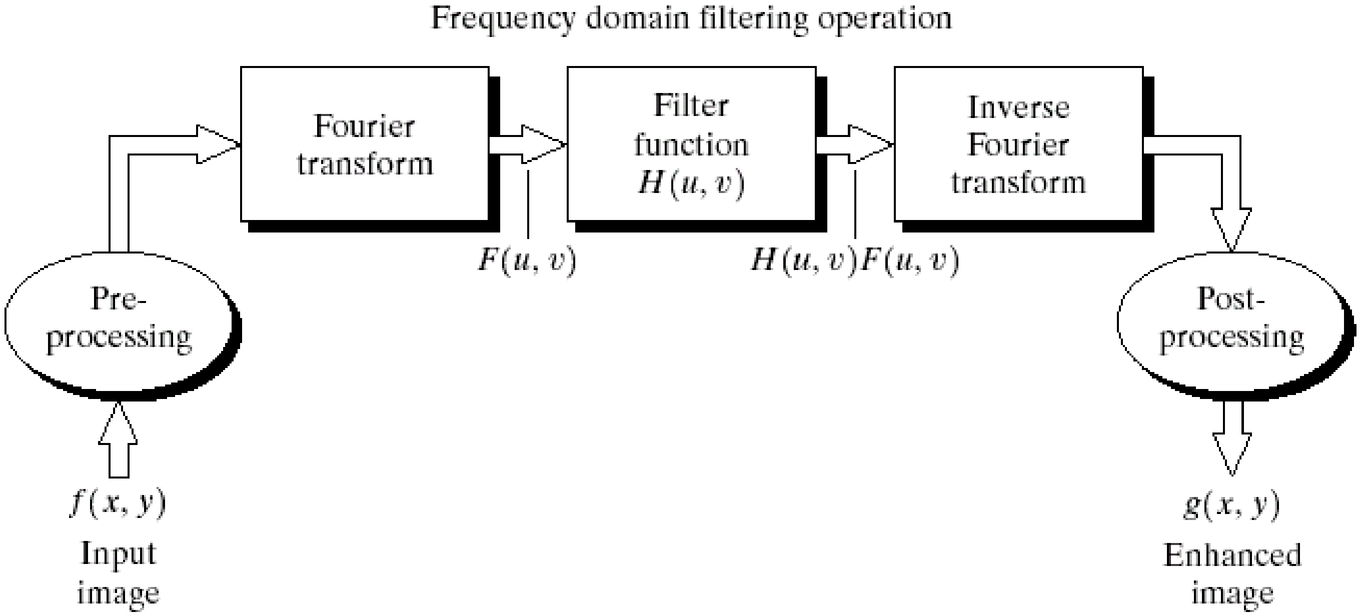
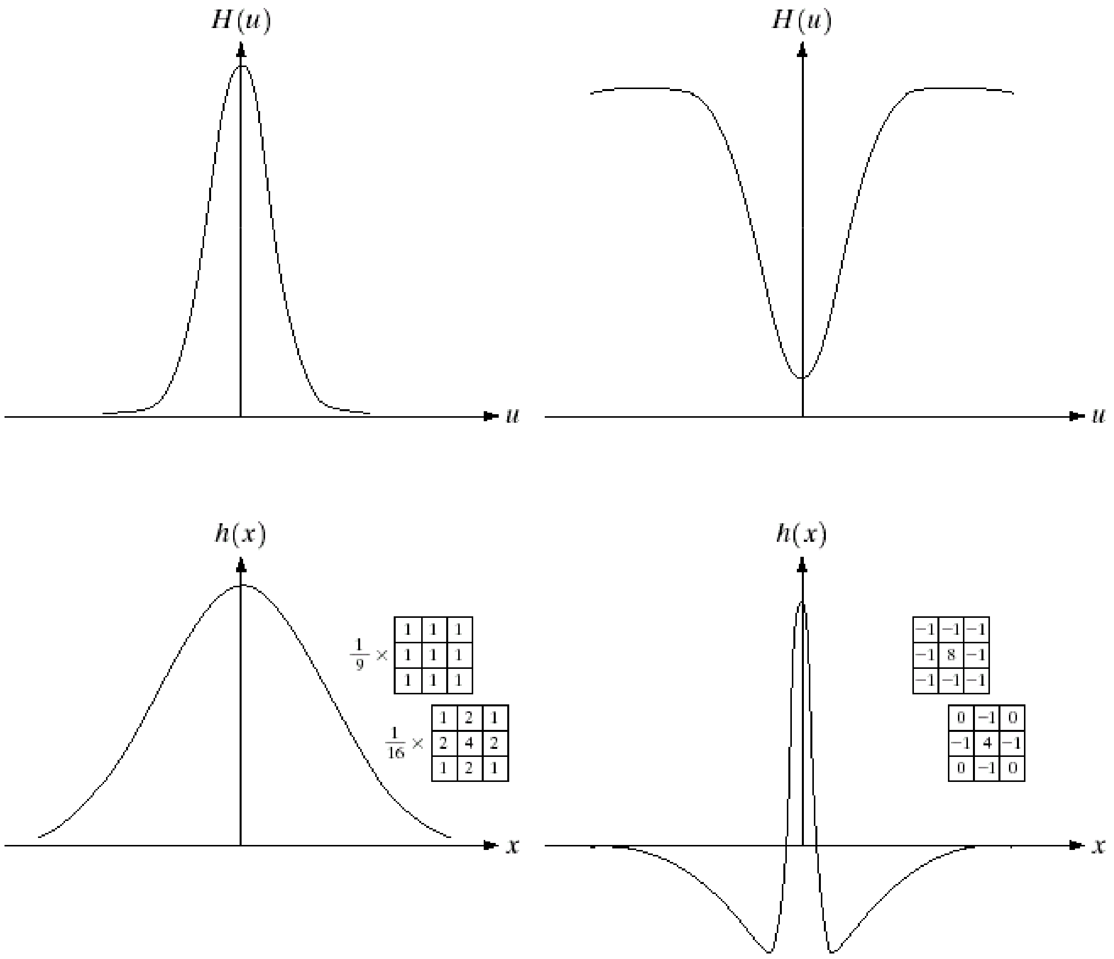


**FIGURE 4.5** Basic steps for filtering in the frequency domain.

Adapted from Shapiro and Stockman, and Gonzales and Woods

# Frequency domain filtering



$\frac{1}{9}\times\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$

$\frac{1}{16}\times\begin{bmatrix}1&2&1\\2&4&2\\1&2&1\end{bmatrix}$

$\begin{bmatrix}-1&-1&-1\\-1&8&-1\\-1&-1&-1\end{bmatrix}$

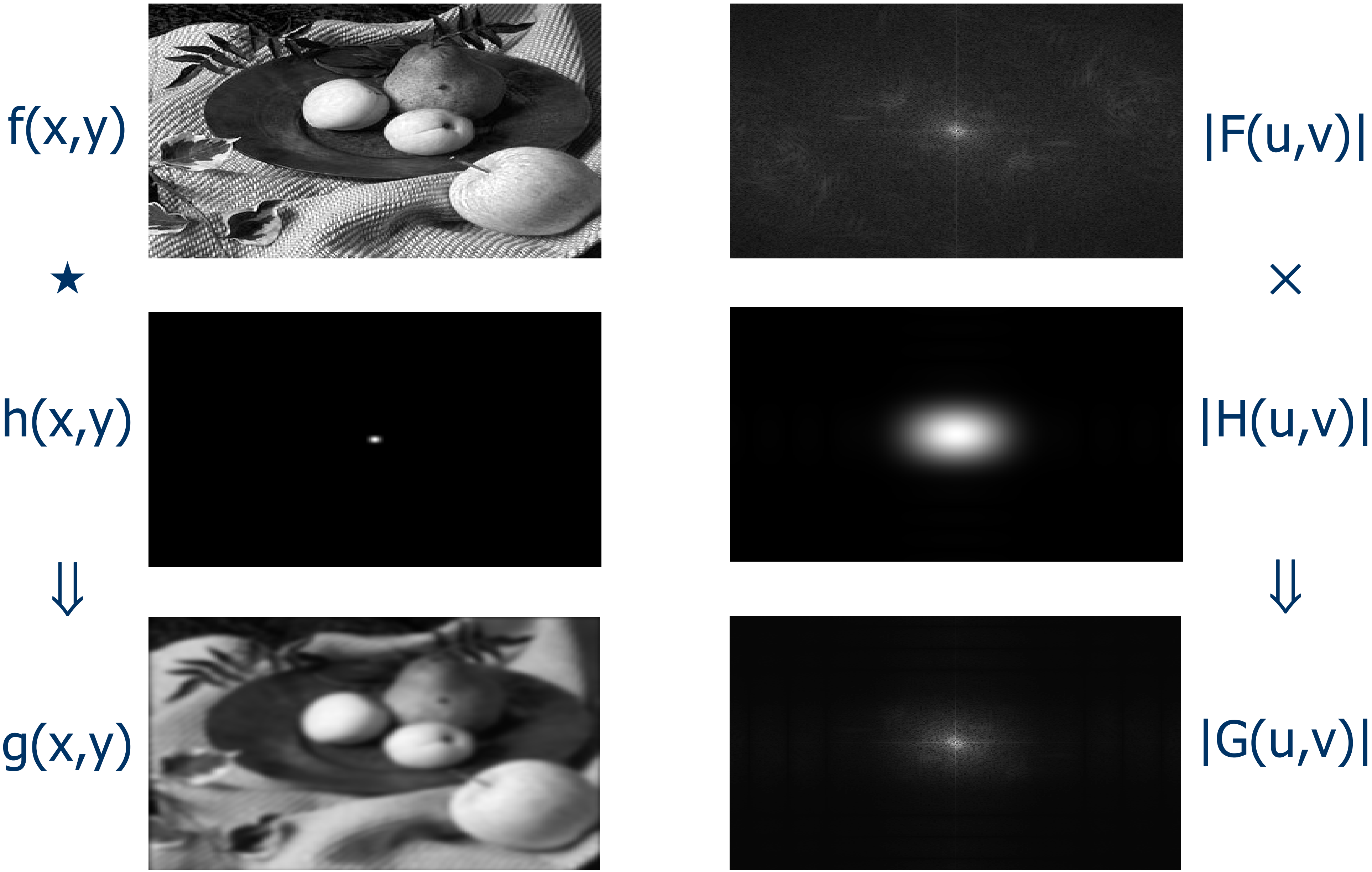
$\begin{bmatrix}0&-1&0\\-1&4&-1\\0&-1&0\end{bmatrix}$

a b
c d

**FIGURE 4.9** (a) Gaussian frequency domain lowpass filter. (b) Gaussian frequency domain highpass filter. (c) Corresponding lowpass spatial filter. (d) Corresponding highpass spatial filter. The masks shown are used in Chapter 3 for lowpass and highpass filtering.

- Since the discrete Fourier transform is periodic, padding is needed in the implementation to avoid aliasing (see section 4.6 in the Gonzales-Woods book for implementation details).

