Introduction to Computer Vision

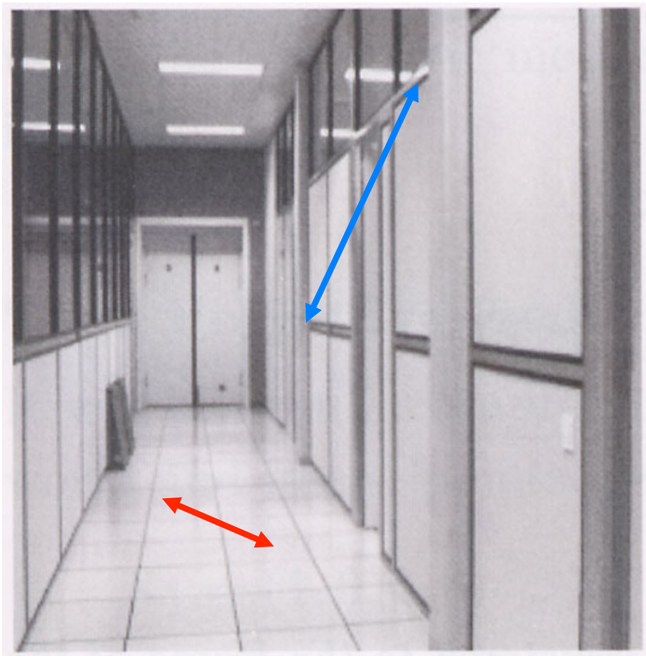
Week 3, Fall 2010

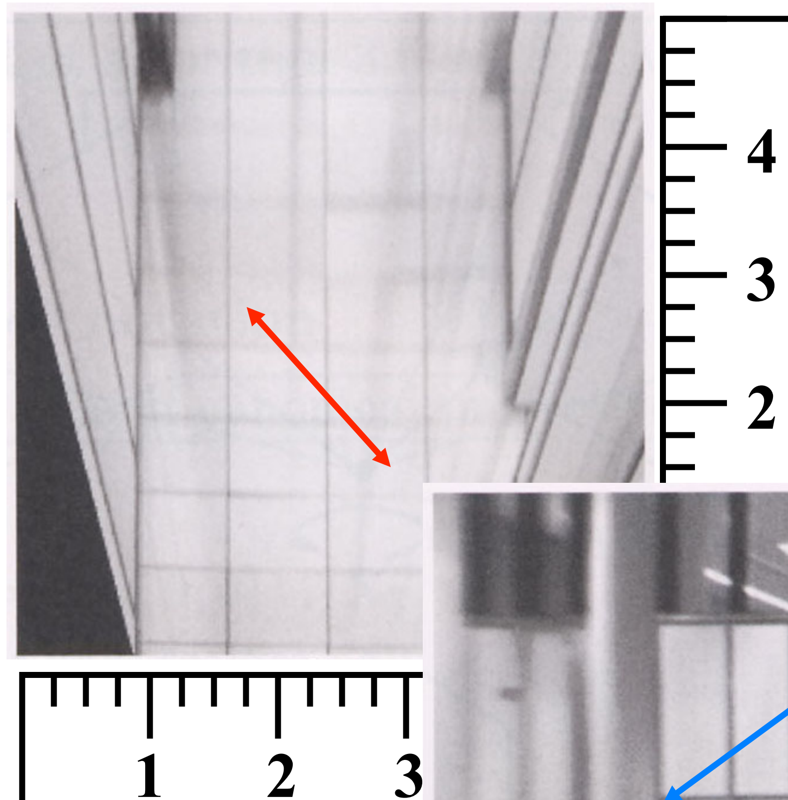
Instructor: Prof. Ko Nishino

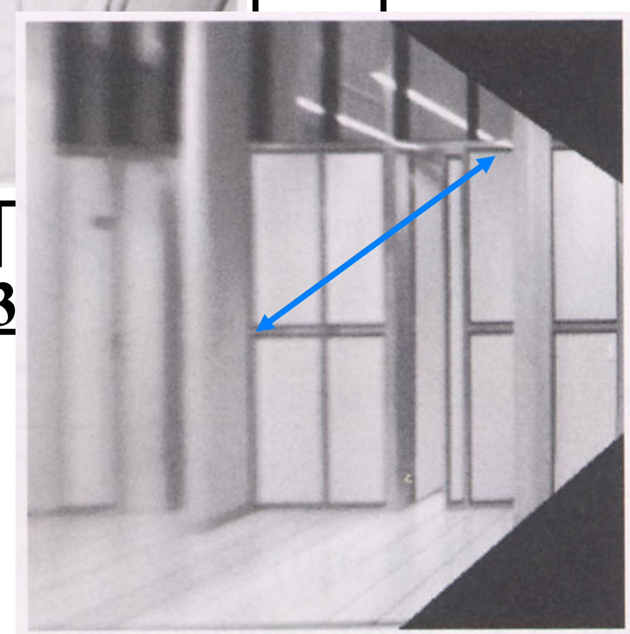
Last Week

- Image Sensing
  - Our eyes: rods and cones…
  - CCD, CMOS, Rolling Shutter
  - Sensing brightness and sensing color
- Projective Geometry/Camera Calibration
  - Projection models: perspective, orthographic, weak-perspective, and affine
  - Homogeneous coordinates (to and from)
  - Camera parameters: intrinsic and extrinsic
  - Camera calibration
    - Direct Linear Calibration: Total Least Square
    - Non-linear calibration

Measurements on Planes



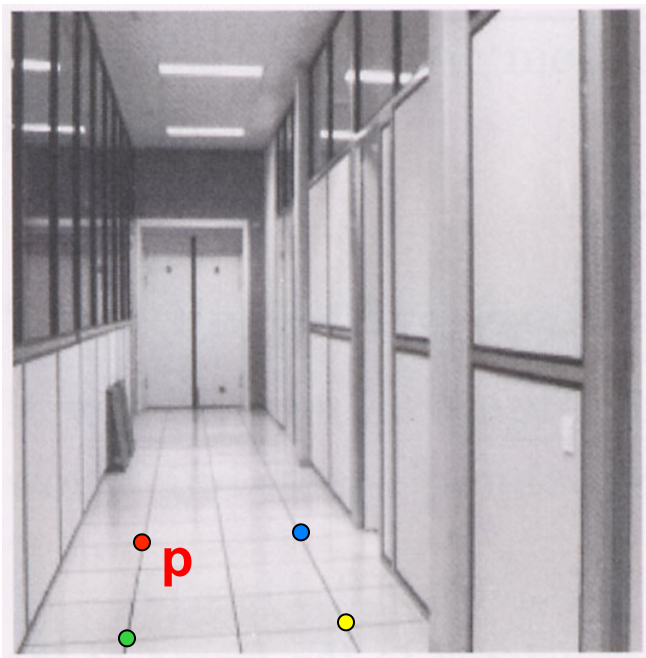


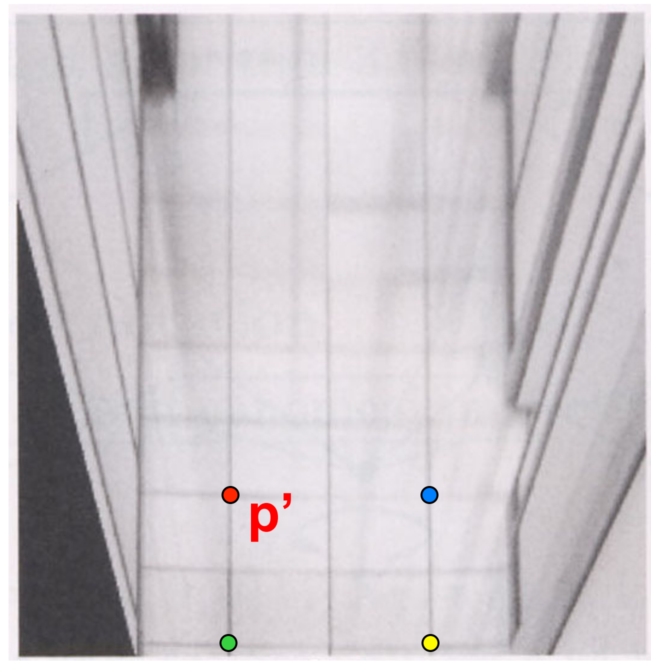


Approach: unwarp then measure

What kind of warp is this?

Image Rectification





To unwarp (rectify) an image

- solve for homography **H** given **p** and **p’**
- solve equations of the form: w**p’** = **Hp**
  - linear in unknowns: w and coefficients of **H**
  - H is defined up to an arbitrary scale factor
  - how many points are necessary to solve for **H**?

Solving for Homographies

$$\begin{bmatrix} x_i' \\ y_i' \\ 1 \end{bmatrix} \cong \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$\begin{aligned} x_i' &= \frac{h_{00}x_i + h_{01}y_i + h_{02}}{h_{20}x_i + h_{21}y_i + h_{22}} \\ y_i' &= \frac{h_{10}x_i + h_{11}y_i + h_{12}}{h_{20}x_i + h_{21}y_i + h_{22}} \end{aligned}$$

$$\begin{aligned} x_i'(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{00}x_i + h_{01}y_i + h_{02} \\ y_i'(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{10}x_i + h_{11}y_i + h_{12} \end{aligned}$$

$$\begin{bmatrix} x_i & y_i & 1 & 0 & 0 & 0 & -x_i'x_i & -x_i'y_i & -x_i' \\ 0 & 0 & 0 & x_i & y_i & 1 & -y_i'x_i & -y_i'y_i & -y_i' \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Solving for Homographies

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1 x_1 & -x'_1 y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1 x_1 & -y'_1 y_1 & -y'_1 \\ & & & & \vdots & & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x'_n x_n & -x'_n y_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -y'_n x_n & -y'_n y_n & -y'_n \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

$$\underset{2n \times 9}{\mathbf{A}} \qquad \underset{9}{\mathbf{h}} \qquad \underset{2n}{\mathbf{0}}$$

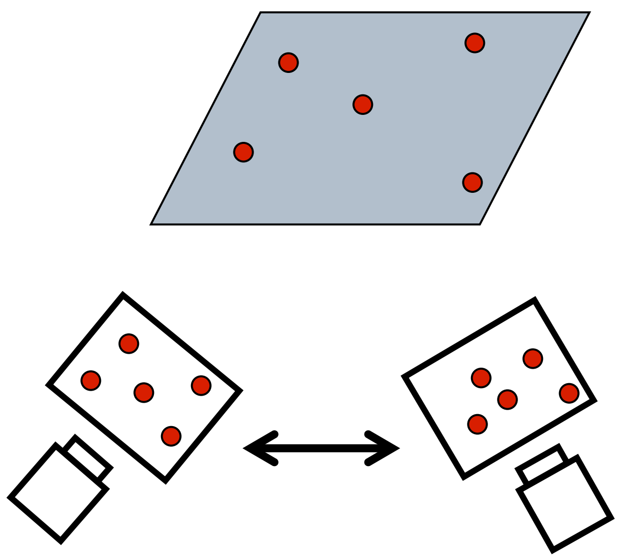
- Total least squares
  - Since $\mathbf{h}$ is only defined up to scale, solve for unit vector $\hat{\mathbf{h}}$
  - Minimize $\|\mathbf{A}\hat{\mathbf{h}}\|^2$

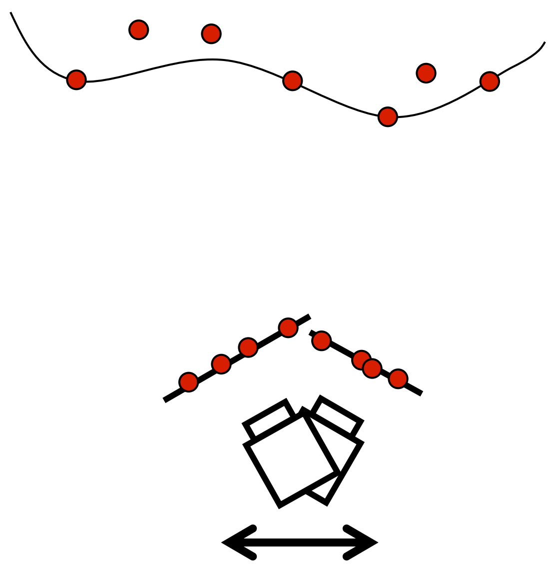
$$\|\mathbf{A}\hat{\mathbf{h}}\|^2 = (\mathbf{A}\hat{\mathbf{h}})^T \mathbf{A}\hat{\mathbf{h}} = \hat{\mathbf{h}}^T \mathbf{A}^T \mathbf{A}\hat{\mathbf{h}}$$

  - Solution: $\hat{\mathbf{h}}$ = eigenvector of $\mathbf{A}^T\mathbf{A}$ with smallest eigenvalue
  - Works with 4 or more points (more points more accurate)

Homography

- Homography is a singular case of the Fundamental Matrix
  - Two views of **coplanar points**
  - Two views that **share the same center of projection**



