

The Frequency Domain, without tears



Somewhere in Cinque Terre, May 2005

CS194: Image Manipulation, Comp. Vision, and Comp. Photo

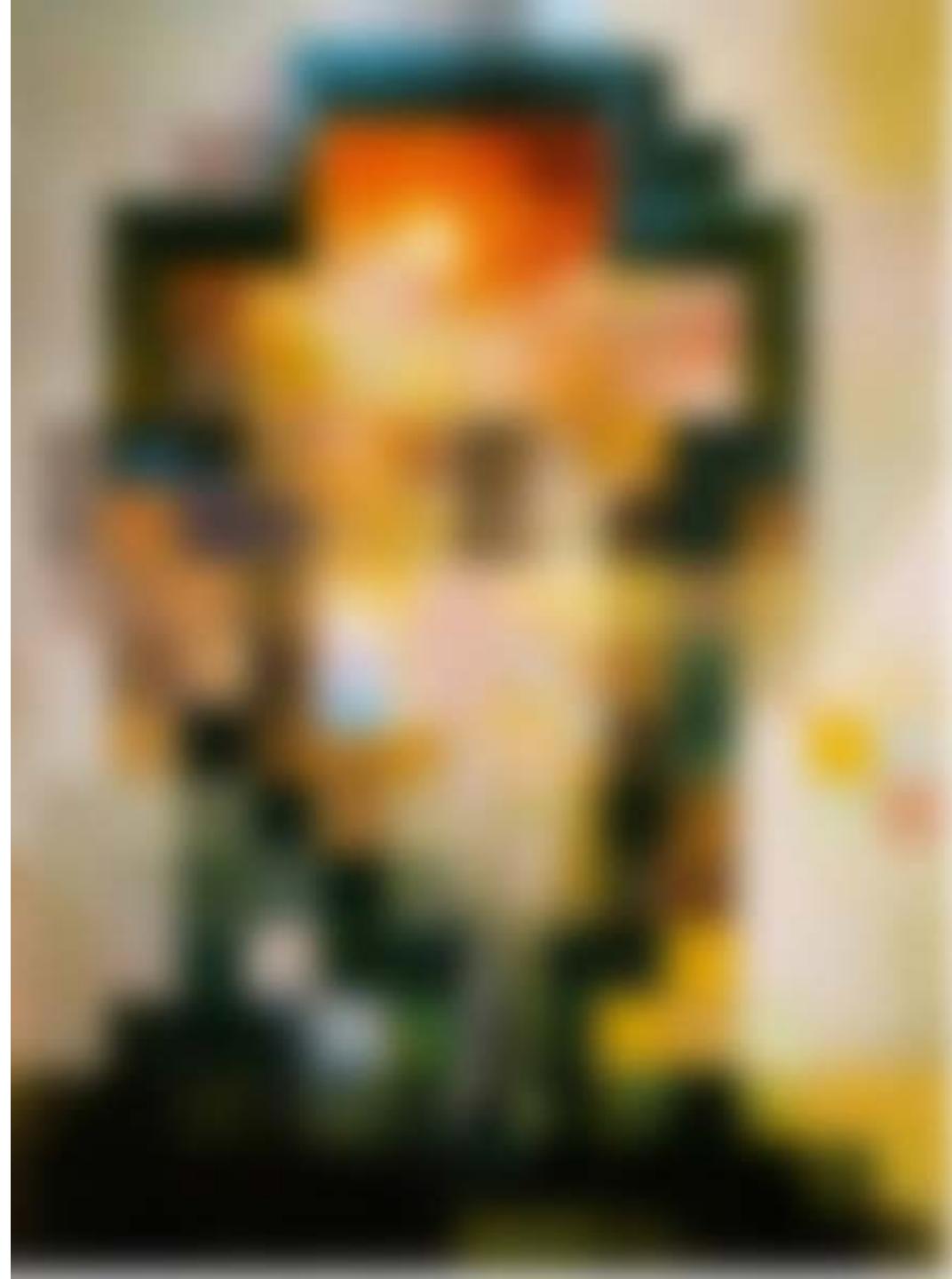
Many slides borrowed
from Steve Seitz

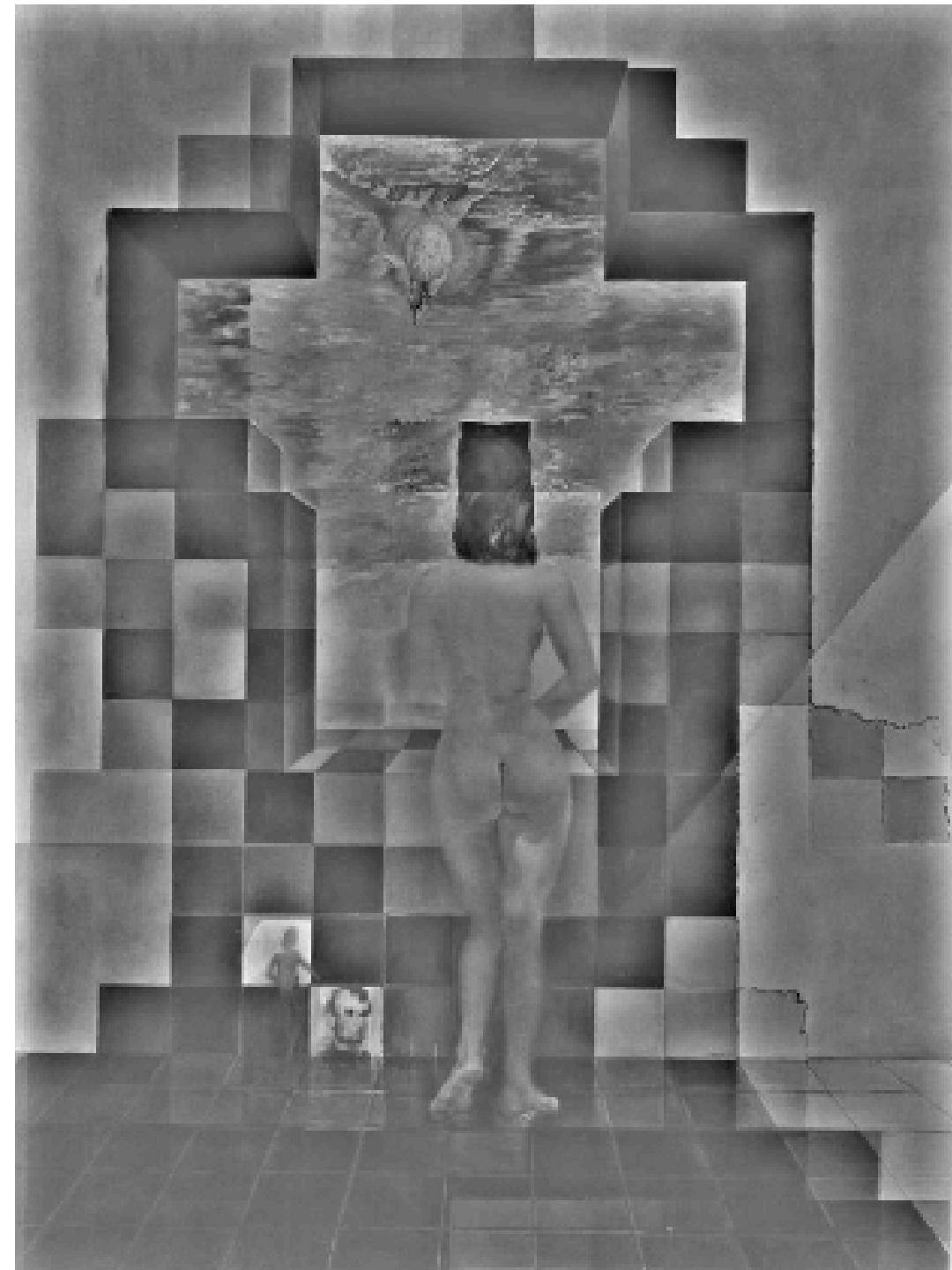
Alexei Efros, UC Berkeley, Spring 2020

Salvador Dali

*“Gala Contemplating the Mediterranean Sea,
which at 30 meters becomes the portrait
of Abraham Lincoln”, 1976*

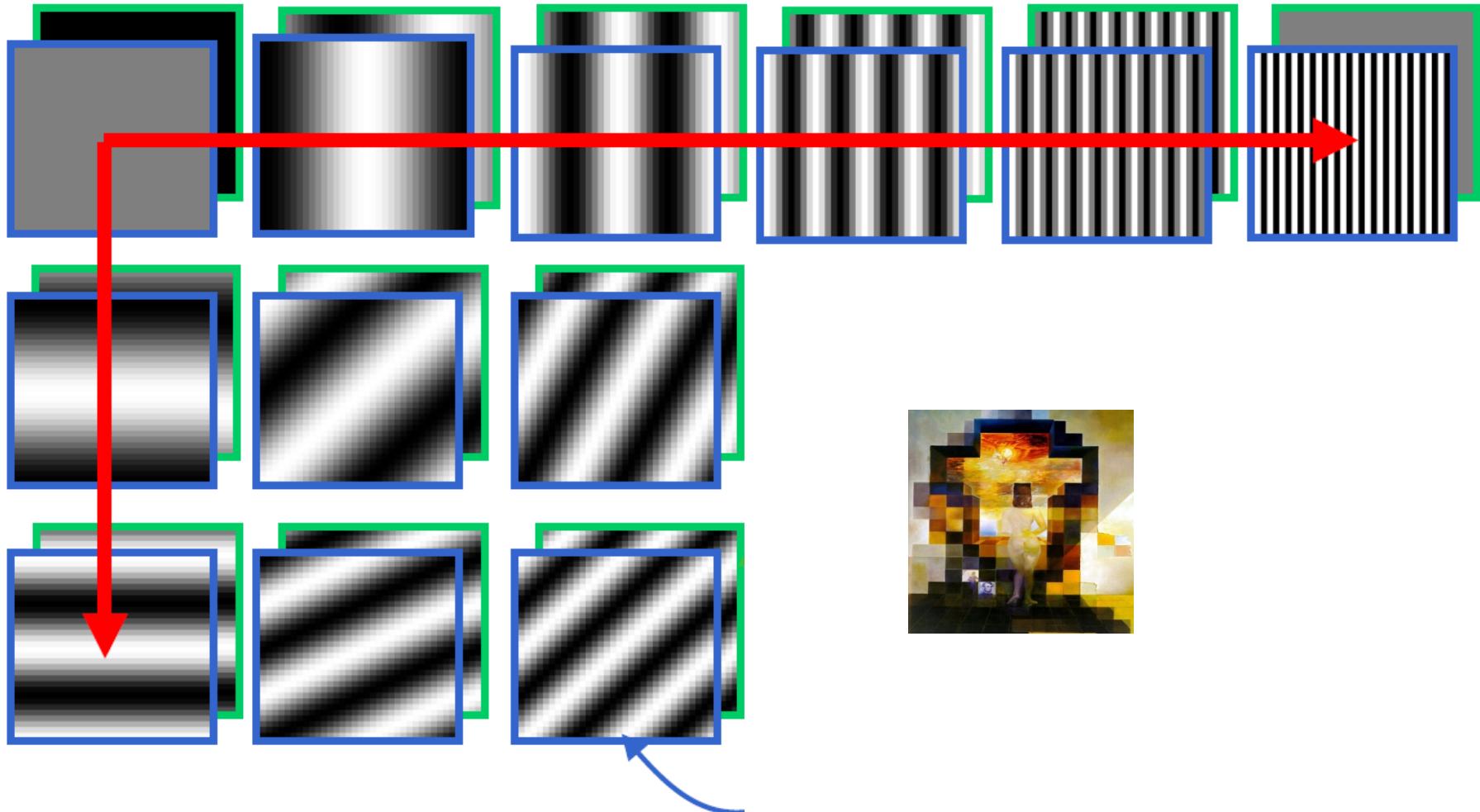






A nice set of basis

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 univariate function can be rewritten as a weighted sum of sines and cosines of different frequencies.

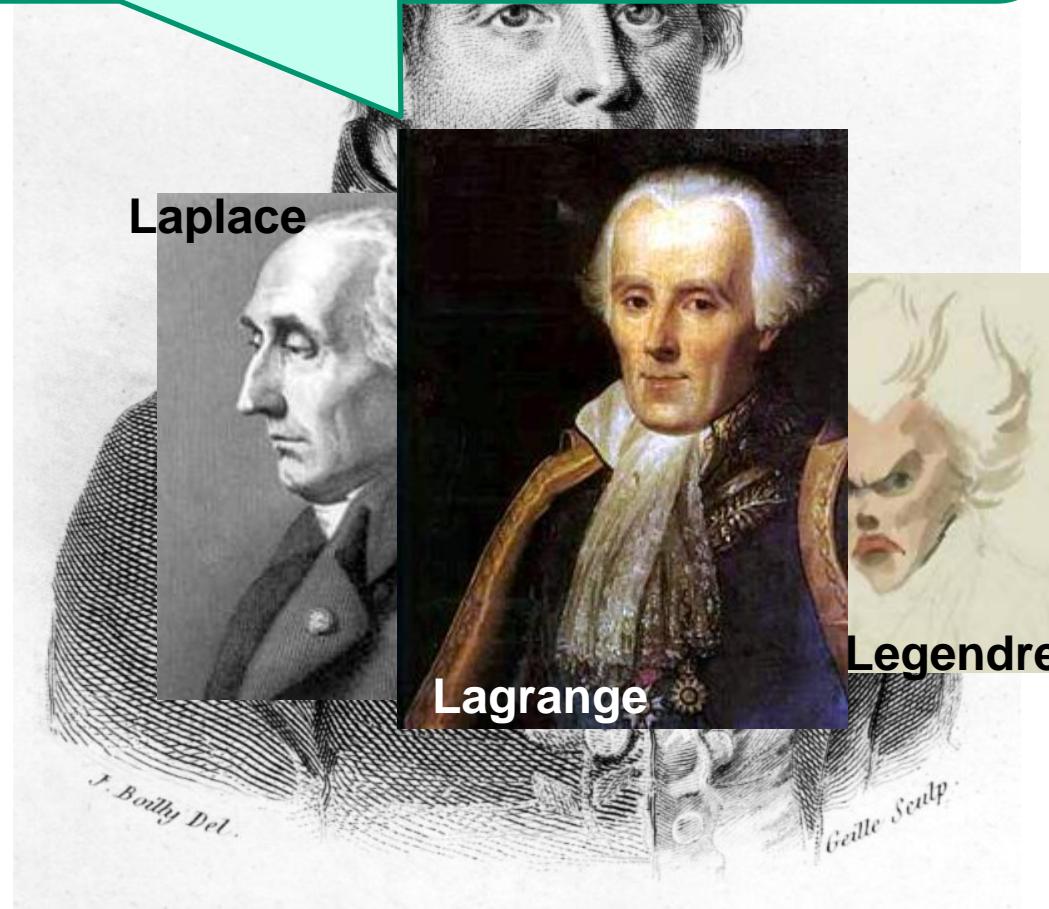
...the manner in which the author arrives at these equations is not exempt of difficulties and...his analysis to integrate them still leaves something to be desired on the score of generality and even rigour.