# Frequency domain filtering

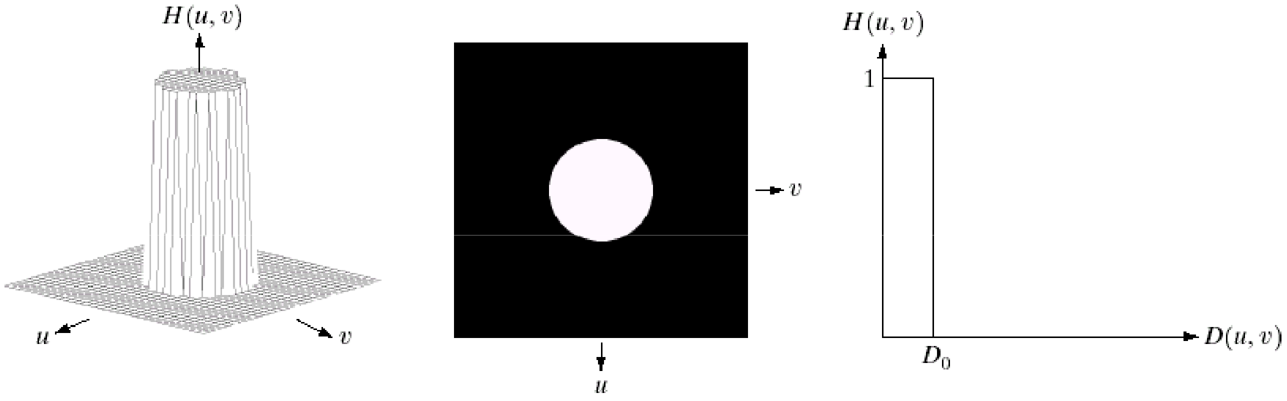


f(x,y) ★ h(x,y) ⇓ g(x,y)

|F(u,v)| × |H(u,v)| ⇓ |G(u,v)|

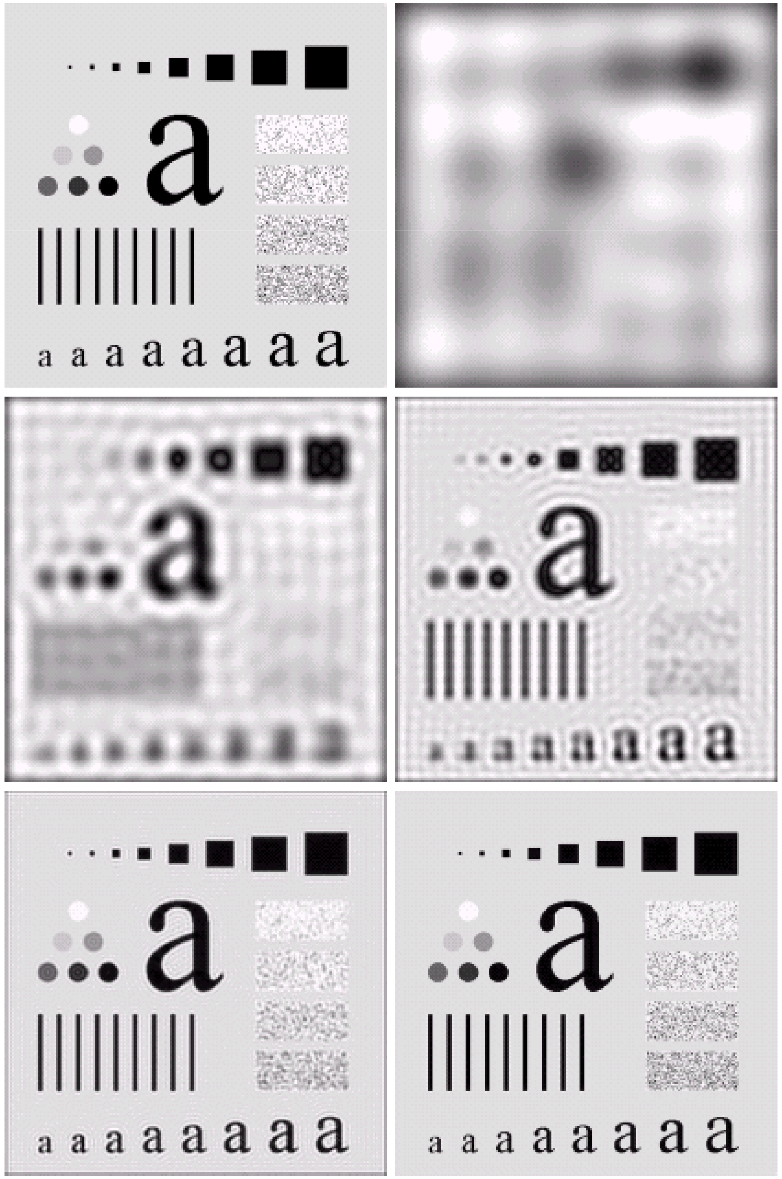
Adapted from Alexei Efros, CMU

# Smoothing frequency domain filters



a b c

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



a b
c d
e f

**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.

# Smoothing frequency domain filters

- The blurring and ringing caused by the ideal low-pass filter can be explained using the convolution theorem where the spatial representation of a filter is given below.

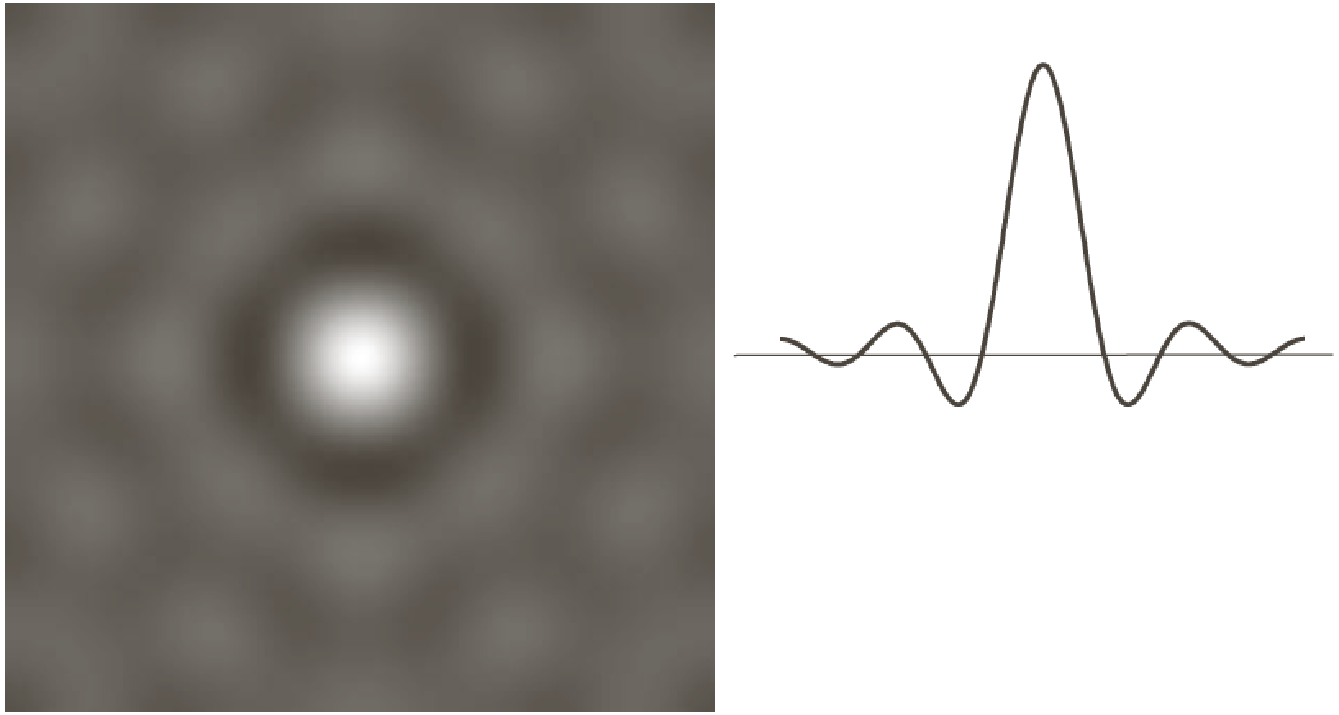
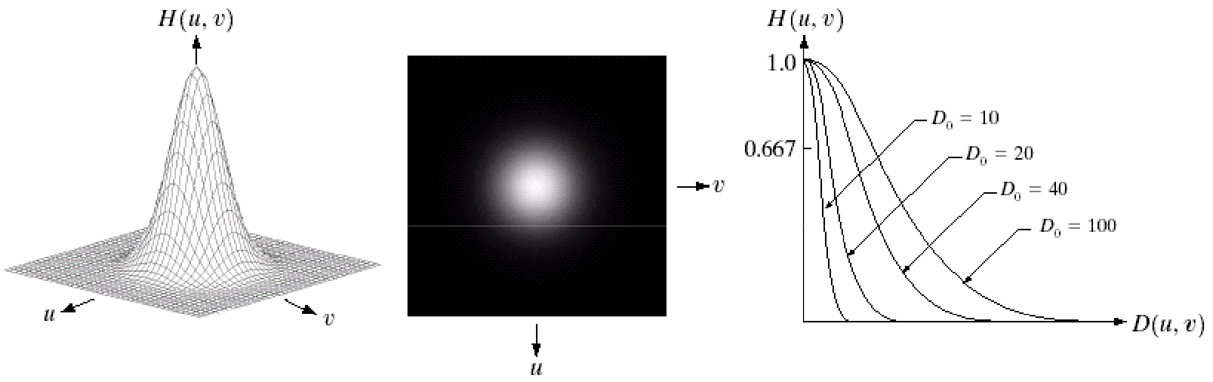


**FIGURE 4.43** (a) Representation in the spatial domain of an ILPF of radius 5 and size 1000 × 1000. (b) Intensity profile of a horizontal line passing through the center of the image.

# Smoothing frequency domain filters



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 $D_0$.