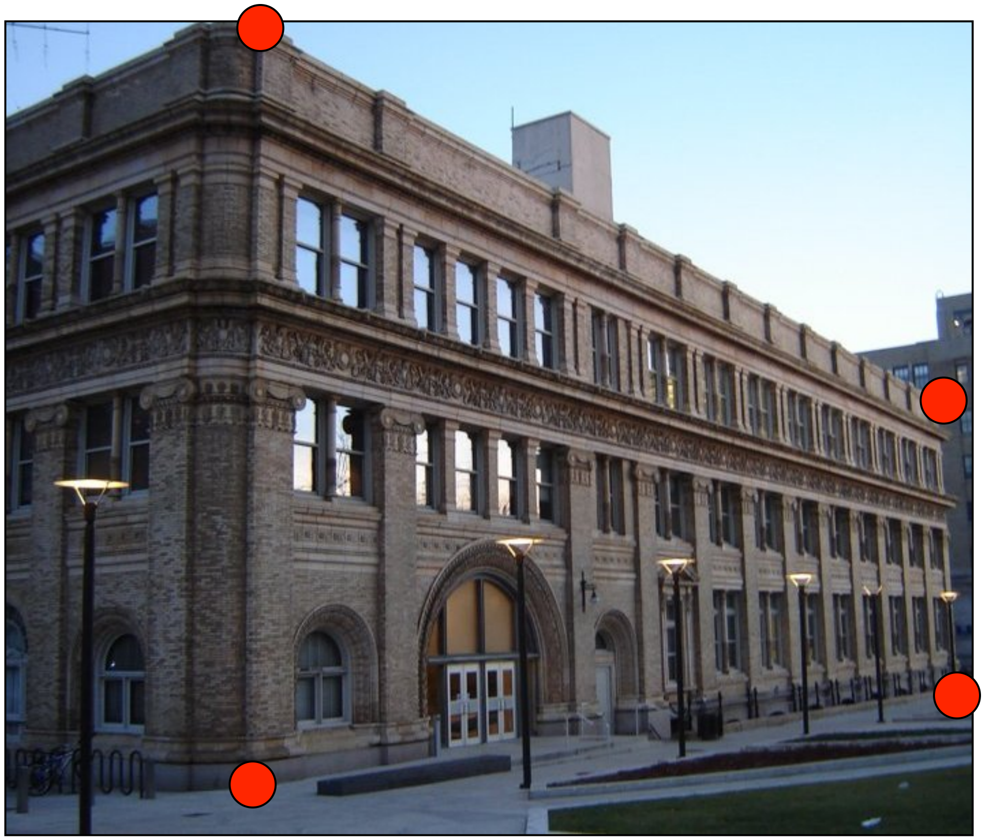


Project #1: Homography

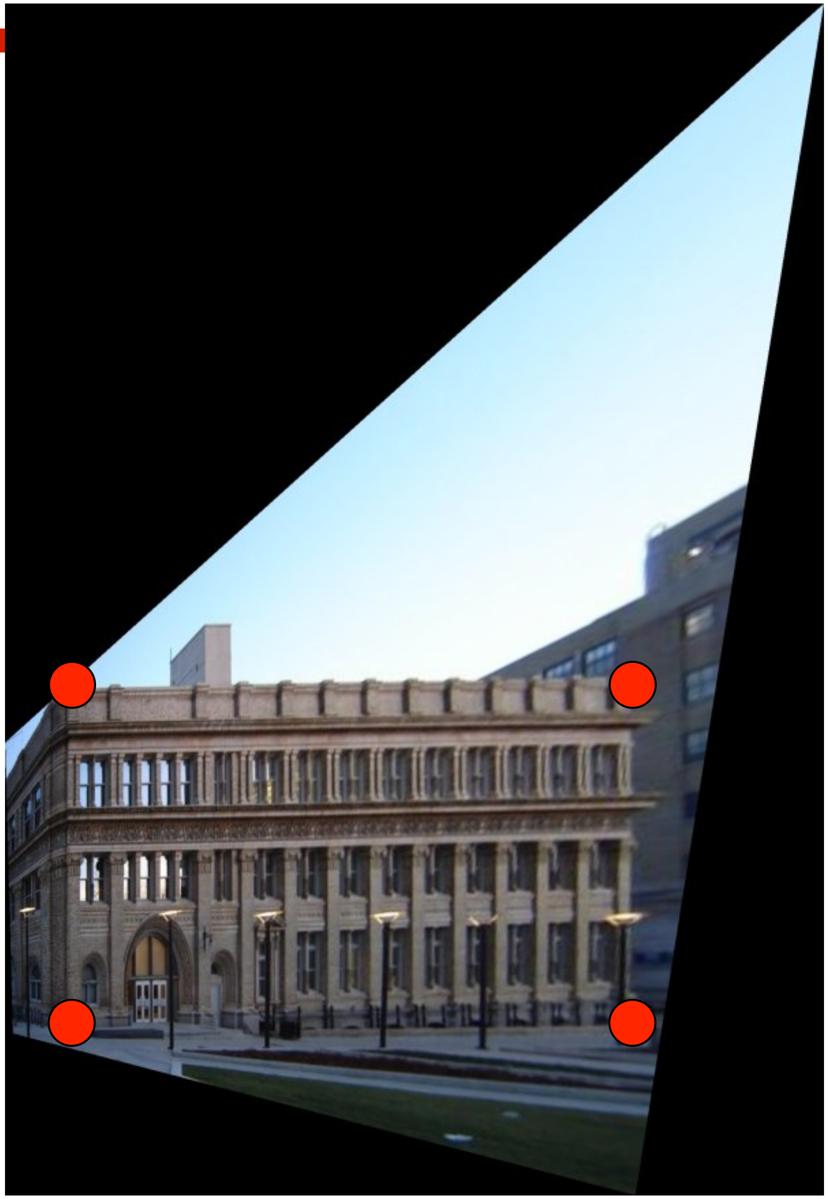
- Rectification
  - Take two images of an object with a planar surface
  - Make a fronto-parallel image of one of the planar surfaces
  - Submit results for three images including the test
- Compositing
  - Take two images
  - Composite the entire or part of one image into another using the homography of corresponding regions
  - Submit results for three images including the test
- Planar Mosaic (Extra Credit)

Example

- Rectification



This is your test image



Example

- Rectification

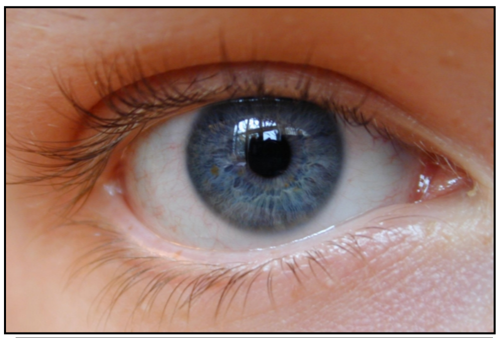
Example

- Compositing

This is your test image set

Example

- Composite
  - Need not be rectangular
  - Masking and Blending



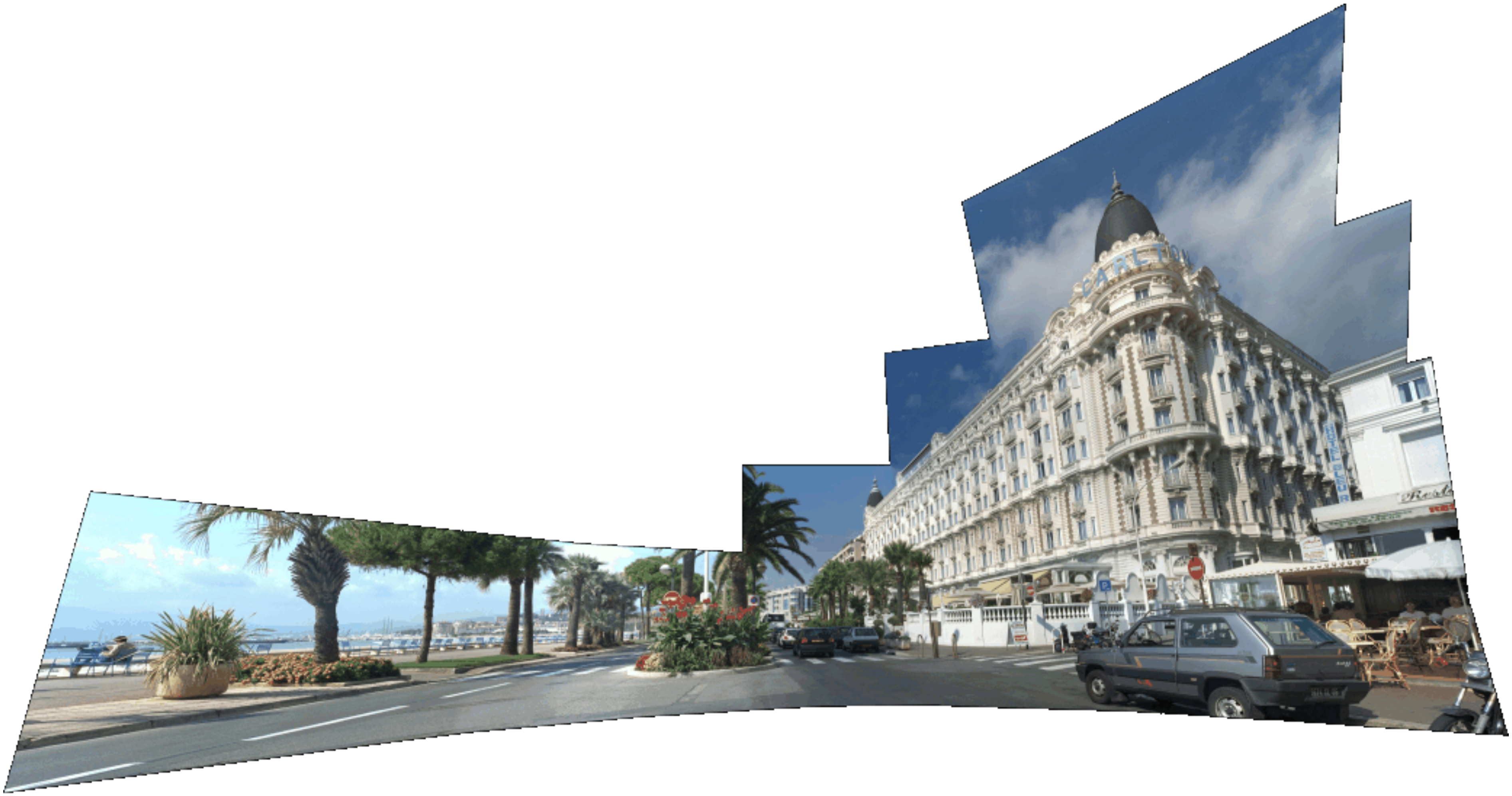
Example

- Planar Mosaic (Extra Credit)



Note that the COP is the same


CARLTON

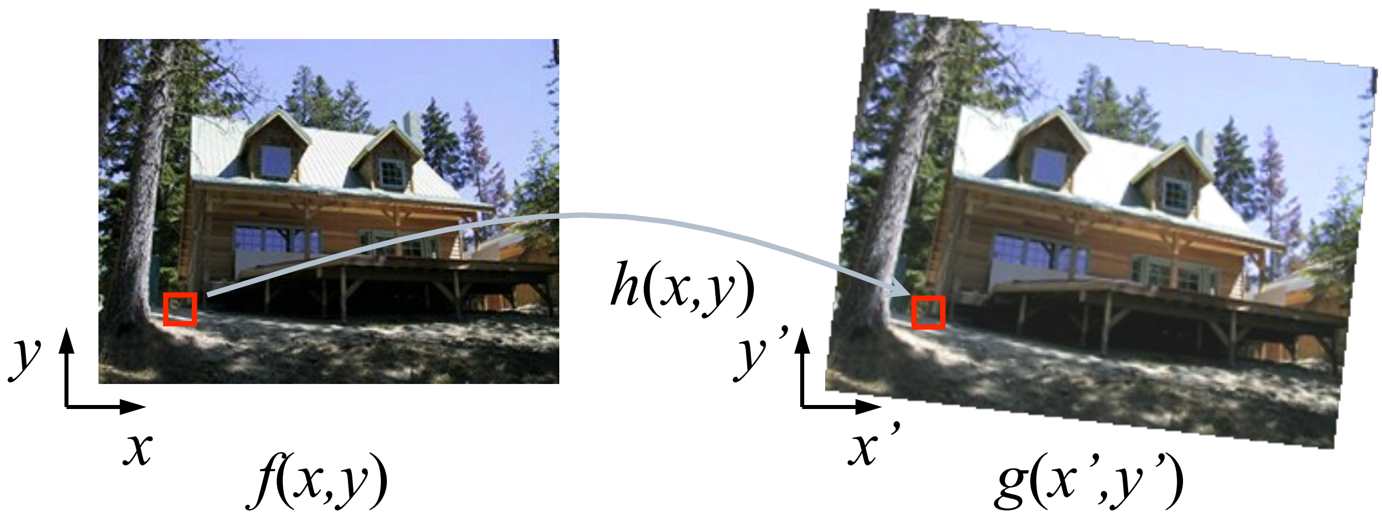


by I. Zoghlami

Ingredients

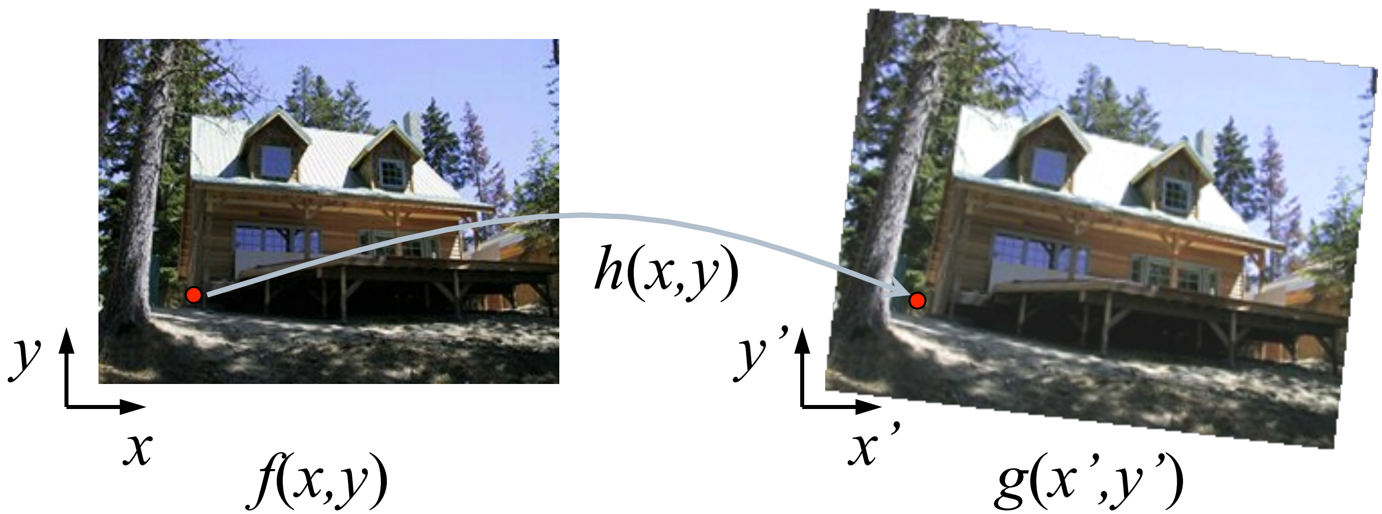
- Take good images
- Specify correspondences (manual)
- Compute homography
  - Solve with eigen decomposition
- Apply homography
  - Warping
  - Interpolation
  - Masking
  - Blending

Image Warping



- Given a coordinate transform $(x',y') = h(x,y)$ and a source image $f(x,y)$, how do we compute a transformed image $g(x',y') = f(h(x,y))$?

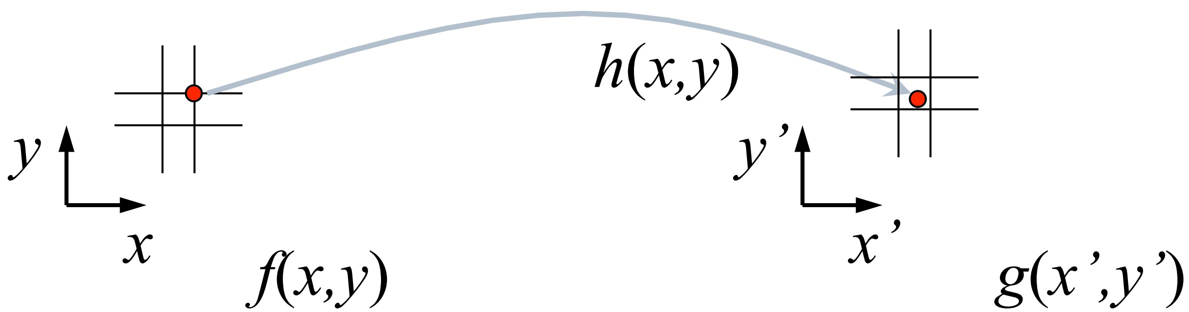
Forward Warping



- Send each pixel $f(x,y)$ to its corresponding location $(x',y') = h(x,y)$ in the second image

Q: what if pixel lands "between" two pixels?

