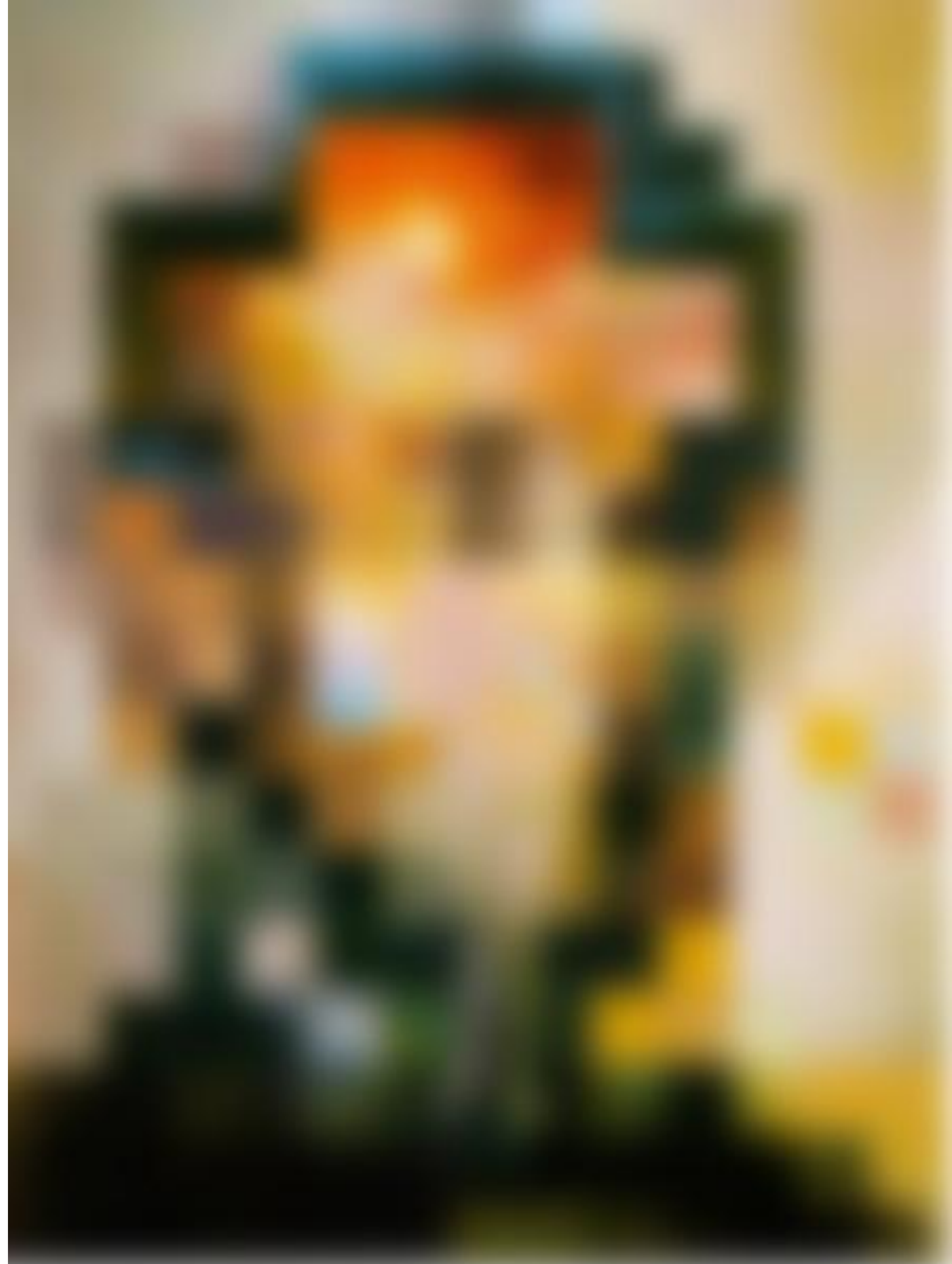
The Frequency Domain

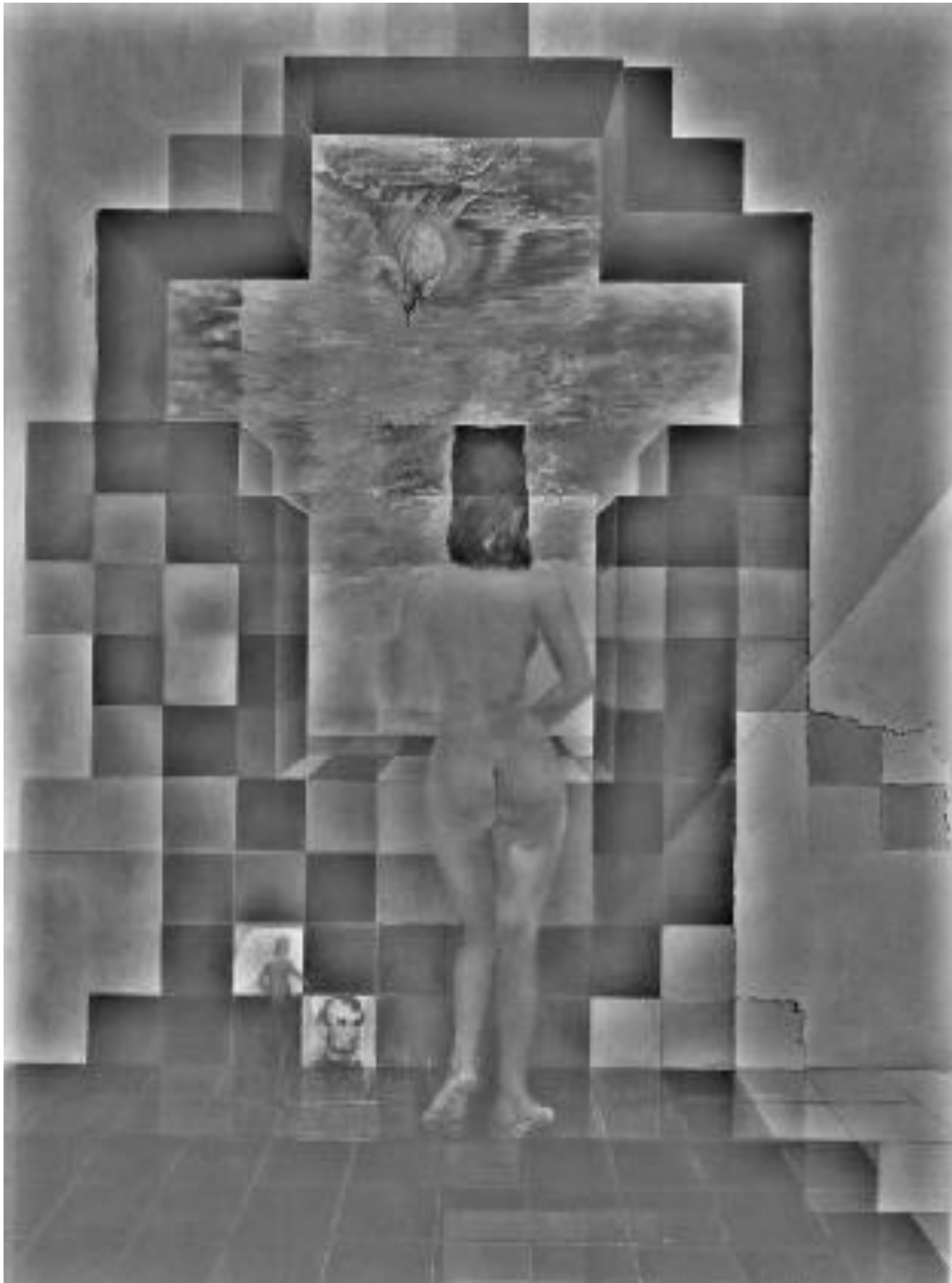
Somewhere in Cinque Terre, May 2005

15-463: Computational Photography
Alexei Efros, CMU, Spring 2010

Many slides borrowed
from Steve Seitz

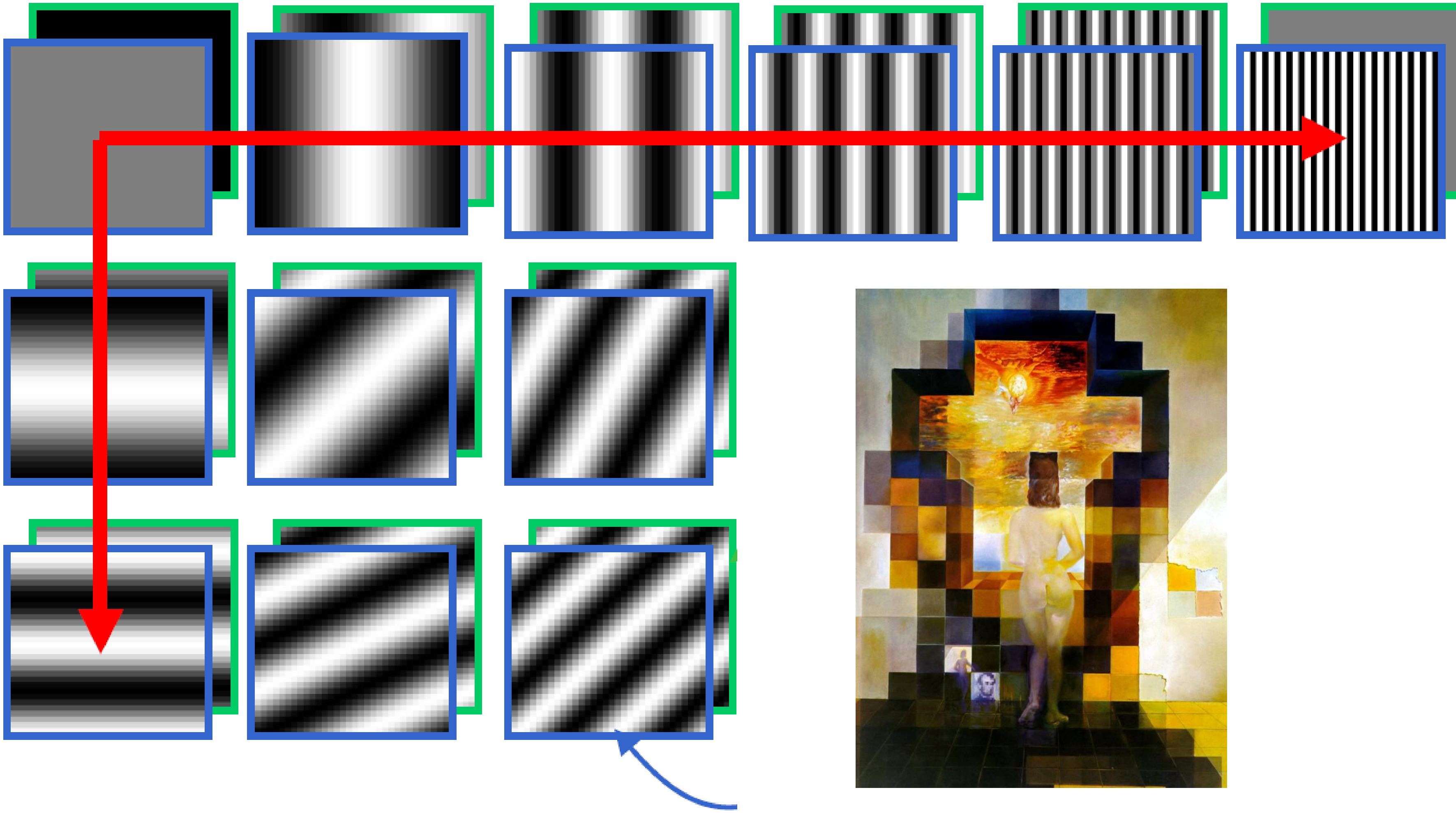
Salvador Dali
"Gala Contemplating the Mediterranean Sea, which at 30 meters becomes the portrait of Abraham Lincoln", 1976





A nice set of basis

Teases away fast vs. slow changes in the image.



This change of basis has a special name…

Jean Baptiste Joseph Fourier (1768-1830)

had crazy idea (1807):

***Any** periodic function can be rewritten as a weighted sum of sines and cosines of different frequencies.*

Don’t believe it?

- Neither did Lagrange, Laplace, Poisson and other big wigs
- Not translated into English until 1878!

But it’s true!

- called Fourier Series

A sum of sines

Our building block:

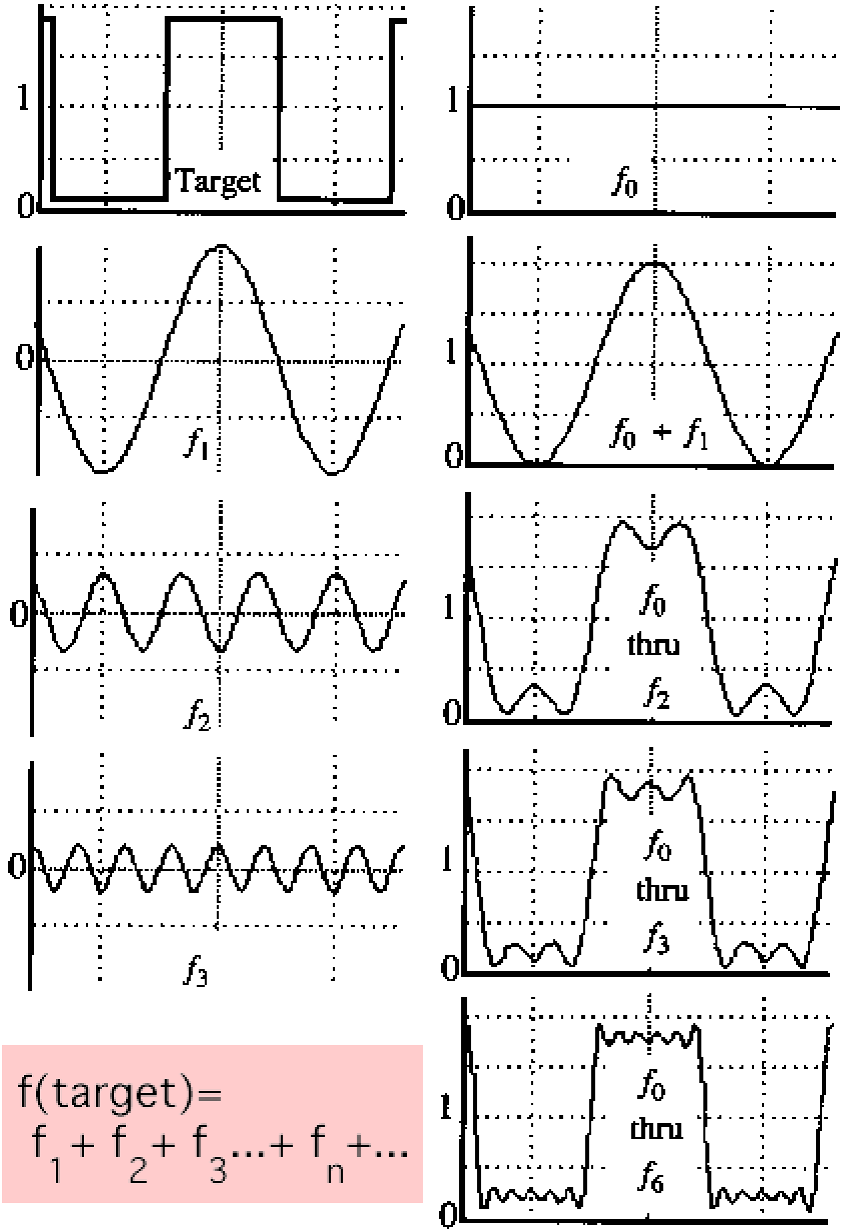
$$A\sin(\omega x + \phi)$$

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

How many degrees of freedom?

What does each control?

Which one encodes the coarse vs. fine structure of the signal?



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

Fourier Transform

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

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

For every ω from 0 to inf, ***F(ω)*** holds the amplitude *A* and phase ϕ of the corresponding sine $A\sin(\omega x + \phi)$

- How can *F* hold both? Complex number trick!