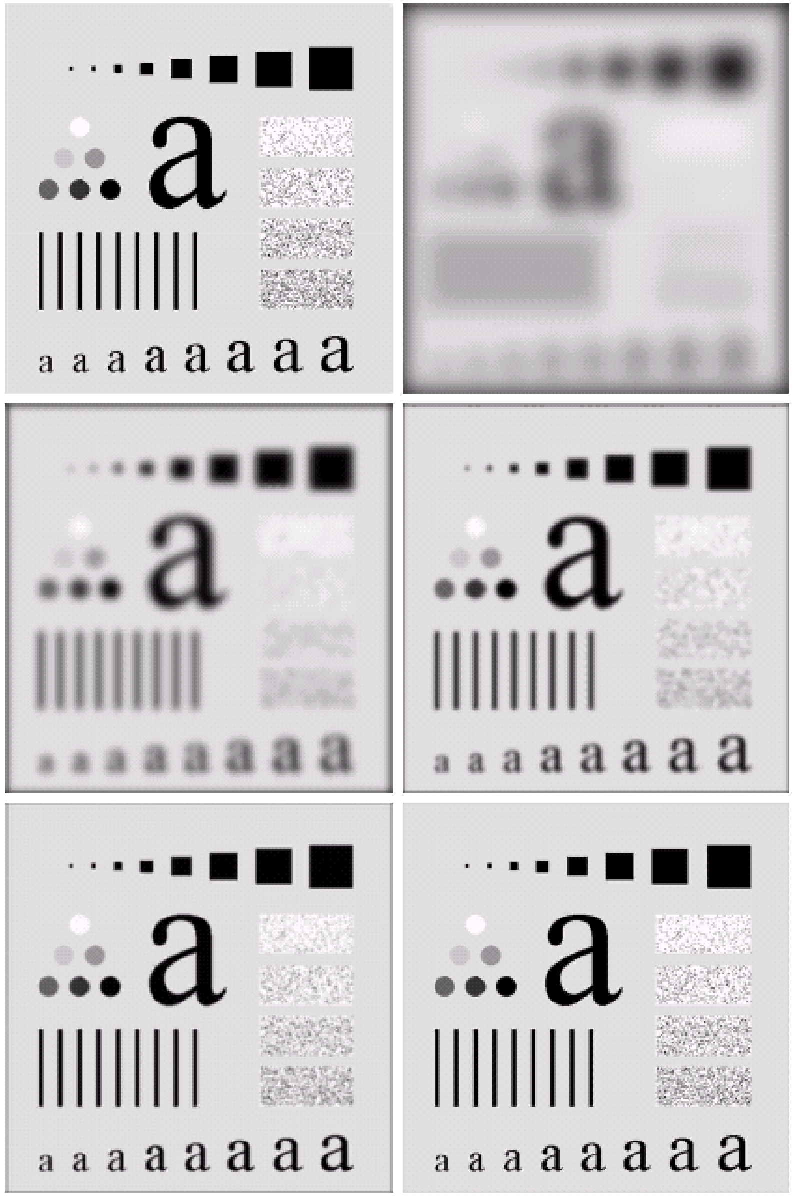


**FIGURE 4.18** (a) Original image. (b)–(f) Results of filtering with Gaussian lowpass filters with cutoff frequencies set at radii values of 5, 15, 30, 80, and 230, as shown in Fig. 4.11(b). Compare with Figs. 4.12 and 4.15.

a b
c d
e f

# Sharpening frequency domain filters

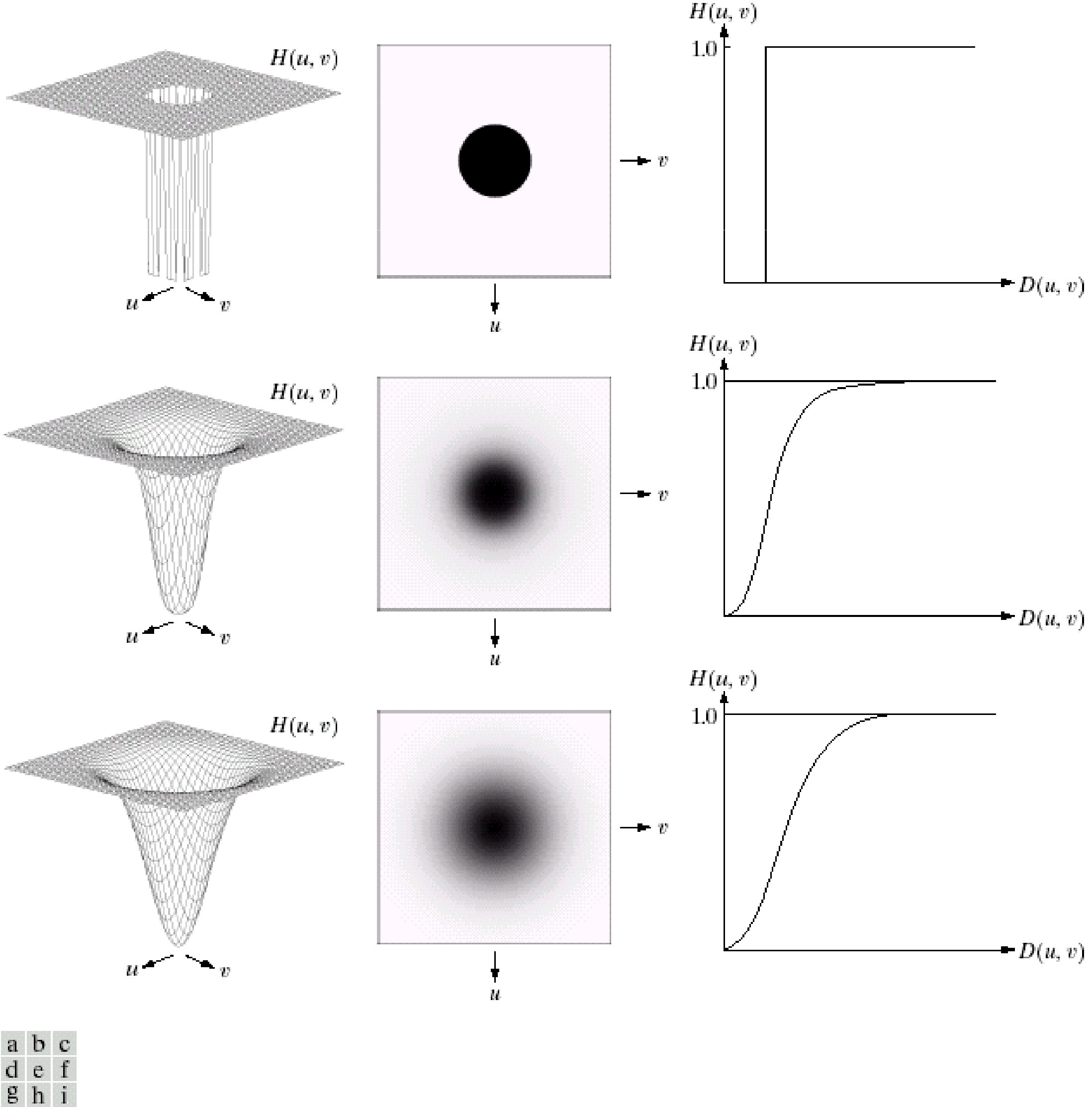


**FIGURE 4.22** Top row: Perspective plot, image representation, and cross section of a typical ideal highpass filter. Middle and bottom rows: The same sequence for typical Butterworth and Gaussian highpass filters.

# Sharpening frequency domain filters

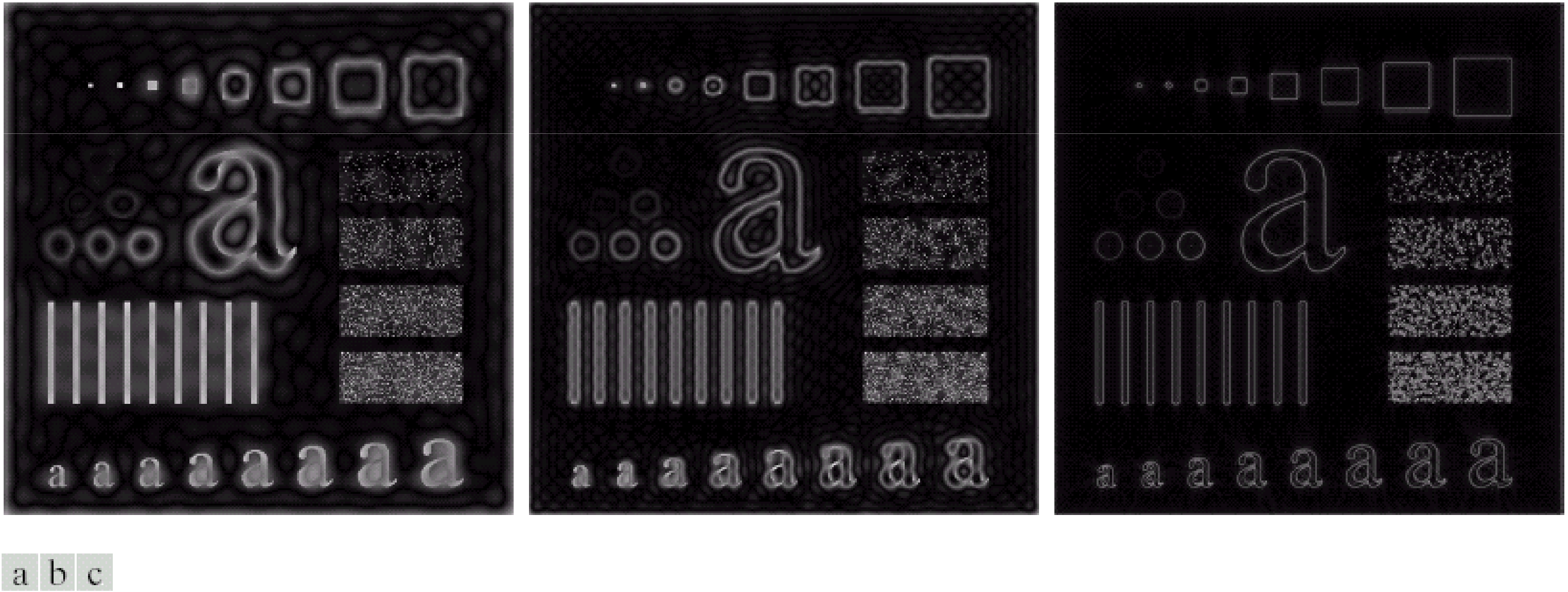


**FIGURE 4.24** Results of ideal highpass filtering the image in Fig. 4.11(a) with $D_0 = 15$, 30, and 80, respectively. Problems with ringing are quite evident in (a) and (b).