Don't believe it?

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

But it's (mostly) true!

- called Fourier Series



A sum of sines

Our building block:

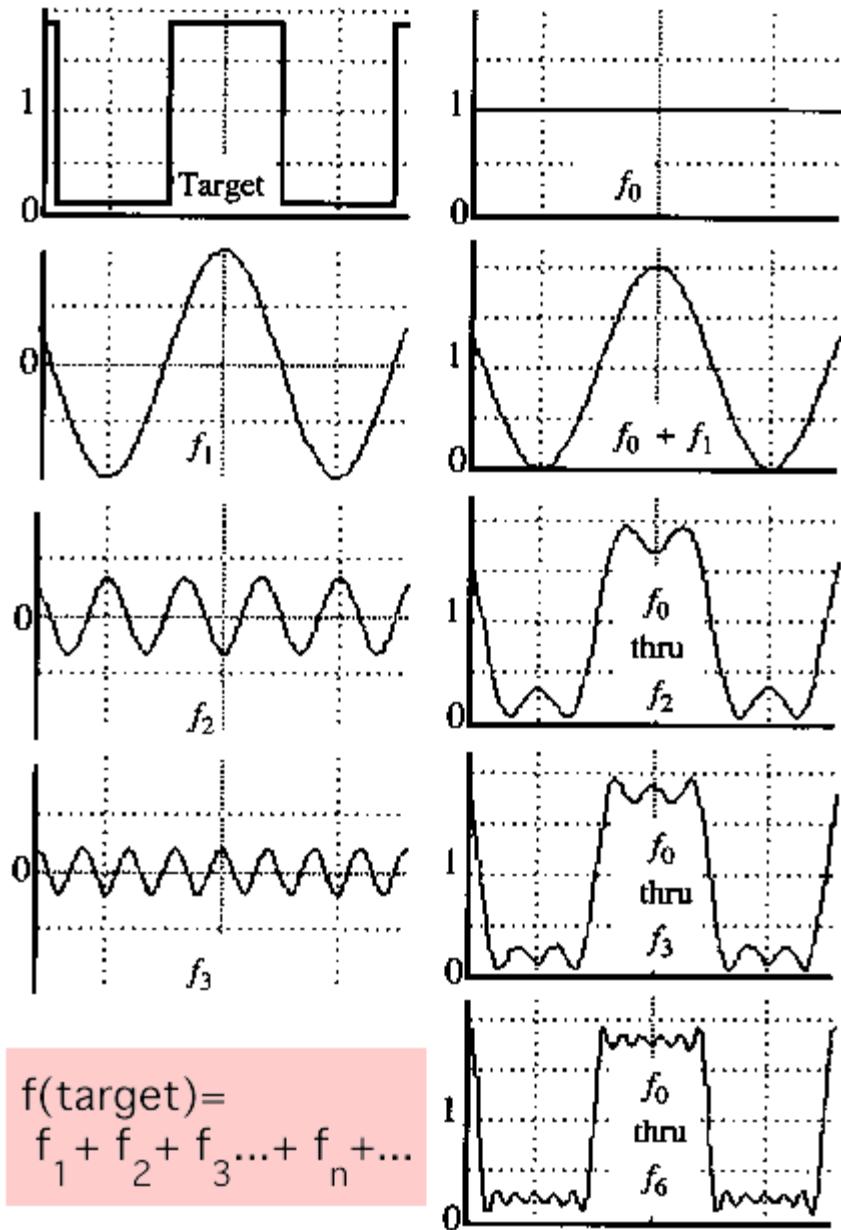
$$A \sin(\omega x + \phi)$$

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?



Fourier Transform

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



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

- How can F hold both?

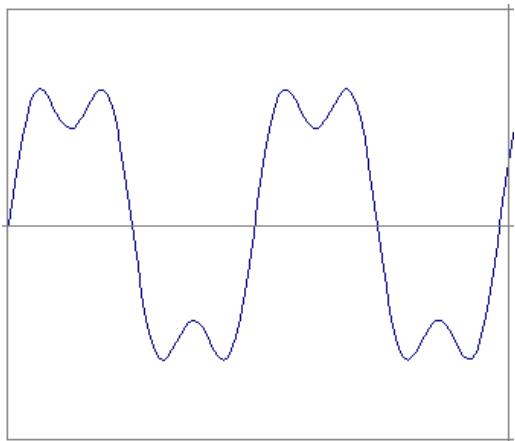
$$F(\omega) = R(\omega) + iI(\omega)$$
$$A = \pm \sqrt{R(\omega)^2 + I(\omega)^2} \quad \phi = \tan^{-1} \frac{I(\omega)}{R(\omega)}$$

We can always go back:



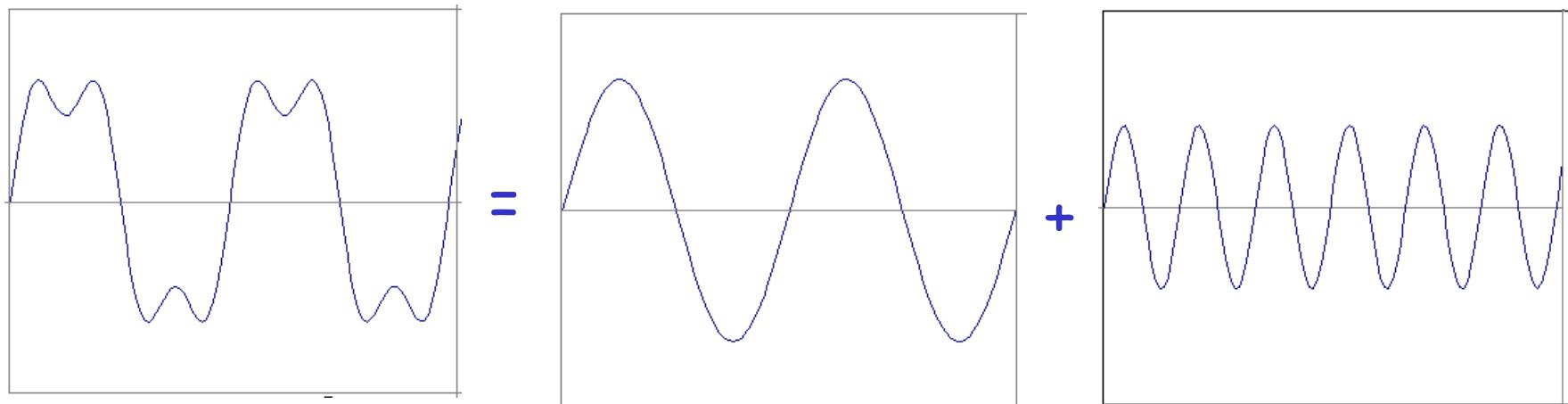
Time and Frequency

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



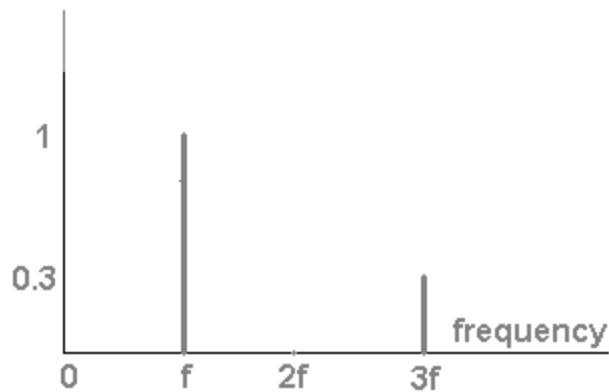
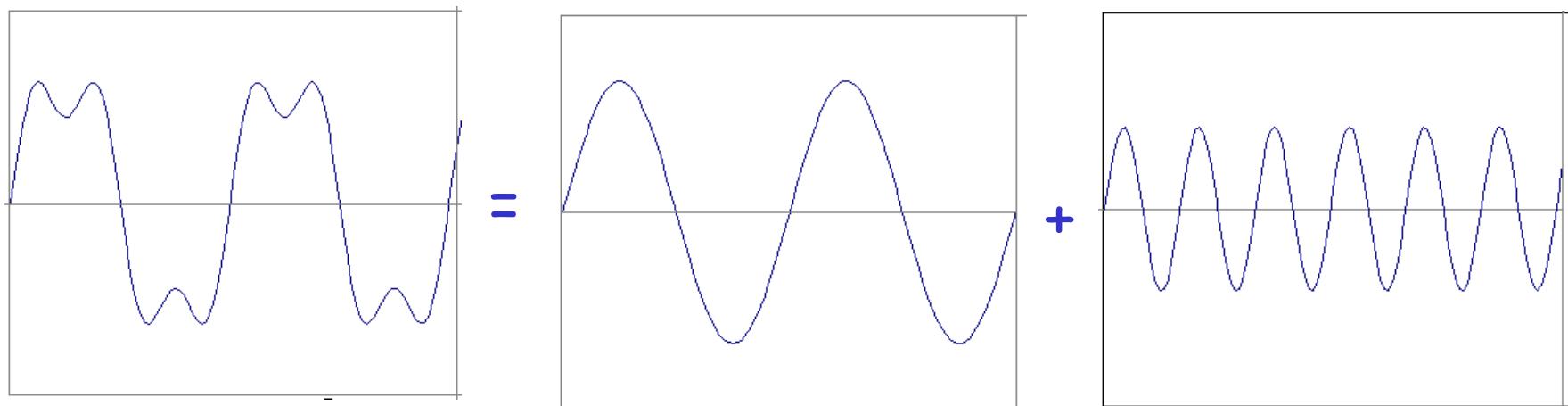
Time and Frequency

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



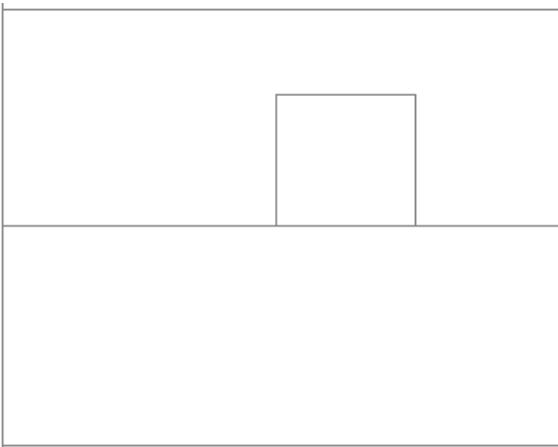
Frequency Spectra

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

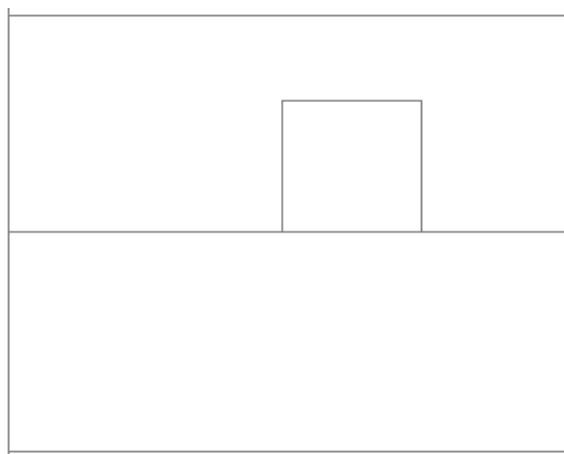


Frequency Spectra

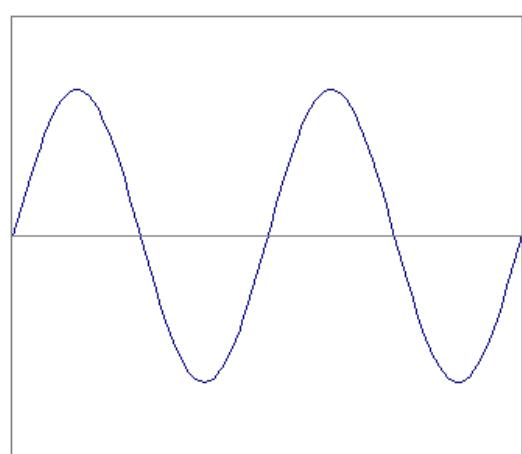
Usually, frequency is more interesting than the phase