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

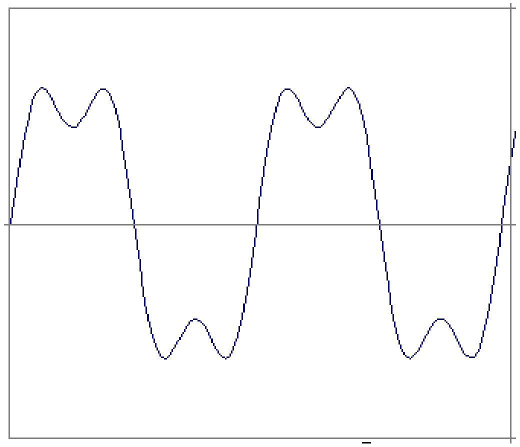
$$A = \pm\sqrt{R(\omega)^2 + I(\omega)^2} \qquad \phi = \tan^{-1}\frac{I(\omega)}{R(\omega)}$$

We can always go back:

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

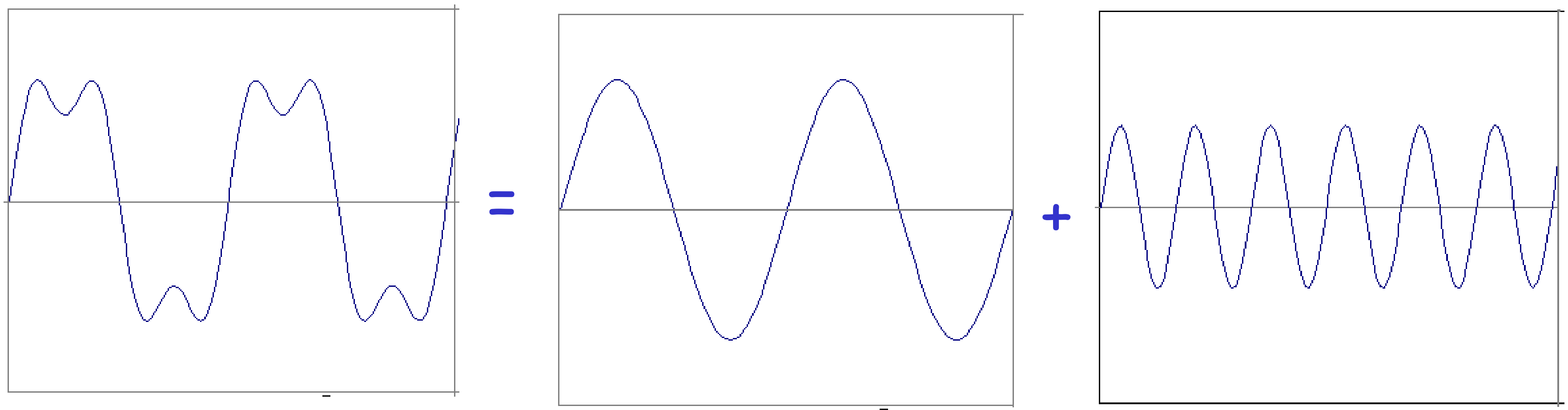
Time and Frequency

example : $g(t) = \sin(2pf\, t) + (1/3)\sin(2p(3f)\, t)$



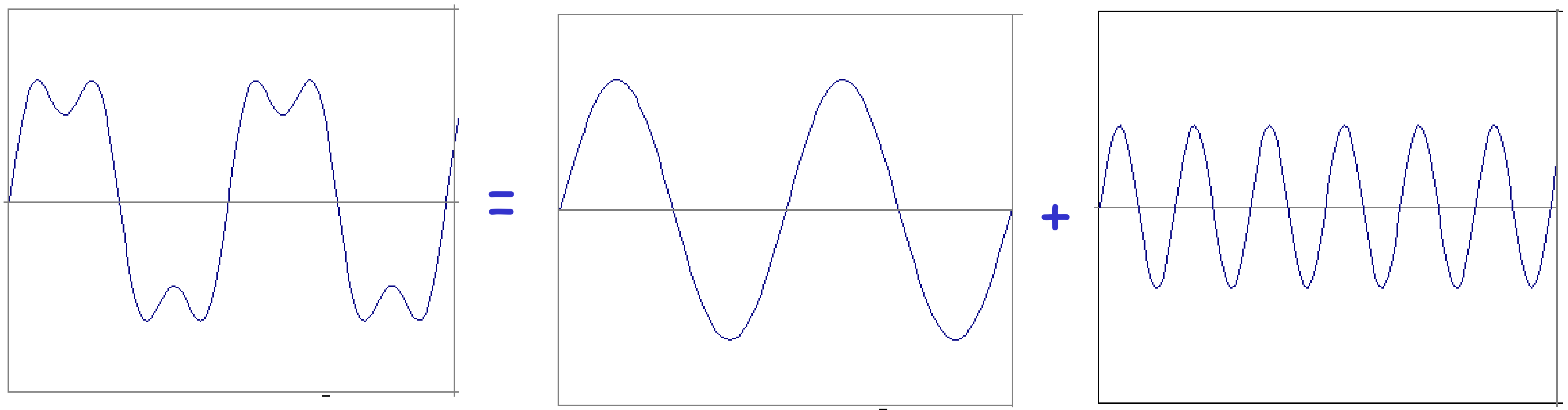
Time and Frequency

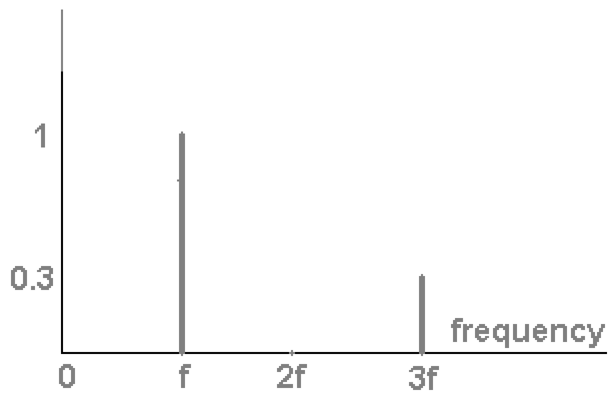
example : $g(t) = \sin(2pf\,t) + (1/3)\sin(2p(3f)\,t)$



Frequency Spectra

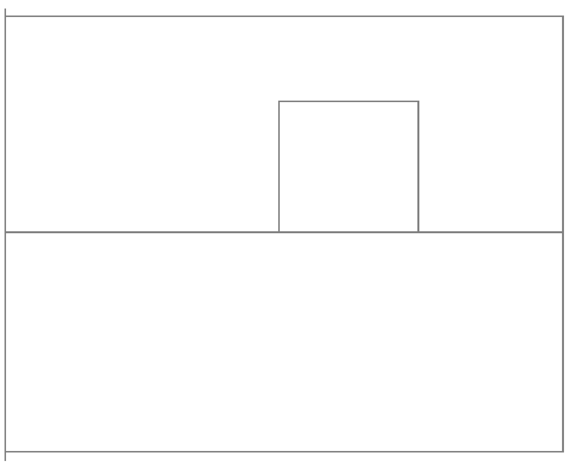
example : $g(t) = \sin(2pf\ t) + (1/3)\sin(2p(3f)\ t)$



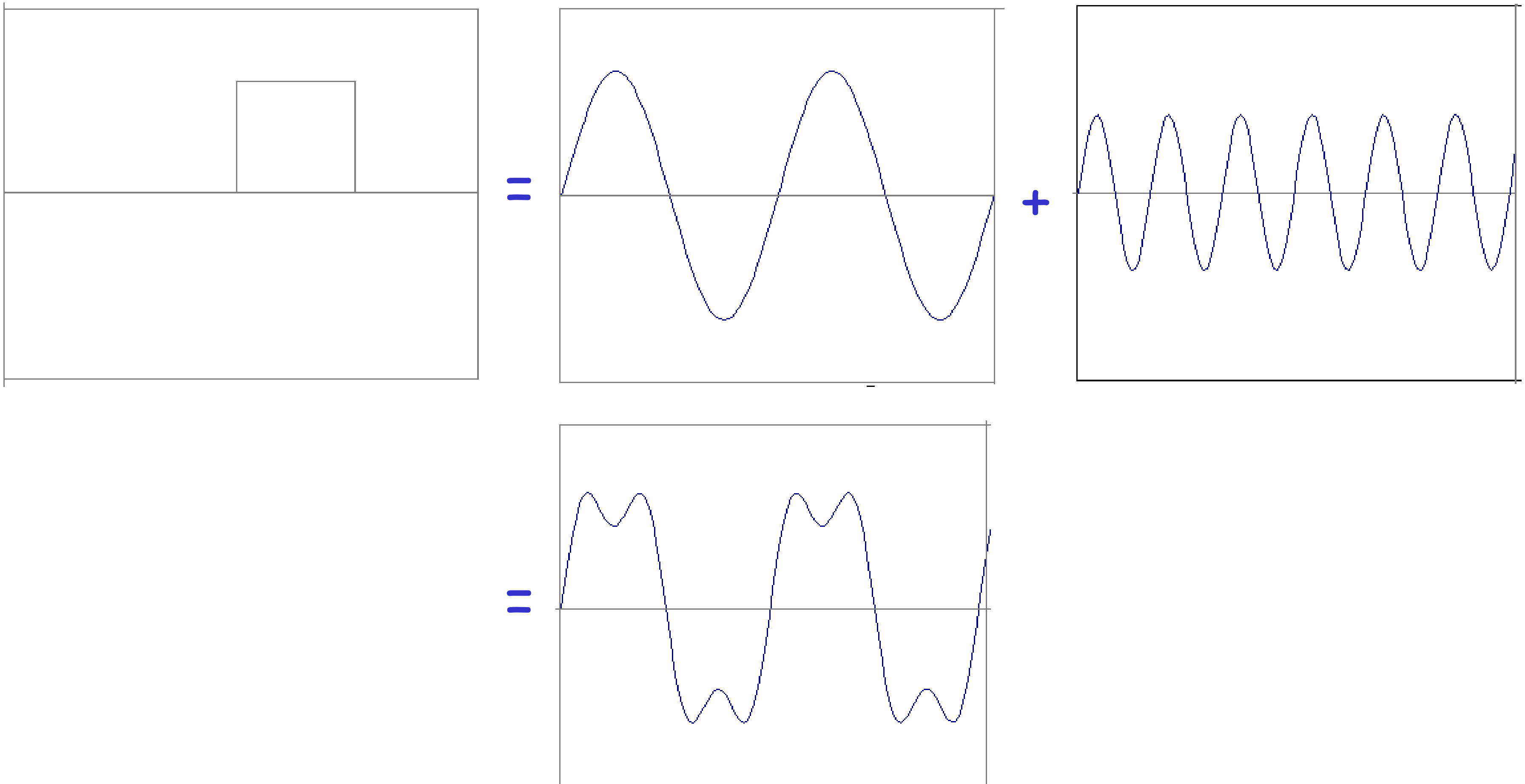


Frequency Spectra

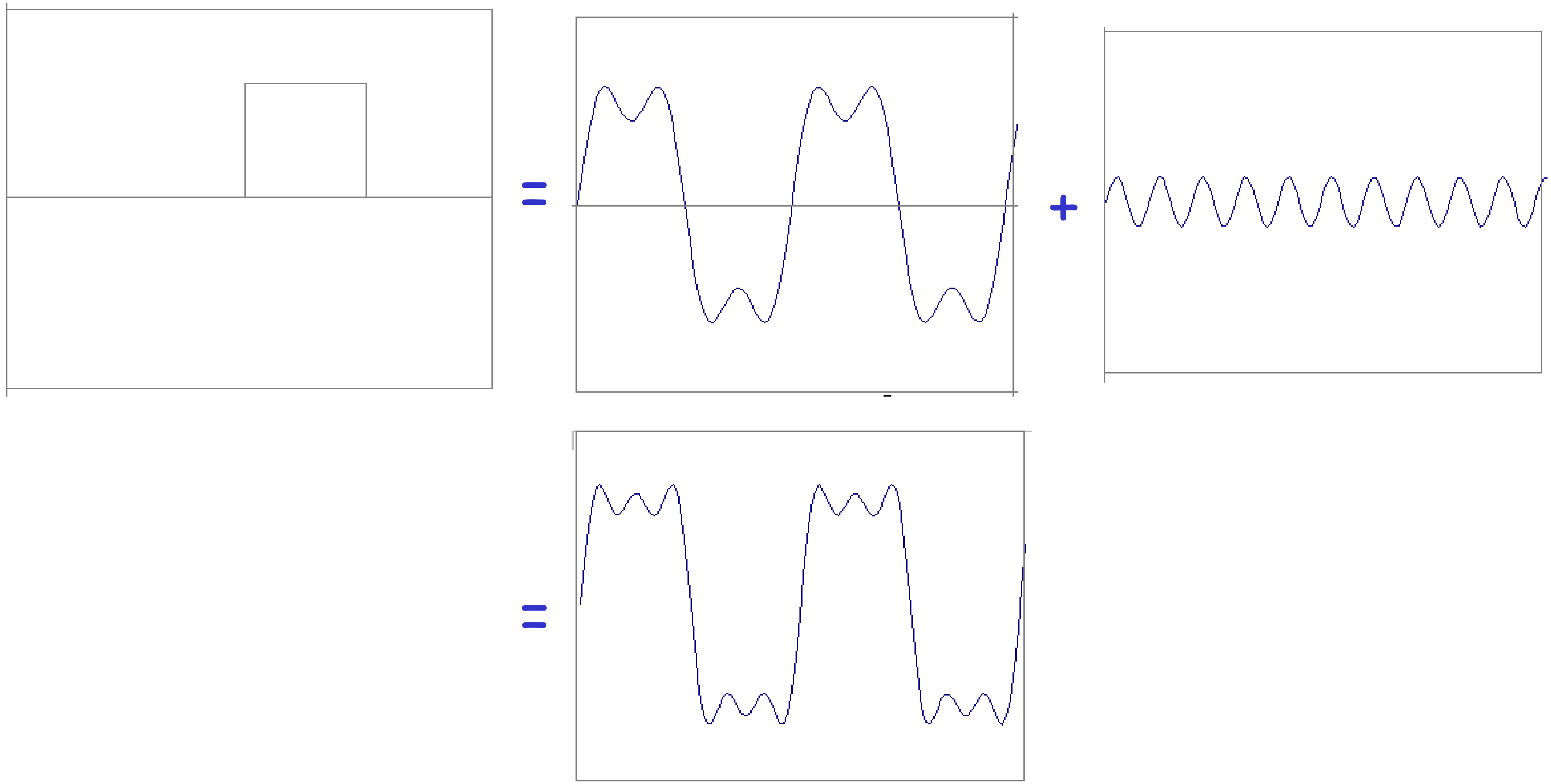
Usually, frequency is more interesting than the phase