Adapted from Gonzales and Woods

# Sharpening frequency domain filters

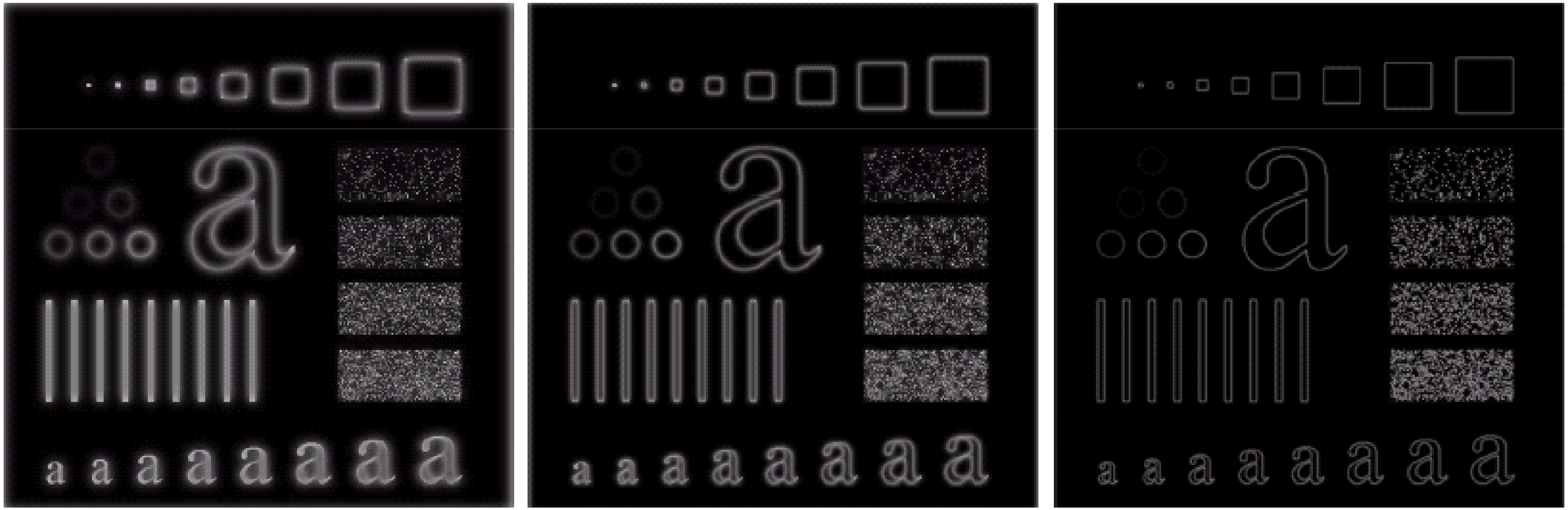


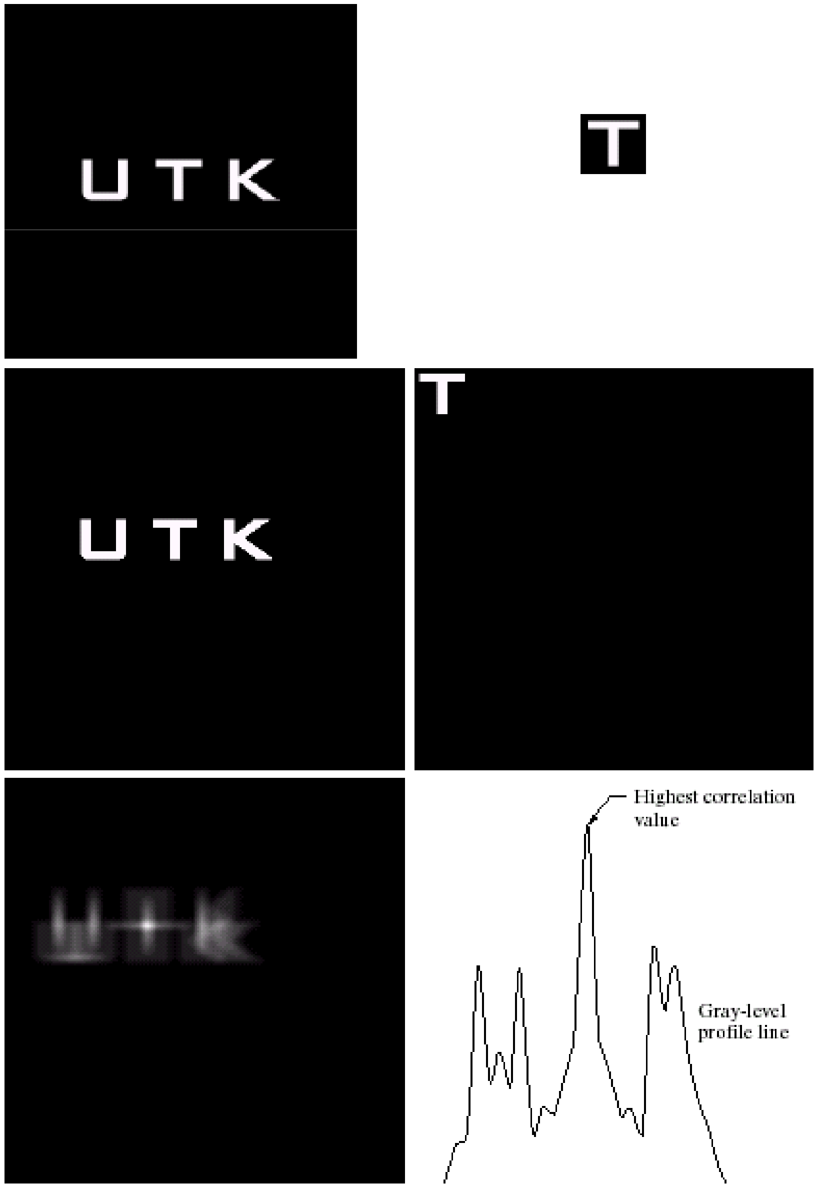
**FIGURE 4.26** Results of highpass filtering the image of Fig. 4.11(a) using a GHPF of order 2 with $D_0 = 15$, 30, and 80, respectively. Compare with Figs. 4.24 and 4.25.

Adapted from Gonzales and Woods

# Template matching

- Correlation can also be used for matching.
- If we want to determine whether an image f contains a particular object, we let h be that object (also called a template) and compute the correlation between f and h.
- If there is a match, the correlation will be maximum at the location where h finds a correspondence in f.
- Preprocessing such as scaling and alignment is necessary in most practical applications.

# Template matching

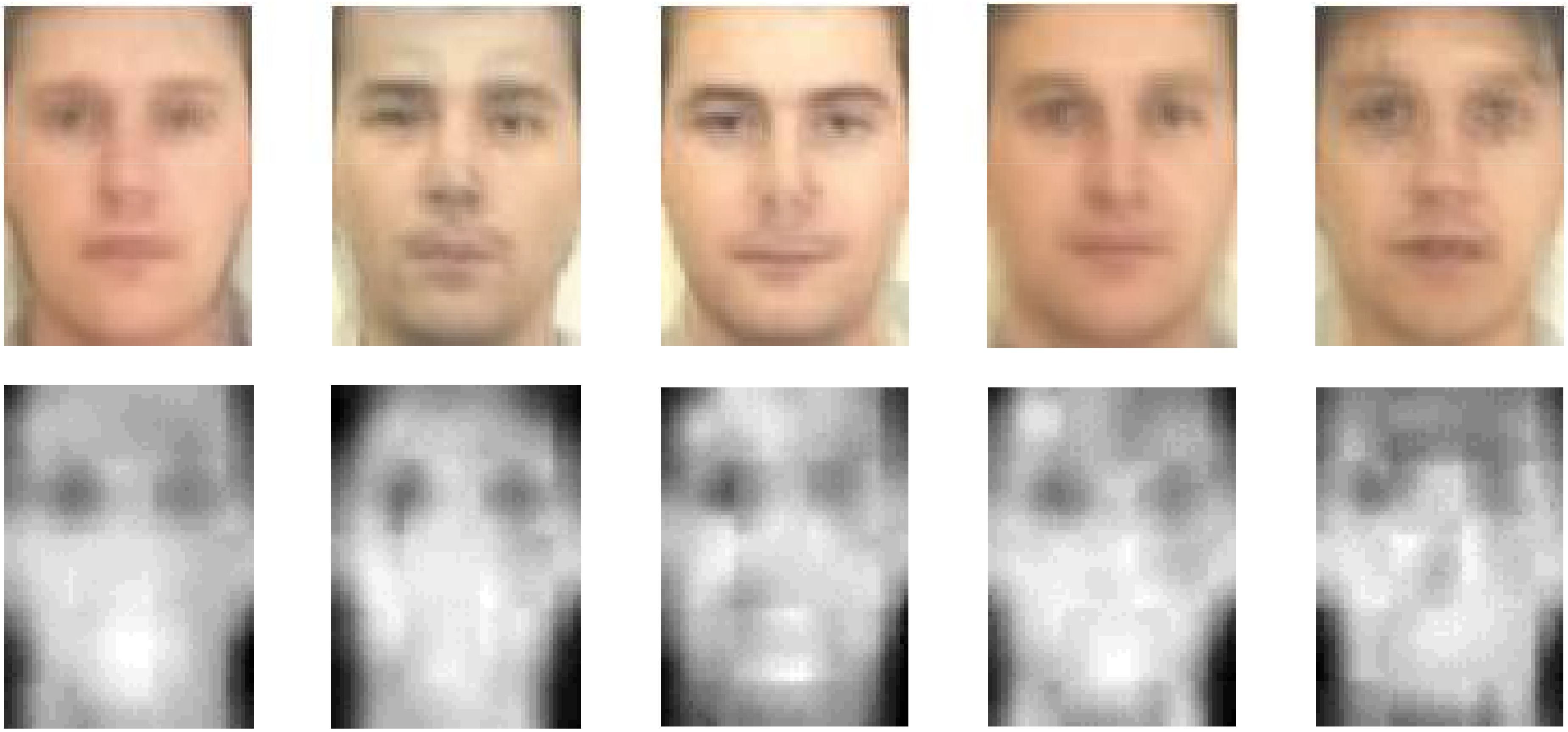


a b
c d
e f

**FIGURE 4.41** (a) Image. (b) Template. (c) and (d) Padded images. (e) Correlation function displayed as an image. (f) Horizontal profile line through the highest value in (e), showing the point at which the best match took place.