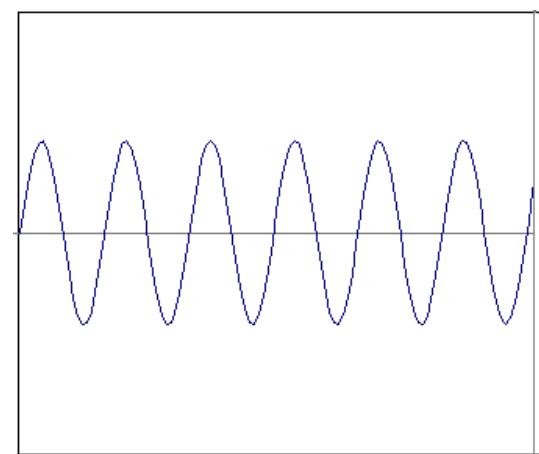
Frequency Spectra



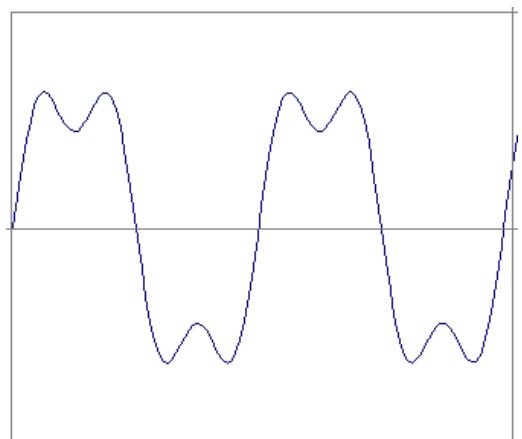
=



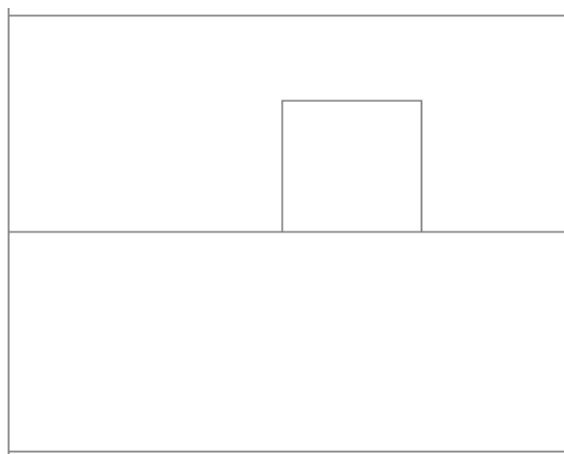
+



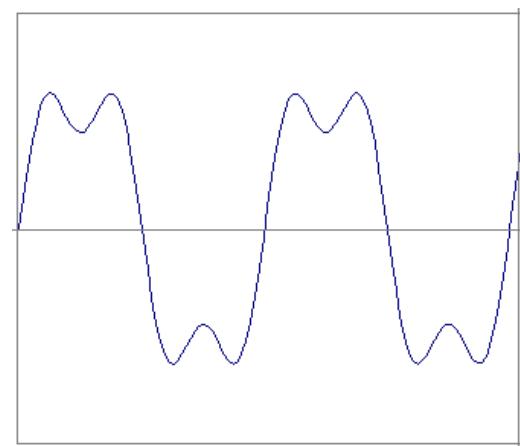
=



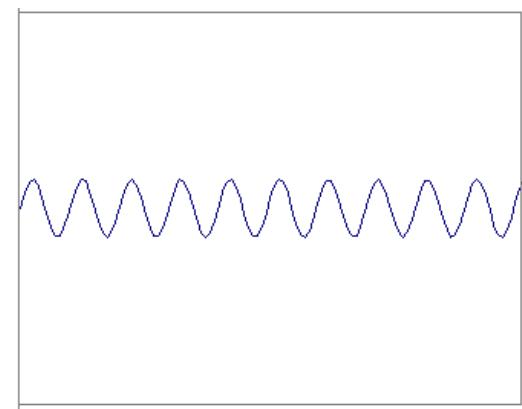
Frequency Spectra



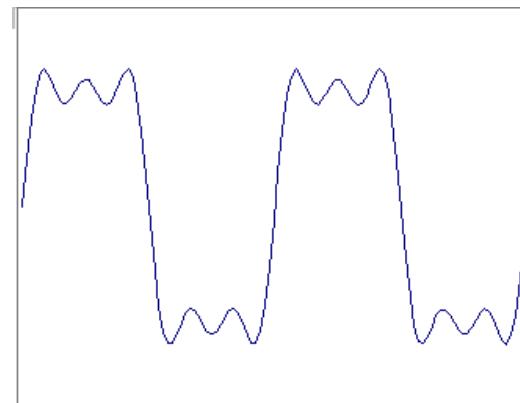
=



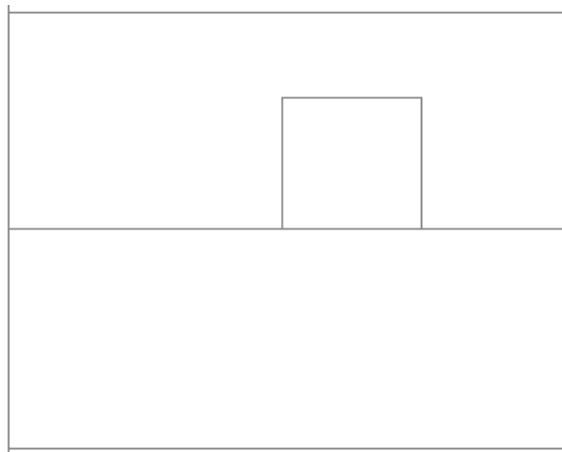
+



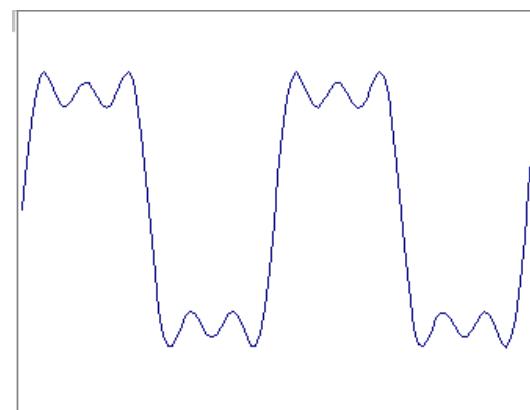
=



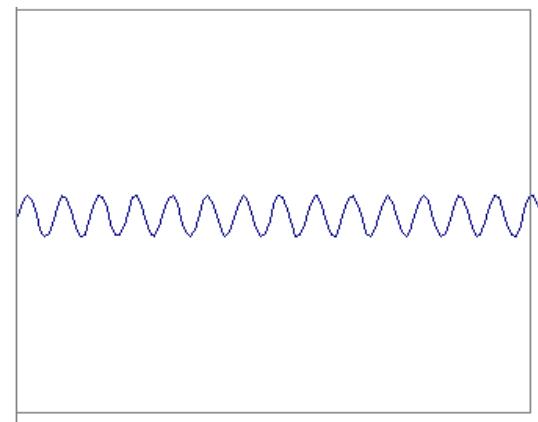
Frequency Spectra



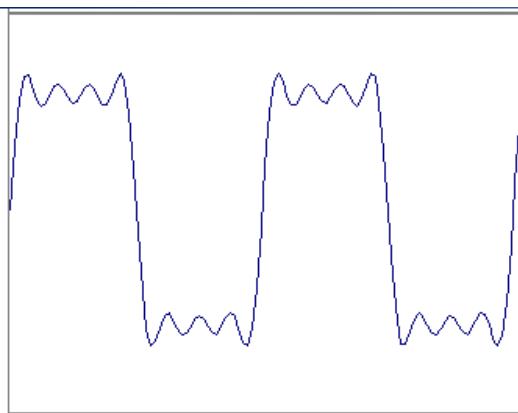
=



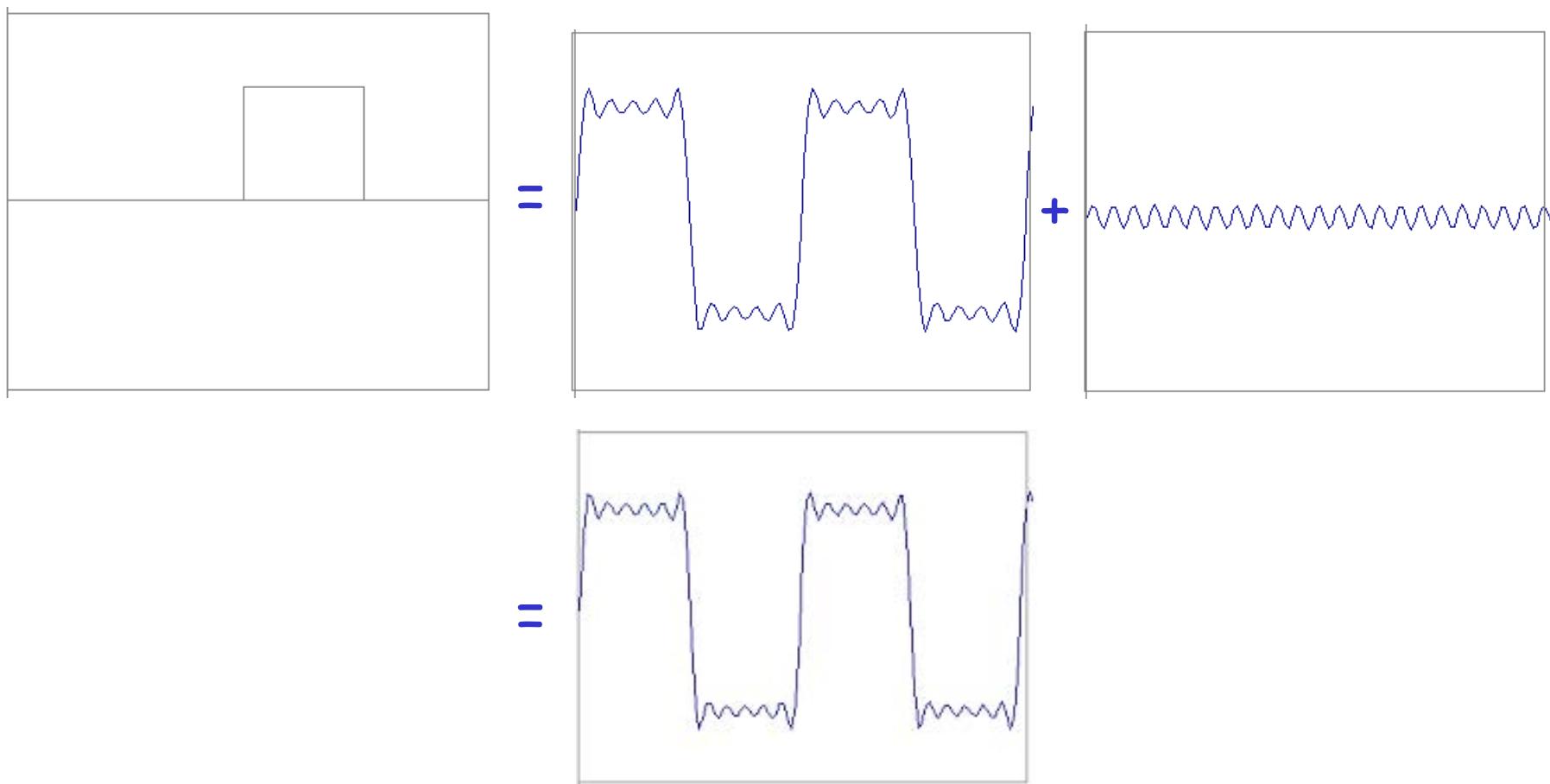
+



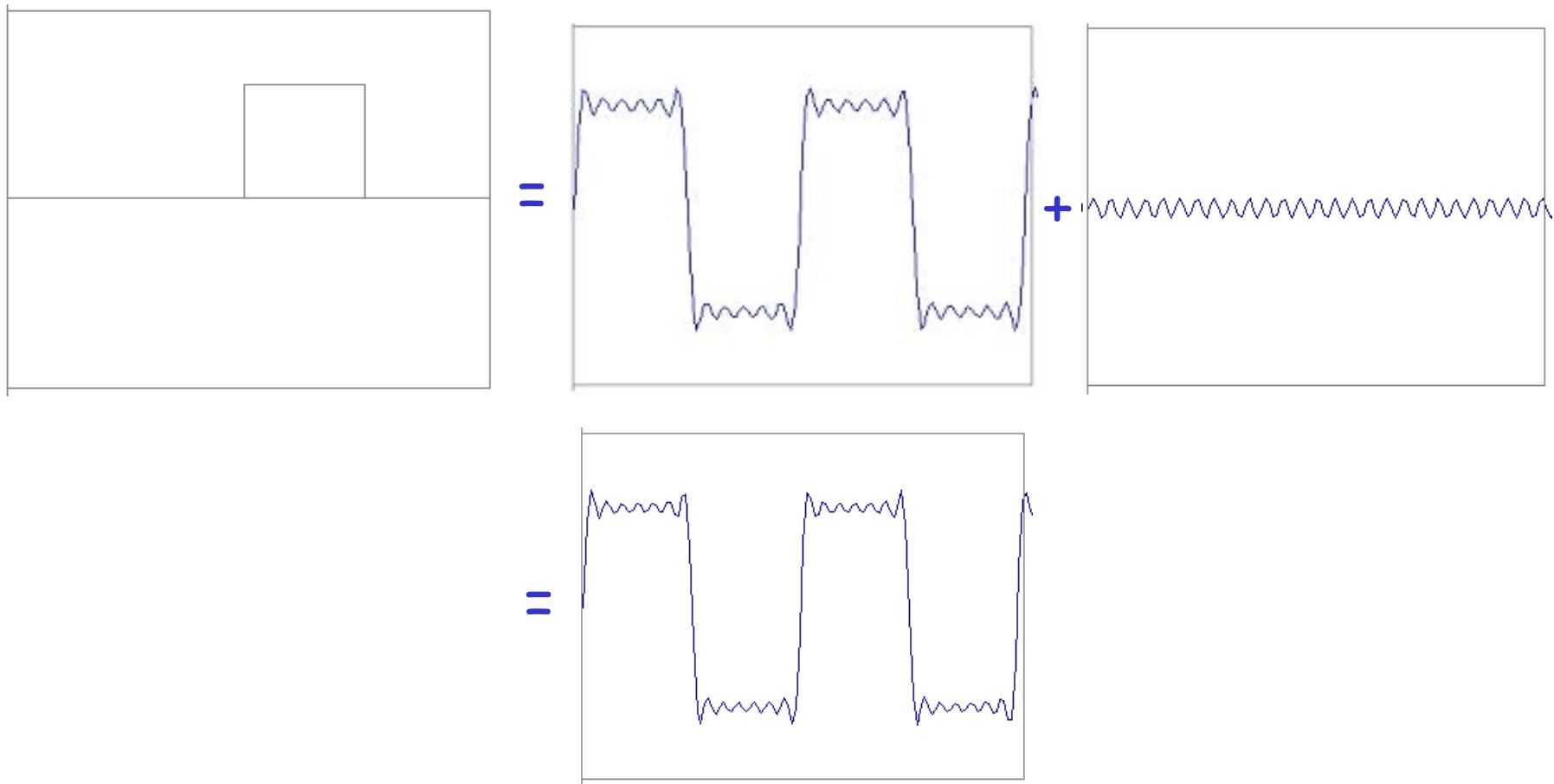
=



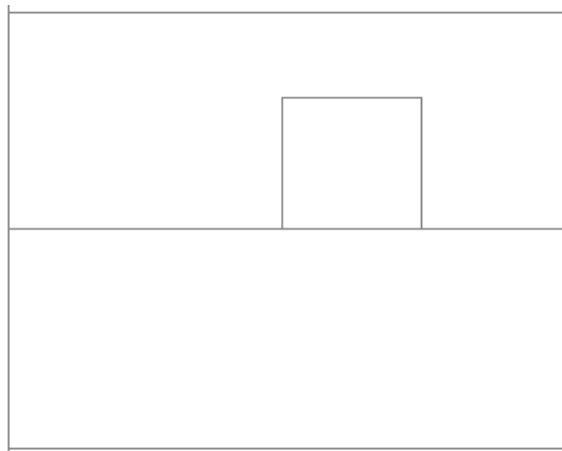
Frequency Spectra



Frequency Spectra

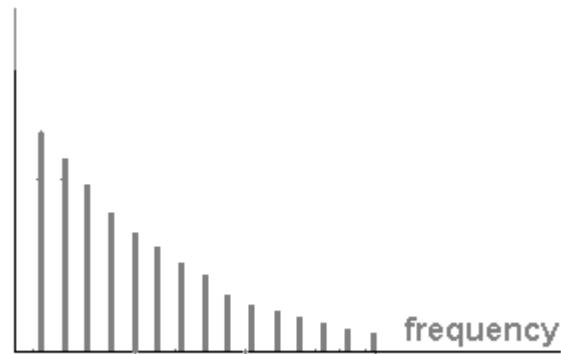


Frequency Spectra

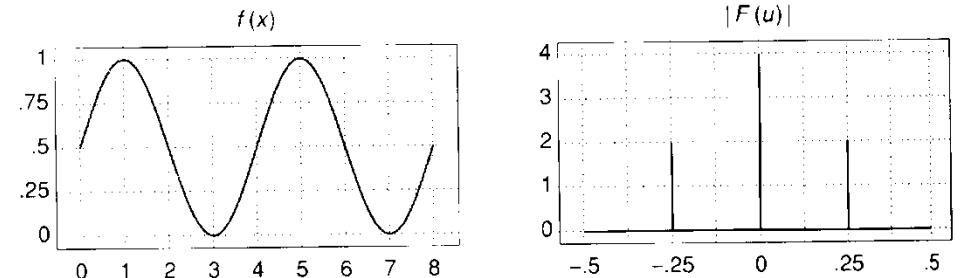


=

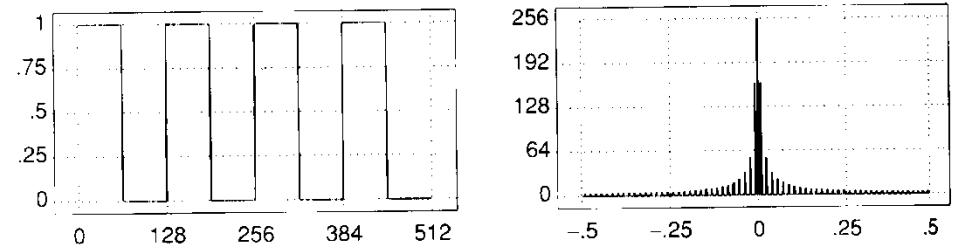
$$A \sum_{k=1}^{\infty} \frac{1}{k} \sin(2\pi kt)$$



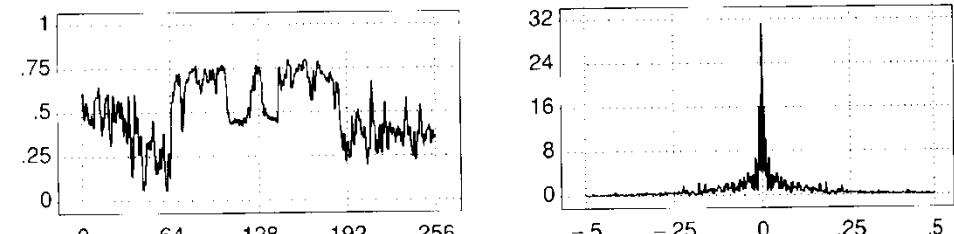
Frequency Spectra



(a)



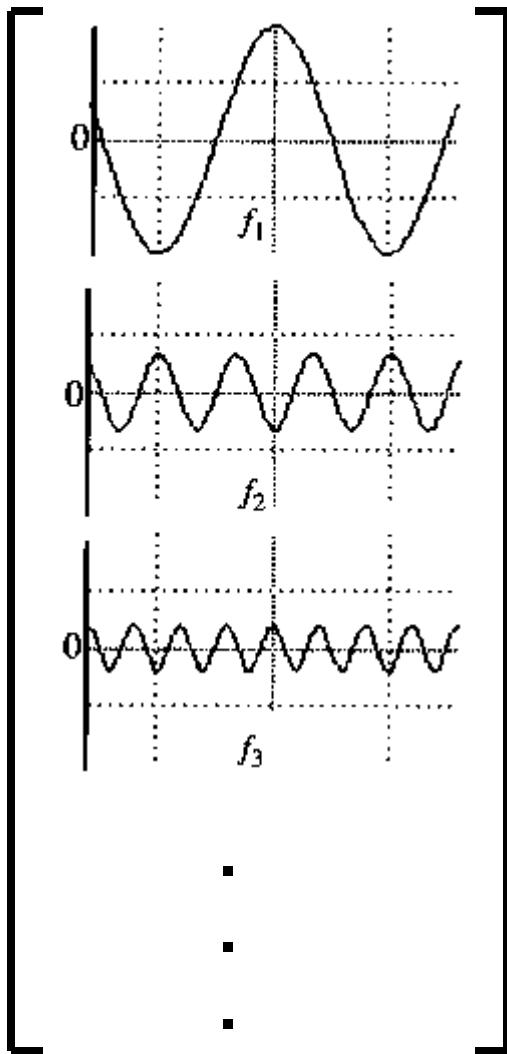
(b)



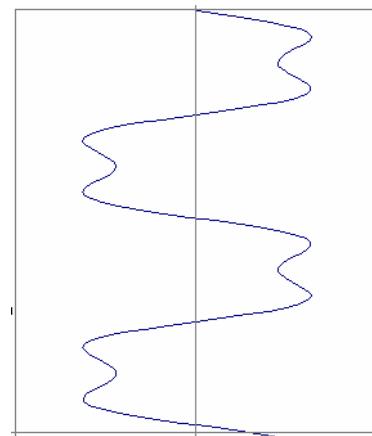
(c)

FT: Just a change of basis