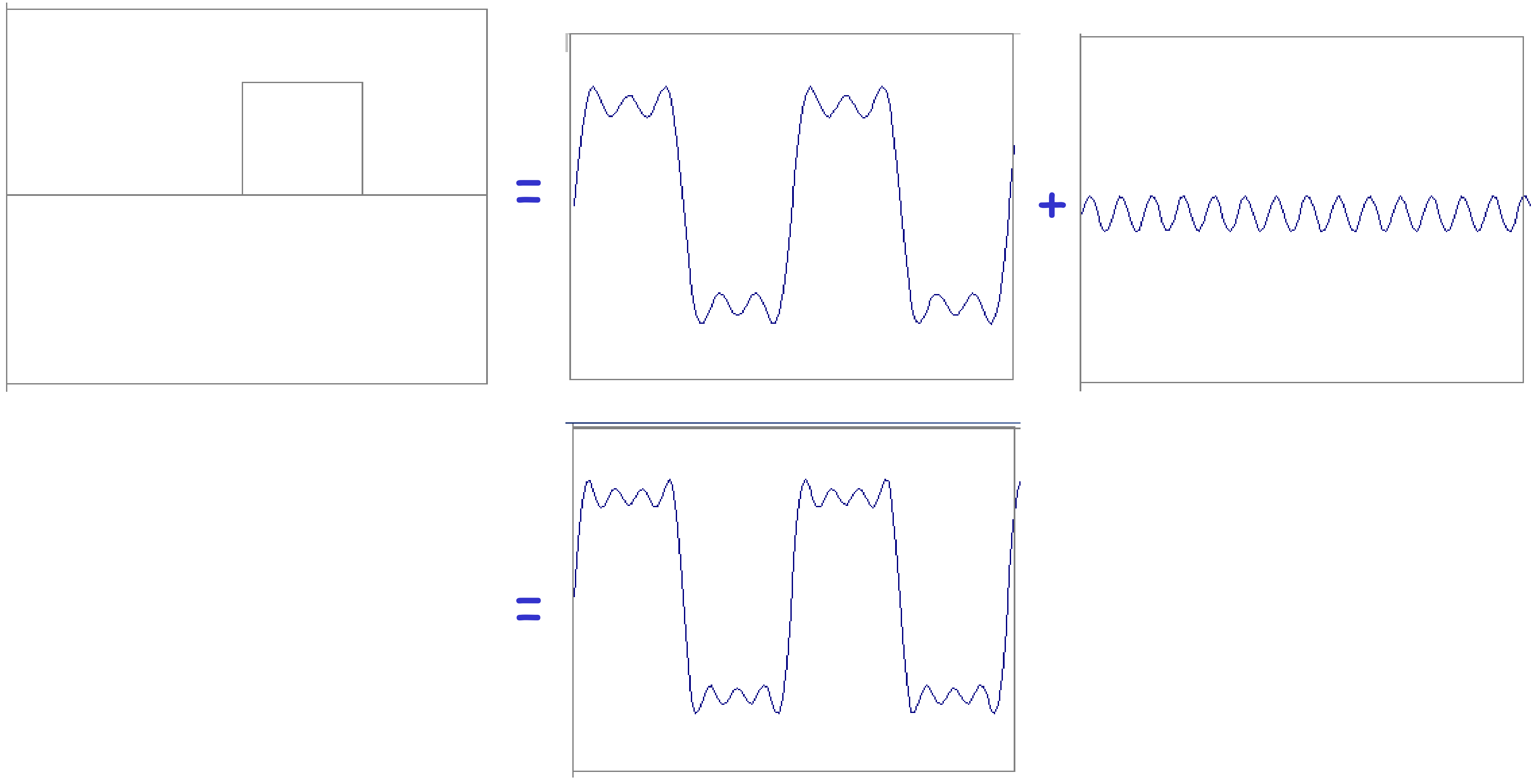
Frequency Spectra



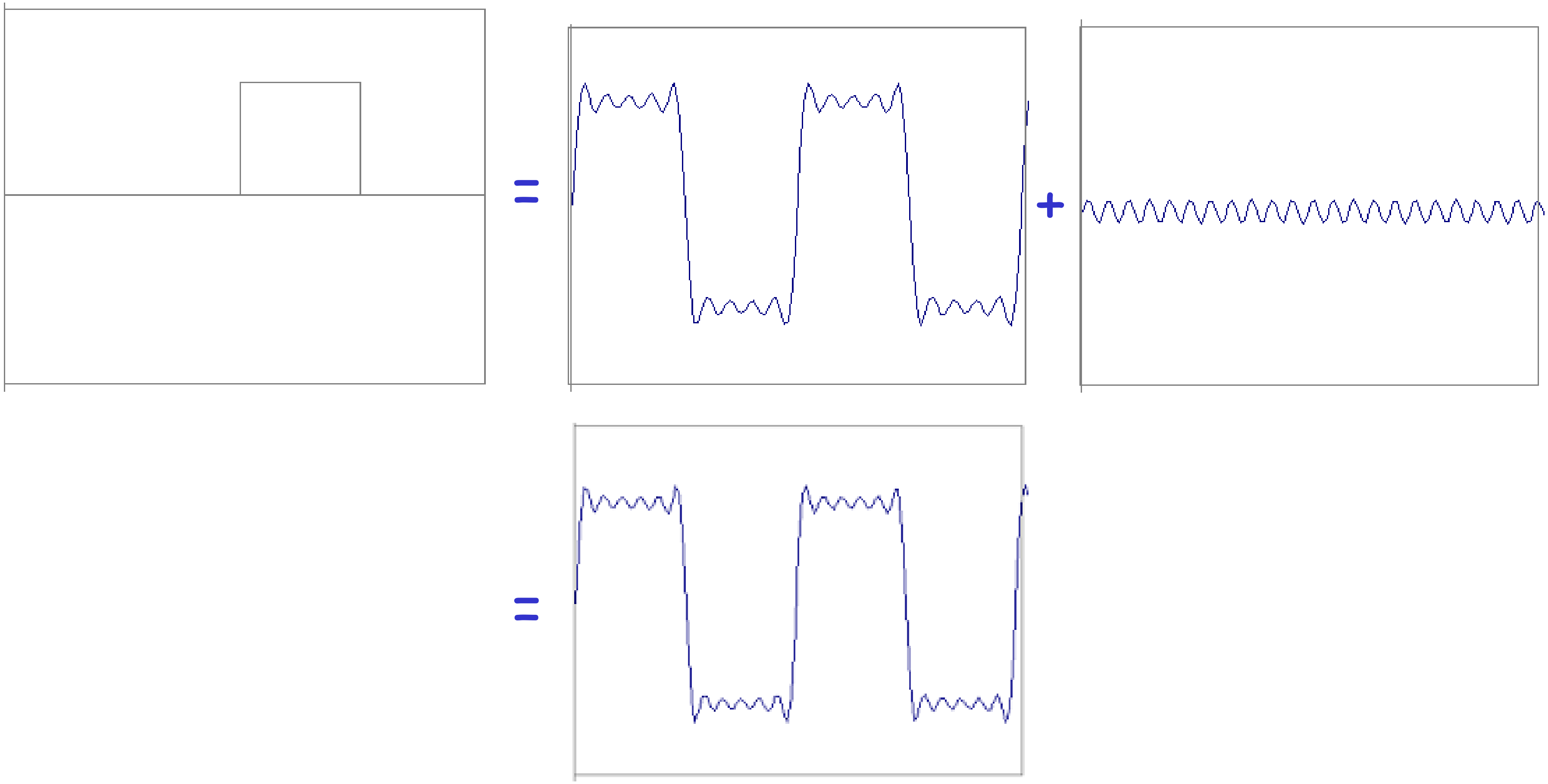
Frequency Spectra



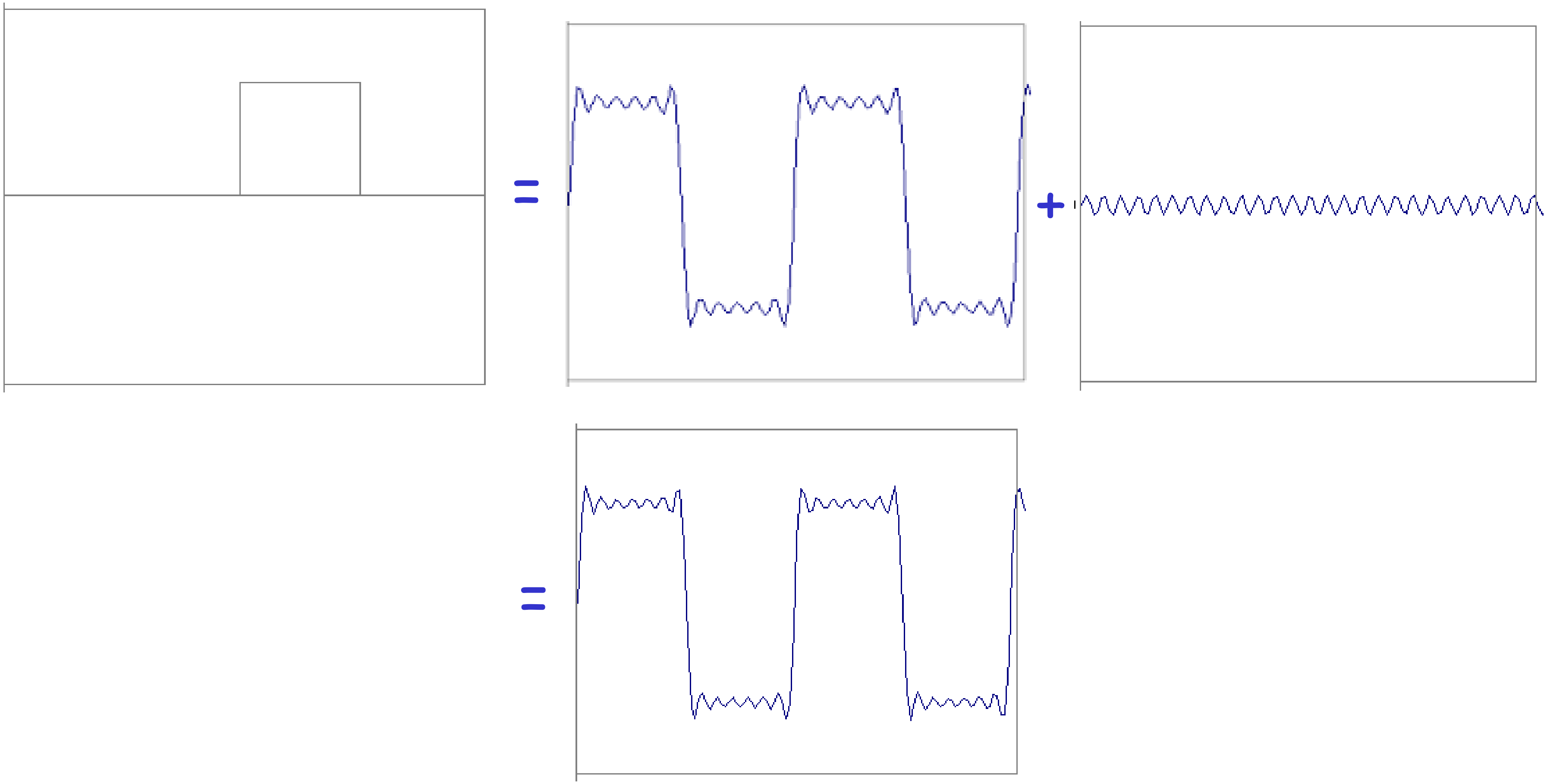
Frequency Spectra



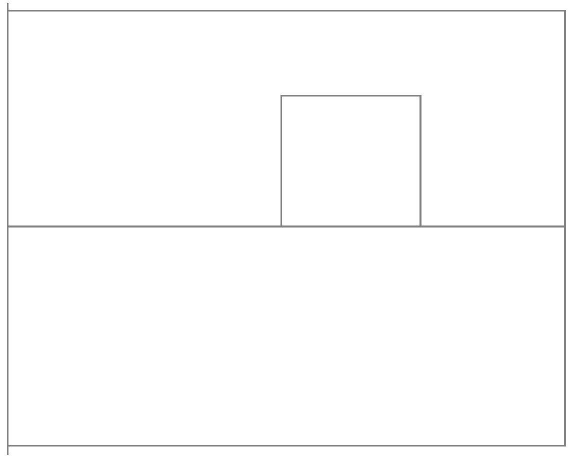
Frequency Spectra



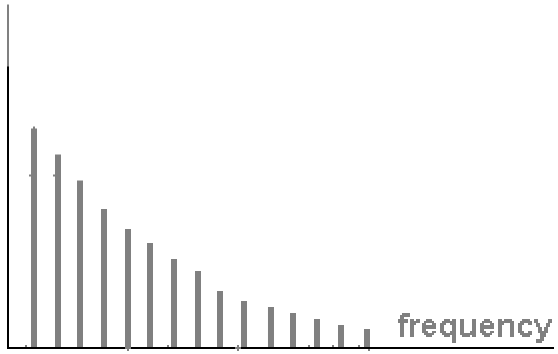
Frequency Spectra



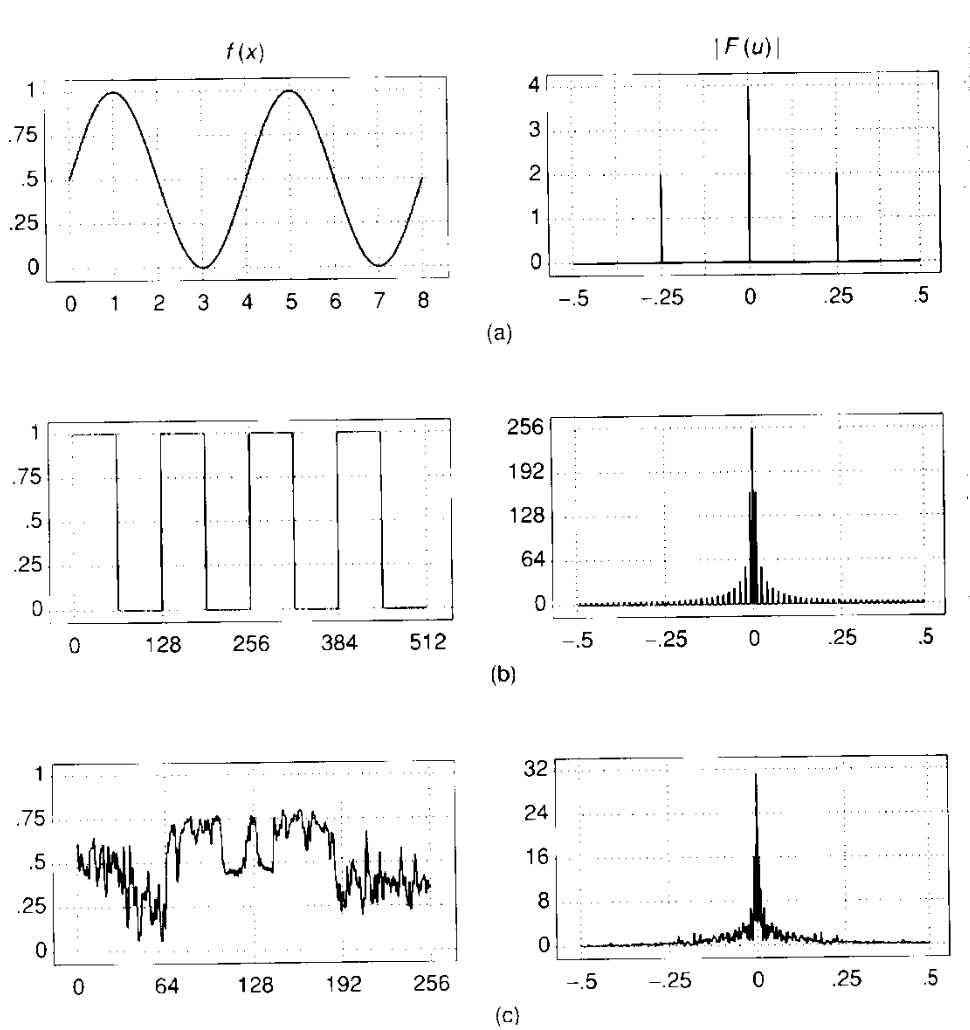
Frequency Spectra



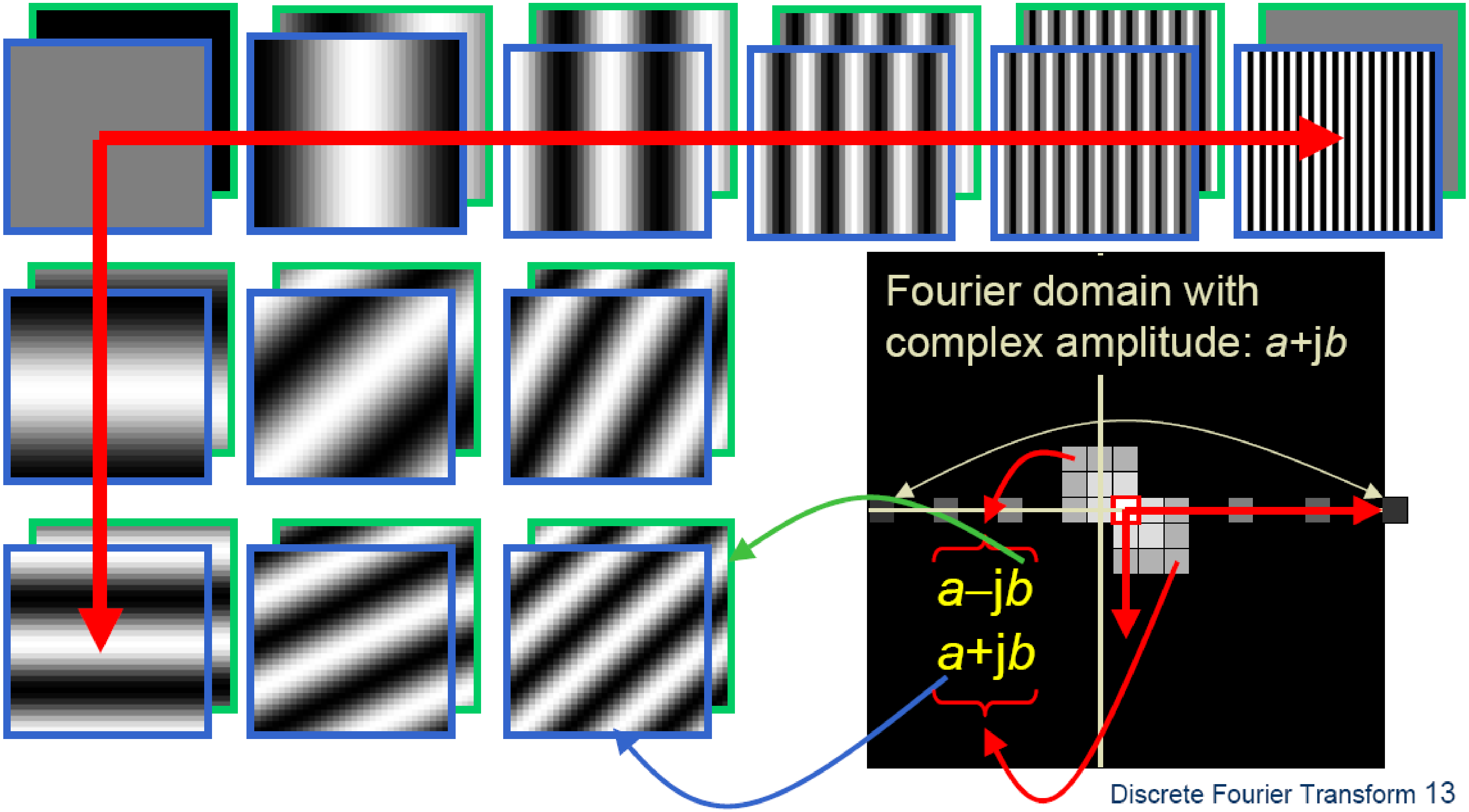
$$= \quad A\sum_{k=1}^{\infty}\frac{1}{k}\sin(2\pi kt)$$