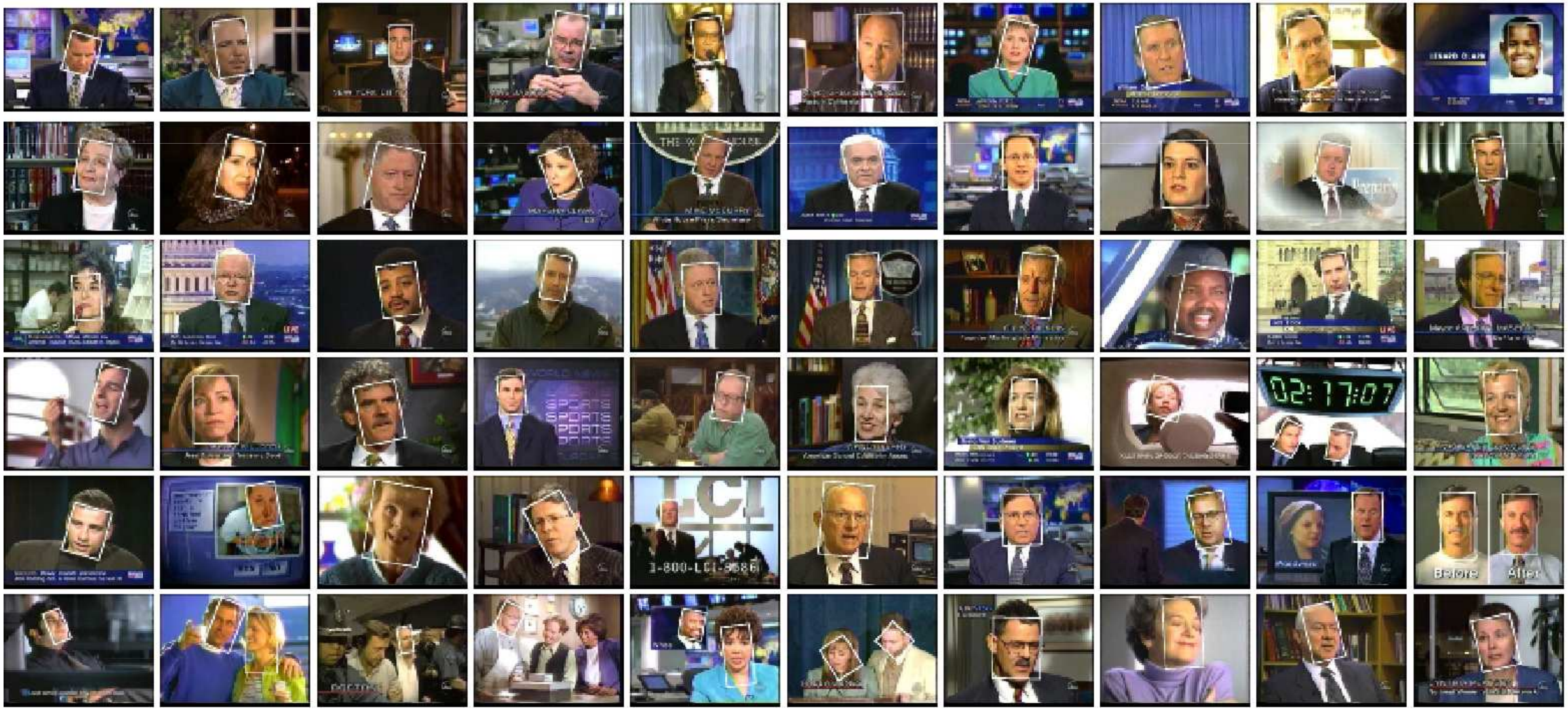
Adapted from Gonzales and Woods

# Template matching



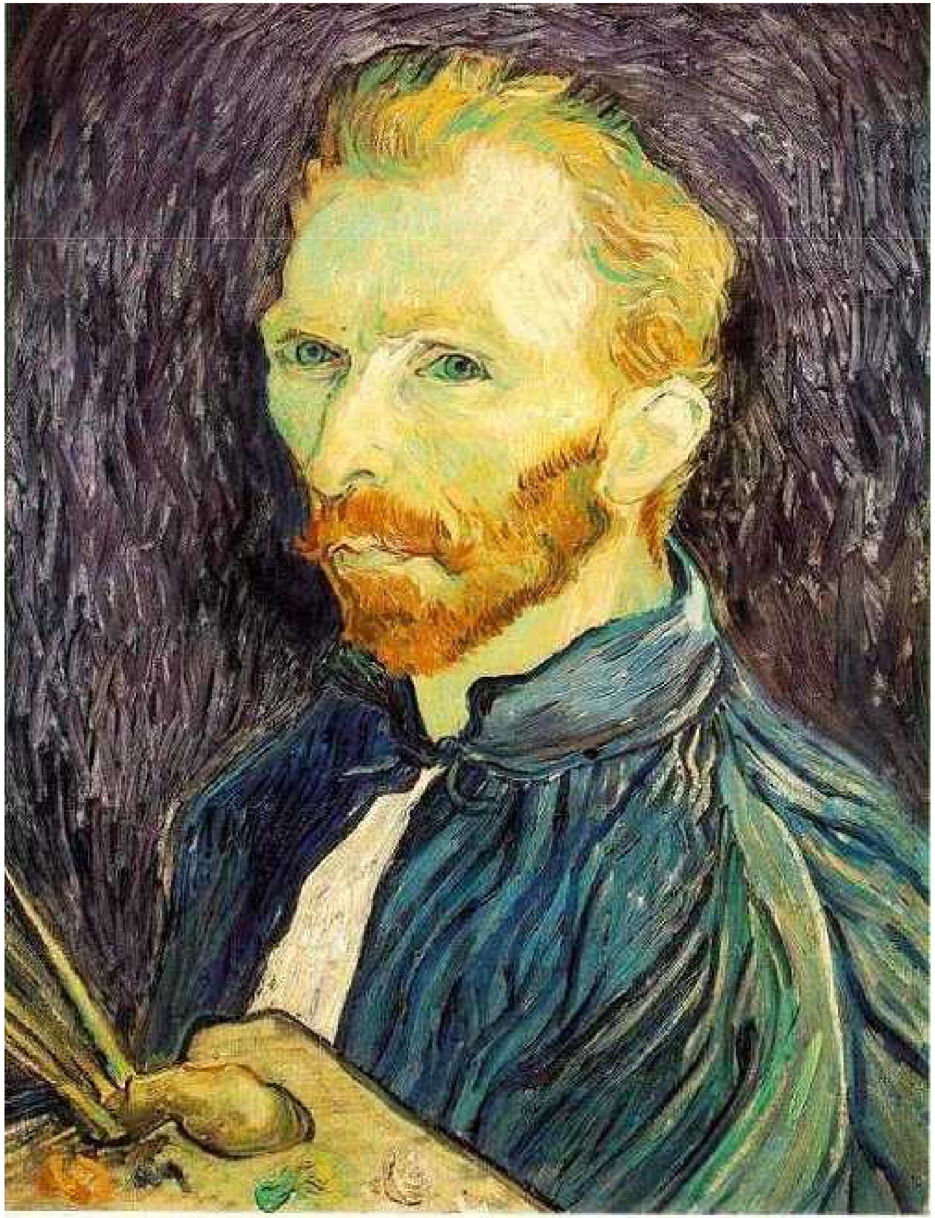
Face detection using template matching: face templates.

# Template matching



Face detection using template matching: detected faces.

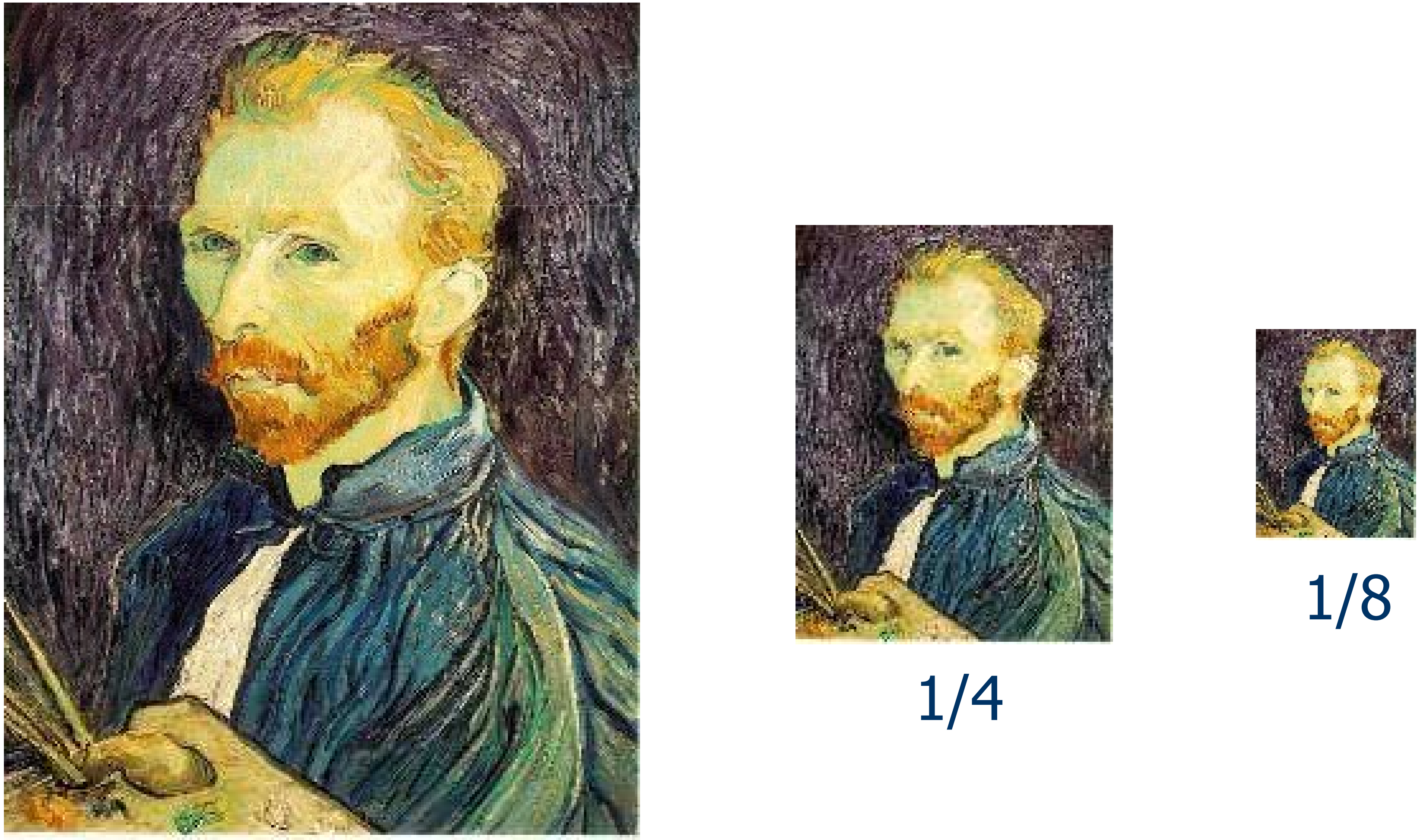
# Resizing images



How can we generate a half-sized version of a large image?