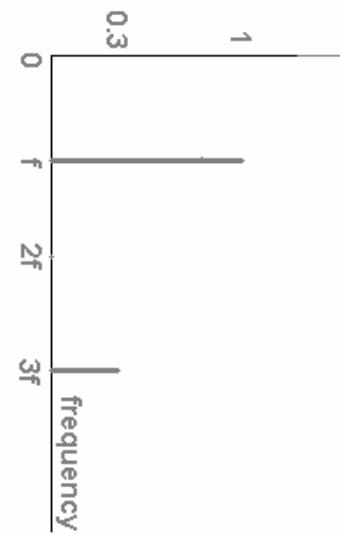
$$\mathbf{M}^* f(x) = F(\omega)$$



*

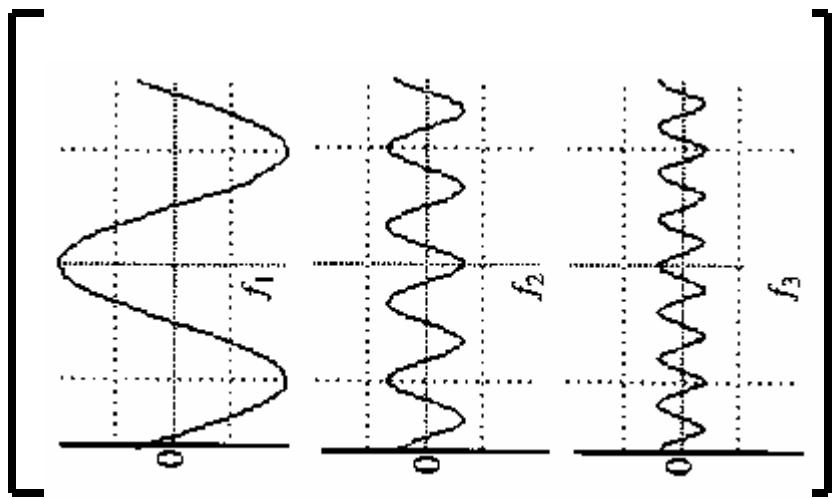


==

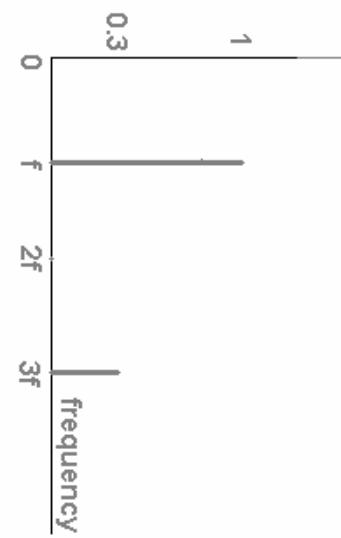


IFT: Just a change of basis

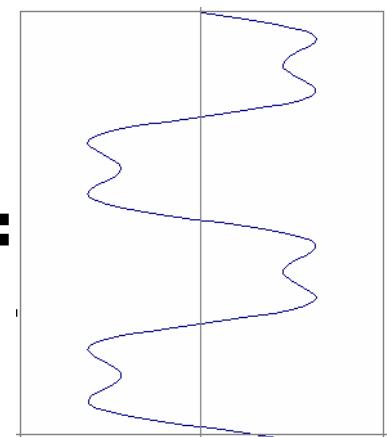
$$M^{-1} * F(\omega) = f(x)$$



*



=



Finally: Scary Math

$$\text{Fourier Transform : } F(\omega) = \int_{-\infty}^{+\infty} f(x)e^{-i\omega x} dx$$

$$\text{Inverse Fourier Transform : } f(x) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} F(\omega)e^{i\omega x} d\omega$$

Finally: Scary Math

$$\text{Fourier Transform : } F(\omega) = \int_{-\infty}^{+\infty} f(x)e^{-i\omega x} dx$$

$$\text{Inverse Fourier Transform : } f(x) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} F(\omega)e^{i\omega x} d\omega$$

...not really scary: $e^{i\omega x} = \cos(\omega x) + i \sin(\omega x)$

is hiding our old friend: $\sin(\omega x + \phi)$

phase can be encoded
by sin/cos pair



$$P \cos(x) + Q \sin(x) = A \sin(x + \phi)$$
$$A = \pm \sqrt{P^2 + Q^2}$$
$$\phi = \tan^{-1} \left(\frac{P}{Q} \right)$$