Forward Warping



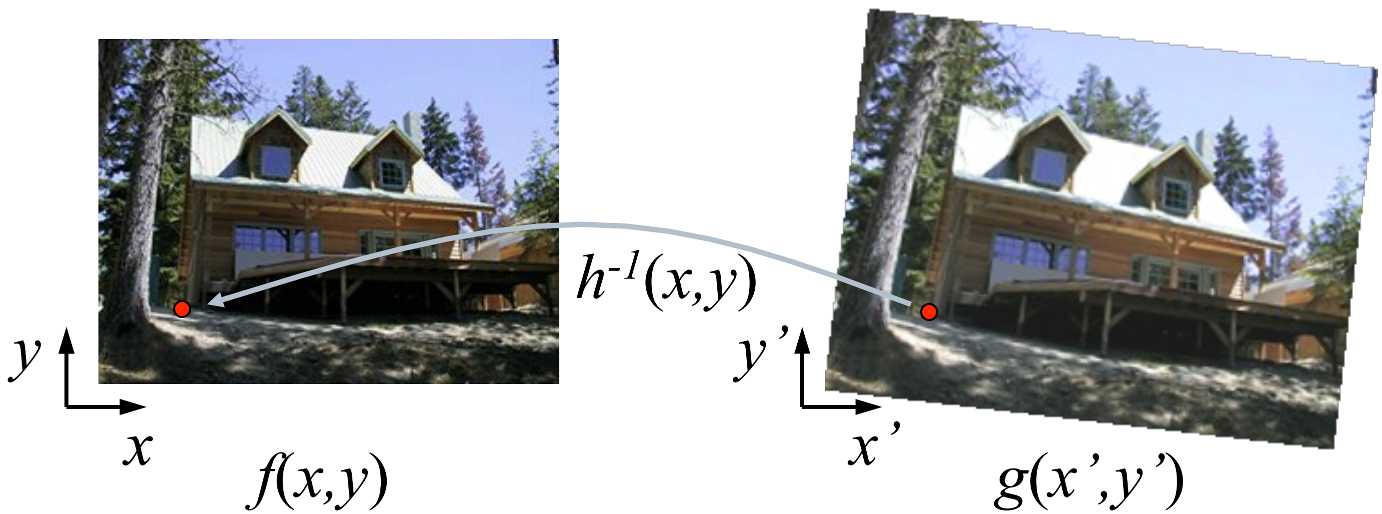
- Send each pixel $f(x,y)$ to its corresponding location $(x',y') = h(x,y)$ in the second image

Q: what if pixel lands "between" two pixels?

A: distribute color among neighboring pixels *(x',y')*

– Known as "splatting"

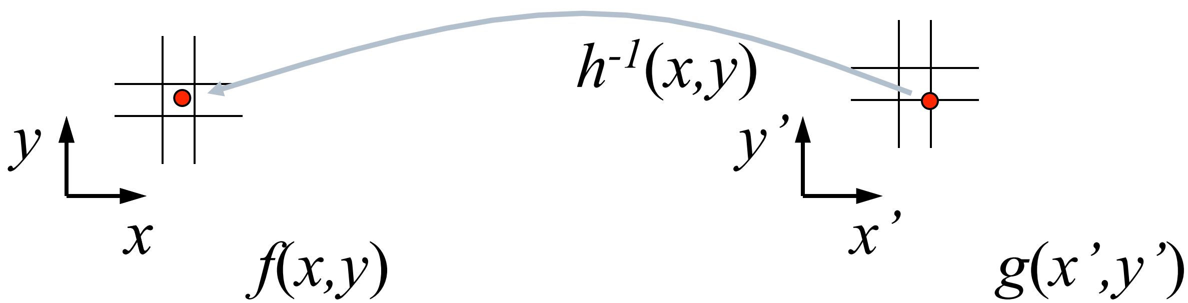
Inverse Warping



- Get each pixel $g(x',y')$ from its corresponding location
- $(x,y) = h^{-1}(x',y')$ in the first image

Q: what if pixel comes from "between" two pixels?

Inverse Warping



- Get each pixel $g(x',y')$ from its corresponding location $(x,y) = h^{-1}(x',y')$ in the first image

Q: what if pixel comes from "between" two pixels?

A: *resample* color value

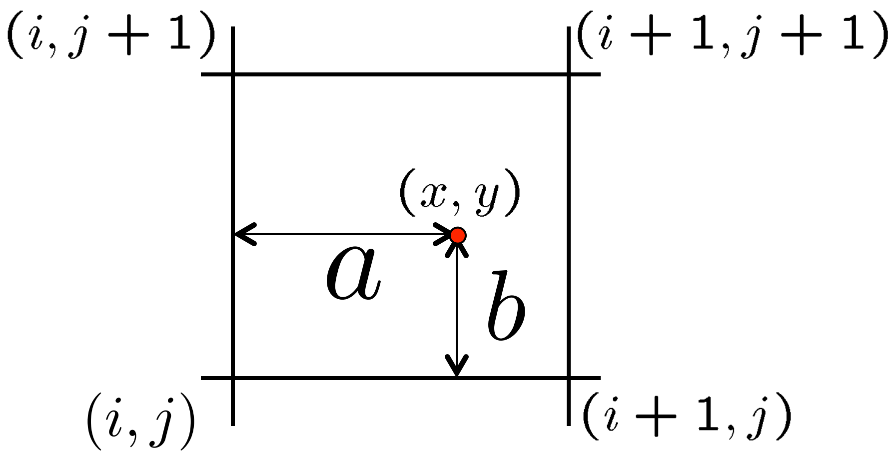
Forward vs. Inverse Warping

- Q: which is better?

- A: usually inverse—eliminates holes
  - however, it requires an invertible warp function—not always possible...

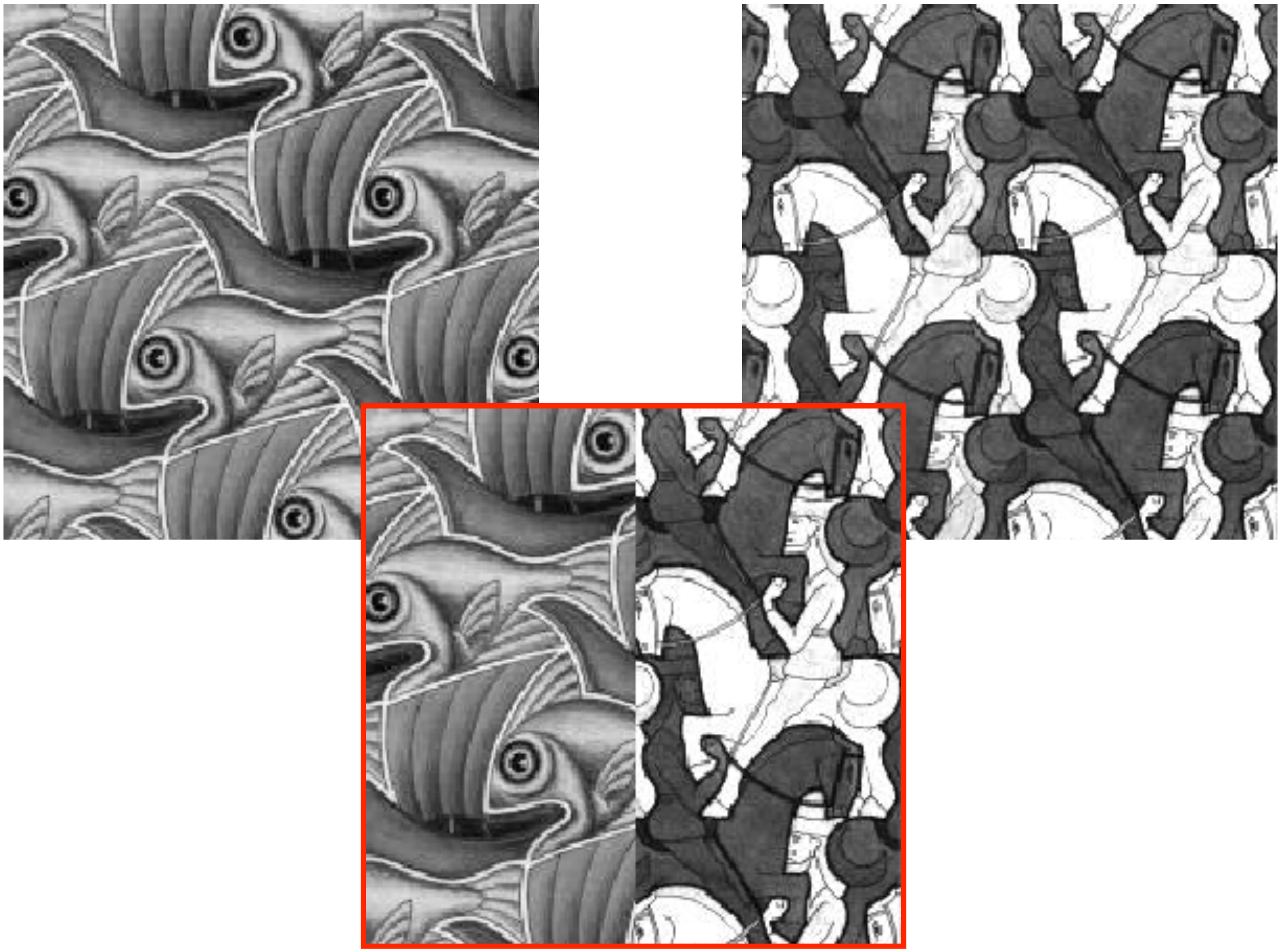
Bilinear Interpolation

- A simple method for resampling images

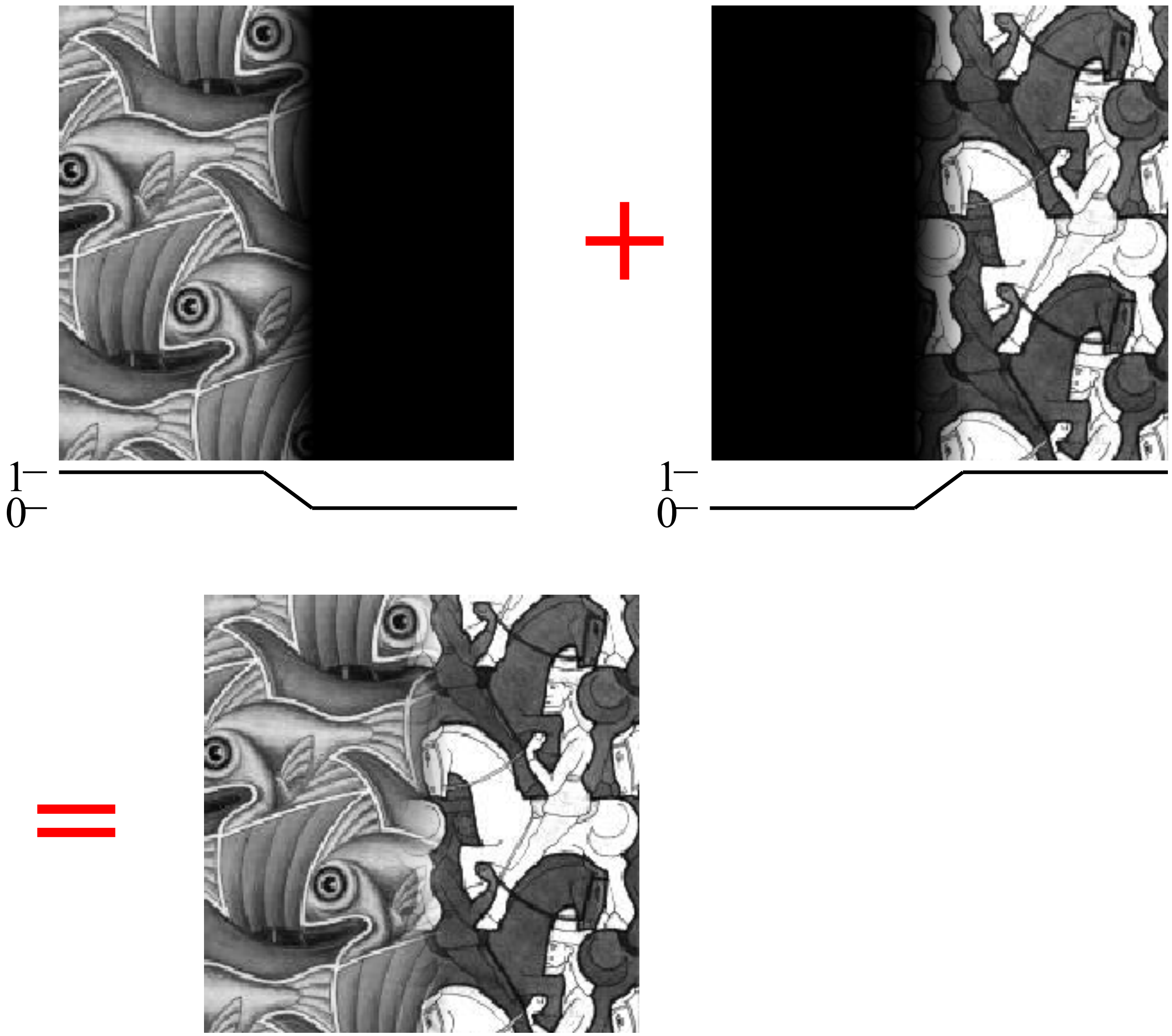


$$
\begin{aligned}
f(x, y) = \ & (1-a)(1-b) && f[i, j] \\
& +a(1-b) && f[i+1, j] \\
& +ab && f[i+1, j+1] \\
& +(1-a)b && f[i, j+1]
\end{aligned}
$$