Adapted from Steve Seitz, U of Washington

# Resizing images



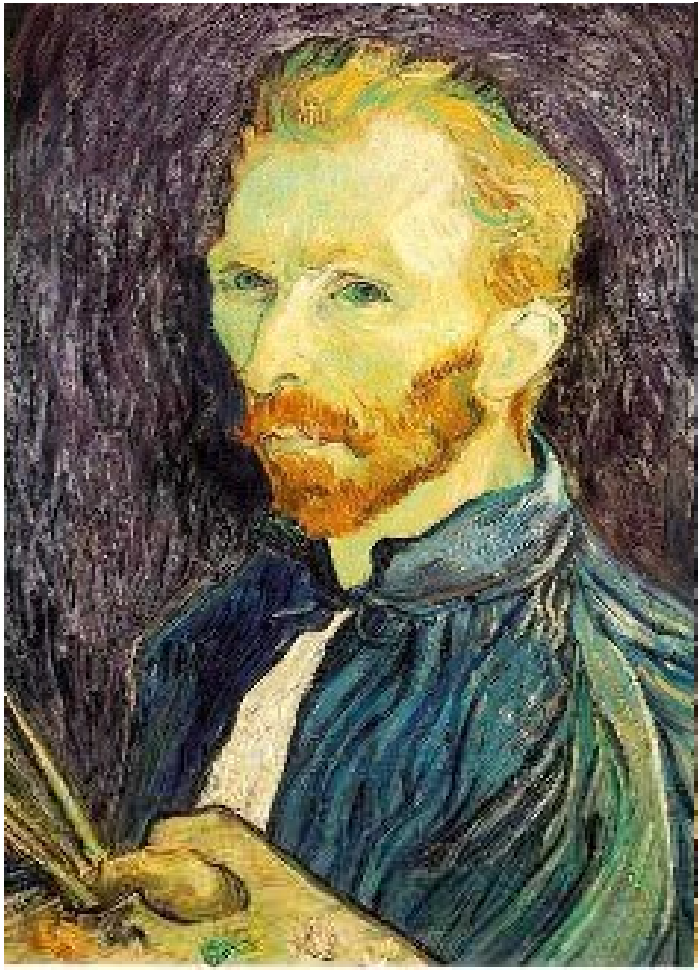
1/4

1/8

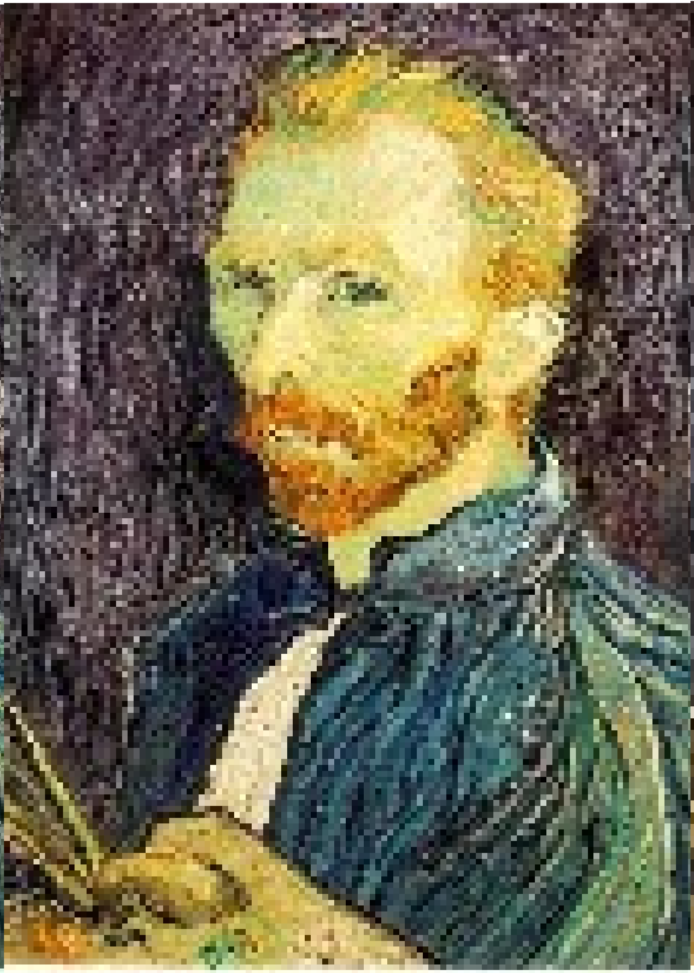
Throw away every other row and column to create a 1/2 size image (also called sub-sampling).

Adapted from Steve Seitz, U of Washington

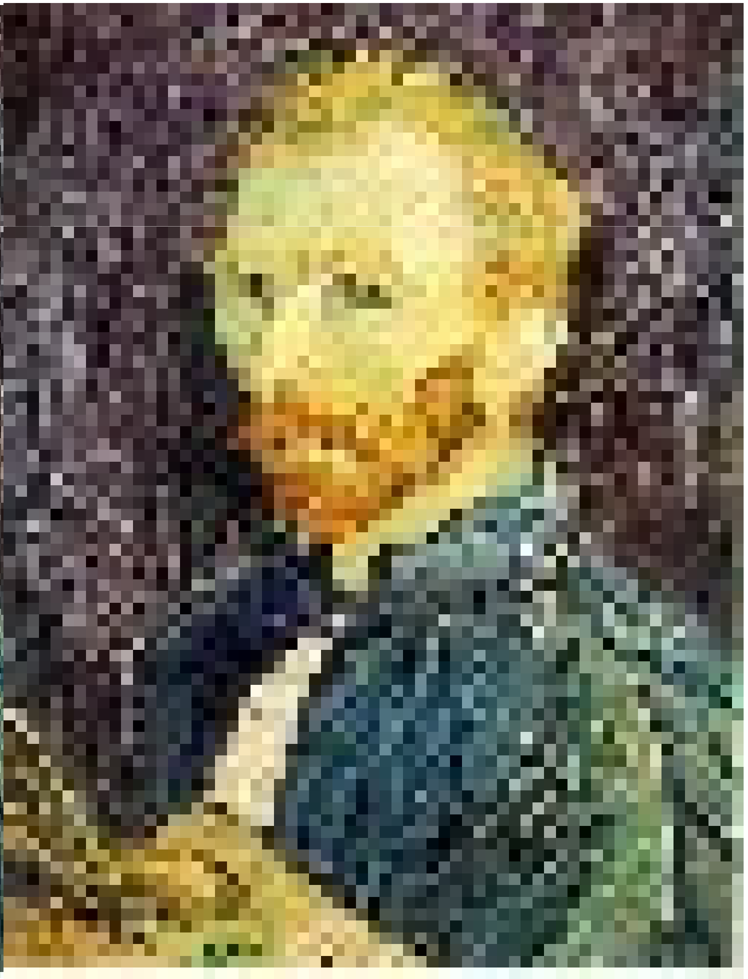
# Resizing images



1/2



1/4 (2x zoom)



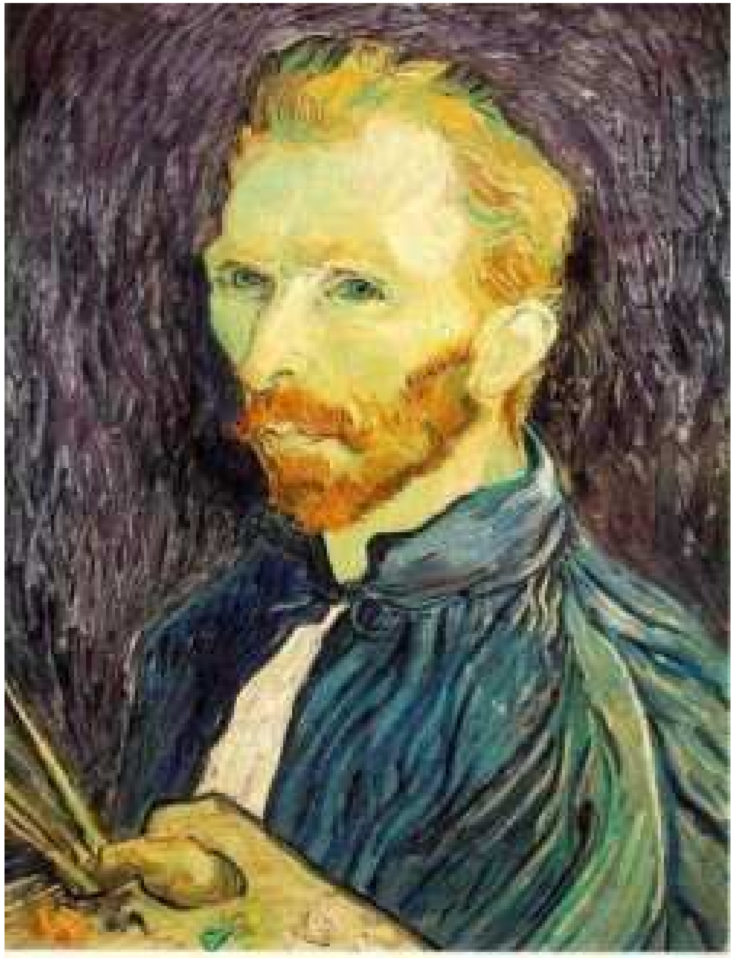
1/8 (4x zoom)

Does this look nice?

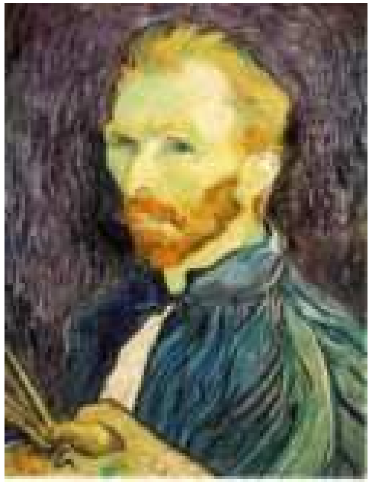
Adapted from Steve Seitz, U of Washington

# Resizing images

- We cannot shrink an image by simply taking every k'th pixel.
- Solution: smooth the image, then sub-sample.