Frequency Spectra

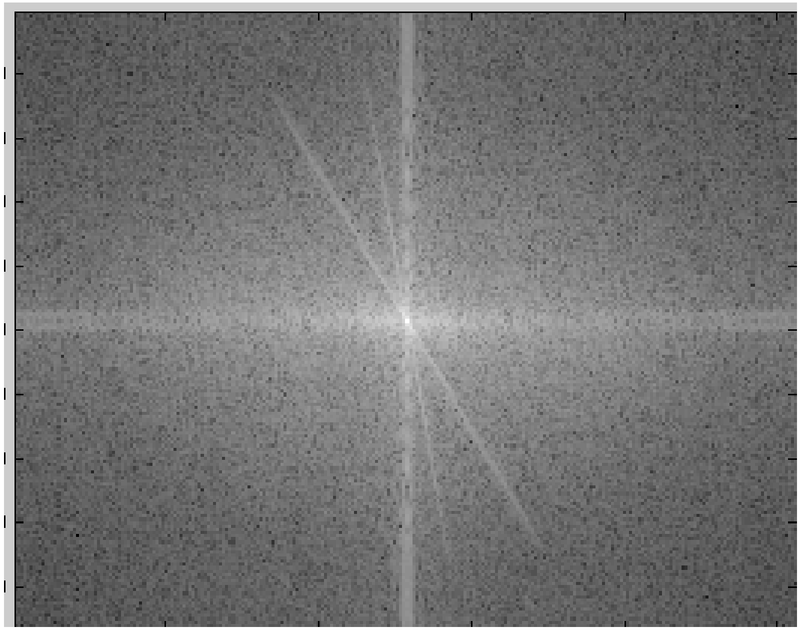


Extension to 2D



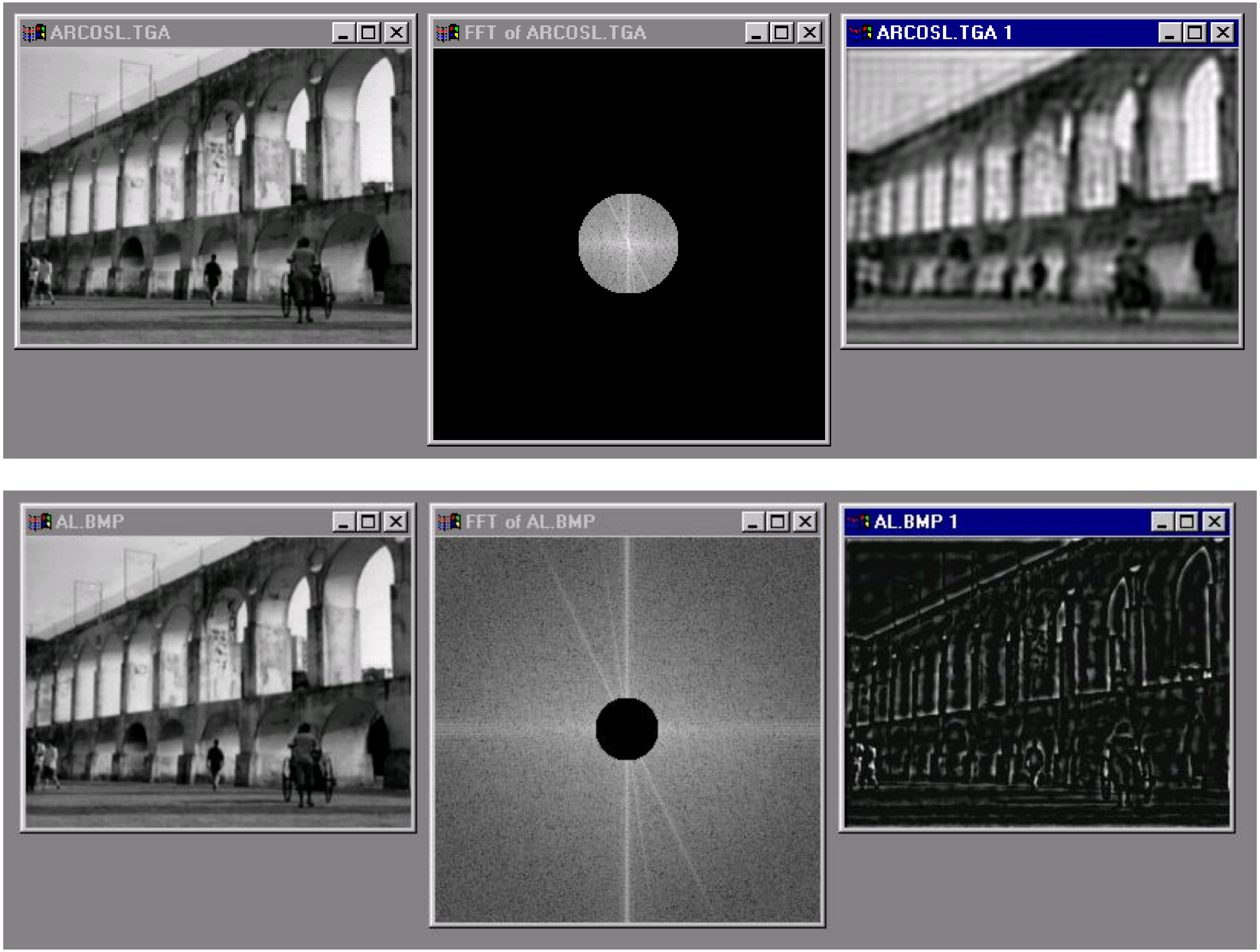
in Matlab, check out: imagesc(log(abs(fftshift(fft2(im)))));

Man-made Scene



Can change spectrum, then reconstruct

Low and High Pass filtering



The Convolution Theorem

The greatest thing since sliced (banana) bread!

- The Fourier transform of the convolution of two functions is the product of their Fourier transforms

$$\mathrm{F}[g * h] = \mathrm{F}[g]\,\mathrm{F}[h]$$

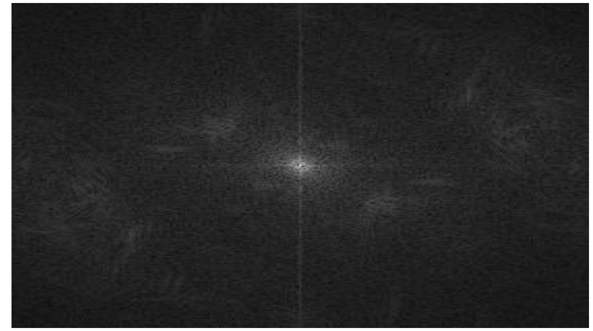
- The inverse Fourier transform of the product of two Fourier transforms is the convolution of the two inverse Fourier transforms

$$\mathrm{F}^{-1}[gh] = \mathrm{F}^{-1}[g] * \mathrm{F}^{-1}[h]$$

- **Convolution** in spatial domain is equivalent to **multiplication** in frequency domain!

2D convolution theorem example

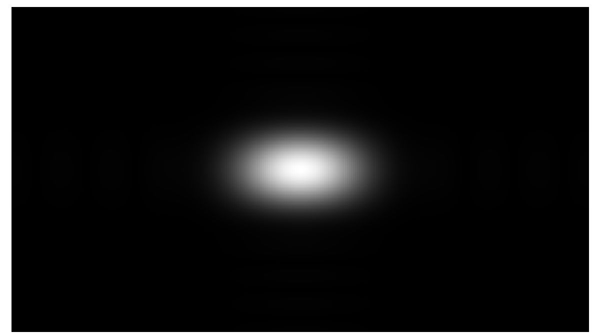
$f(x,y)$



$|F(s_x,s_y)|$

$*$ $\times$

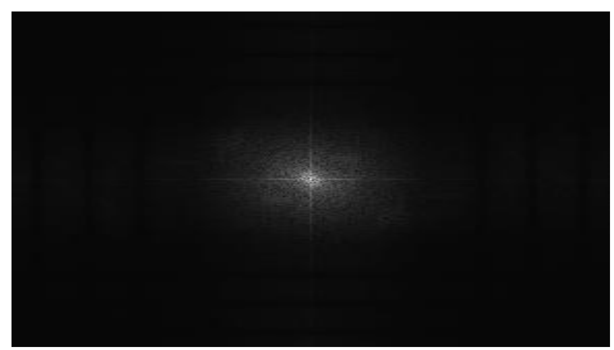
$h(x,y)$



$|H(s_x,s_y)|$

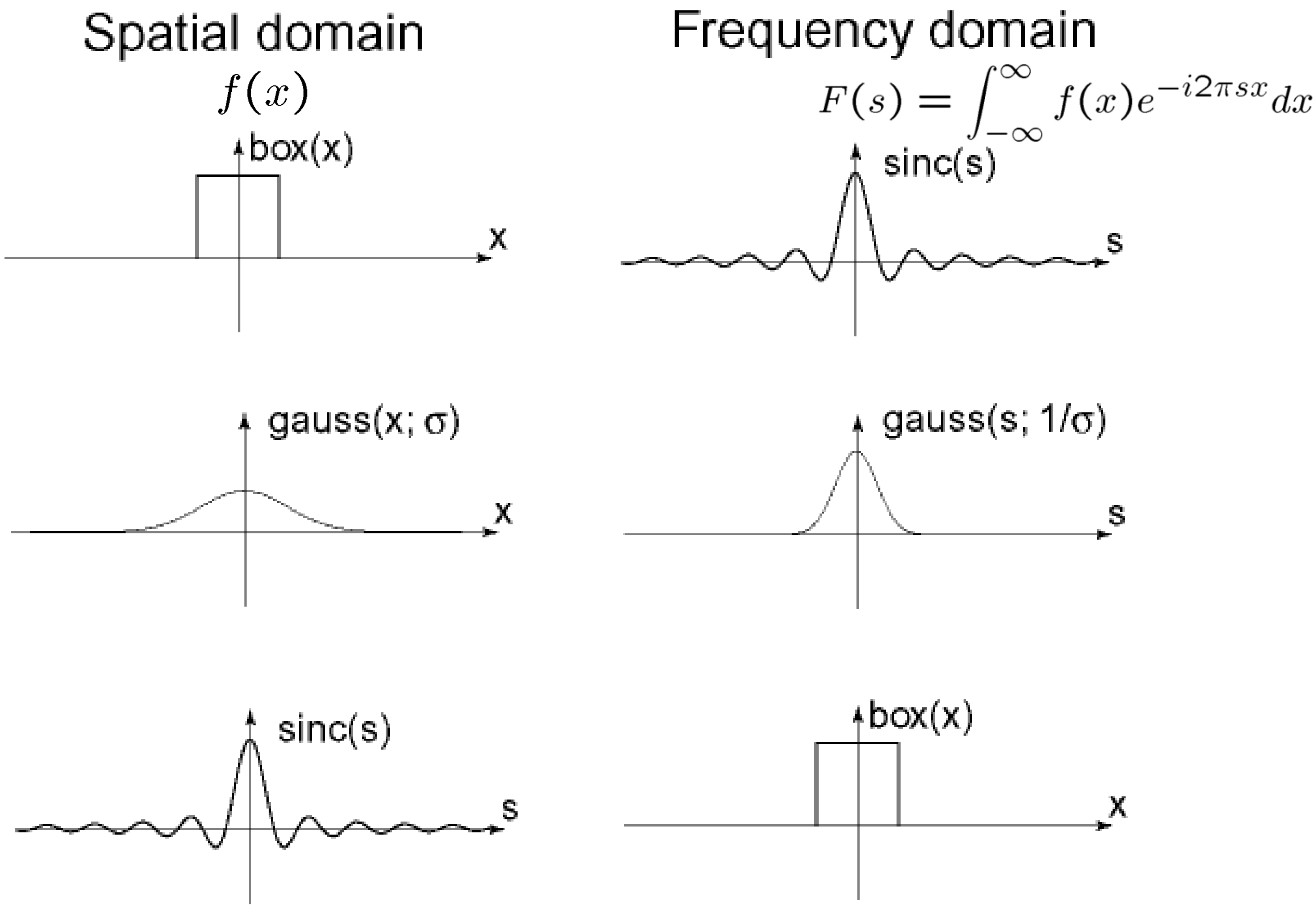
$\Downarrow$ $\Downarrow$

$g(x,y)$



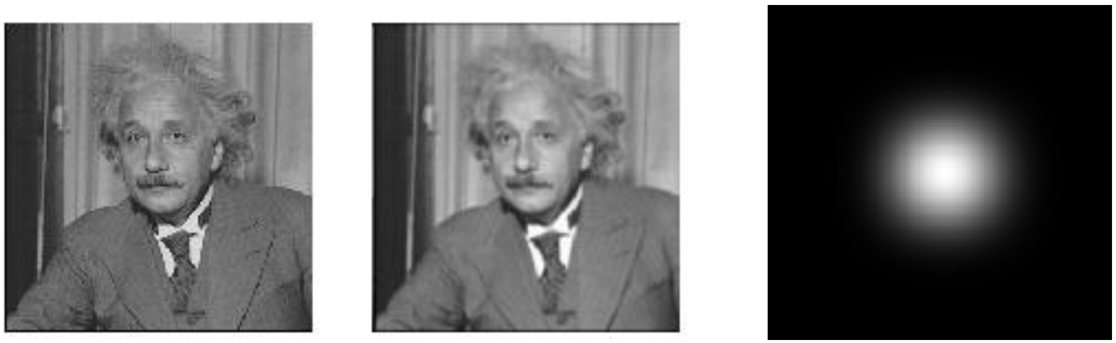
$|G(s_x,s_y)|$

Fourier Transform pairs



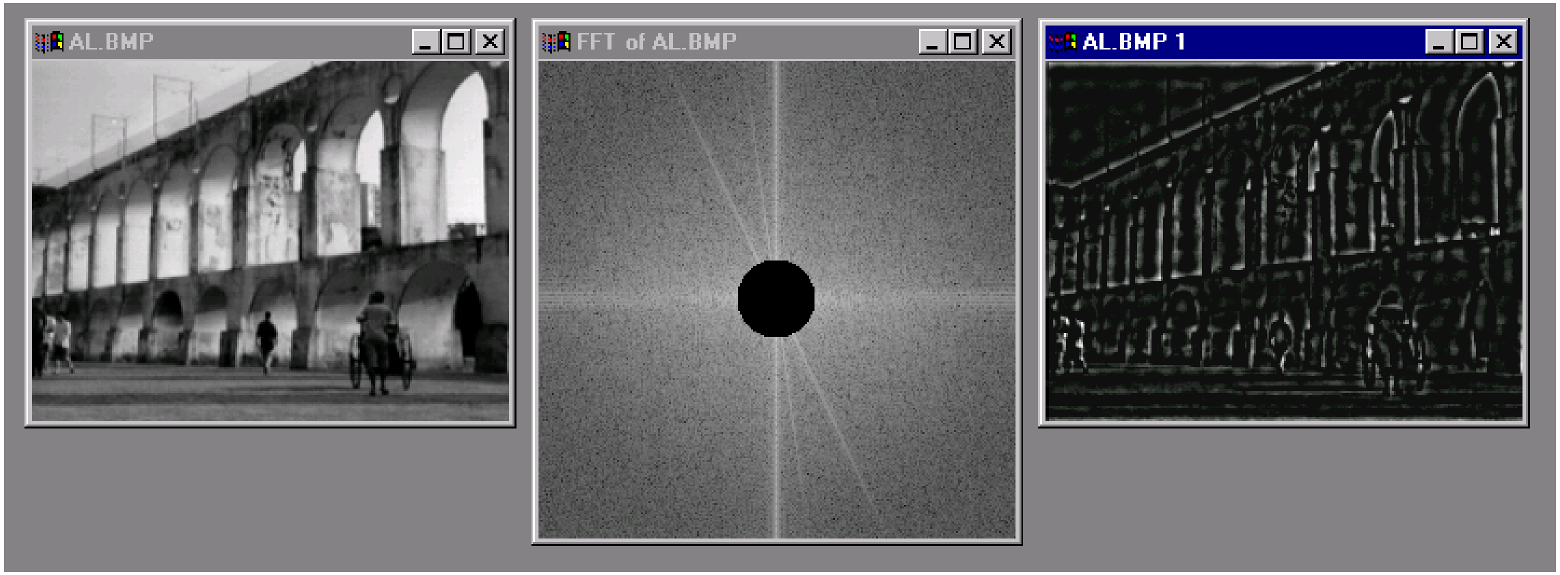
Low-pass, Band-pass, High-pass filters

low-pass:



High-pass / band-pass:

Edges in images



What does blurring take away?

original

What does blurring take away?

smoothed (5x5 Gaussian)