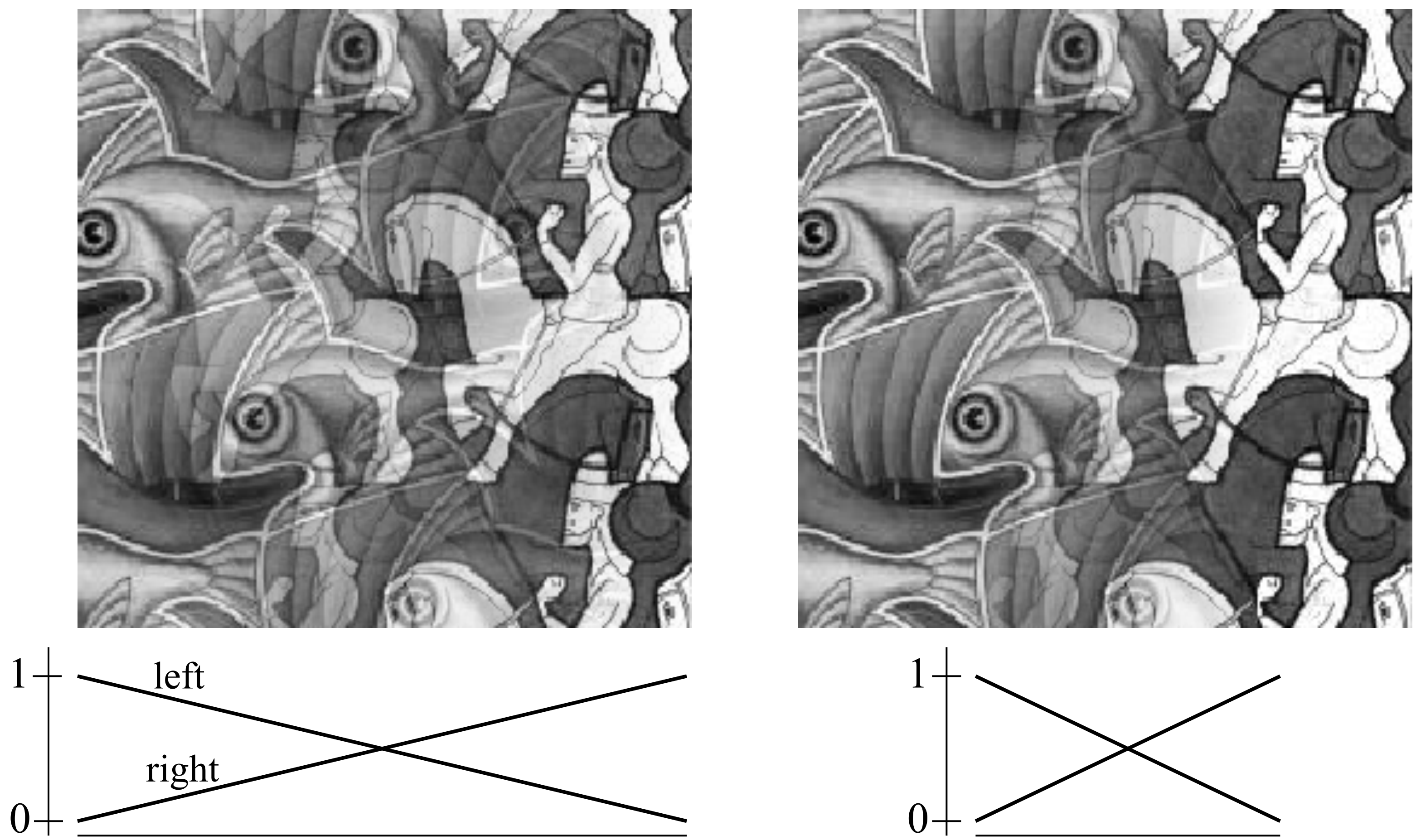
Image Blending



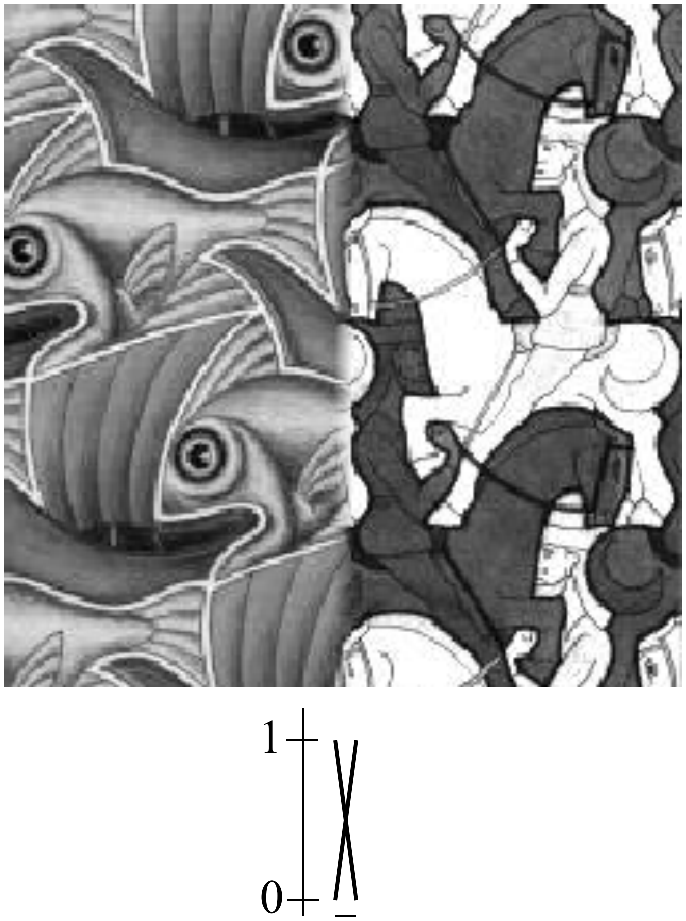
Feathering

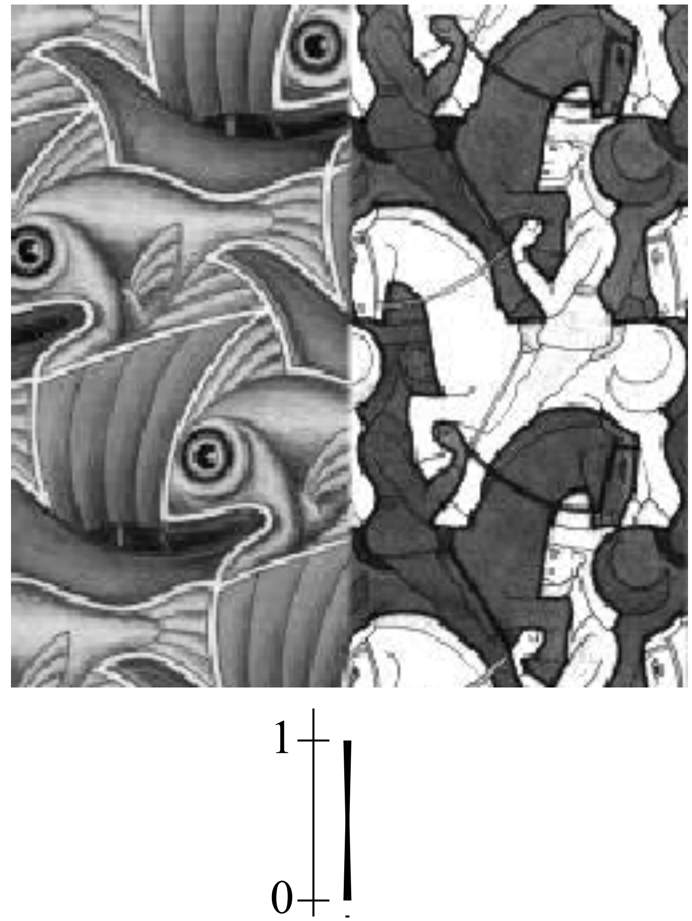


Effect of Window Size

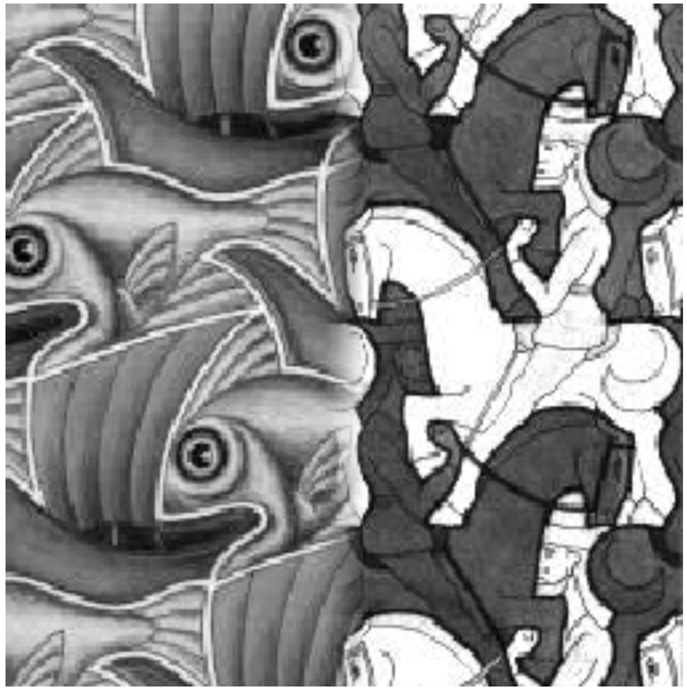


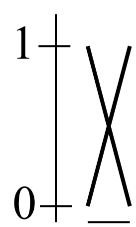
Effect of Window Size





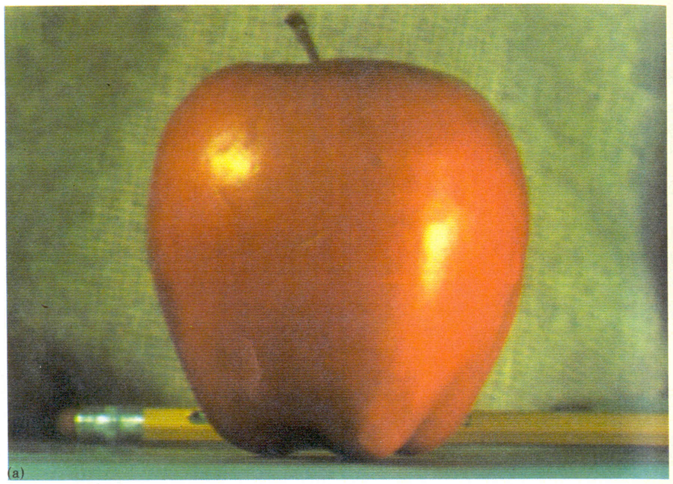
Good Window Size

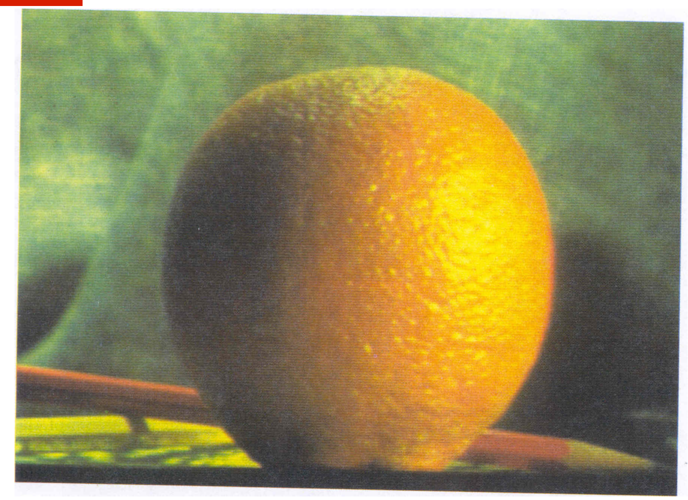


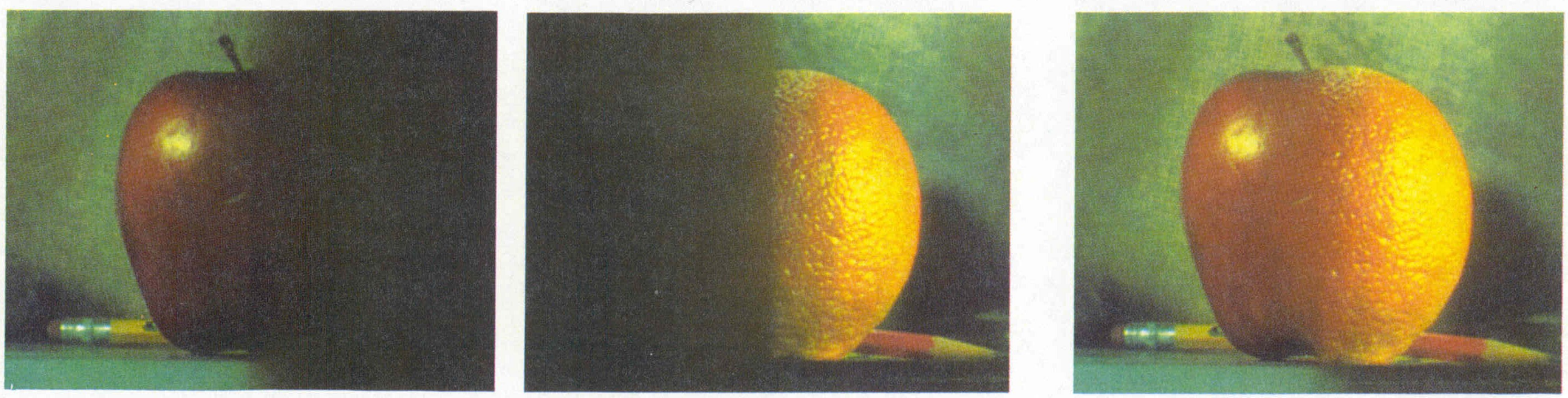


- "Optimal" window: smooth but not ghosted
  - Doesn't always work...

Pyramid Blending





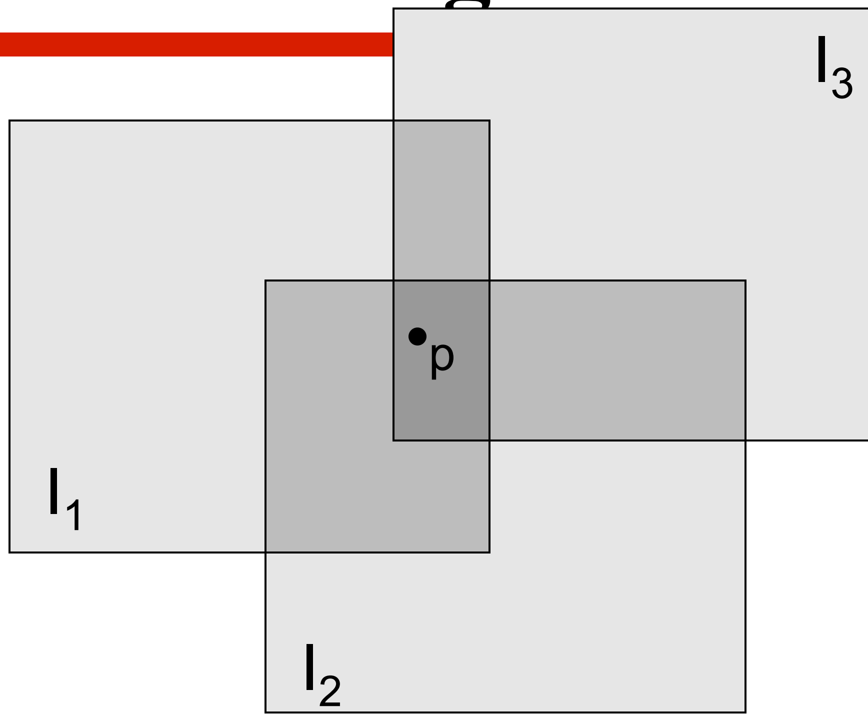


(d) (h) (l)

- Create a Laplacian pyramid, blend each level

Burt, P. J. and Adelson, E. H., [A multiresolution spline with applications to image mosaics](), ACM Transactions on Graphics, 42(4), October 1983, 217-236.

Alpha Blending



Encoding blend weights: $I(x,y) = (\alpha R,\ \alpha G,\ \alpha B,\ \alpha)$

color at $p = \dfrac{(\alpha_1 R_1,\ \alpha_1 G_1,\ \alpha_1 B_1) + (\alpha_2 R_2,\ \alpha_2 G_2,\ \alpha_2 B_2) + (\alpha_3 R_3,\ \alpha_3 G_3,\ \alpha_3 B_3)}{\alpha_1 + \alpha_2 + \alpha_3}$

Implement this in two steps:

1. accumulate: add up the (α premultiplied) RGBα values at each pixel
2. normalize: divide each pixel's accumulated RGB by its α value

   Q: what if $\alpha = 0$?

Poisson Image Editing

- For more info: Perez et al, SIGGRAPH 2003
  - http://research.microsoft.com/vision/cambridge/papers/perez_siggraph03.pdf

Project #1: Homography

- **Due 10/17 Sunday Midnight**
- See the assignment web page for details (3 artifacts for each task)
- Skelton Code on the web
  - Fill in the empty functions
  - Write additional functions for extra credits
- Cameras!

Image Filtering (Continuous)

Reading: Robot Vision Chapter 6

What is an Image?