So it's just our signal $f(x)$ times sine at frequency ω

Extension to 2D

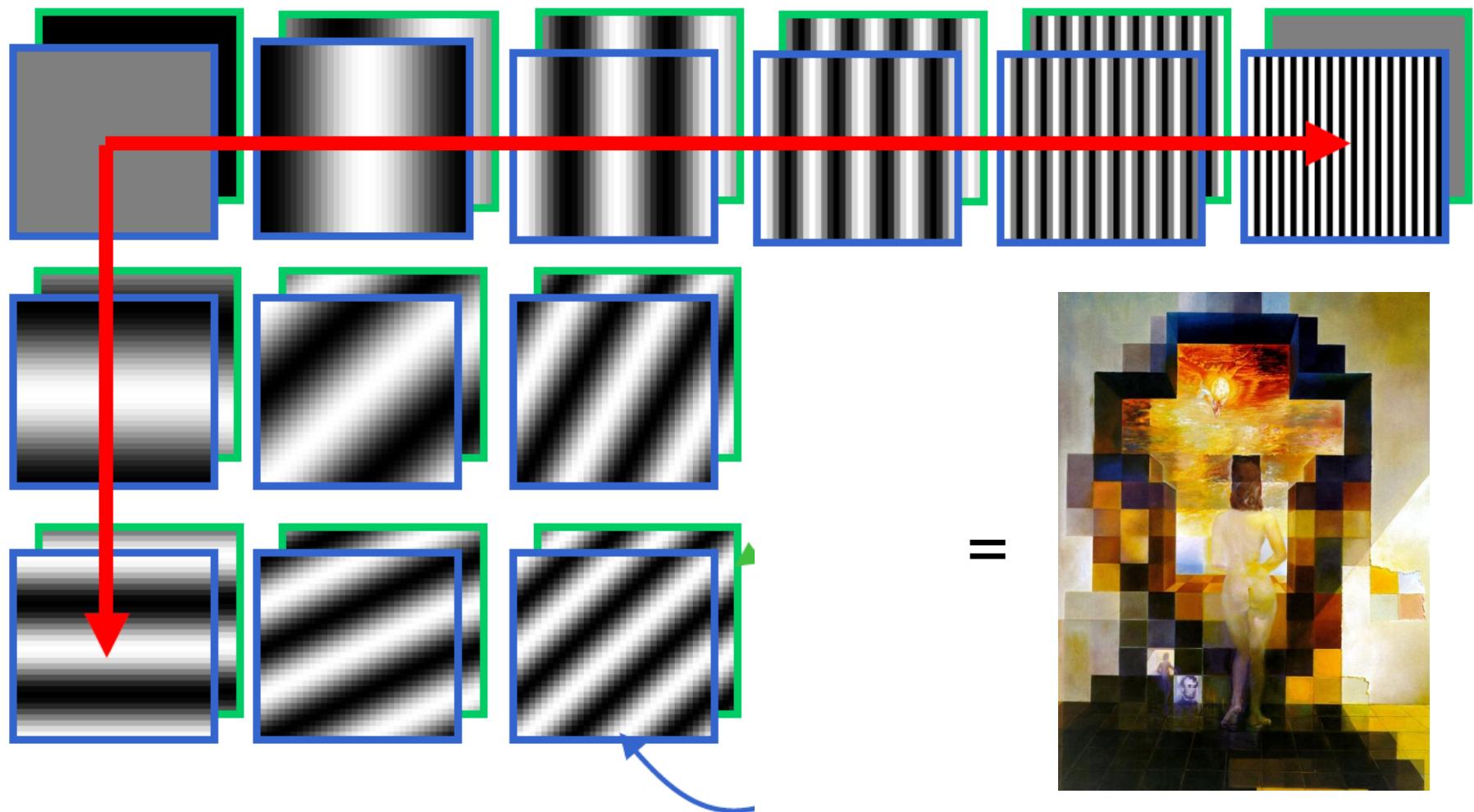
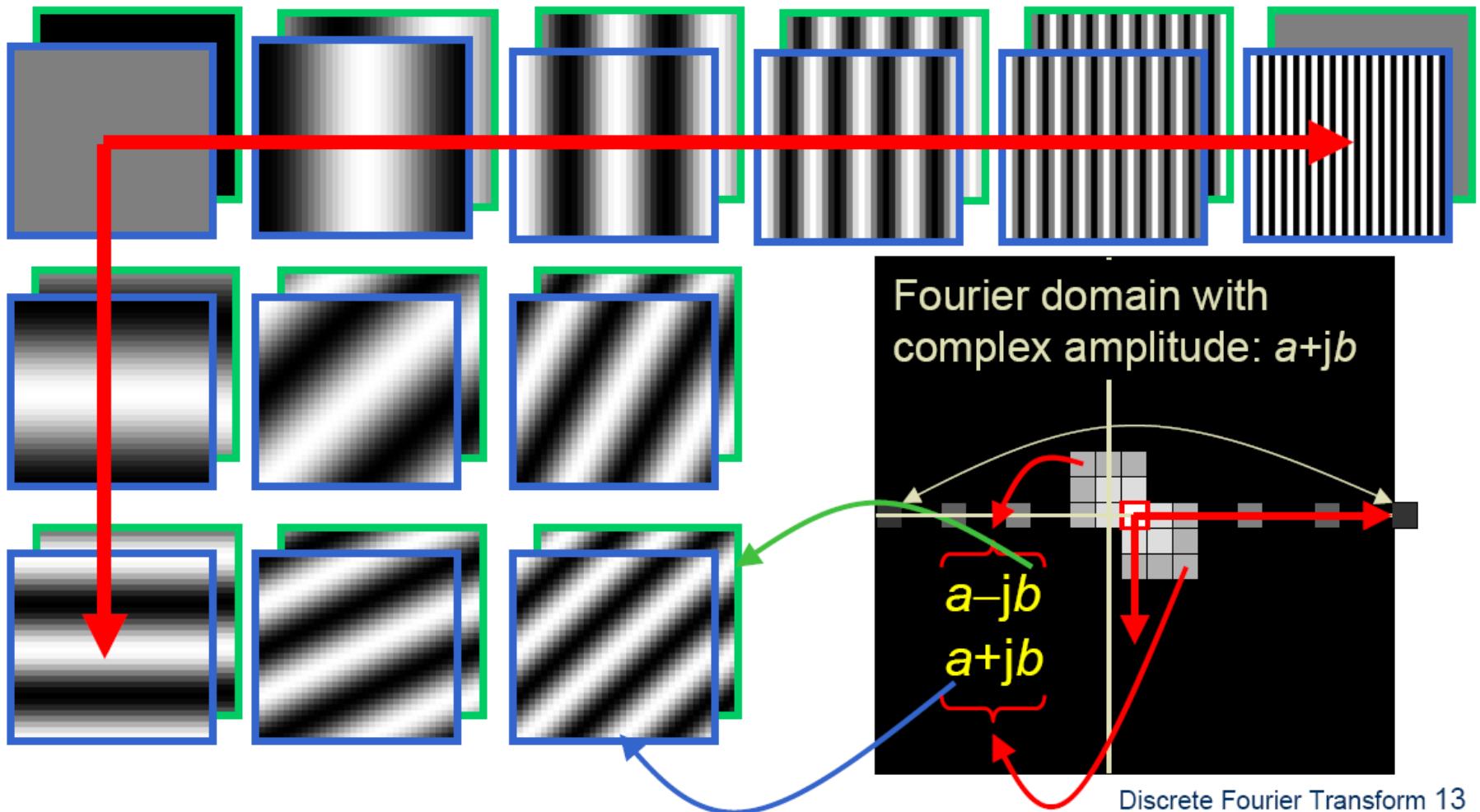


Image as a sum of basis images

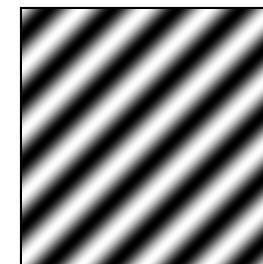
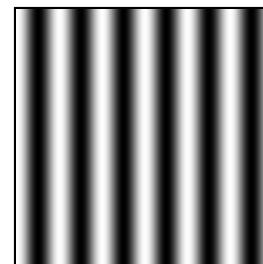
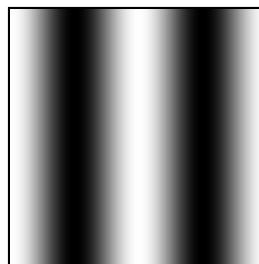
Extension to 2D



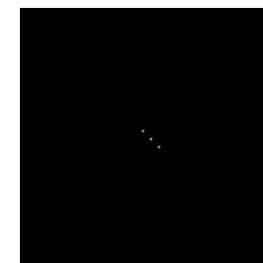
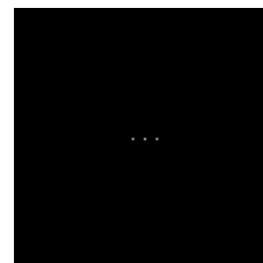
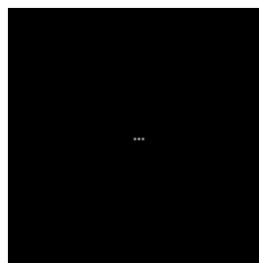
in Matlab, check out: `imagesc(log(abs(fftshift(fft2(im)))));`