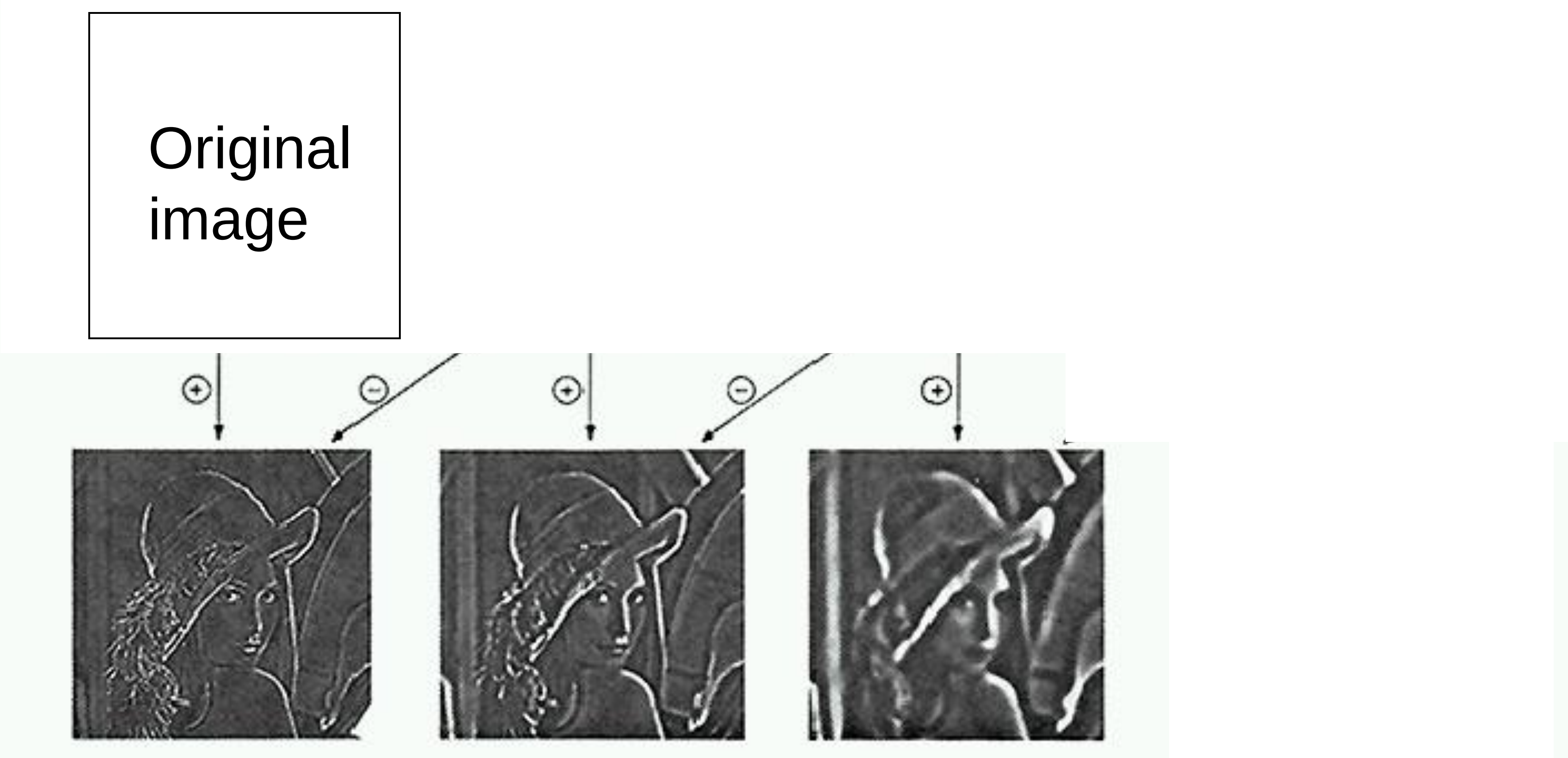
High-Pass filter

smoothed – original

Band-pass filtering

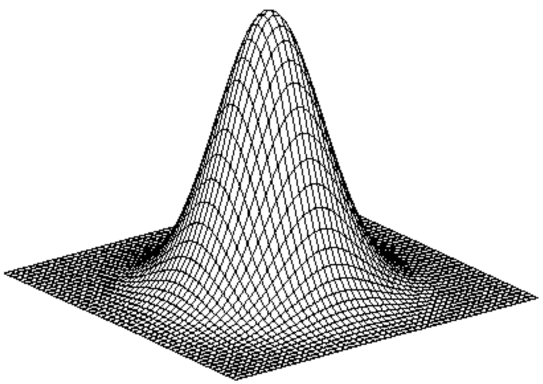
Gaussian Pyramid (low-pass images)

Laplacian Pyramid



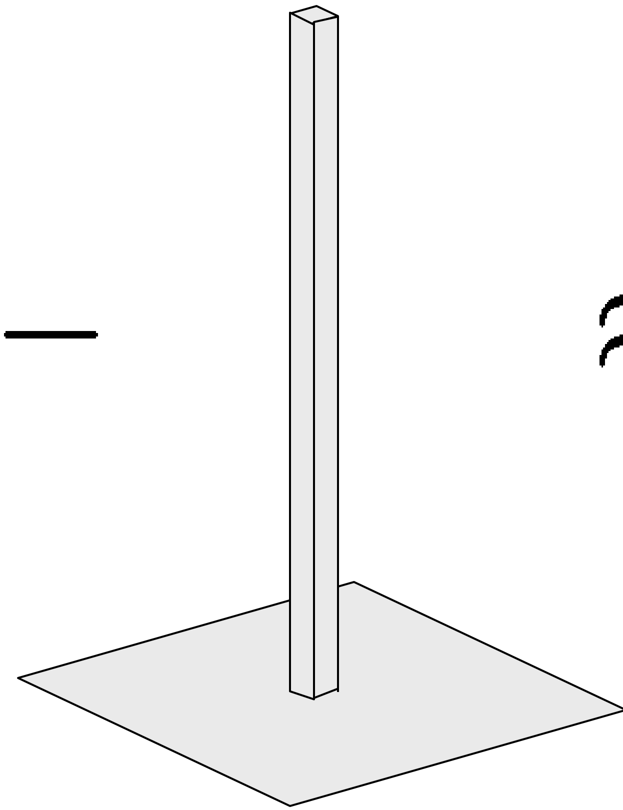
How can we reconstruct (collapse) this pyramid into the original image?

Why Laplacian?



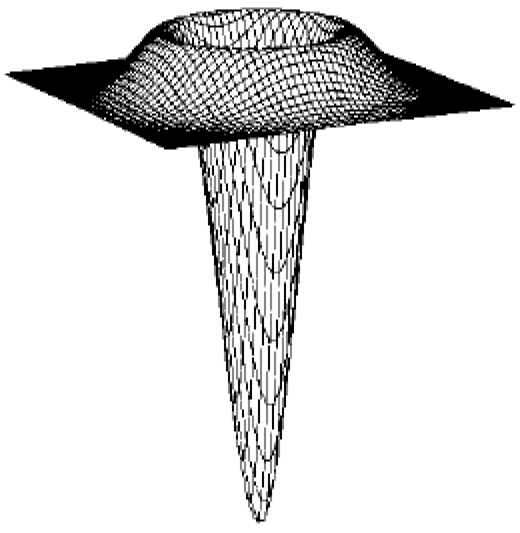
Gaussian

$-$



delta function

$\approx$



Laplacian of Gaussian

Unsharp Masking

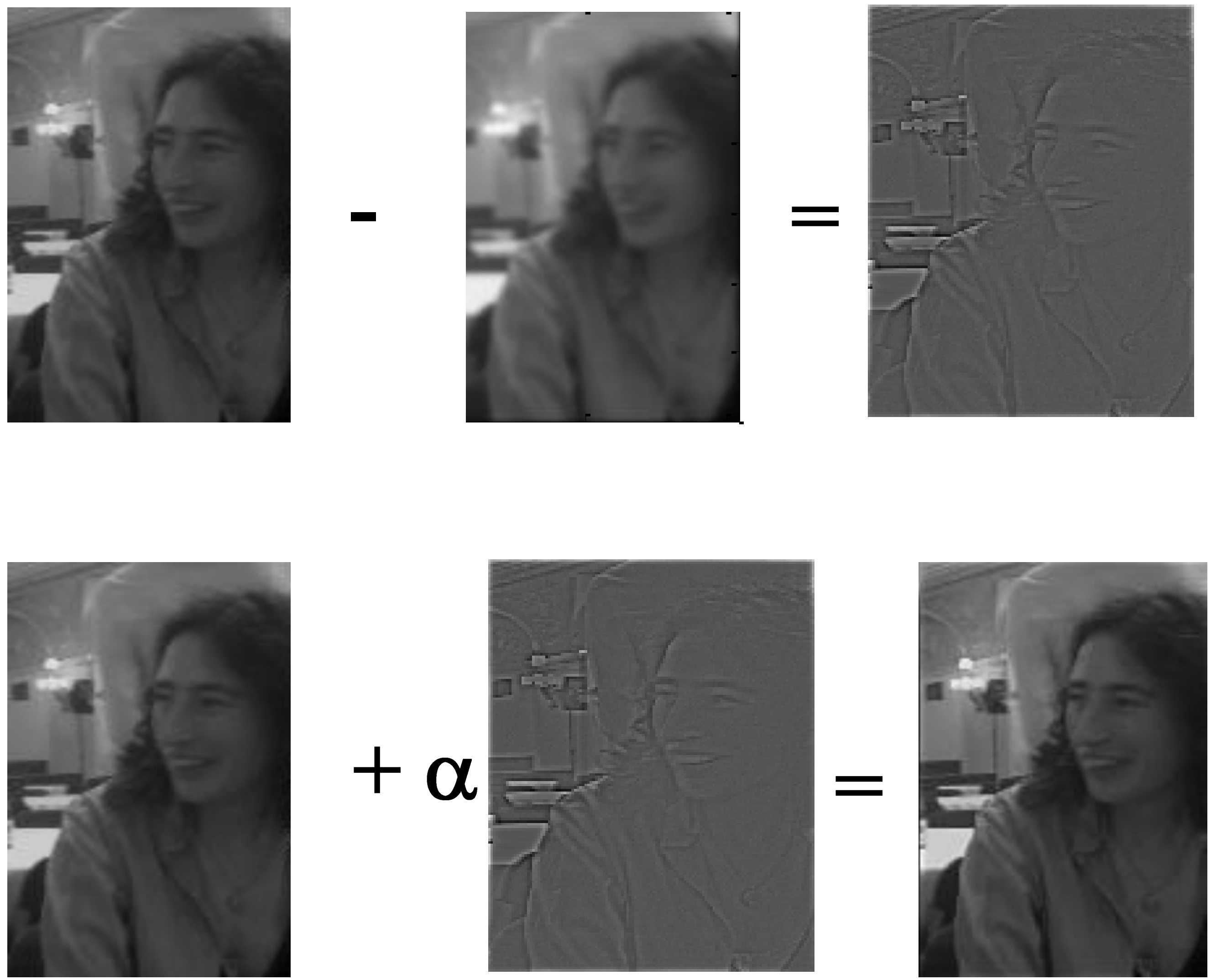
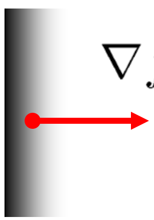


Image gradient

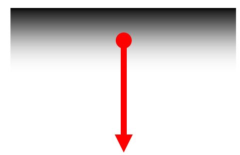
The gradient of an image:

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

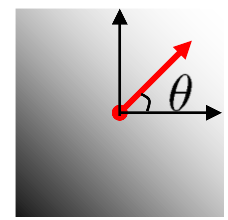
The gradient points in the direction of most rapid change in intensity



$$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$$



$$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$$



$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

The gradient direction is given by:

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

- how does this relate to the direction of the edge?

The *edge strength* is given by the gradient magnitude

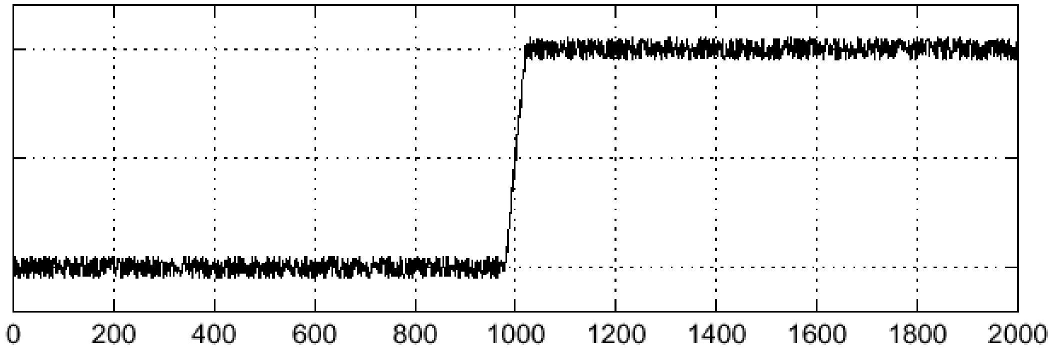
$$||\nabla f|| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Effects of noise

Consider a single row or column of the image

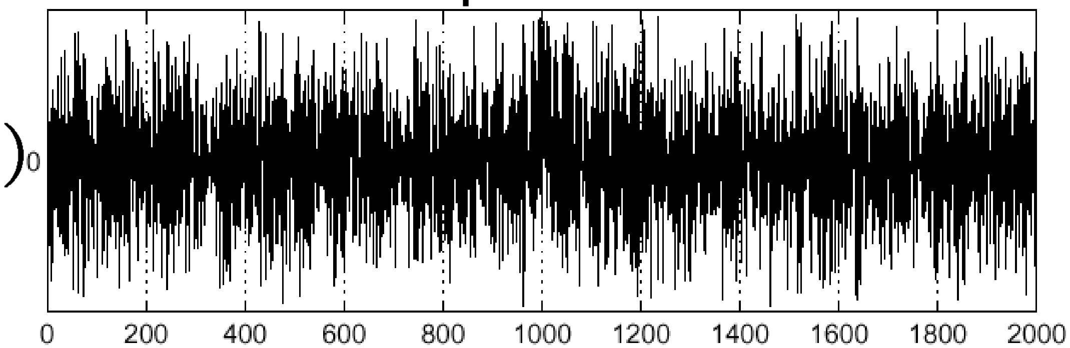
- Plotting intensity as a function of position gives a signal

$f(x)$



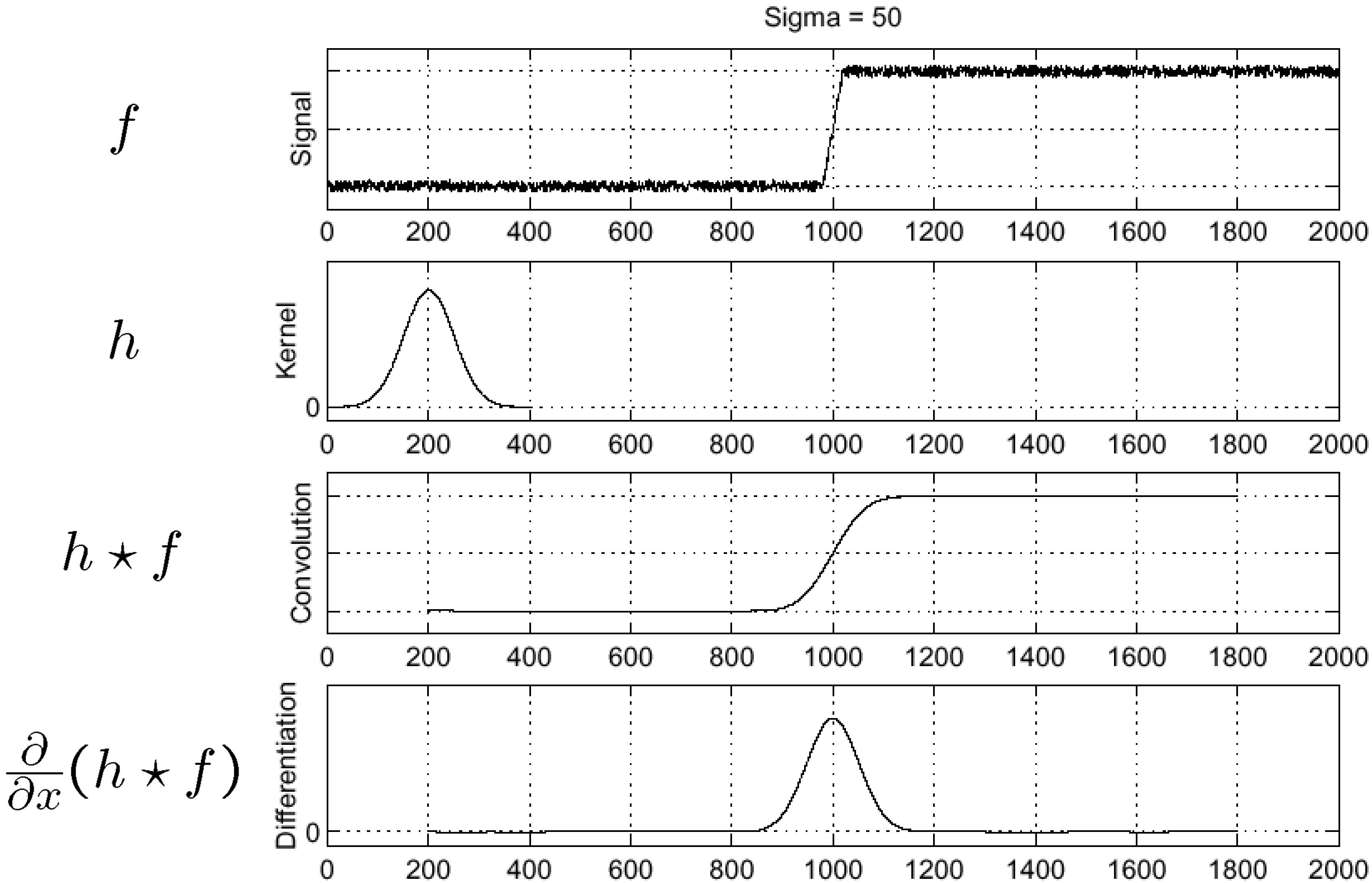
How to compute a derivative?