Image as a Function

- We can think of an **image** as a function, f, from R^2 to R:
  - $f(x, y)$ gives the **intensity** at position (x, y)
  - Realistically, we expect the image only to be defined over a rectangle, with a finite range:
    - $f: [a,b] \times [c,d] \rightarrow [0,1]$

- A color image is just three functions pasted together. We can write this as a "vector-valued" function:

$$f(x,y) = \begin{bmatrix} r(x,y) \\ g(x,y) \\ b(x,y) \end{bmatrix}$$

Image as a Function

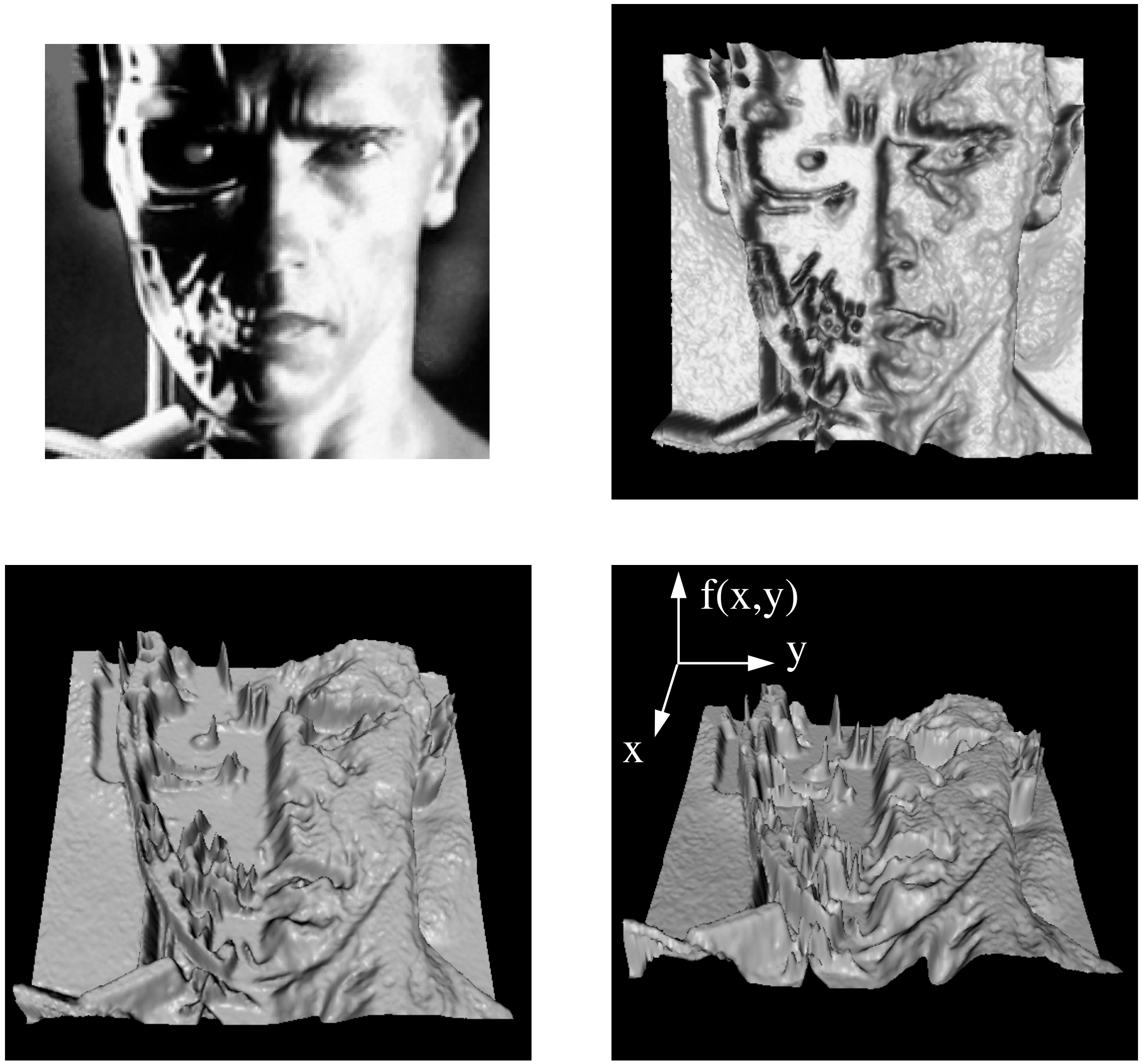


Image Processing

- Define a new image g in terms of an existing image f
  - We can transform either the domain or the range of f
- Range transformation:

$$g(x, y) = t(f(x, y))$$

What kinds of operations can this perform?

Image Processing

- Some operations preserve the range but change the domain of f:

$$g(x, y) = f(t_x(x, y), t_y(x, y))$$

What kinds of operations can this perform?

Image Processing

- Still other operations operate on both the domain *and* the range of f.

Linear Shift Invariant Systems

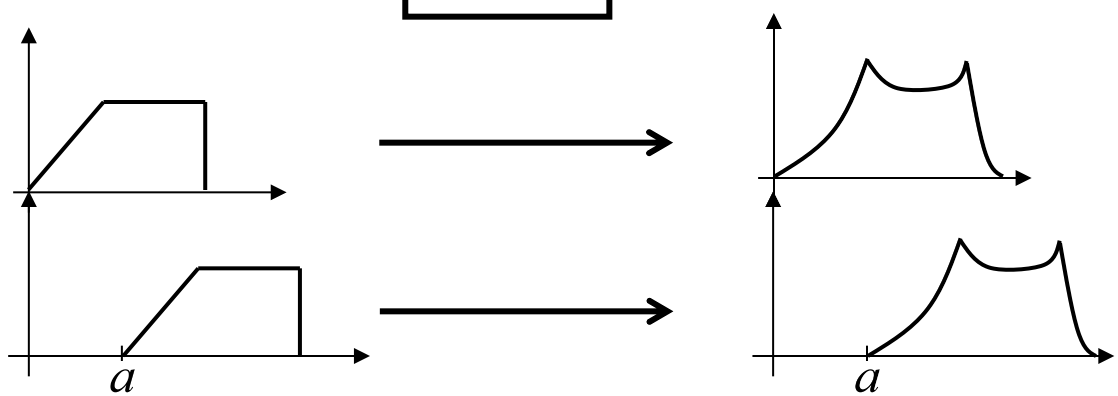
Linearity:

$$f_1 \longrightarrow \boxed{} \longrightarrow g_1 \qquad f_2 \longrightarrow \boxed{} \longrightarrow g_2$$

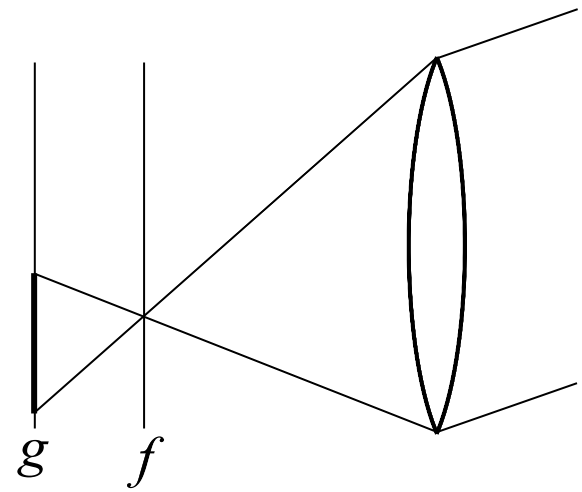
$$\alpha f_1 + \beta f_2 \longrightarrow \boxed{} \longrightarrow \alpha g_1 + \beta g_2$$

Shift invariance:

$$f(x-a) \longrightarrow \boxed{} \longrightarrow g(x-a)$$



Example of LSIS



Defocused image (g) is a processed version of the focused image (f)

Ideal lens is a LSIS

$$f(x) \longrightarrow \boxed{\text{LSIS}} \longrightarrow g(x)$$

Linearity: Brightness variation

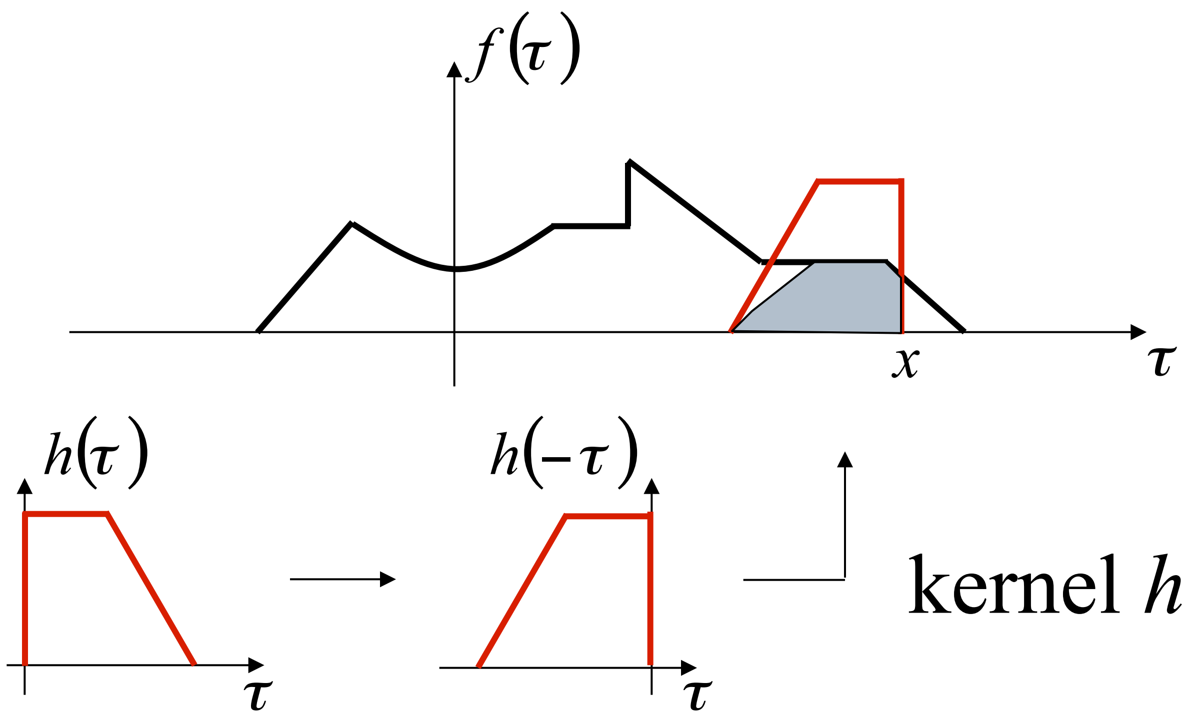
Shift invariance: Scene movement

(not valid for lenses with non-linear distortions)

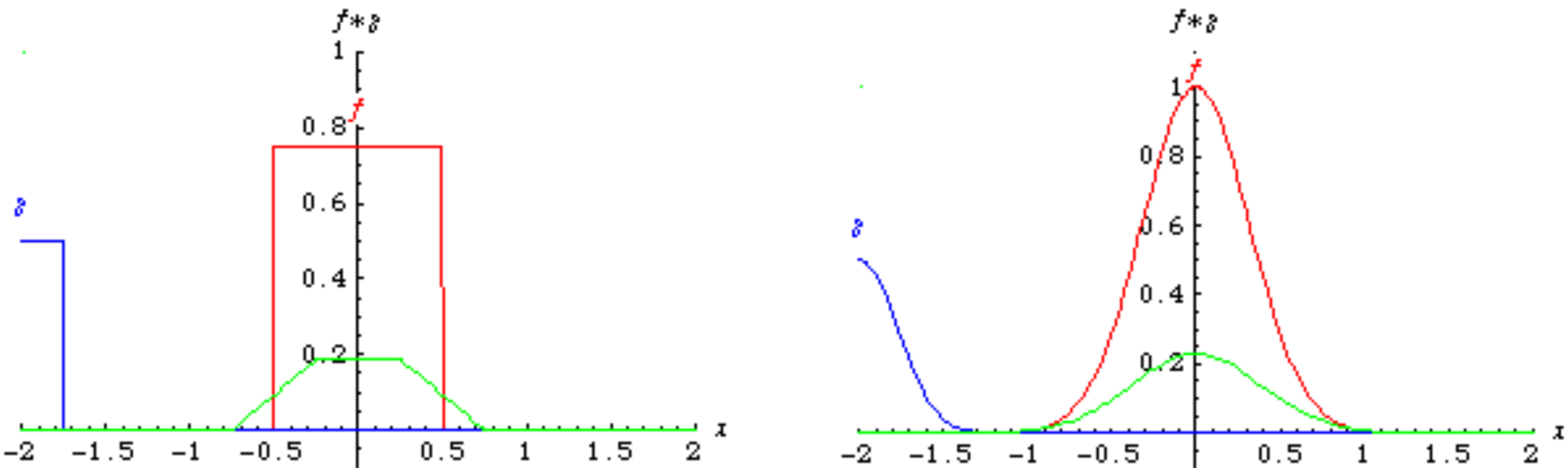
Convolution

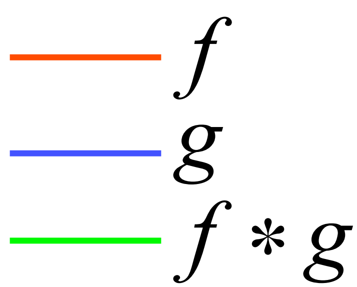
LSIS is doing convolution; convolution is linear and shift invariant

$$g(x) = \int_{-\infty}^{\infty} f(\tau)h(x-\tau)d\tau \qquad g = f * h$$



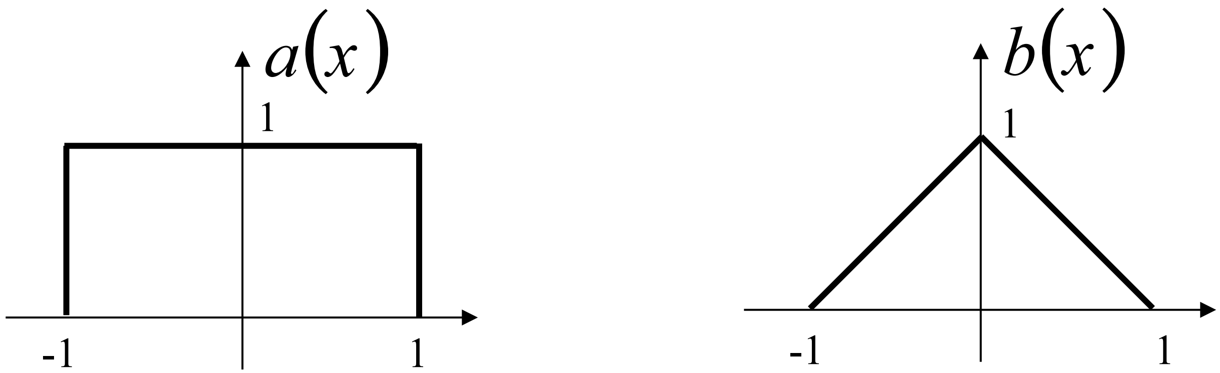
Convolution



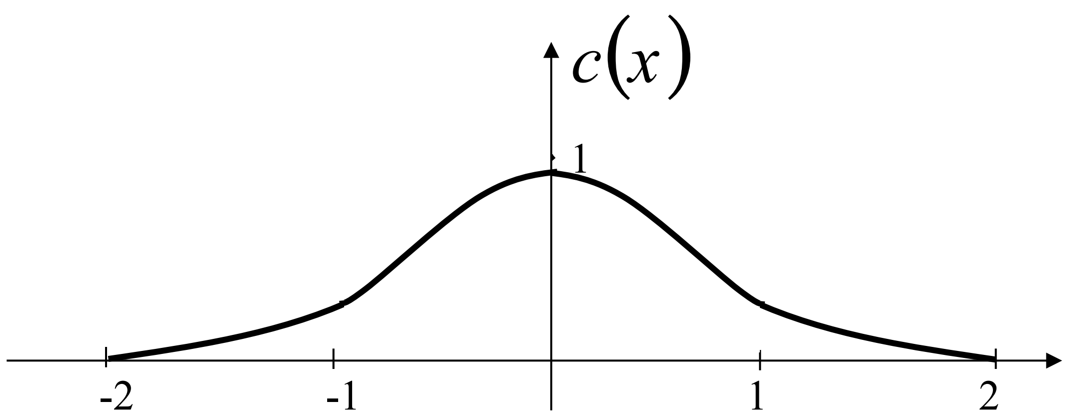


Eric Weisstein's Math World

Example of Convolution



$$c = a * b$$



Convolution Kernel

$$f \longrightarrow \boxed{h} \longrightarrow g \qquad\qquad g = f * h$$

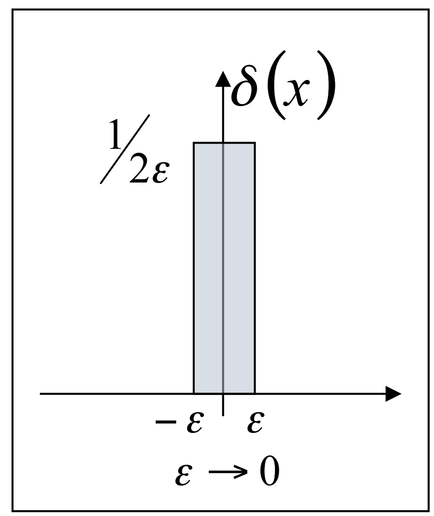
- What h will give us $g = f$?