Fourier analysis in images

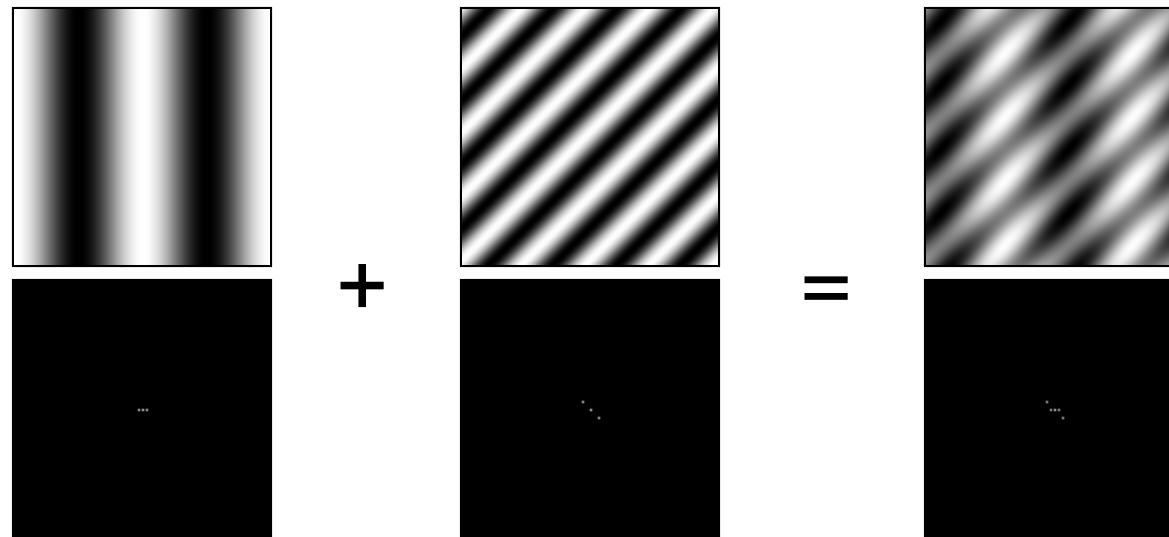
Intensity Image



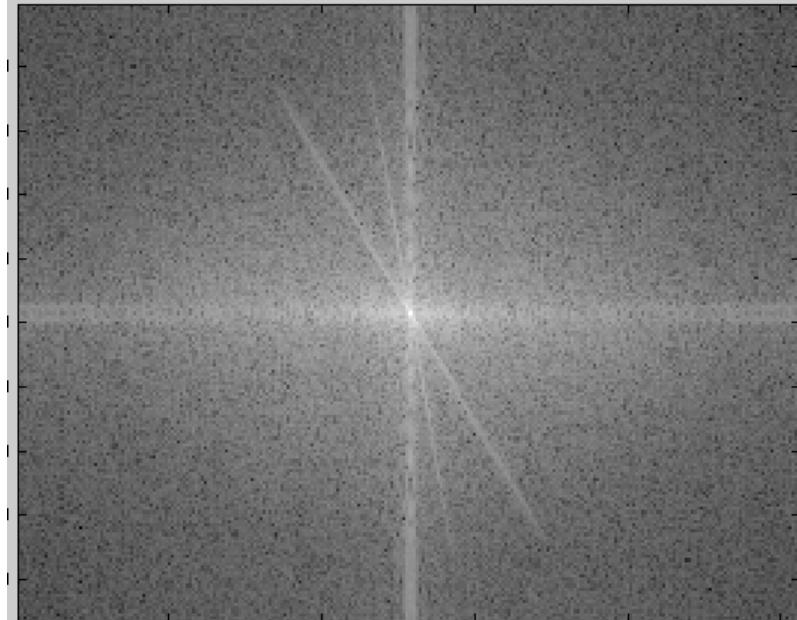
Fourier Image



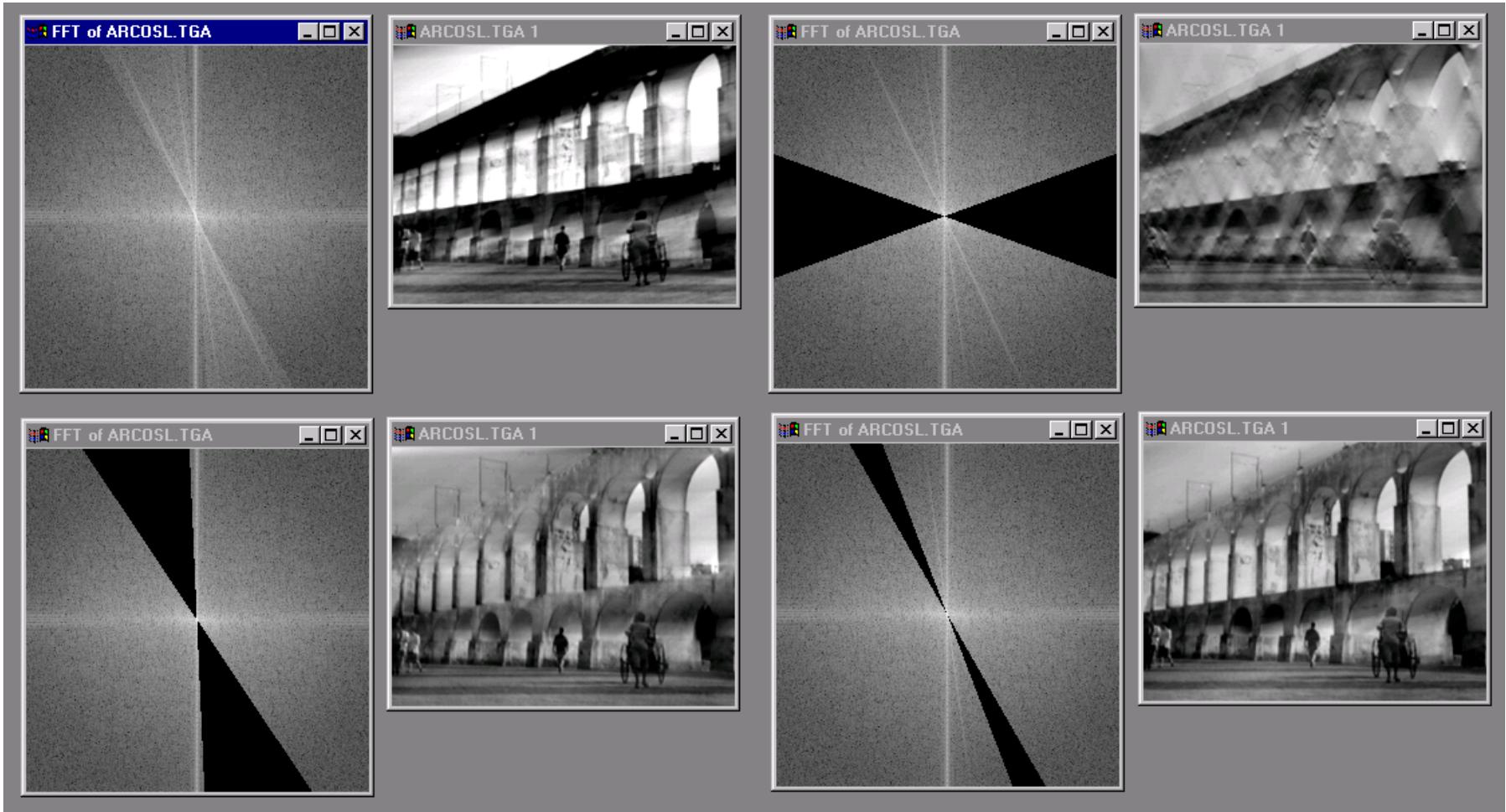
Signals can be composed



Man-made Scene

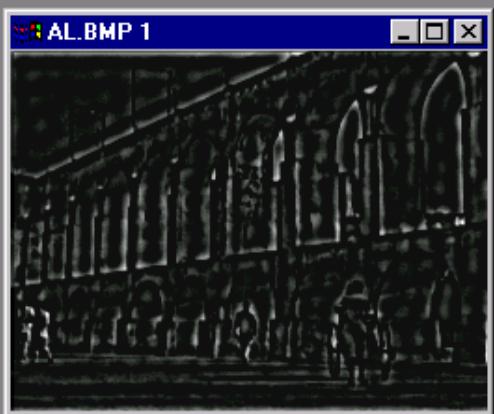
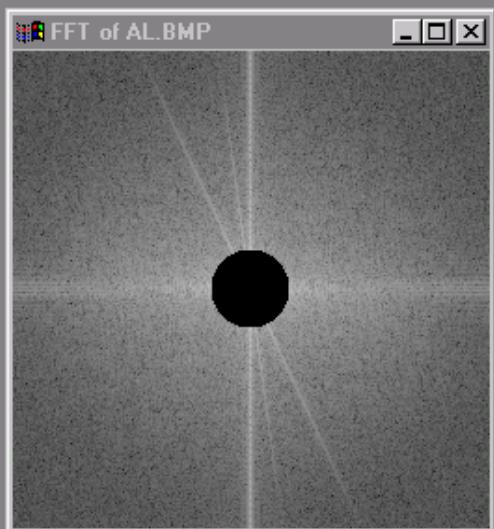
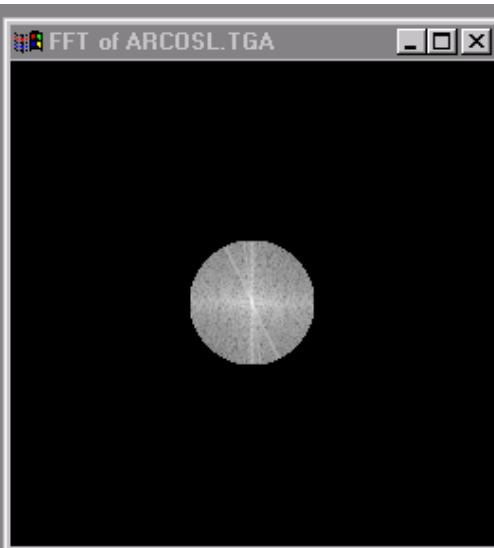


Can change spectrum, then reconstruct



Local change in one domain, courses global change in the other

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

$$\mathcal{F}[g * h] = \mathcal{F}[g]\mathcal{F}[h]$$

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

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

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

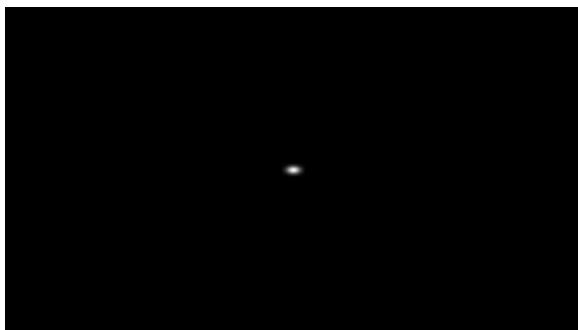
2D convolution theorem example

$f(x,y)$



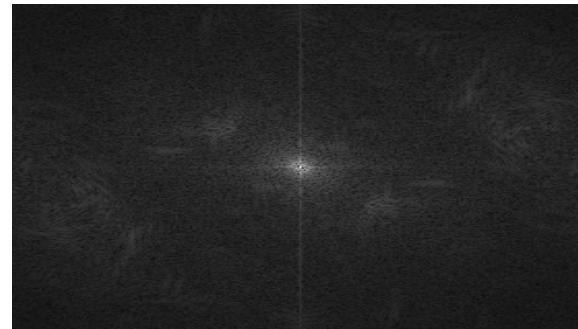
*

$h(x,y)$



↓↓

$g(x,y)$



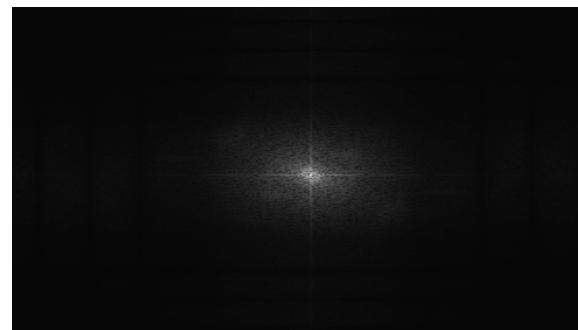
×

$|F(s_x, s_y)|$



↓↓

$|H(s_x, s_y)|$

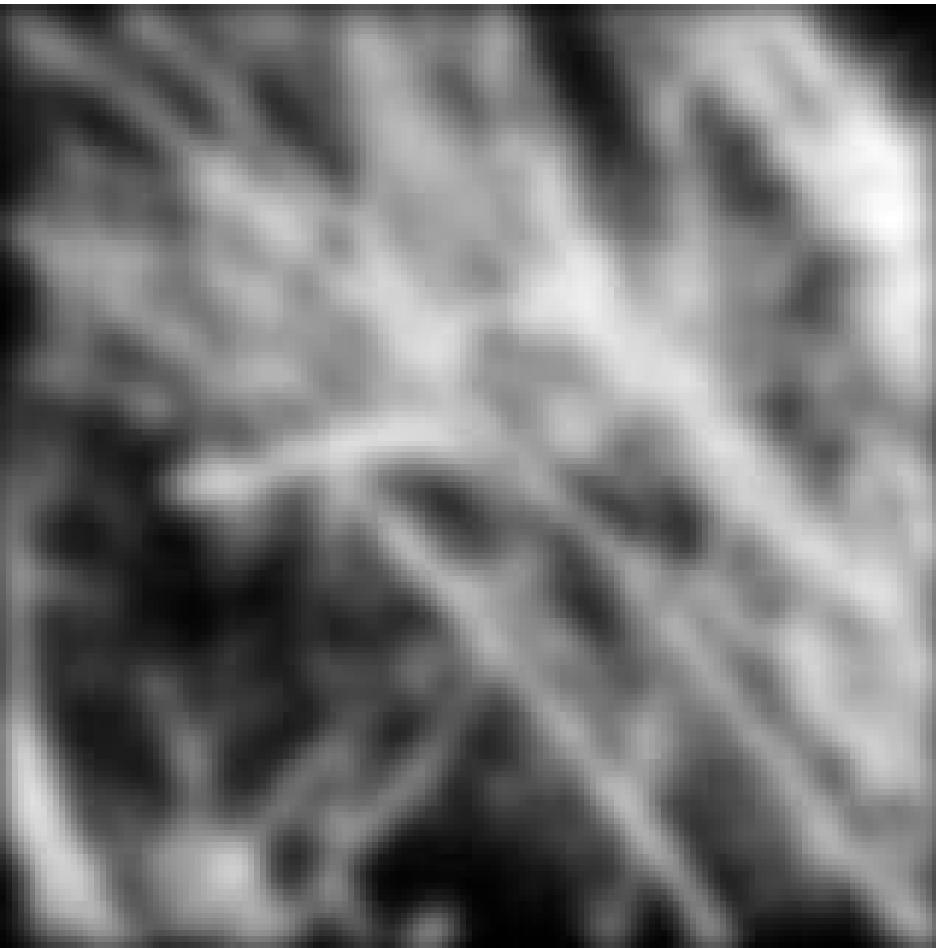


$|G(s_x, s_y)|$

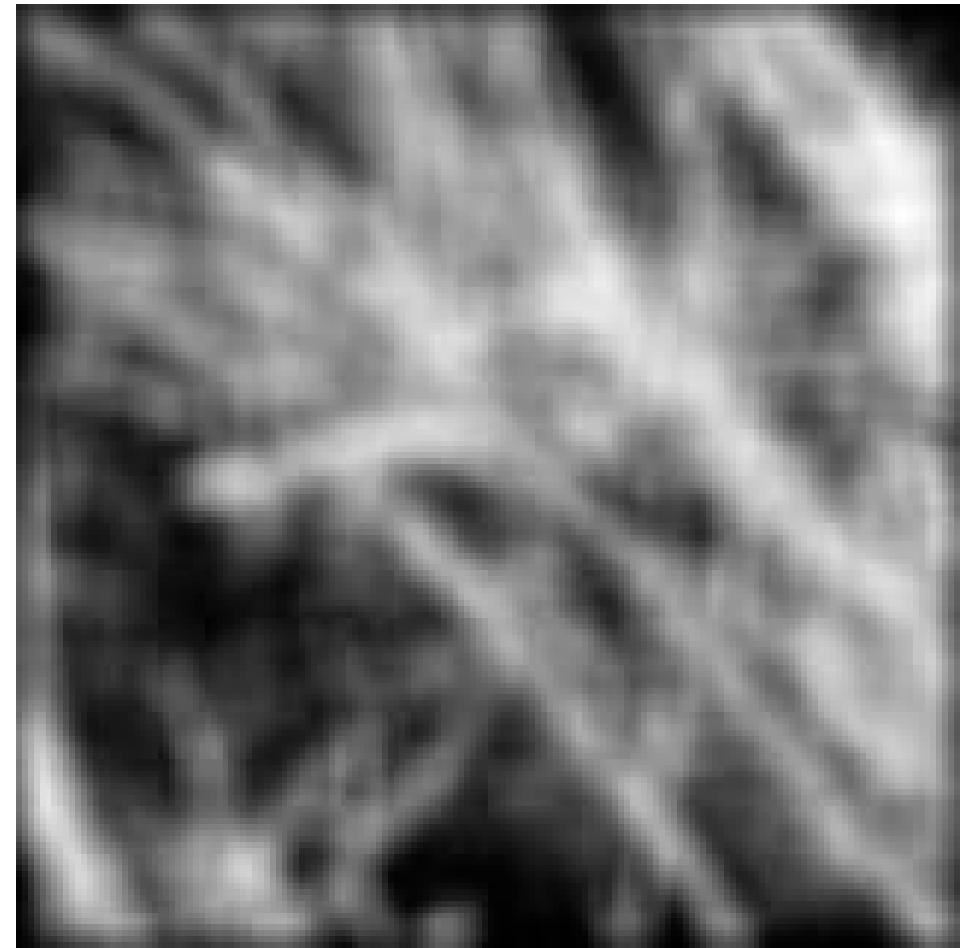
Filtering

Why does the Gaussian give a nice smooth image, but the square filter give edgy artifacts?