$\frac{d}{dx}f(x)$



Where is the edge?

Solution: smooth first



Where is the edge? Look for peaks in $\frac{\partial}{\partial x}(h \star f)$

Derivative theorem of convolution

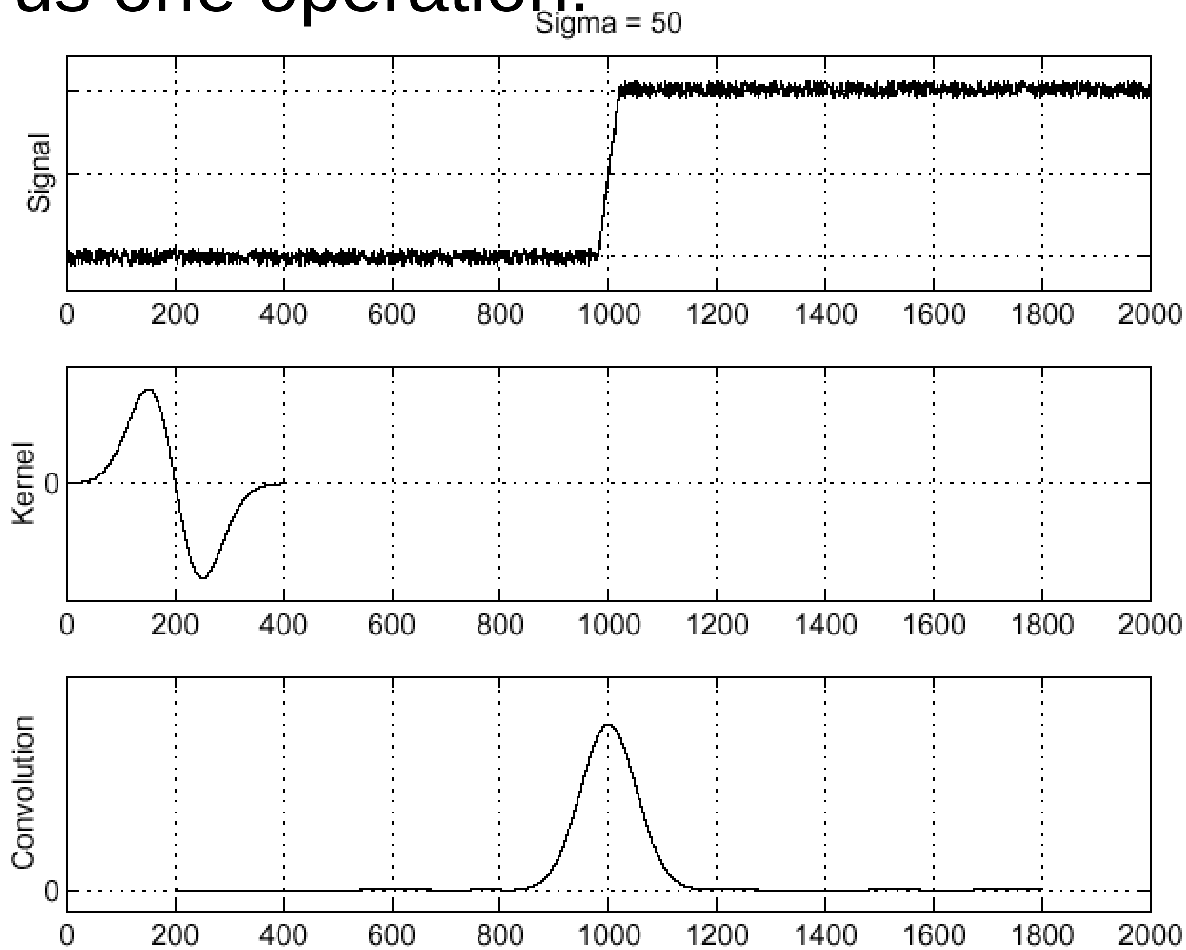
$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$

This saves us one operation:

f

$\frac{\partial}{\partial x}h$

$(\frac{\partial}{\partial x}h) \star f$



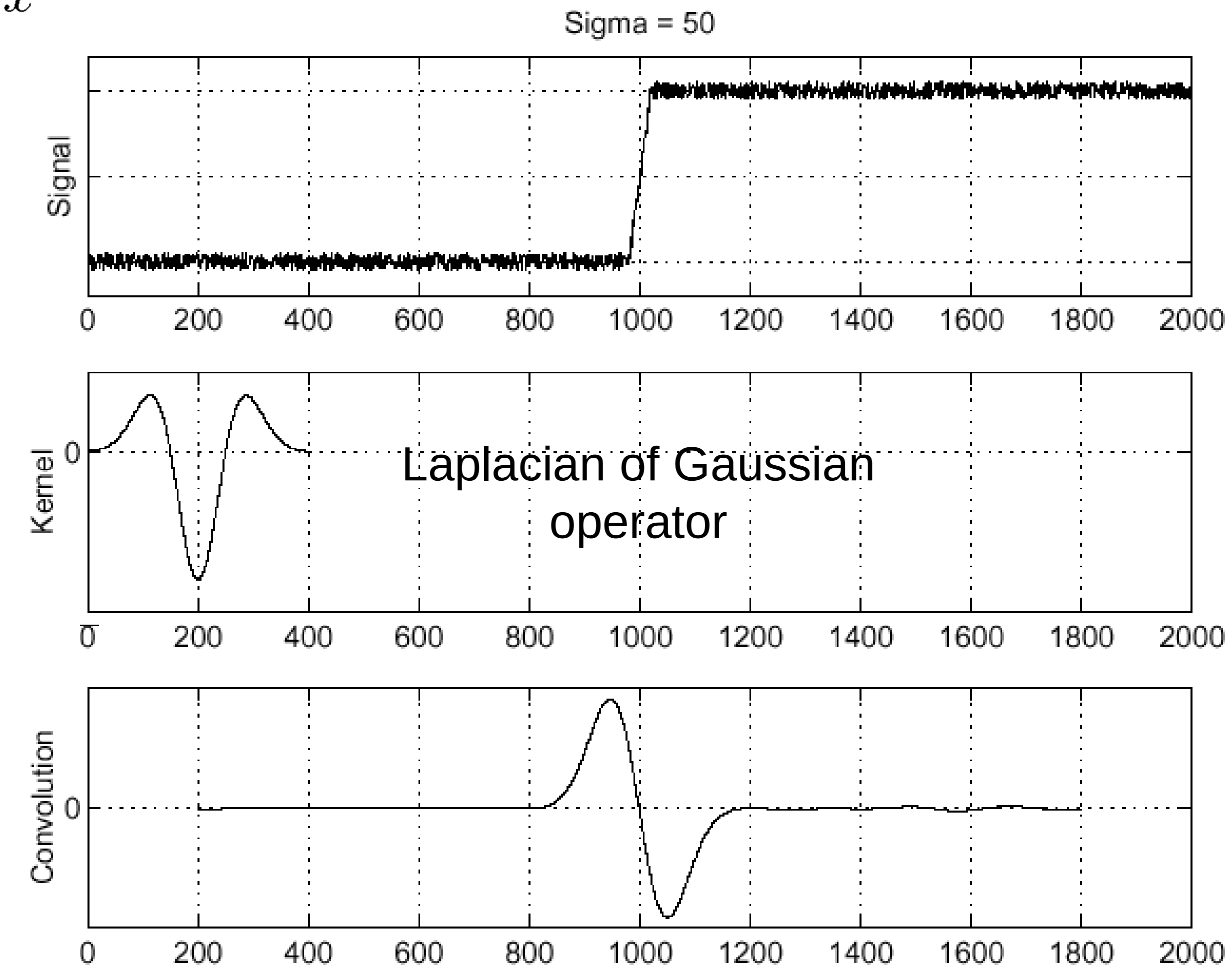
Laplacian of Gaussian

Consider $\frac{\partial^2}{\partial x^2}(h \star f)$

f

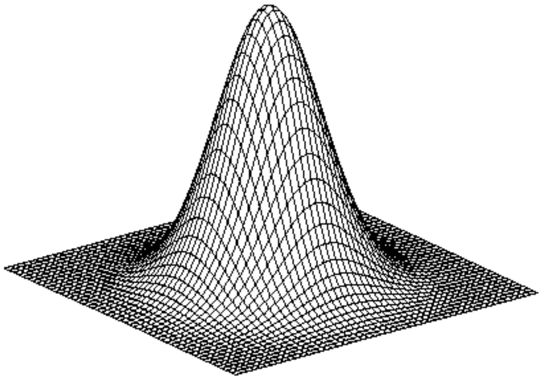
$\frac{\partial^2}{\partial x^2}h$

$(\frac{\partial^2}{\partial x^2}h) \star f$



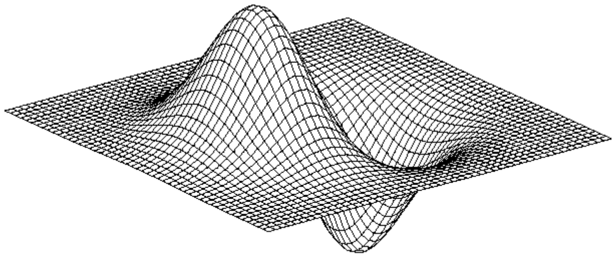
Where is the edge? Zero-crossings of bottom graph

2D edge detection filters



Gaussian

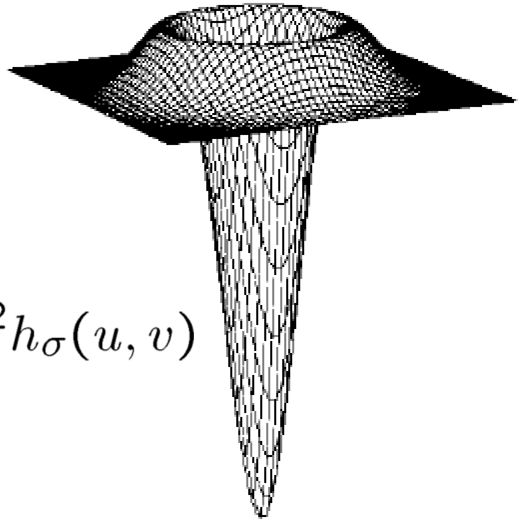
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



derivative of Gaussian

$$\frac{\partial}{\partial x} h_\sigma(u,v)$$

Laplacian of Gaussian



$$\nabla^2 h_\sigma(u,v)$$

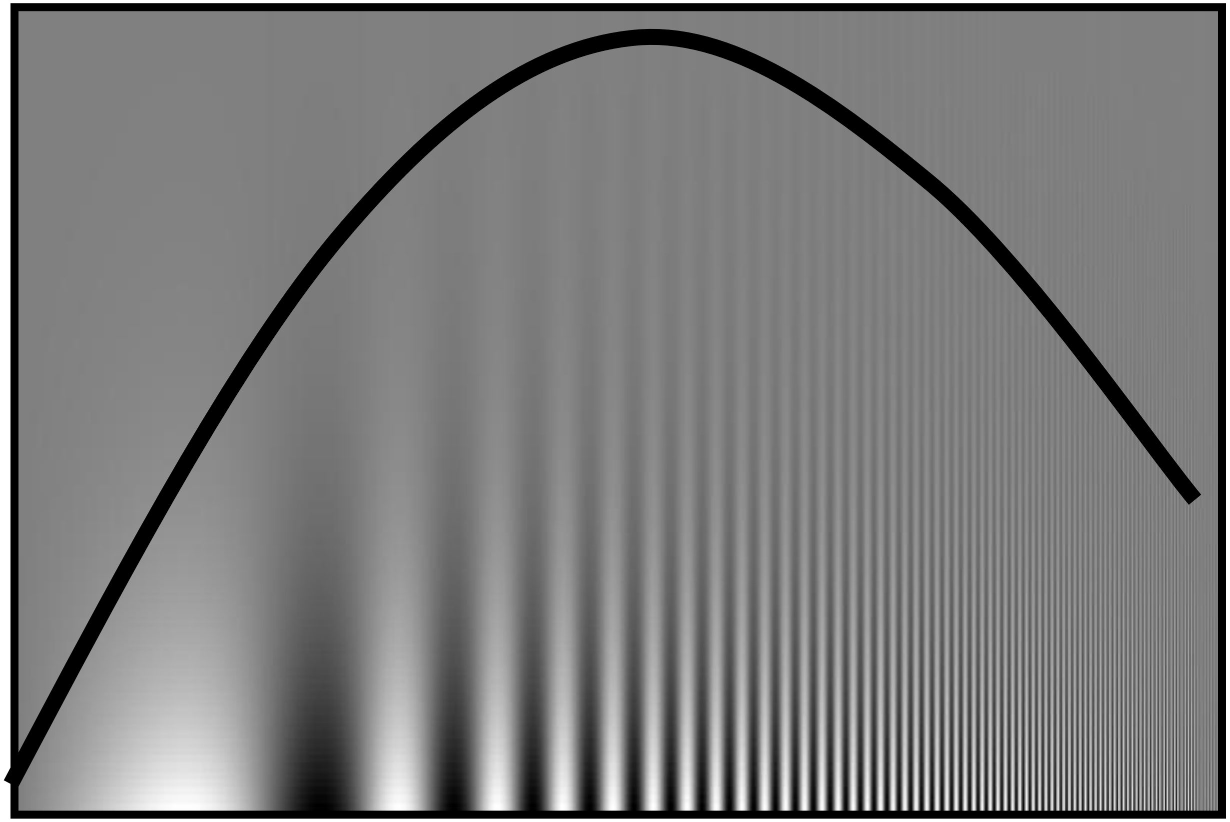
∇^2 is the **Laplacian** operator:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Try this in MATLAB

```
g = fspecial('gaussian',15,2);
imagesc(g);  colormap(gray);
surfl(g)
gclown = conv2(clown,g,'same');
imagesc(conv2(clown,[-1 1],'same'));
imagesc(conv2(gclown,[-1 1],'same'));
dx = conv2(g,[-1 1],'same');
imagesc(conv2(clown,dx,'same'));
lg = fspecial('log',15,2);
lclown = conv2(clown,lg,'same');
imagesc(lclown)
imagesc(clown + .2*lclown)
```