Dirac Delta Function (Unit Impulse Function)

Sifting property:

$$\int_{-\infty}^{\infty} f(x)\delta(x)dx = \int_{-\infty}^{\infty} f(0)\delta(x)dx$$

$$= f(0)\int_{-\infty}^{\infty} \delta(x)dx = f(0)$$



$$g(x) = \int_{-\infty}^{\infty} f(\tau)\delta(x-\tau)d\tau = f(x)$$

$$= \int_{-\infty}^{\infty} \delta(\tau)h(x-\tau)d\tau = h(x)$$

Point Spread Function

scene $\rightarrow$ Optical System $\rightarrow$ image

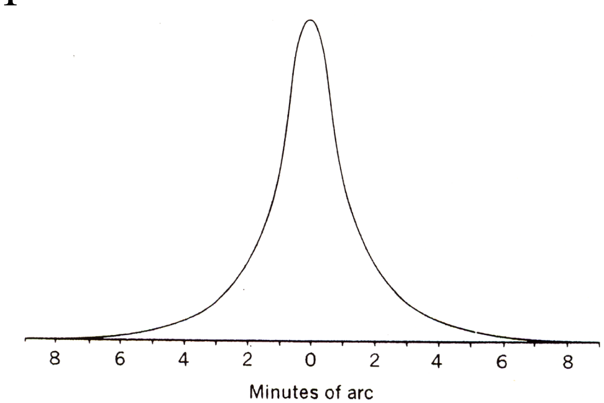
Ideally, the optical system is a dirac delta function.

However, optical systems are never ideal.

$\delta(x)$ $\rightarrow$ Optical System $\rightarrow$ $PSF(x)$

point source — point spread func.

Human eyes' point spread function



Point Spread Function

normal vision

myopia

hyperopia

astigmatism

Images by Richmond Eye Associates

Properties of Convolution

- Commutative

$$a * b = b * a$$

- Associative

$$(a * b) * c = a * (b * c)$$

Cascade system

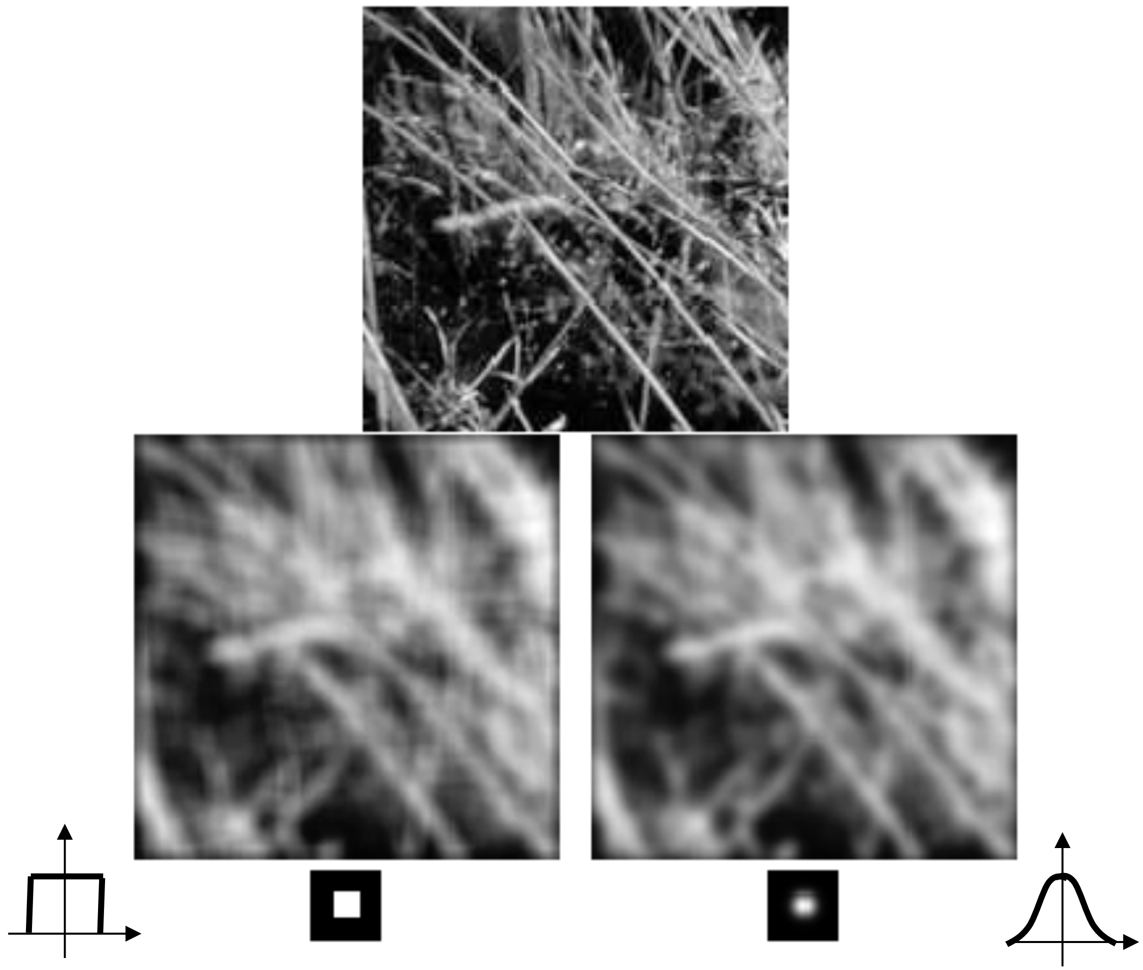
$$f \longrightarrow \boxed{h_1} \longrightarrow \boxed{h_2} \longrightarrow g$$

$$= f \longrightarrow \boxed{h_1 * h_2} \longrightarrow g$$

$$= f \longrightarrow \boxed{h_2 * h_1} \longrightarrow g$$

2D Convolution

$$g(x) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f\left(\tau_x, \tau_y\right) h\left(x - \tau_x, y - \tau_y\right) d\tau_x \, d\tau_y$$



Jean Baptiste Joseph Fourier (1768-1830)

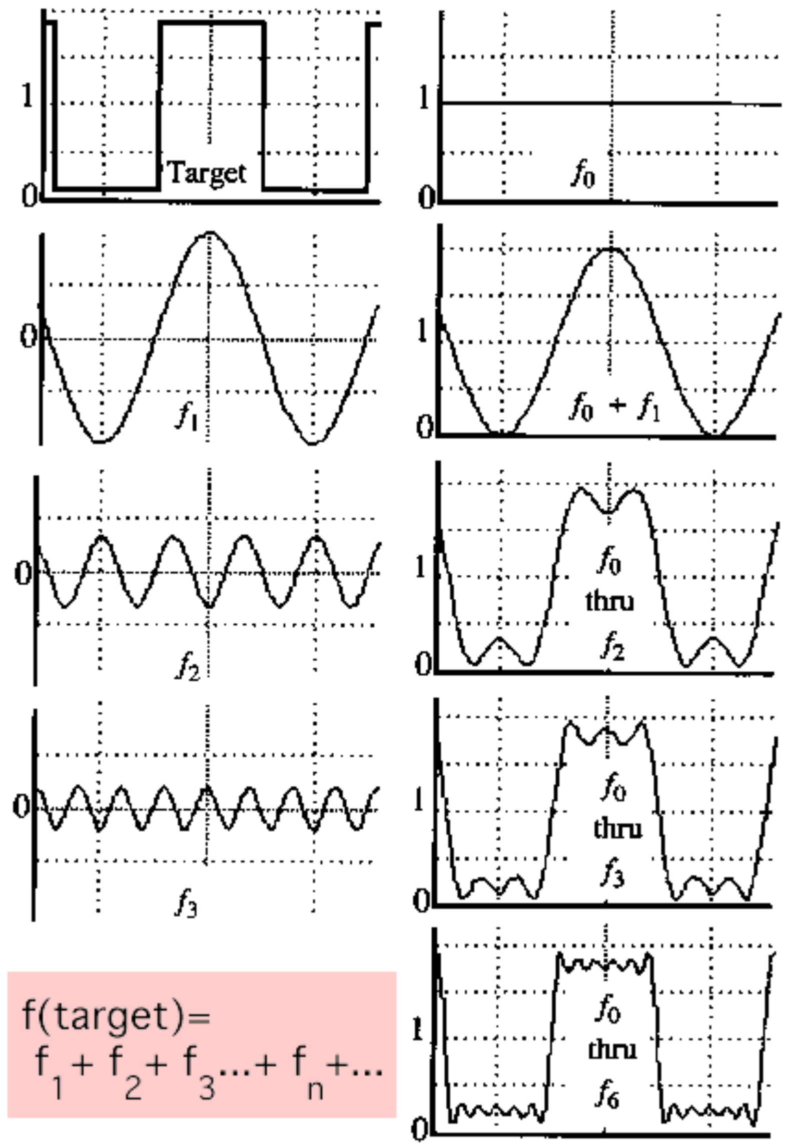
- Had crazy idea (1807):
  "**Any** periodic function can be rewritten as a weighted sum of Sines and Cosines of different frequencies. "
- Don't believe it?
  - Neither did Lagrange, Laplace, Poisson and other big wigs
  - Not translated into English until 1878!
- But it's true!
  - called Fourier Series
  - Possibly the greatest tool used in Engineering

A Sum of Sinusoids

- Our building block:

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

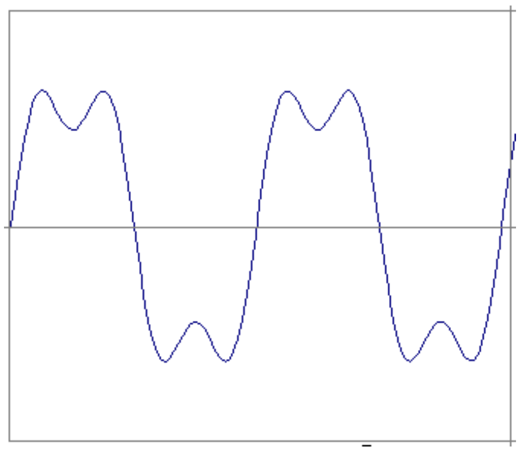
- Add enough of them to get any signal *f(x)* you want!



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

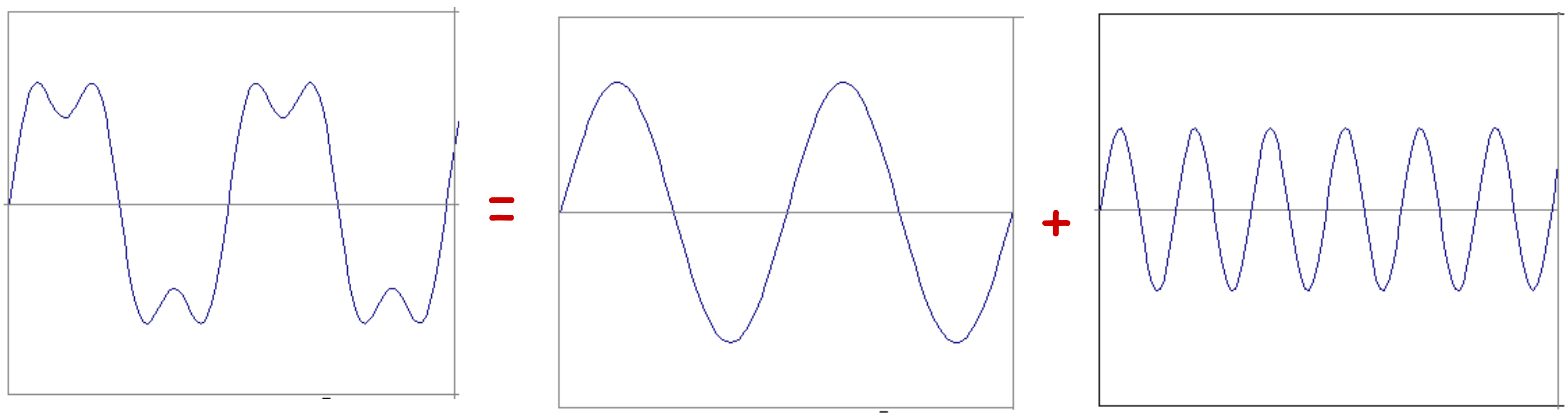
Time and Frequency

$$g(t) = \sin(2\pi f t) + \frac{1}{3}\sin(2\pi(3f)t)$$



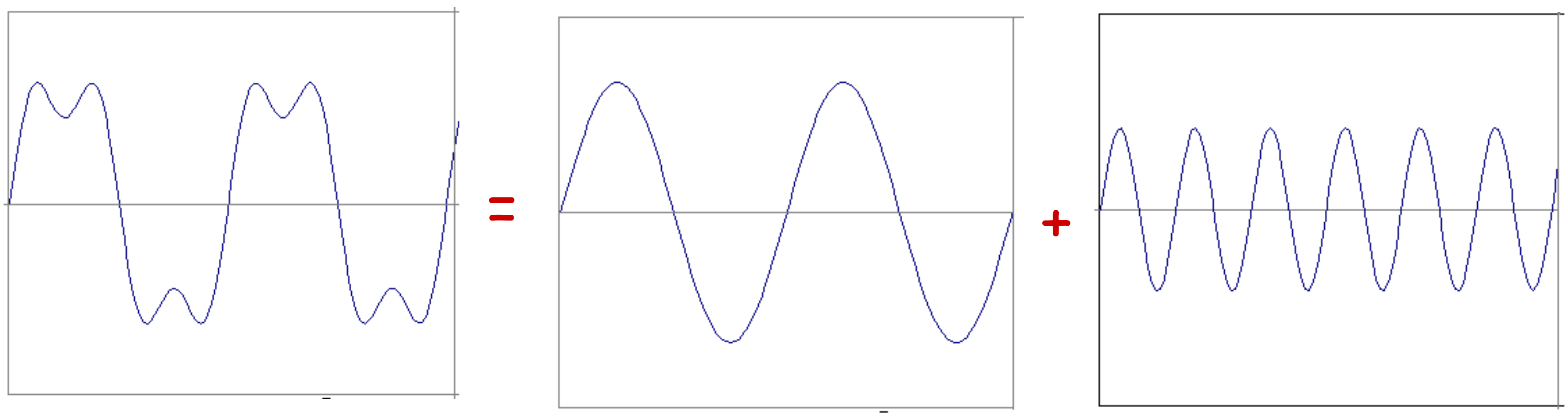
Time and Frequency

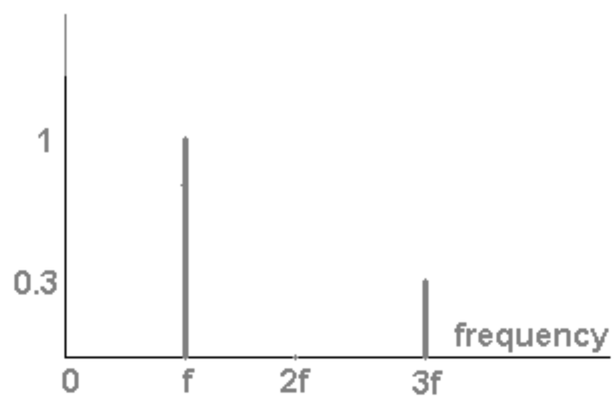
$$g(t) = \sin(2\pi f t) + \frac{1}{3}\sin(2\pi(3f)t)$$