Gaussian 1/2

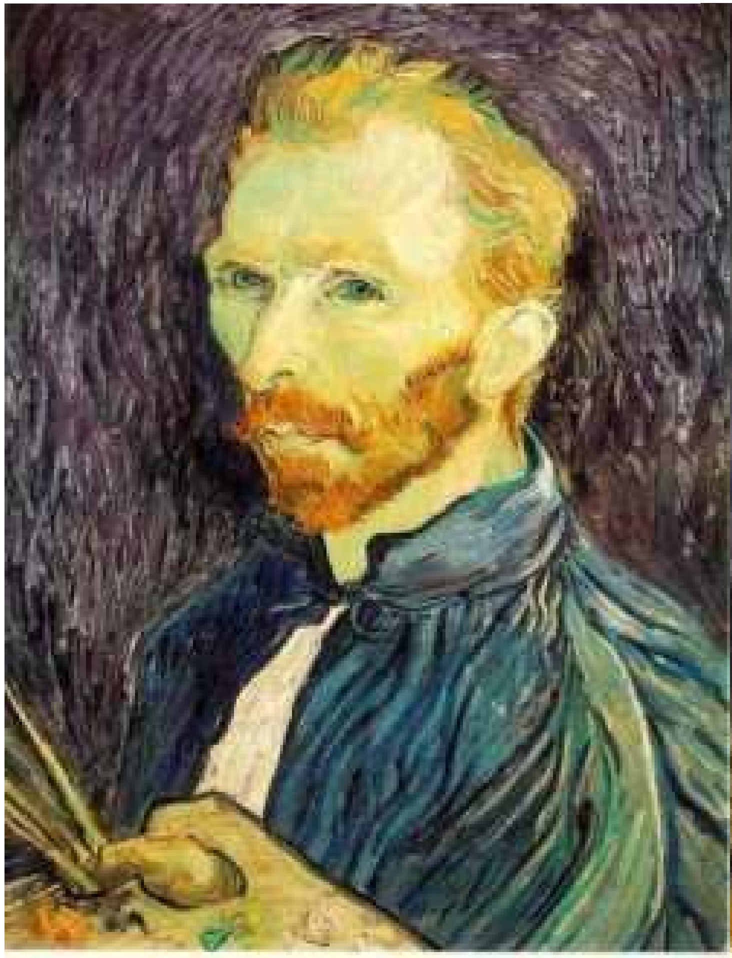


Gaussian 1/4

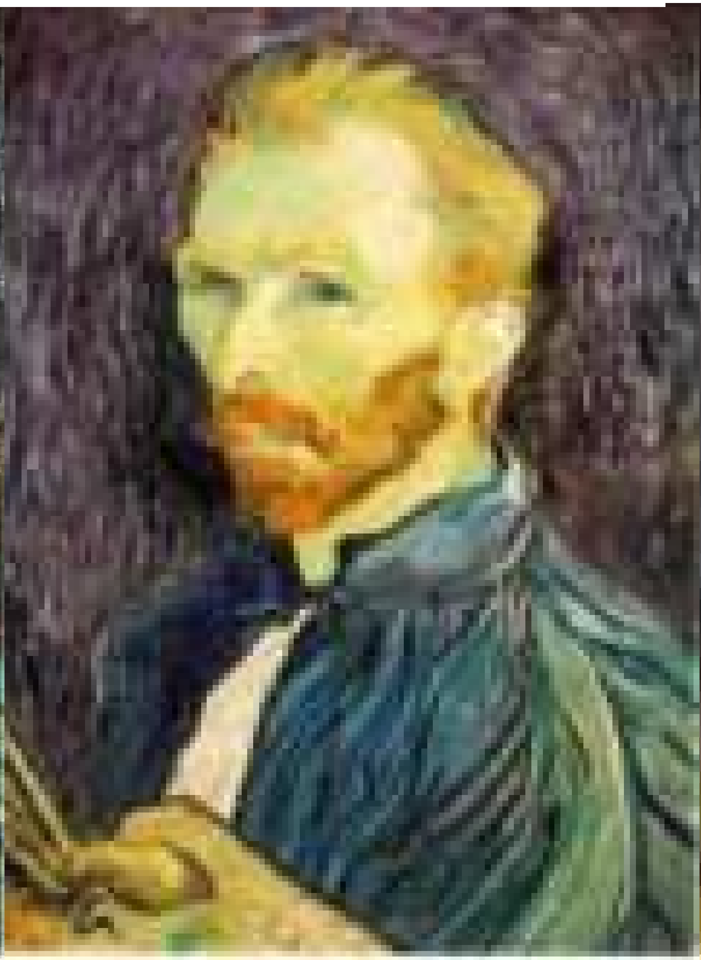
Gaussian 1/8

Adapted from Steve Seitz, U of Washington

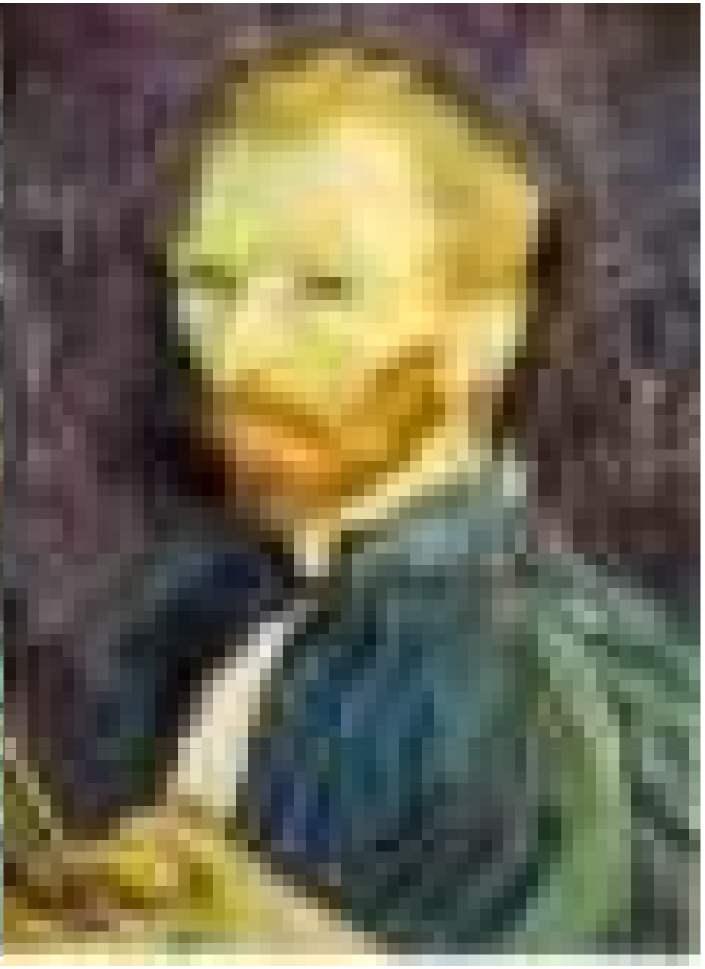
# Resizing images



Gaussian 1/2



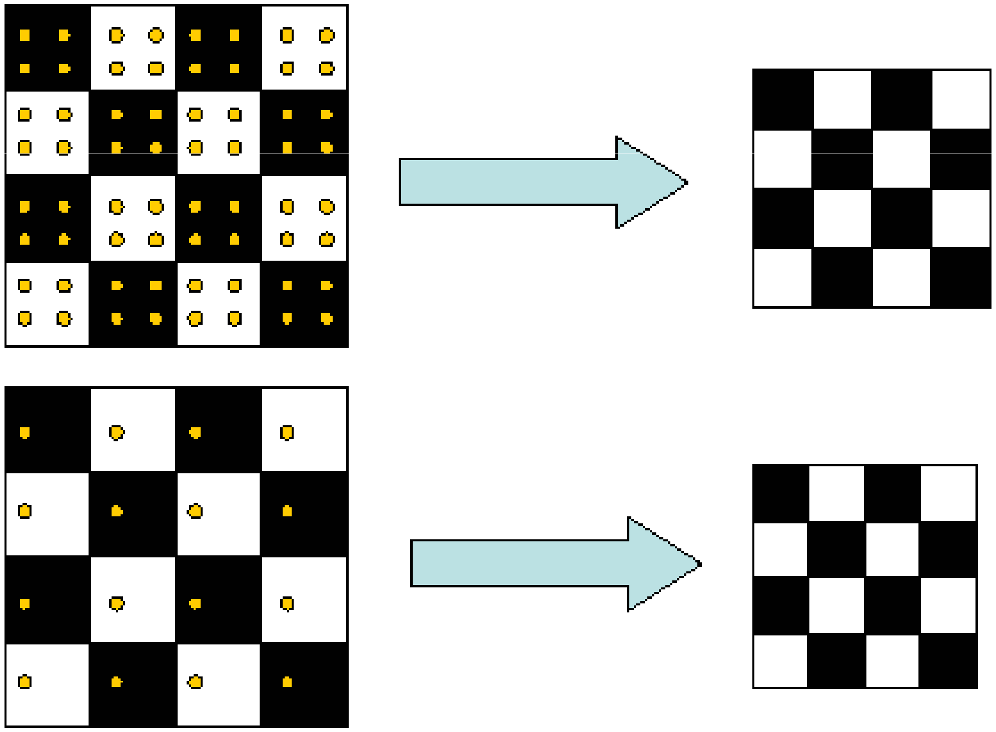
Gaussian 1/4
(2x zoom)



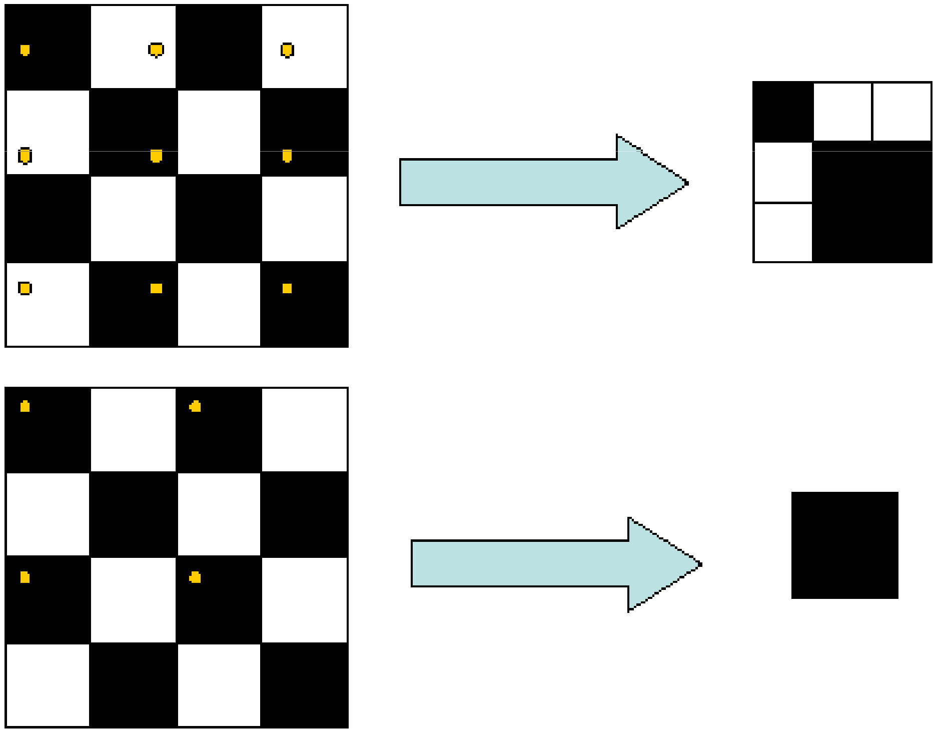
Gaussian 1/8
(4x zoom)