Gaussian



Box filter



Fourier Transform pairs

Spatial domain

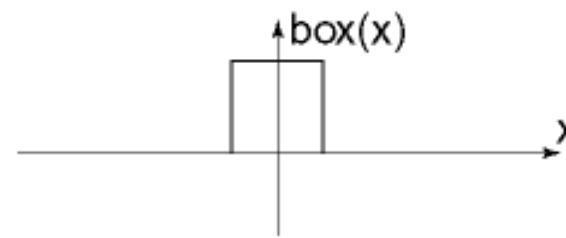
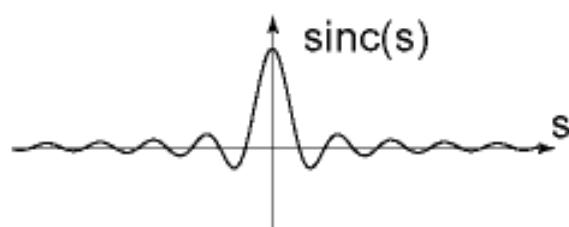
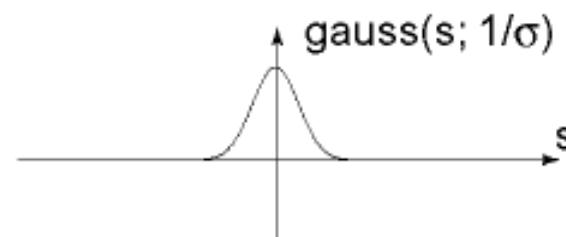
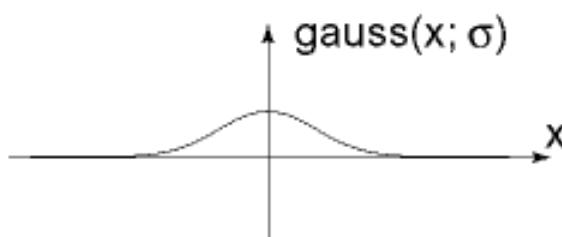
$$f(x)$$

↑
box(x)

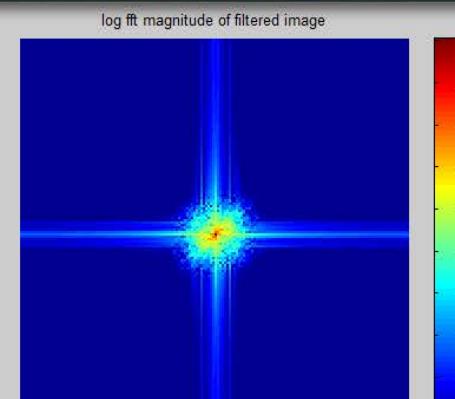
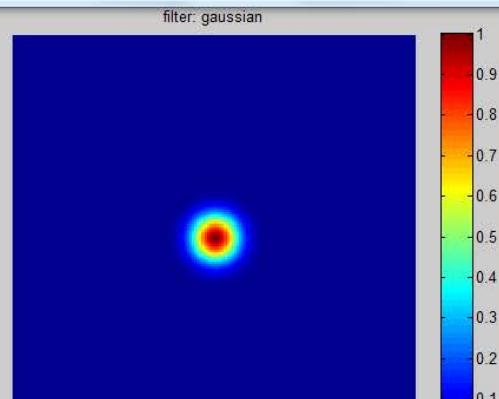
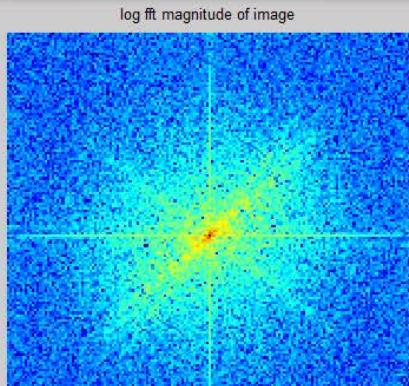
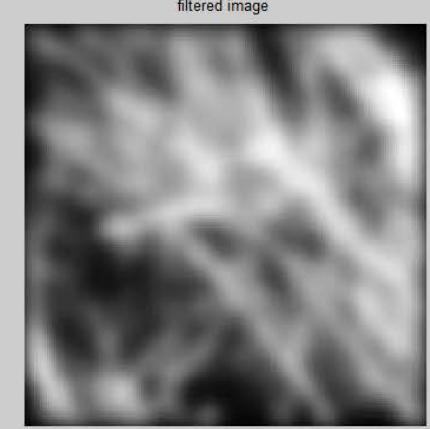
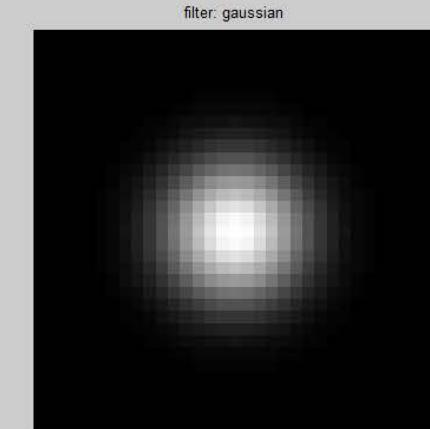
Frequency domain

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

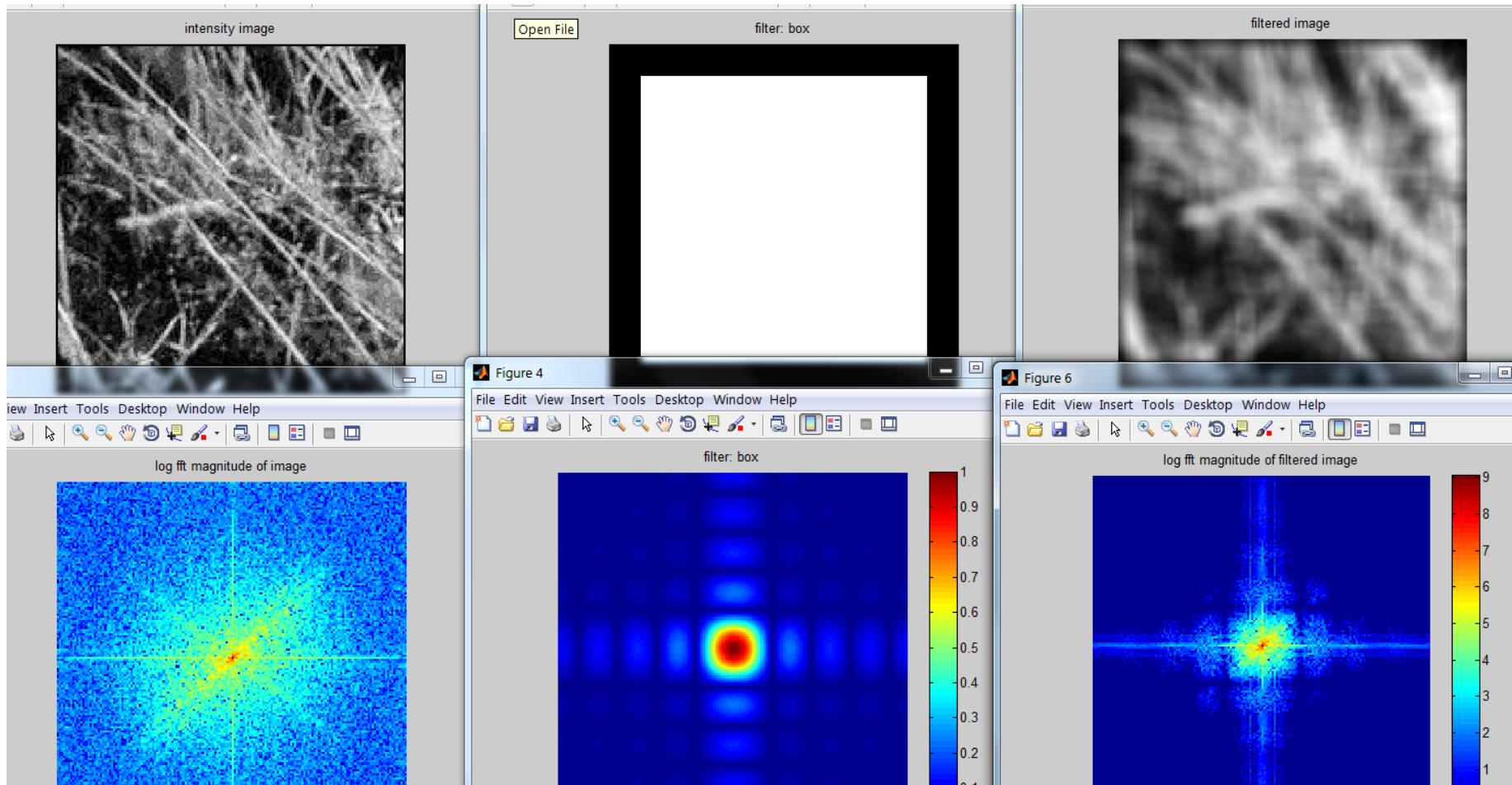
sinc(s)



Gaussian

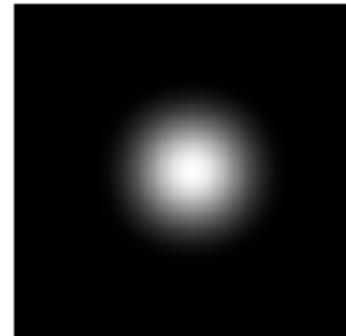
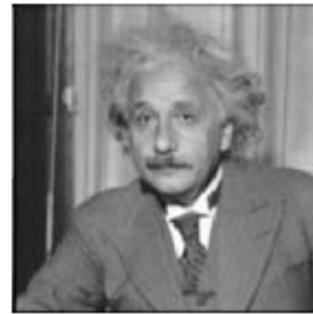
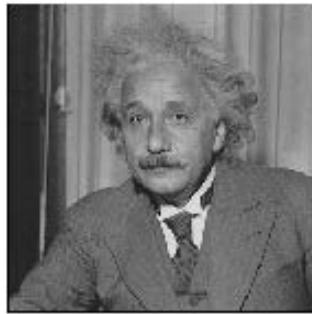


Box Filter

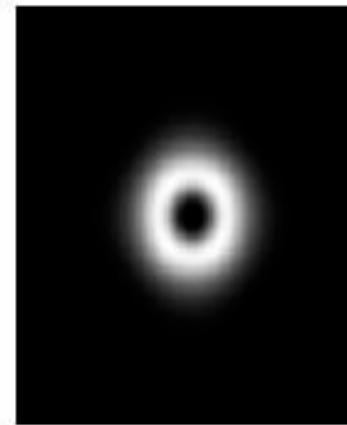
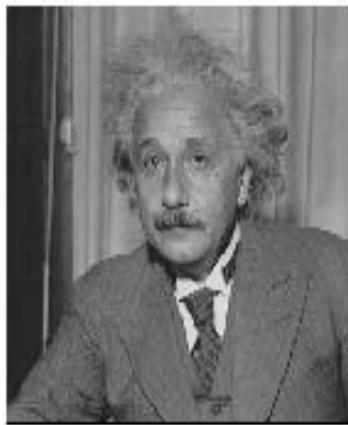


Low-pass, Band-pass, High-pass filters

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

Image “Sharpening”

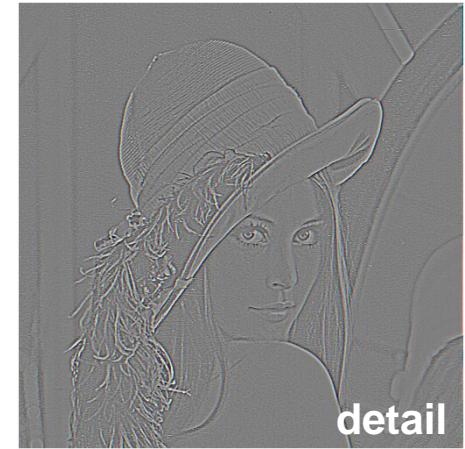
What does blurring take away?



-



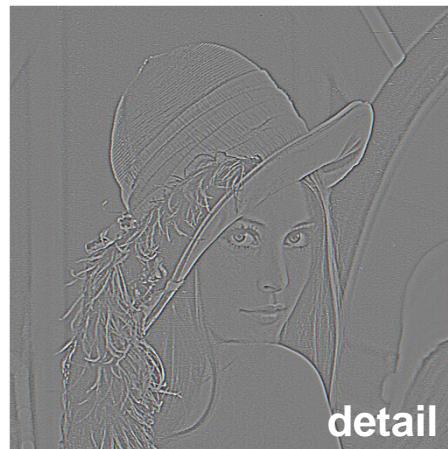
=



Let's add it back:



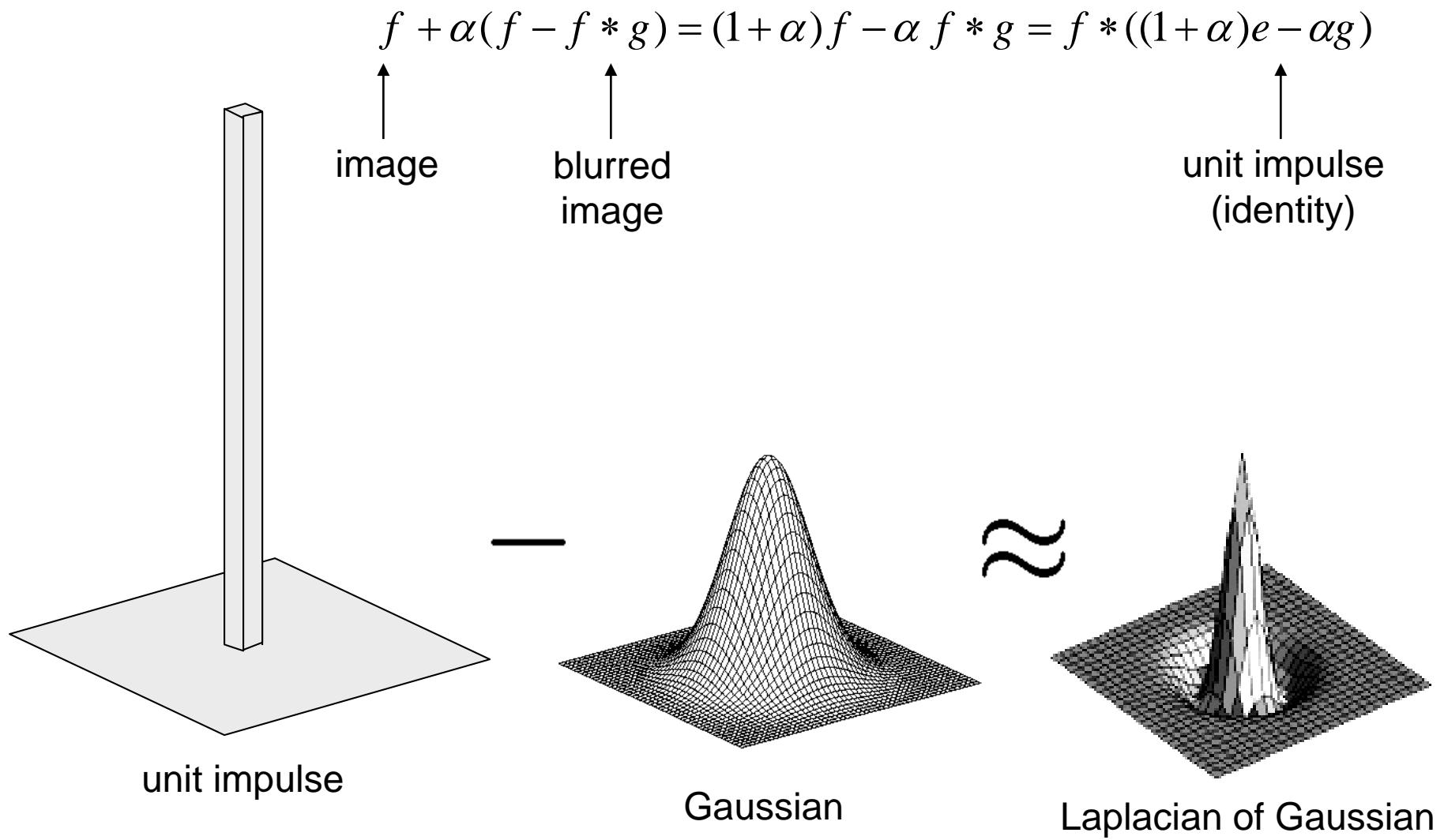
+ α



=



Unsharp mask filter



application: Hybrid Images

What you see...