Frequency Spectra

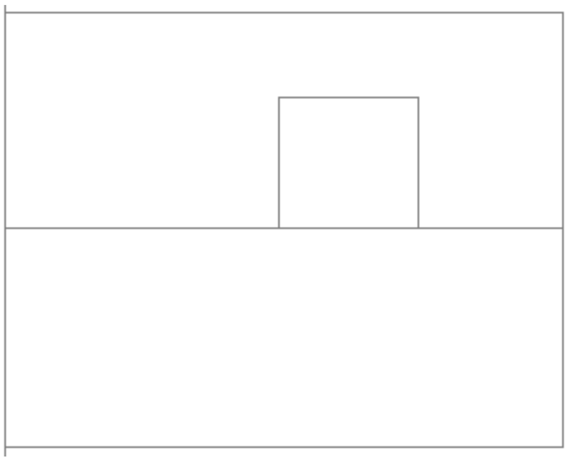
$$g(t) = \sin(2\pi f t) + \frac{1}{3}\sin(2\pi(3f)t)$$



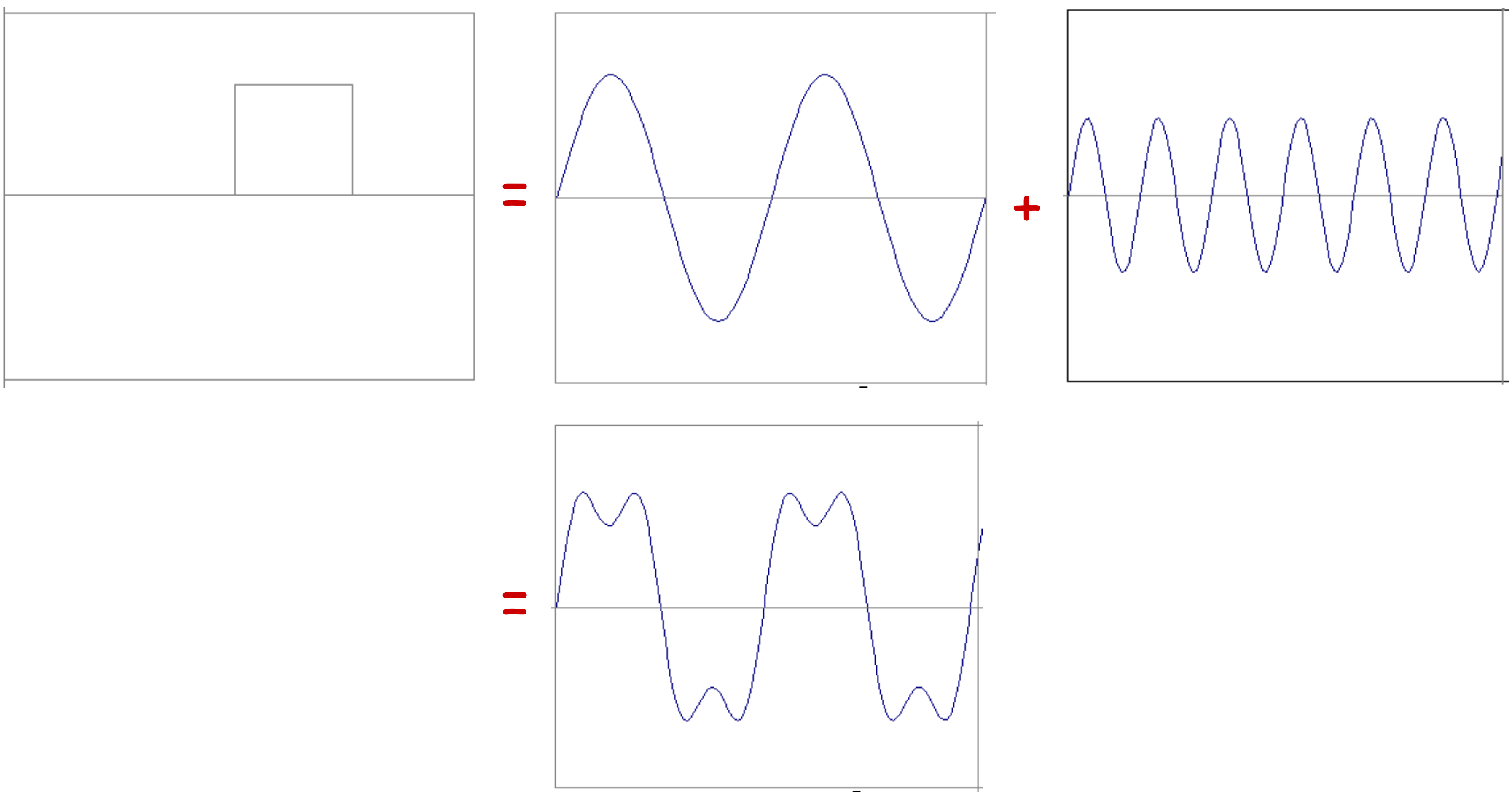


Frequency Spectra

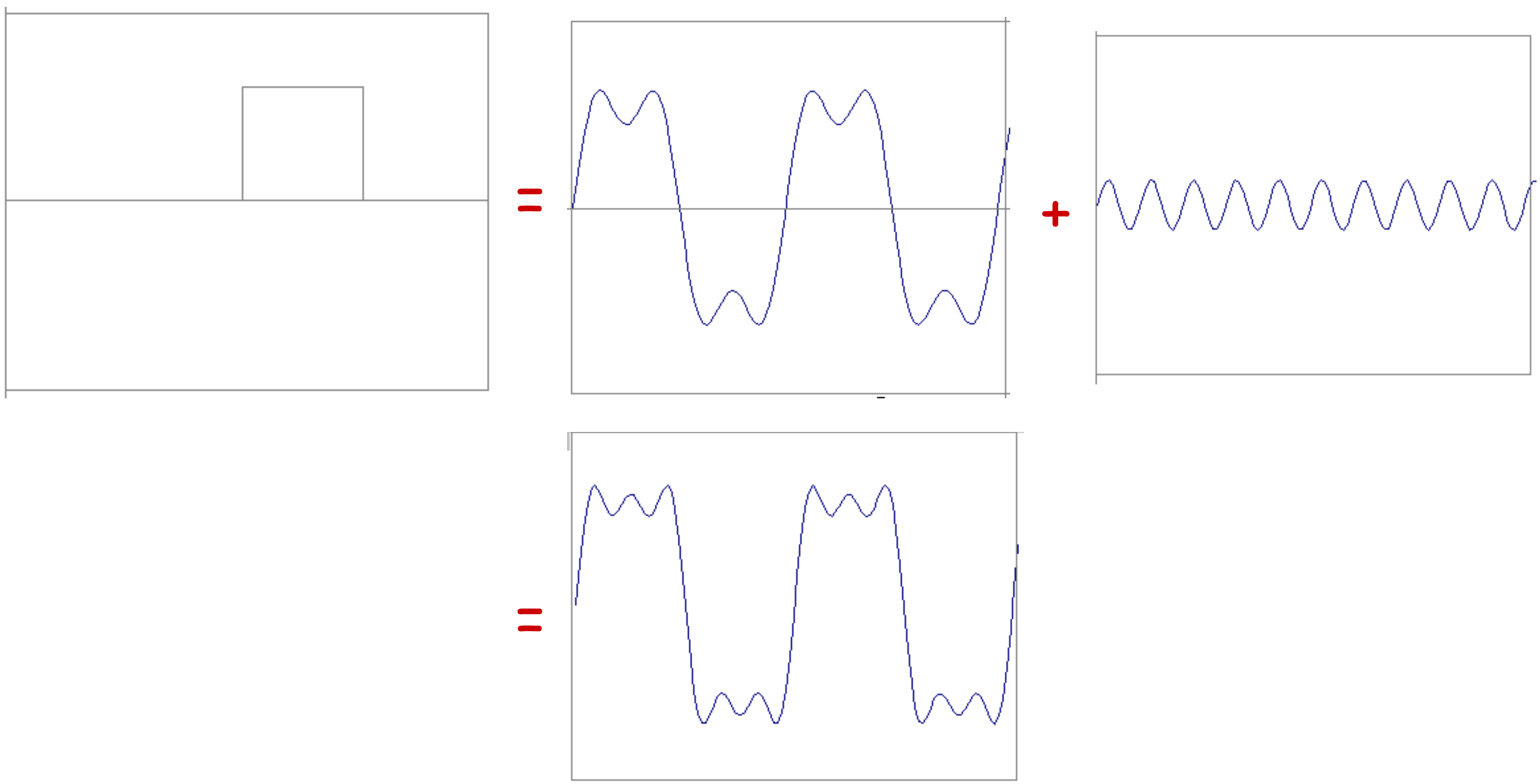
- Usually, frequency is more interesting than the phase