Campbell-Robson contrast sensitivity curve



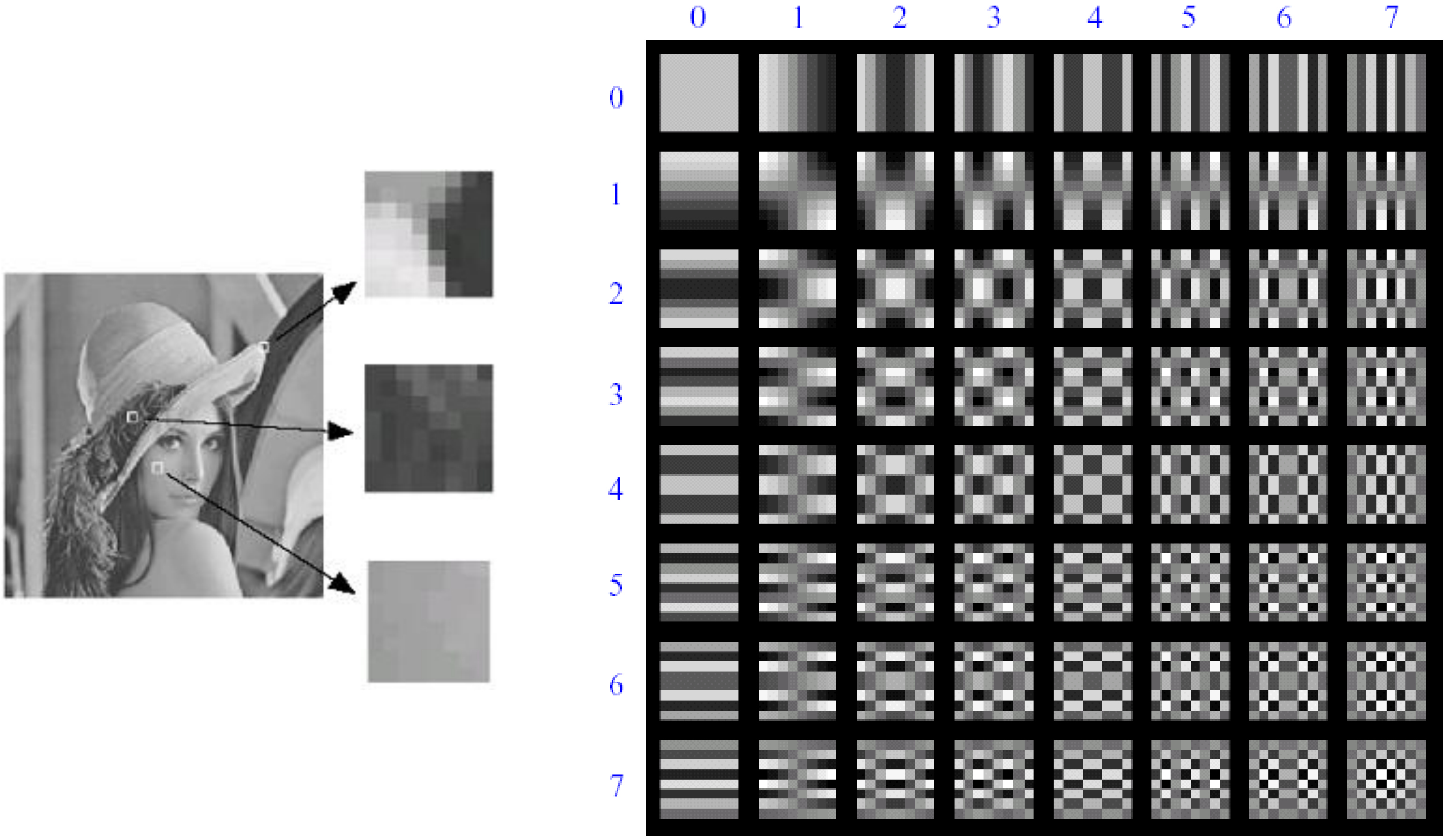
Depends on Color

R

G

B

Lossy Image Compression (JPEG)



Block-based Discrete Cosine Transform (DCT)

Using DCT in JPEG

The first coefficient $B(0,0)$ is the DC component, the average intensity

The top-left coeffs represent low frequencies, the bottom right – high frequencies

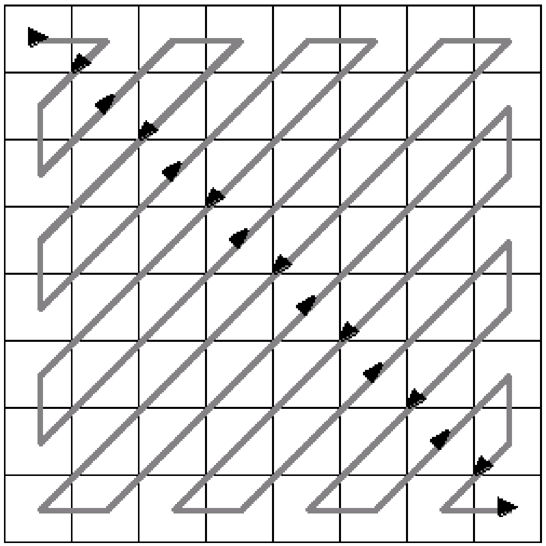


Image compression using DCT

DCT enables image compression by concentrating most image information in the low frequencies

Loose unimportant image info (high frequencies) by cutting $B(u,v)$ at bottom right

The decoder computes the inverse DCT – IDCT

- Quantization Table

3	5	7	9	11	13	15	17
5	7	9	11	13	15	17	19
7	9	11	13	15	17	19	21
9	11	13	15	17	19	21	23
11	13	15	17	19	21	23	25
13	15	17	19	21	23	25	27
15	17	19	21	23	25	27	29
17	19	21	23	25	27	29	31

Block size in JPEG

Block size

- small block
  - faster
  - correlation exists between neighboring pixels
- large block
  - better compression in smooth regions
- It’s 8x8 in standard JPEG

JPEG compression comparison

89k

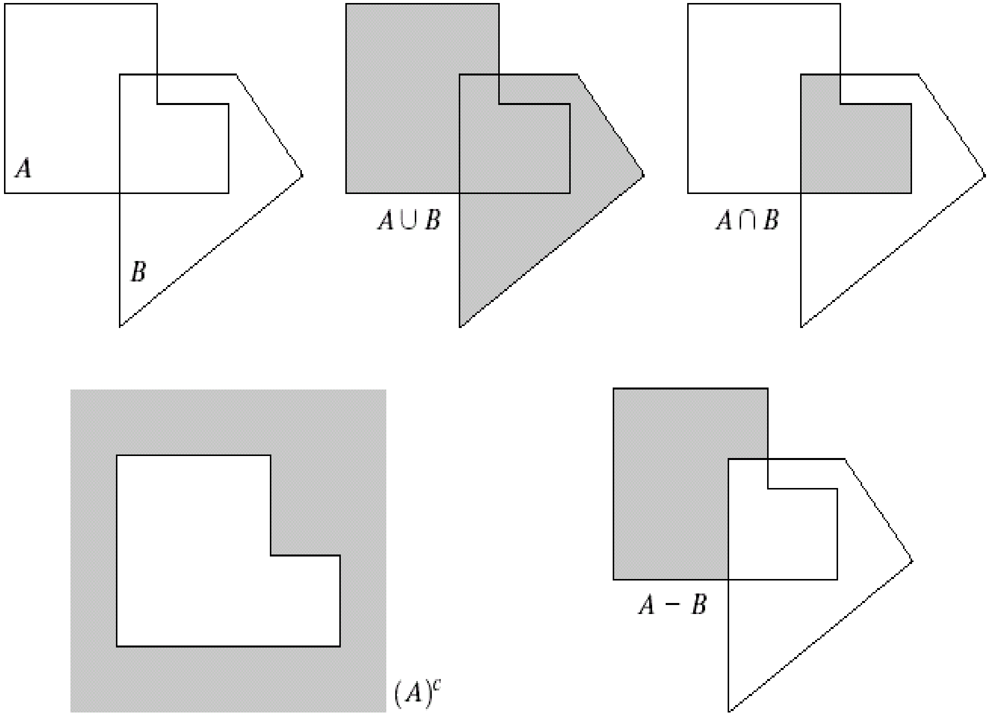
12k

Morphological Operation

What if your images are binary masks?

Binary image processing is a well-studied field, based on set theory, called Mathematical Morphology

Preliminaries



a b c
d e

FIGURE 9.1
(a) Two sets A and B. (b) The union of A and B. (c) The intersection of A and B. (d) The complement of A. (e) The difference between A and B.

Preliminaries

TABLE 9.1
The three basic logical operations.

p	q	p AND q (also $p \cdot q$)	p OR q (also $p + q$)	NOT (p) (also $\bar{p}$)
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Preliminaries

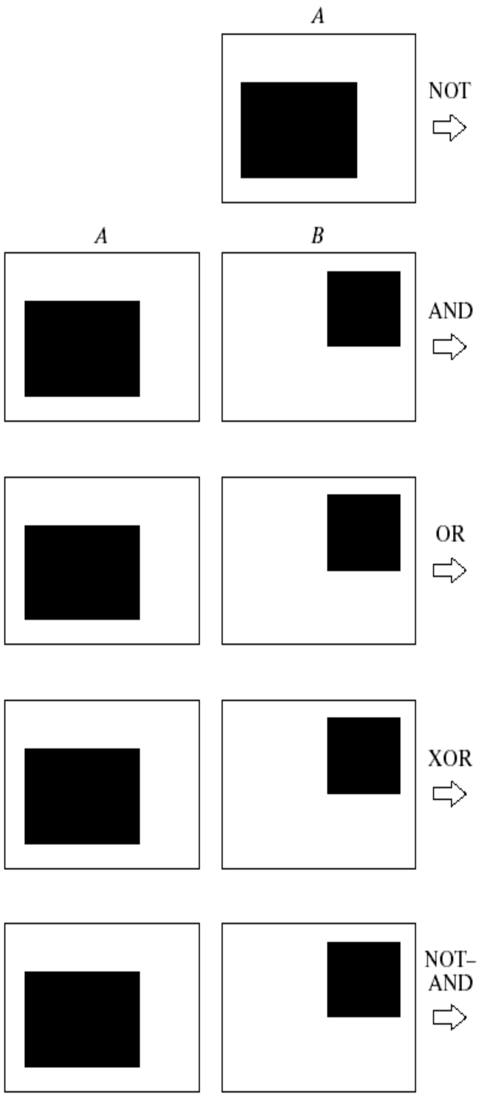


FIGURE 9.3 Some logic operations between binary images. Black represents binary 1s and white binary 0s in this example.

Basic Concepts in Set Theory

A is a set in Z^2, a=(a_1,a_2) an element of A, a$\in$A

If not, then a$\notin$A

$\emptyset$: null (empty) set

Typical set specification: C={w|w=-d, for d $\in$ D}

A subset of B: A$\subseteq$B

Union of A and B: C=A$\cup$B

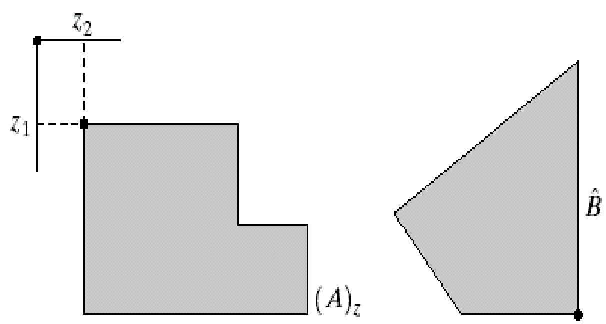
Intersection of A and B: D=A$\cap$B

Disjoint sets: A$\cap$B= $\emptyset$

Complement of A: $A^c = \{w \mid w \notin A\}$

Difference of A and B: A-B={w|w $\in$ A, w $\notin$ B}= $A \cap B^c$

Preliminaries



a b

FIGURE 9.2 (a) Translation of A by z. (b) Reflection of B. The sets A and B are from Fig. 9.1.

$$\hat{B} = \{w \mid w = -b, \ for \ \ b \in B\}$$

$$(A)_z = \{c \mid c = a + z, \ for \ \ a \in A\}$$

Dilation and Erosion

0	1	0
1	1	1
0	1	0

Two basic operations:

- A is the image, B is the "structural element", a mask akin to a kernel in convolution

Dilation :

$$A \oplus B = \{z \mid (\hat{B})_z \cap A \neq \phi\}$$

$$A \oplus B = \{z \mid [(\hat{B})_z \cap A] \subseteq A\}$$

(all shifts of B that have a non-empty overlap with A)

$$A \Theta B = \{z \mid (B)_z \subseteq A\}$$

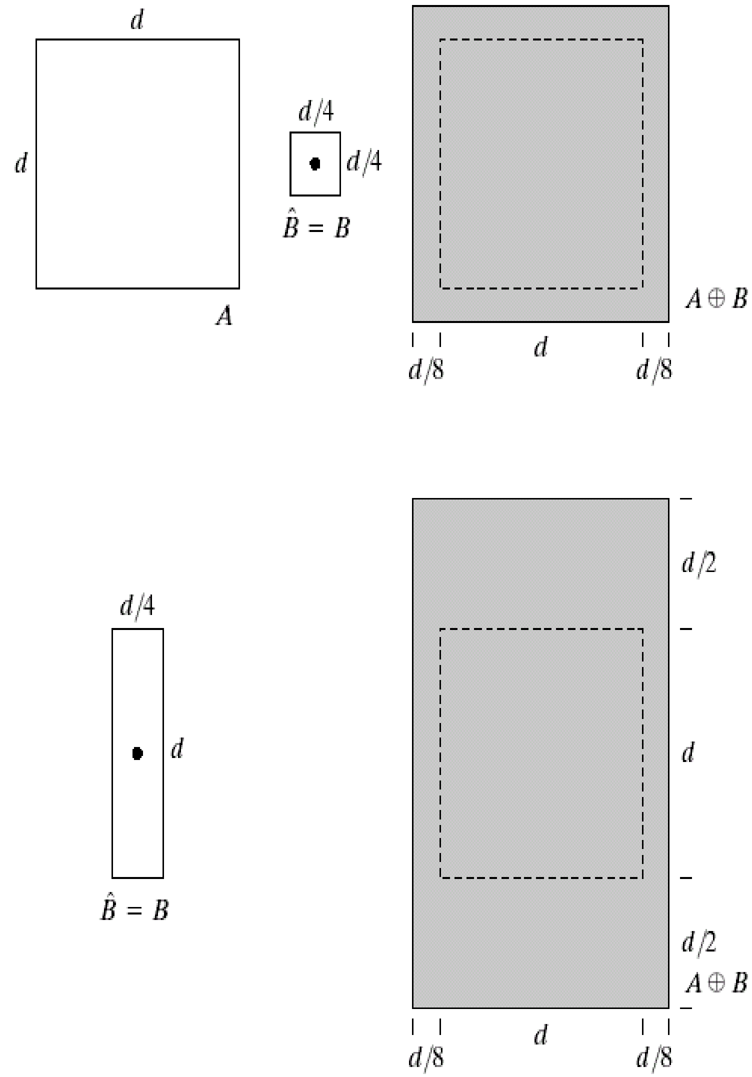
Erosion :

(all shifts of B that are fully contained within A)

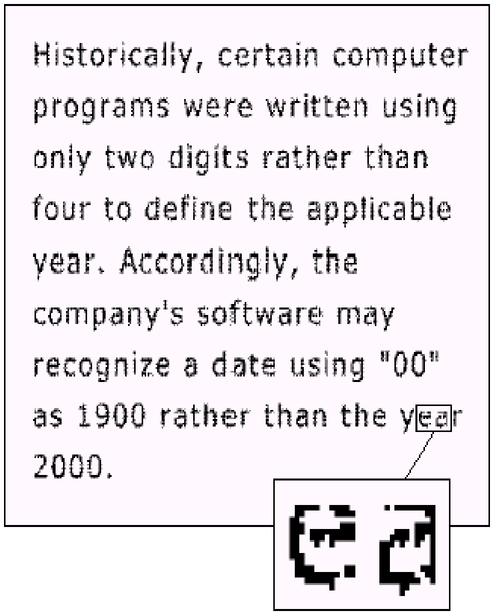
Dilation

a b c
d e

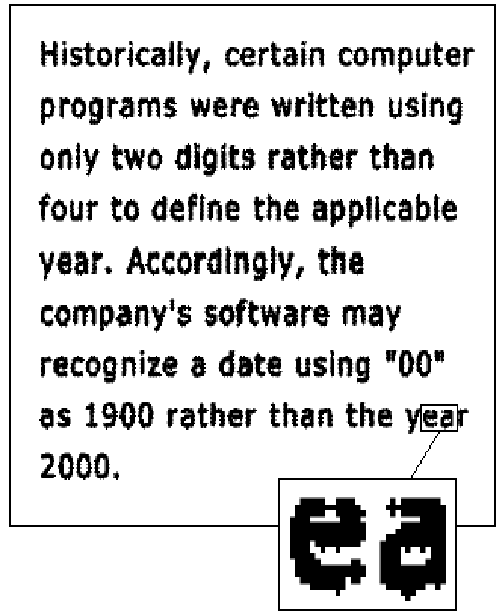
FIGURE 9.4
(a) Set A.
(b) Square structuring element (dot is the center).
(c) Dilation of A by B, shown shaded.
(d) Elongated structuring element.
(e) Dilation of A using this element.



Dilation



Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.



Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.

0	1	0
1	1	1
0	1	0

a c
b

FIGURE 9.5
(a) Sample text of poor resolution with broken characters (magnified view).
(b) Structuring element.
(c) Dilation of (a) by (b). Broken segments were joined.