Adapted from Steve Seitz, U of Washington

# Sampling and aliasing



Examples of GOOD sampling



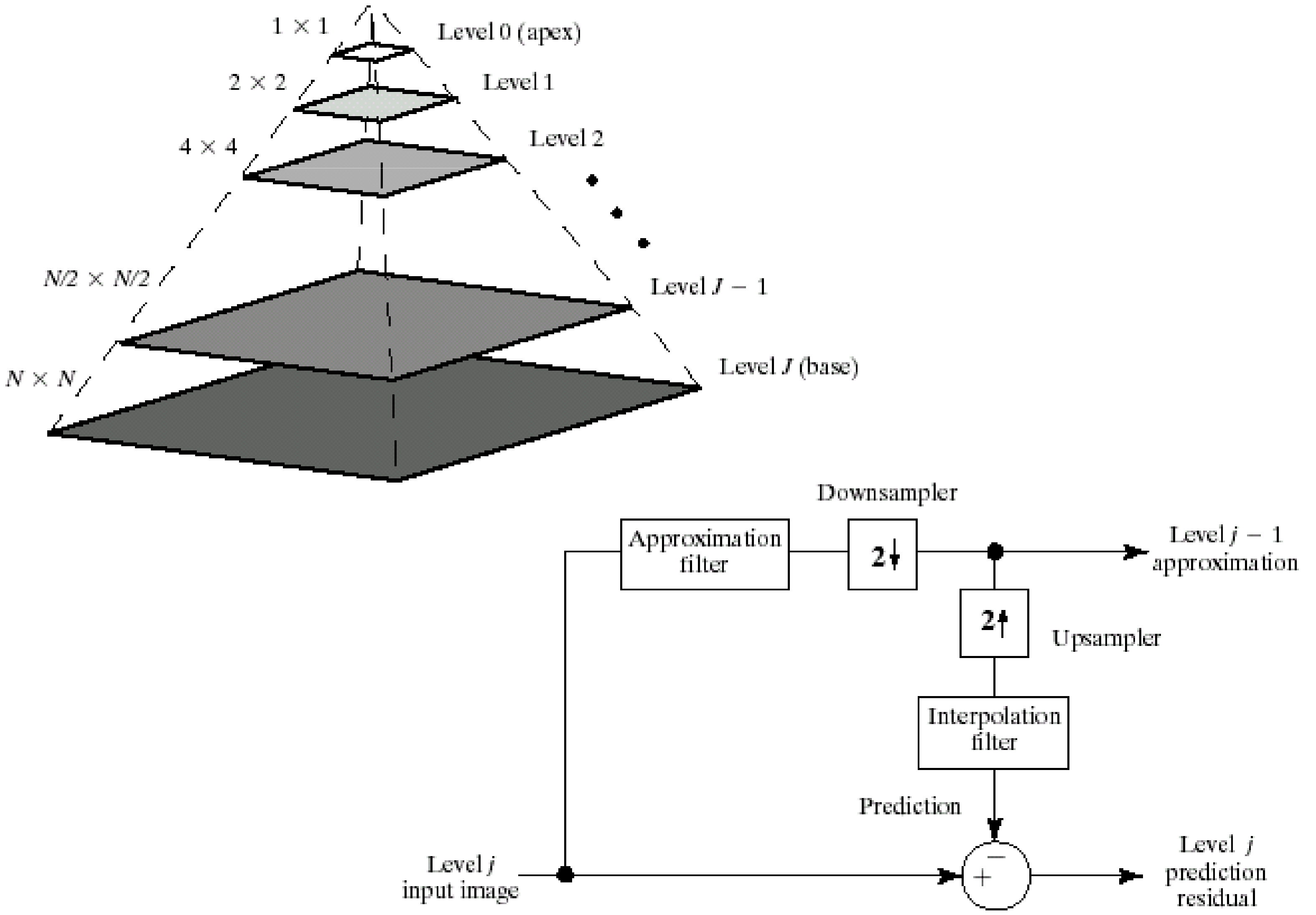
Examples of BAD sampling -> Aliasing

Adapted from Steve Seitz, U of Washington

# Sampling and aliasing

- Errors appear if we do not sample properly.
- Common phenomenon:
  - High spatial frequency components of the image appear as low spatial frequency components.
- Examples:
  - Wagon wheels rolling the wrong way in movies.
  - Checkerboards misrepresented in ray tracing.
  - Striped shirts look funny on color television.

# Gaussian pyramids

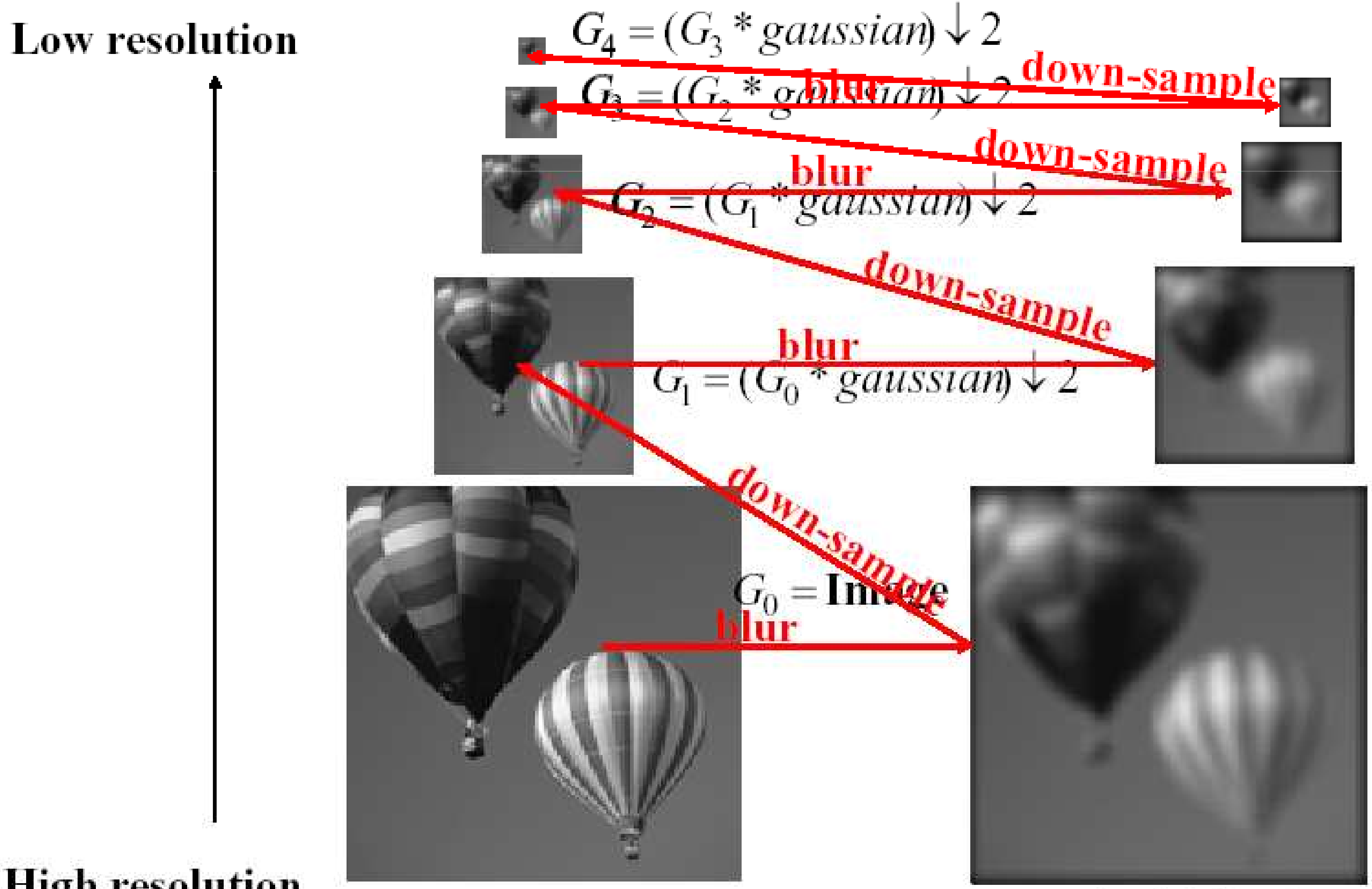


a b

**FIGURE 7.2** (a) A pyramidal image structure and (b) system block diagram for creating it.

Adapted from Gonzales and Woods

# Gaussian pyramids



Low resolution

High resolution

$G_4 = (G_3 * gaussian) \downarrow 2$

$G_3 = (G_2 * gaussian) \downarrow 2$

$G_2 = (G_1 * gaussian) \downarrow 2$

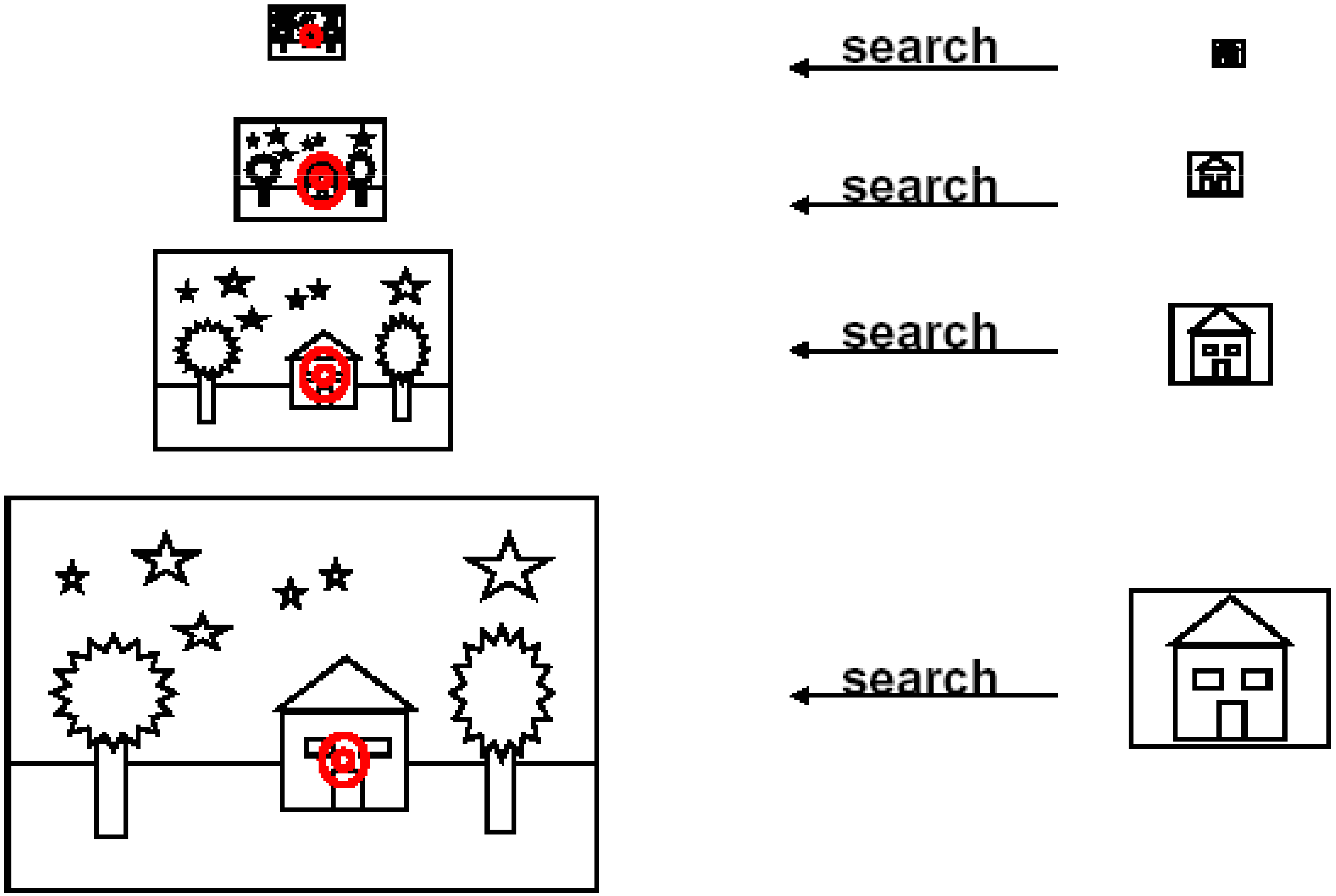
$G_1 = (G_0 * gaussian) \downarrow 2$

$G_0 =$ **Image**

Irani & Basri

Adapted from Michael Black, Brown University

# Gaussian pyramids



Irani & Basri

Adapted from Michael Black, Brown University