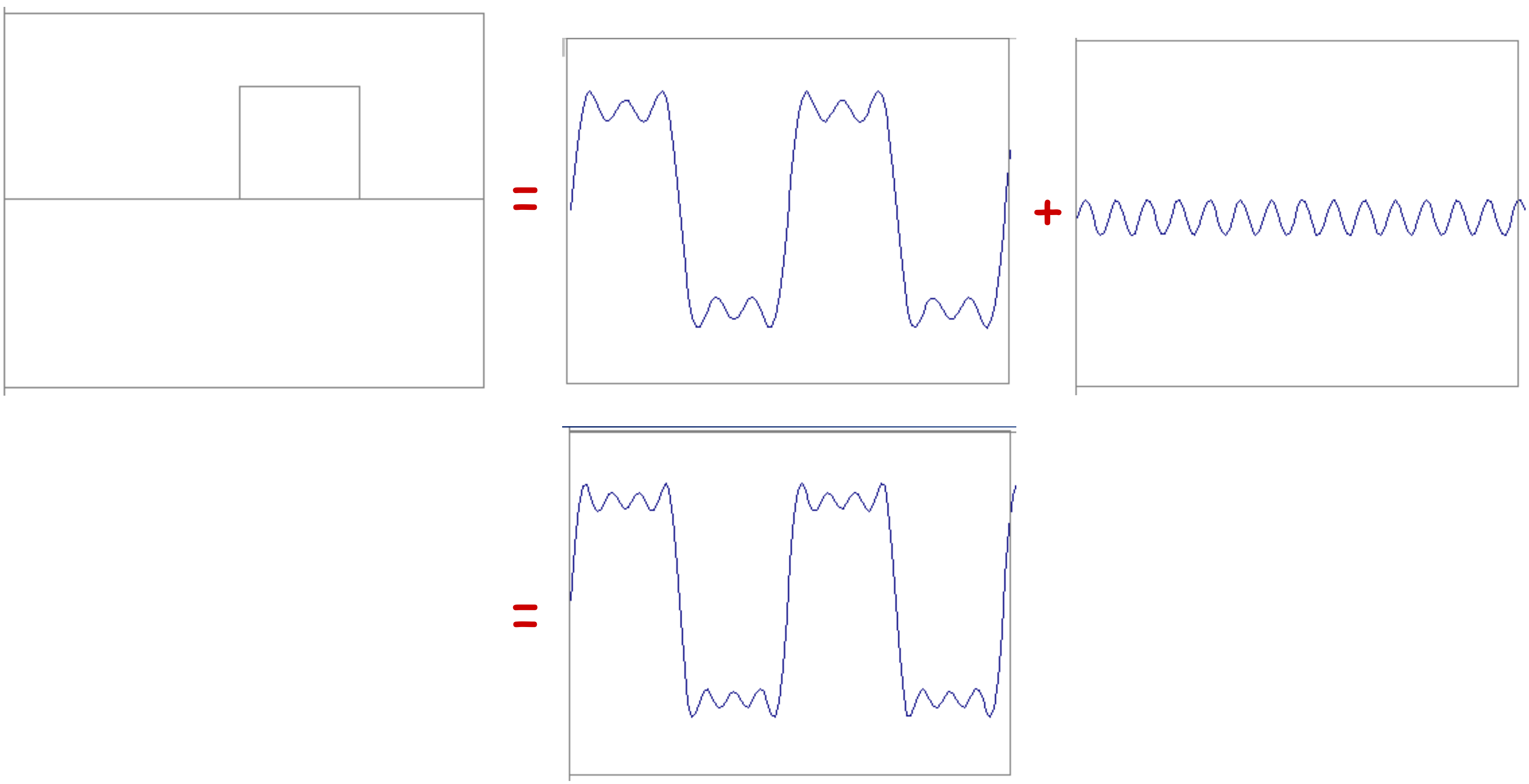
Frequency Spectra



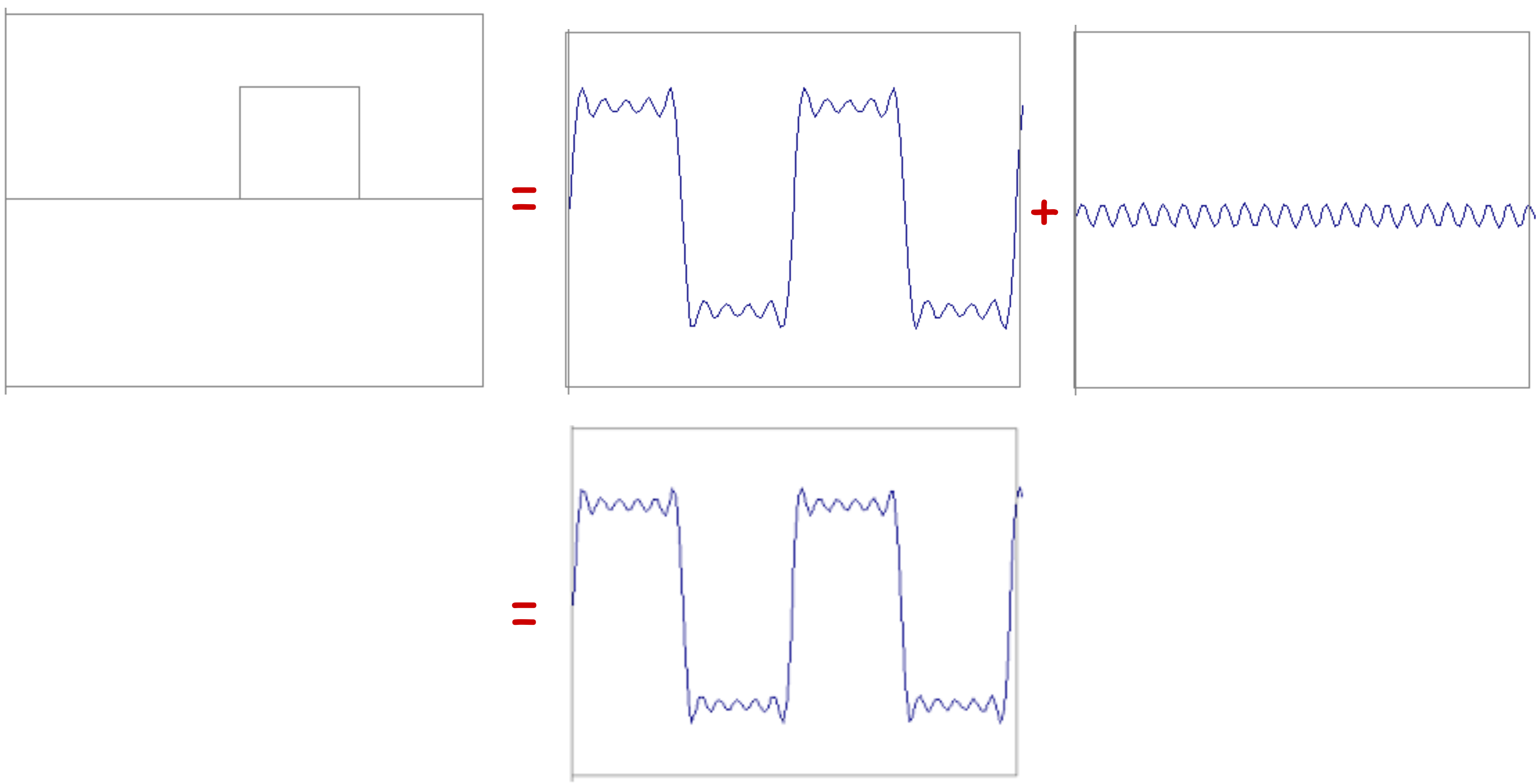
Frequency Spectra



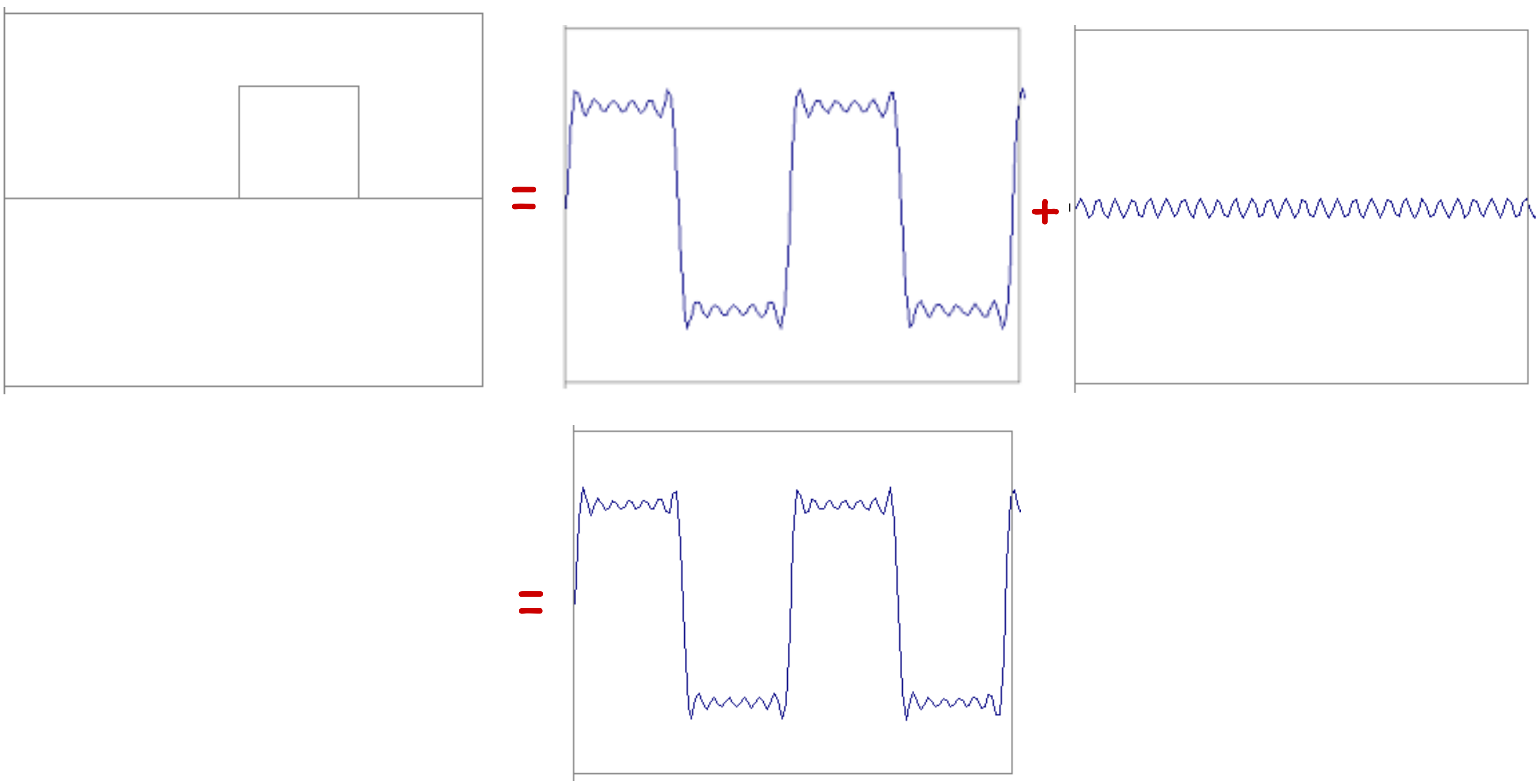
Frequency Spectra



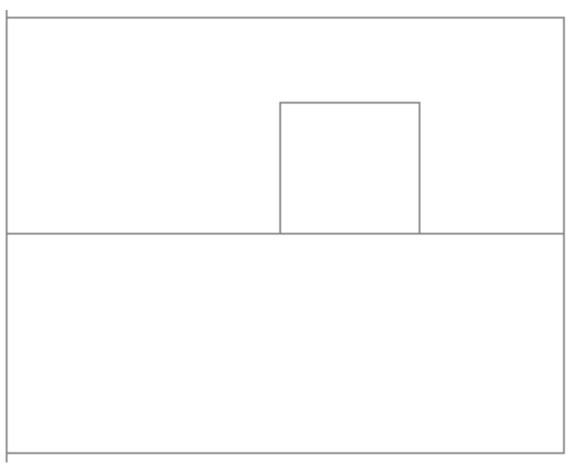
Frequency Spectra



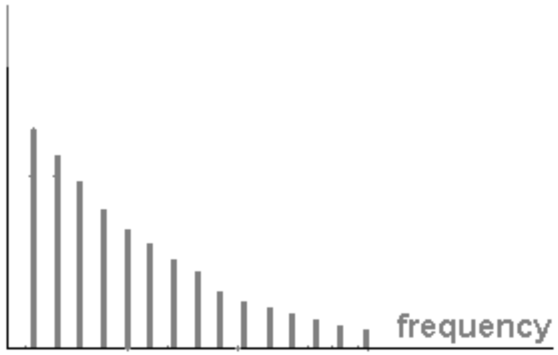
Frequency Spectra



Frequency Spectra



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



Fourier Transform

Represent the signal as an infinite weighted sum
of an infinite number of sinusoids (u: oscillation frequency)

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

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

$P\cos(x) + Q\sin(x) = A\sin(x+\phi)$

phase is encoded by sin/cos pair $\longrightarrow$ $A = \pm\sqrt{P^2 + Q^2} \qquad \phi = \tan^{-1}\left(\frac{P}{Q}\right)$

Arbitrary function $\longrightarrow$ Single Analytic Expression

Spatial Domain (x) $\longrightarrow$ Frequency Domain (u)

(Frequency Spectrum $F(u)$)

<u>Inverse Fourier Transform (IFT)</u>

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

Fourier Transform (Physicists' Definition)

Represent the signal as an infinite weighted sum
of an infinite number of sinusoids (u: angular frequency)

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

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

Arbitrary function $\longrightarrow$ Single Analytic Expression

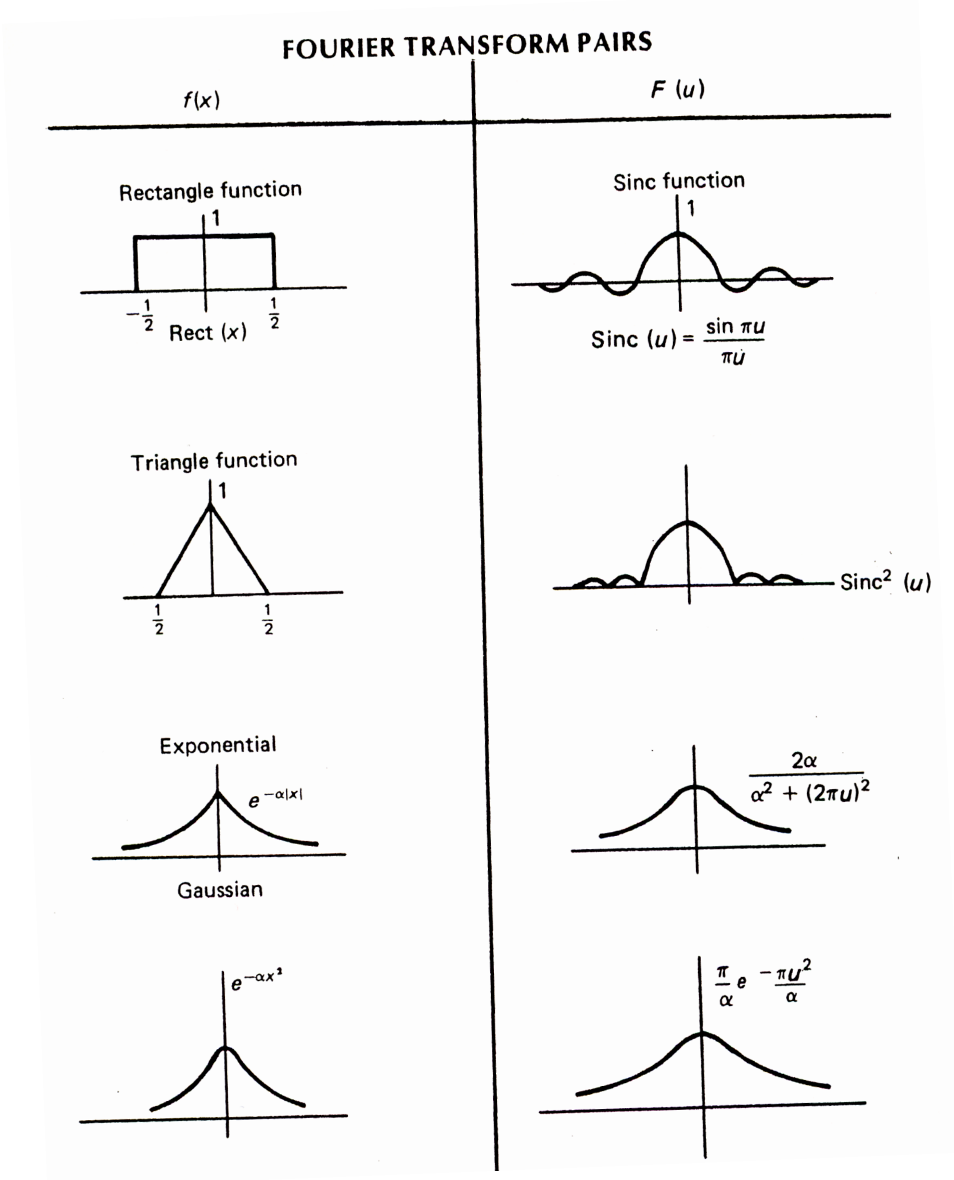
Spatial Domain (x) $\longrightarrow$ Frequency Domain (u)

(Frequency Spectrum $F(u)$)

<u>Inverse Fourier Transform (IFT)</u>

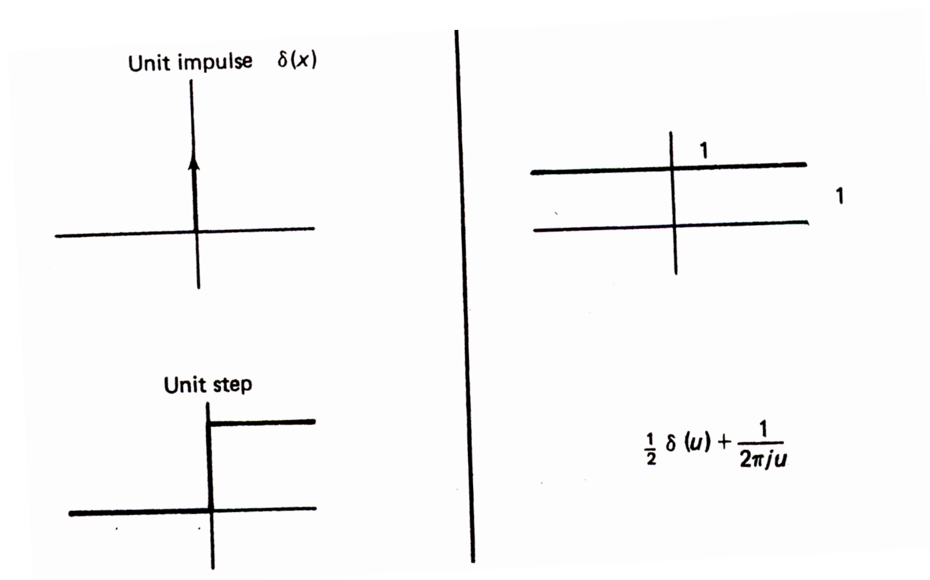
$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(u) e^{iux} dx$$

Fourier Transform Pairs (I)



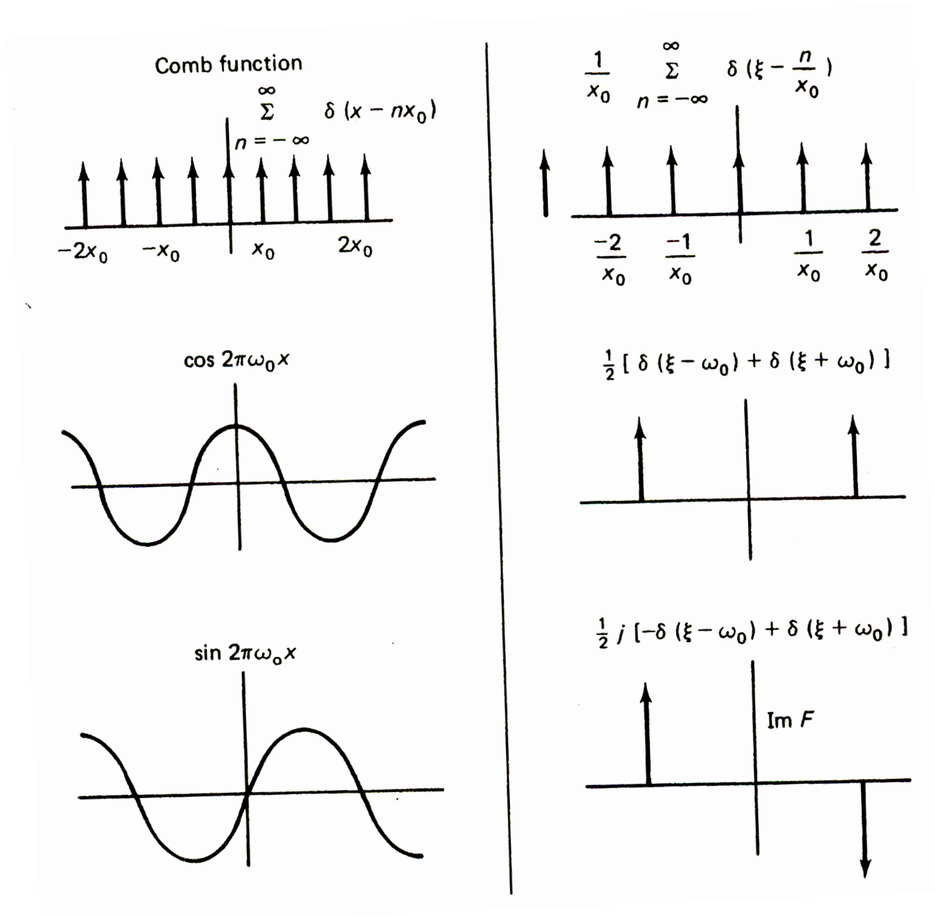
Play with

http://www.falstad.com/fourier/



Note that these are derived using angular frequency ($e^{-i\omega x}$)

Fourier Transform Pairs (II)



Note that these are derived using angular frequency ($e^{-i\omega x}$)

Fourier Transform and Convolution

Let $\quad g = f * h$

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

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

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

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

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

Convolution in spatial domain

$\Leftrightarrow$ Multiplication in frequency domain

Fourier Transform and Convolution

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

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

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