Erosion

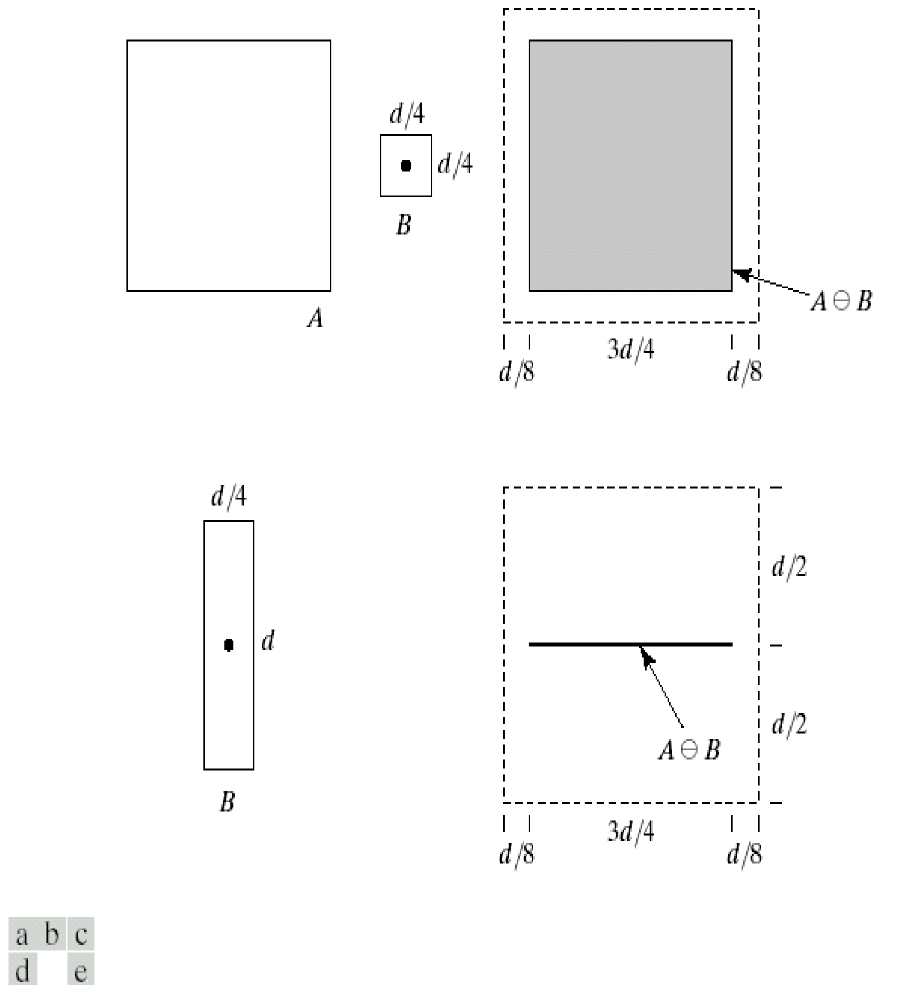
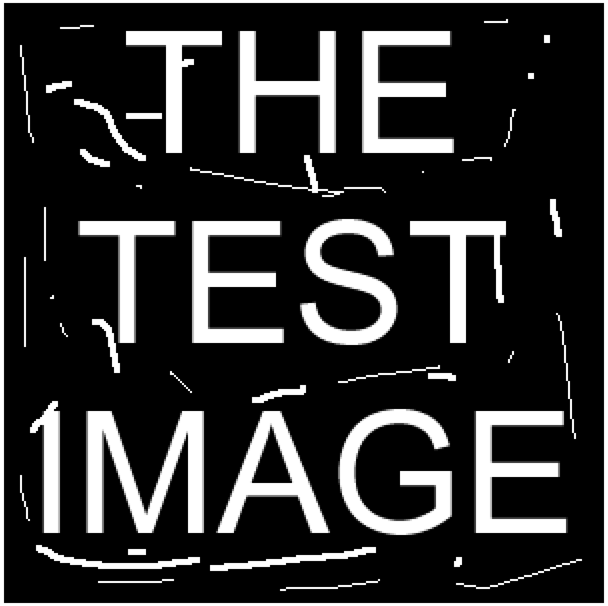
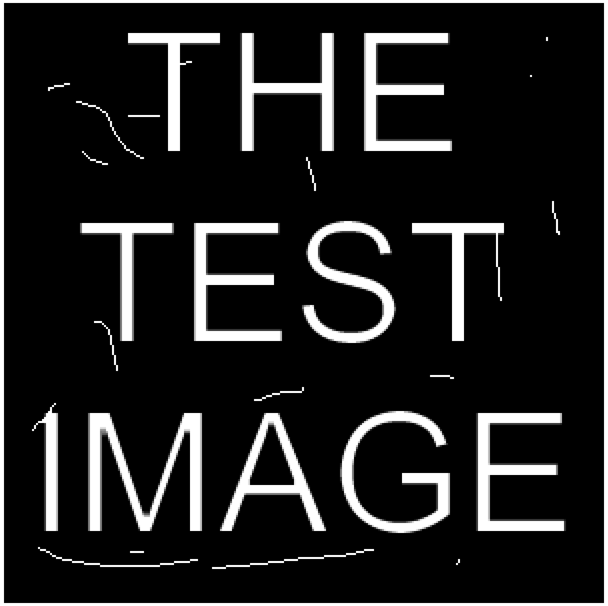


FIGURE 9.6 (a) Set A. (b) Square structuring element. (c) Erosion of A by B, shown shaded. (d) Elongated structuring element. (e) Erosion of A using this element.

Erosion

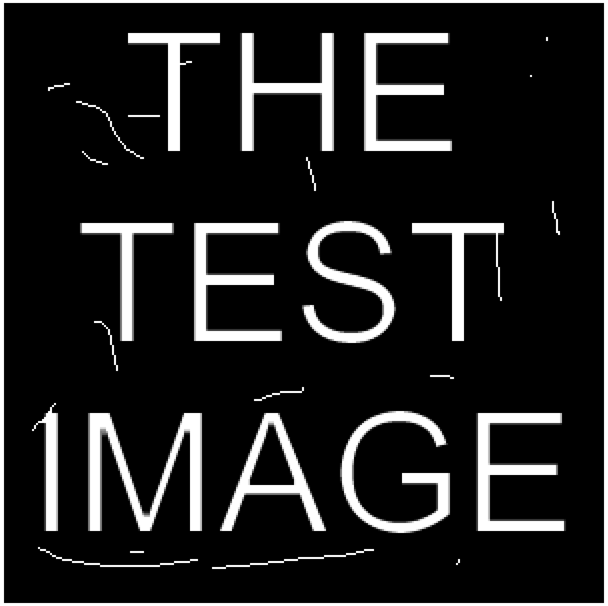


Original image



Eroded image

Erosion



Eroded once

Eroded twice

Opening and Closing

Opening : smoothes the contour of an object, breaks narrow isthmuses, and eliminates thin protrusions

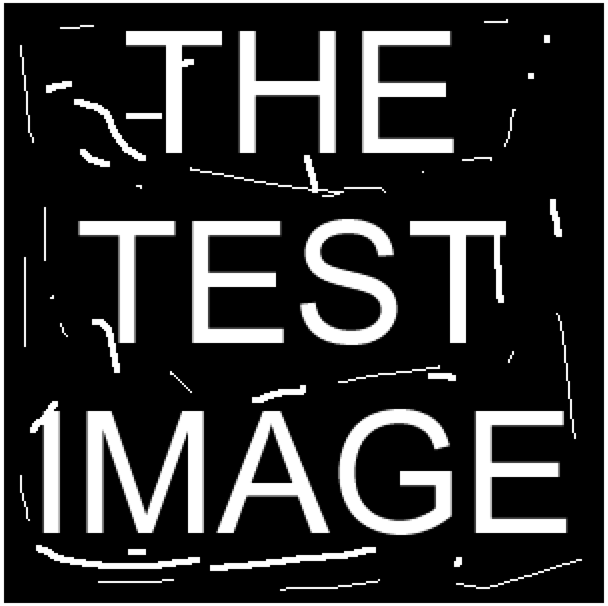
$$A \circ B = (A \Theta B) \oplus B$$

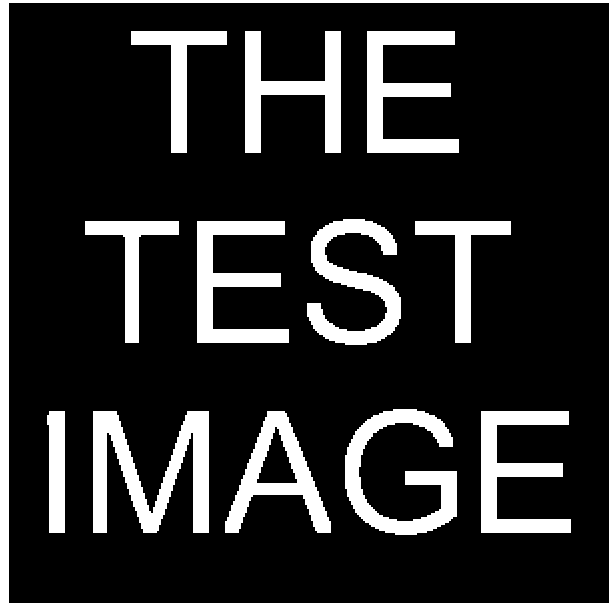
Closing : smooth sections of contours but, as opposed to opning, it generally fuses narrow breaks and long thin gulfs, eliminates small holes, and fills gaps in the contour

$$A \bullet B = (A \oplus B) \Theta B$$

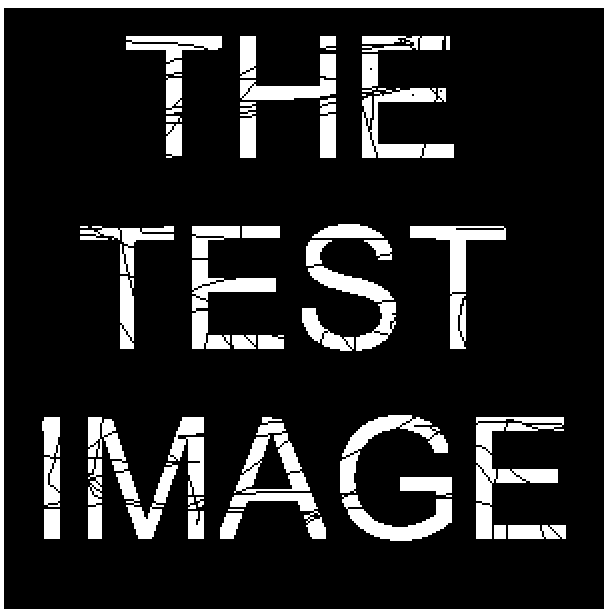
Prove to yourself that they are not the same thing. Play around with `bwmorph` in Matlab.

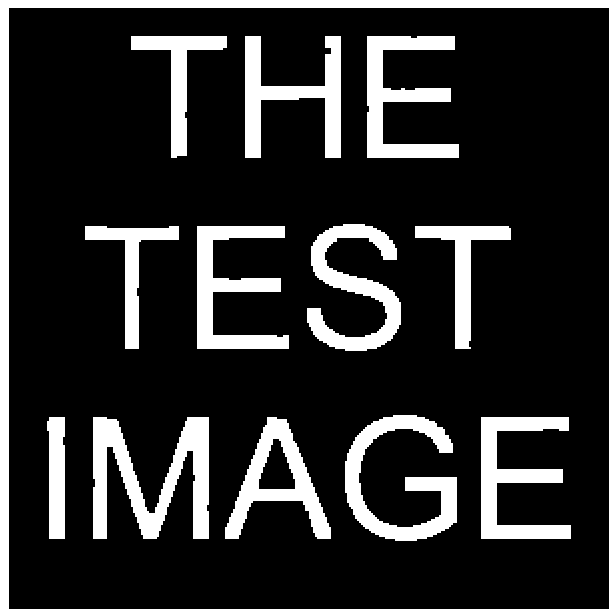
Opening and Closing





OPENING: The original image eroded twice and dilated twice (opened). Most noise is removed



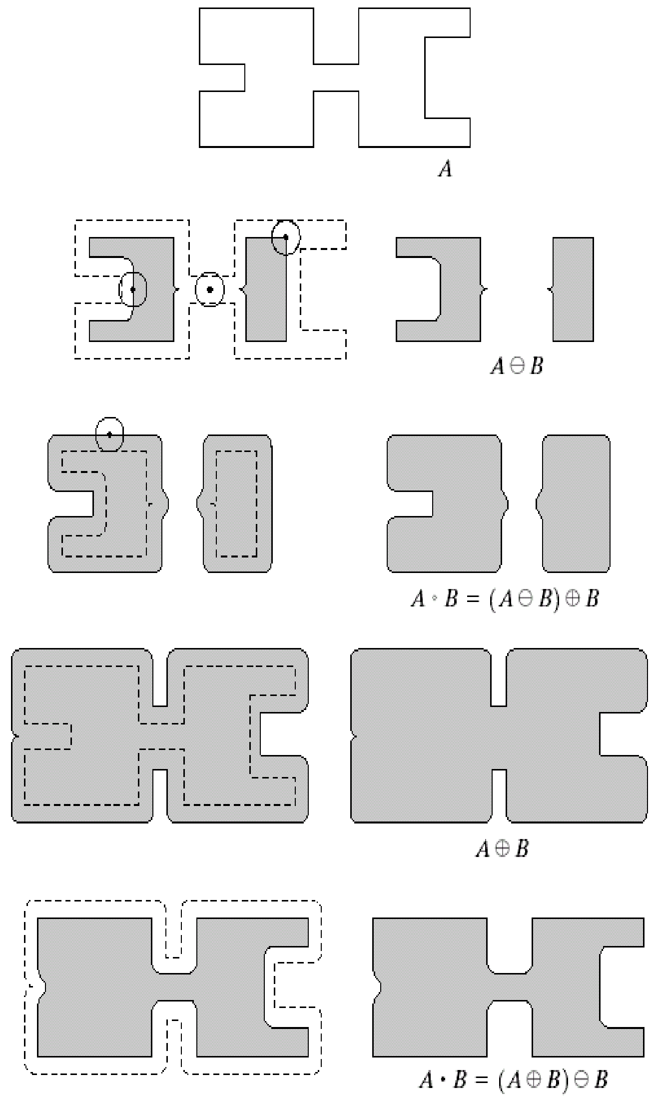


CLOSING: The original image dilated and then eroded. Most holes are filled.

Opening and Closing

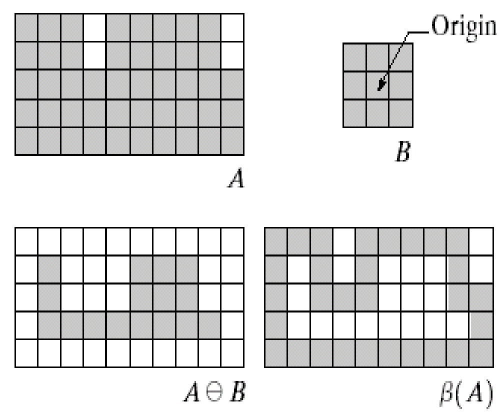
a	
b	c
d	e
f	g
h	i

FIGURE 9.10 Morphological opening and closing. The structuring element is the small circle shown in various positions in (b). The dark dot is the center of the structuring element.



Boundary Extraction

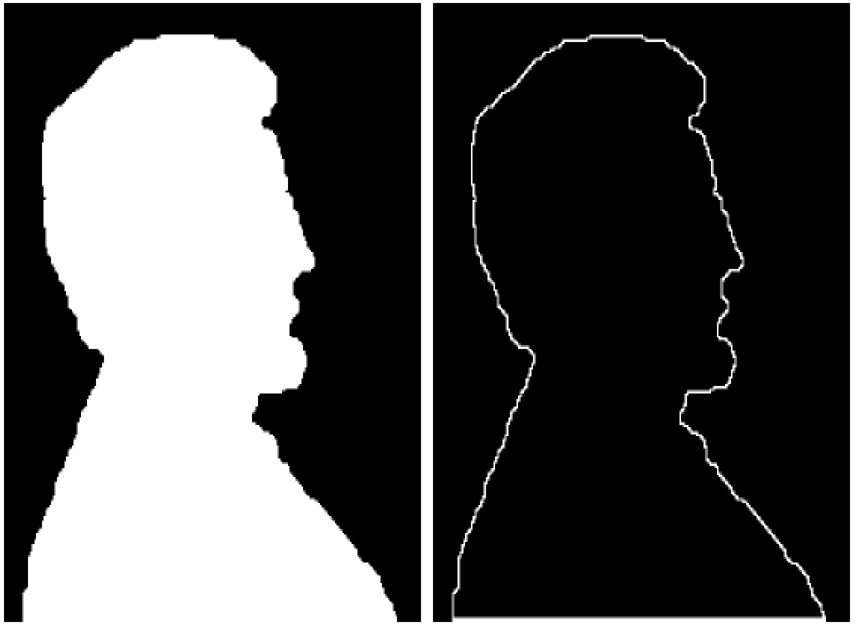
$$\beta(A) = A - (A \ominus B)$$



a b
c d

FIGURE 9.13 (a) Set A. (b) Structuring element B. (c) A eroded by B. (d) Boundary, given by the set difference between A and its erosion.

Boundary Extraction



a b

FIGURE 9.14
(a) A simple binary image, with 1's represented in white. (b) Result of using Eq. (9.5-1) with the structuring element in Fig. 9.13(b).

Project #2: Miniatures!

Project #2: Fake Miniatures!