I see an
angry guy

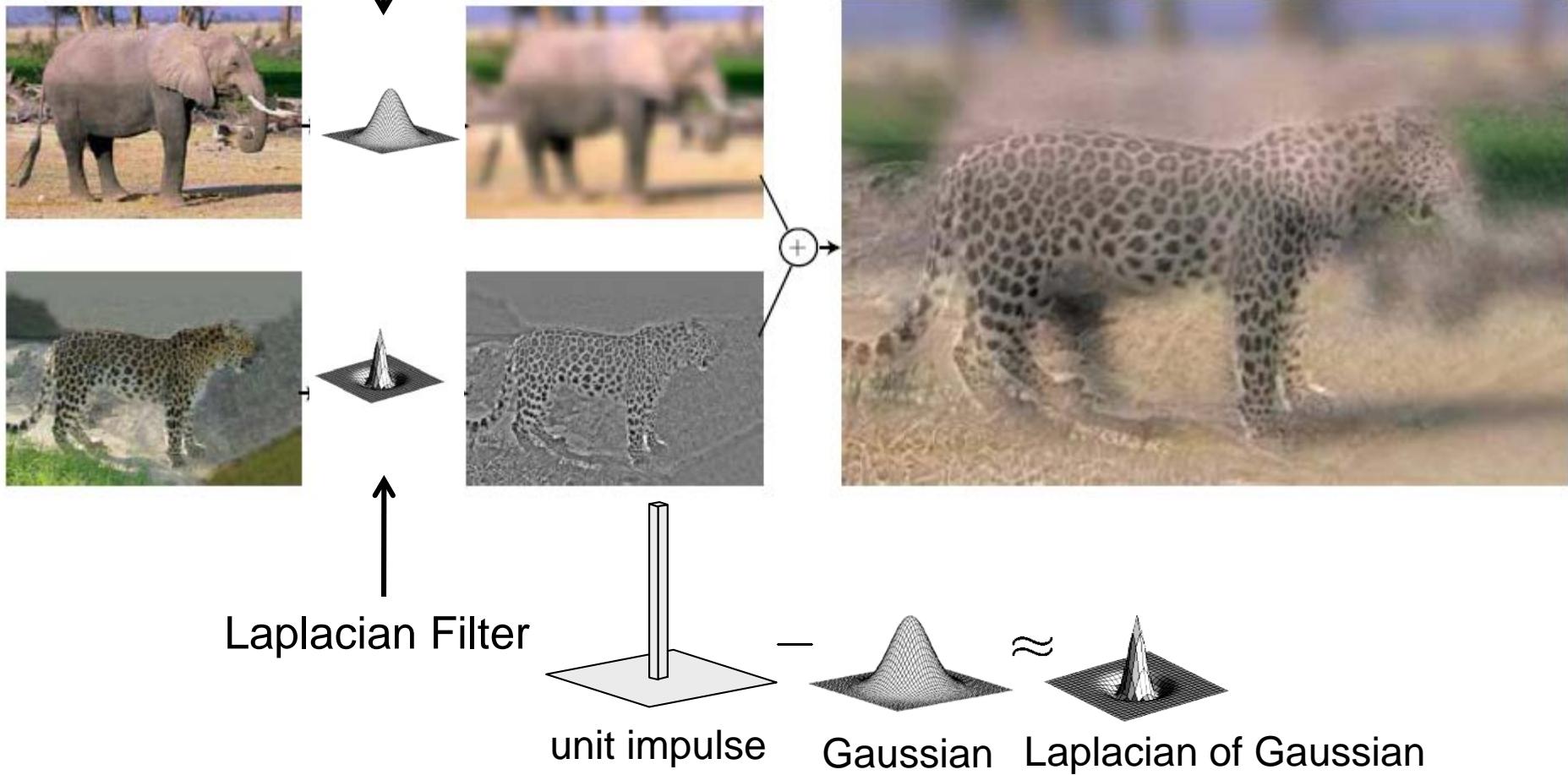
It's a woman!

From Far Away **Up Close**

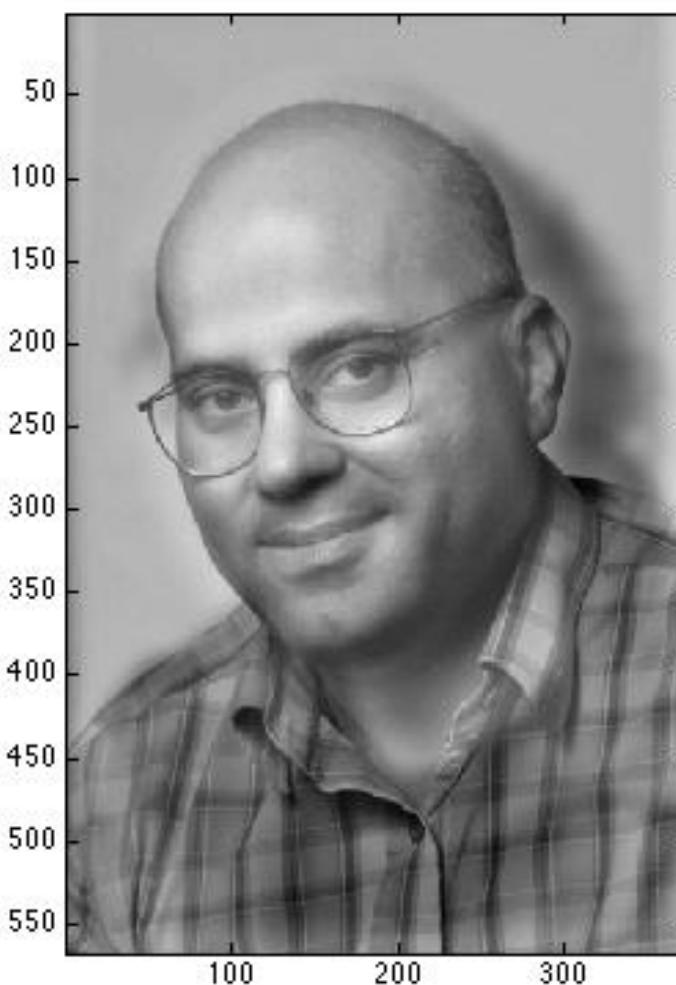
Application: Hybrid Images

Gaussian Filter

A. Oliva, A. Torralba, P.G. Schyns,
[“Hybrid Images,”](#) SIGGRAPH 2006



Yestaryear's homework (CS194-26: Riyaz Faizullabhoj)



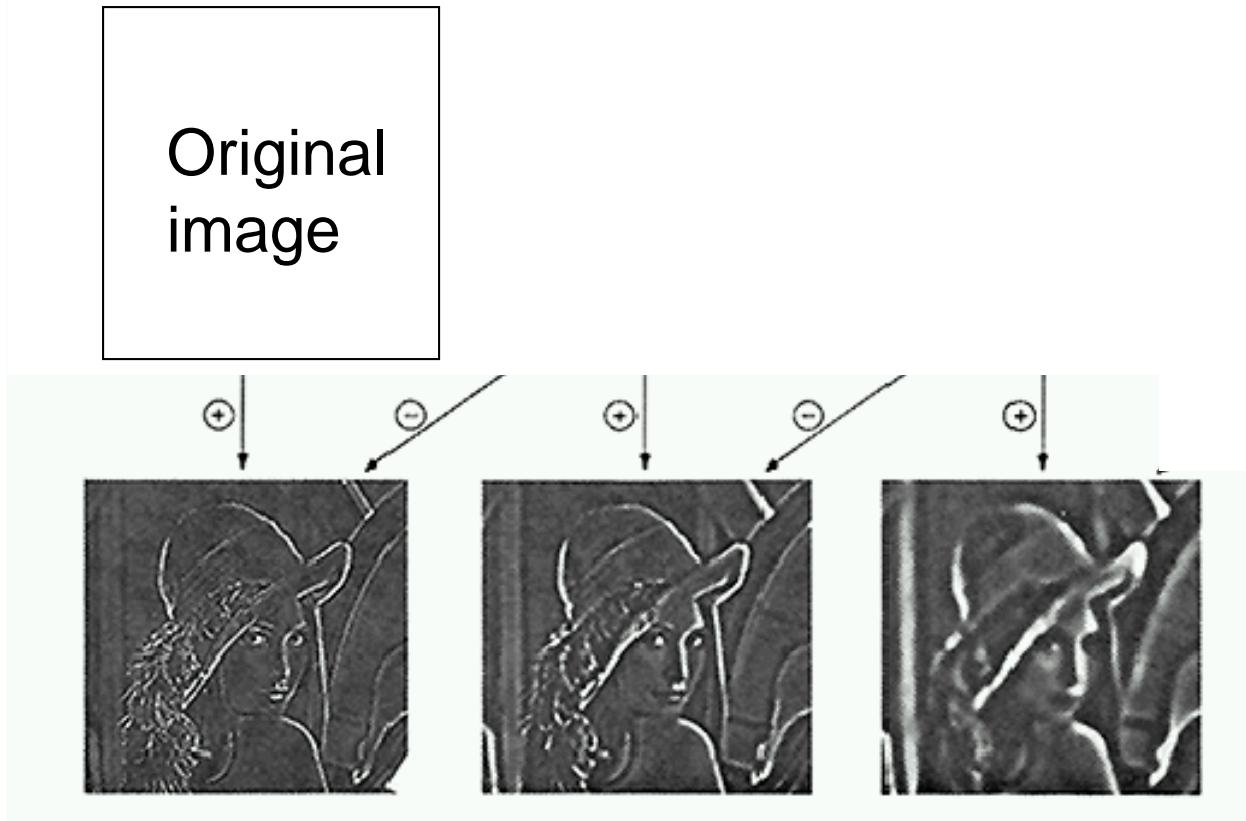
Prof. Jitendros Papadimalik

Band-pass filtering

Gaussian Pyramid (low-pass images)

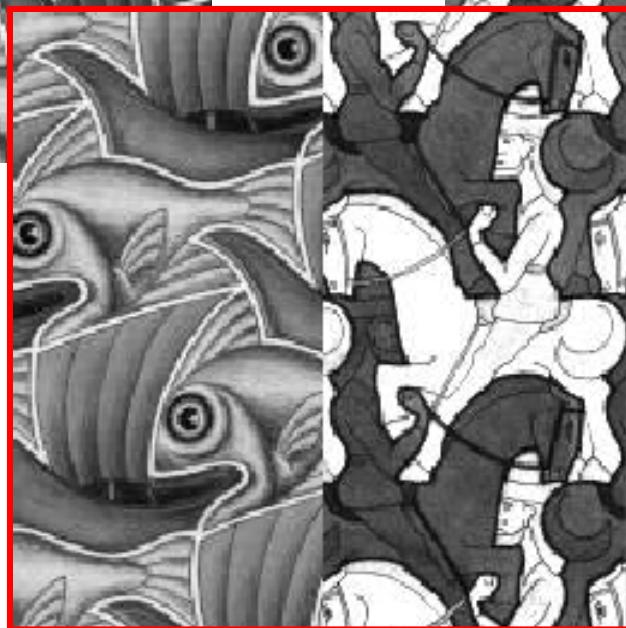
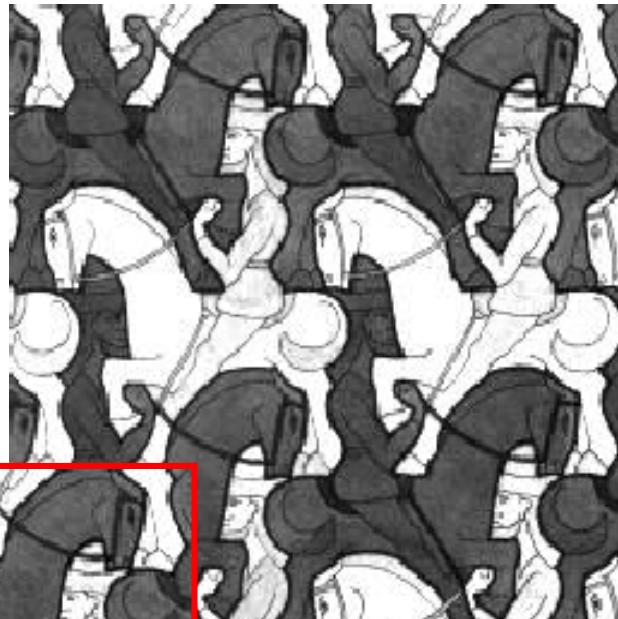
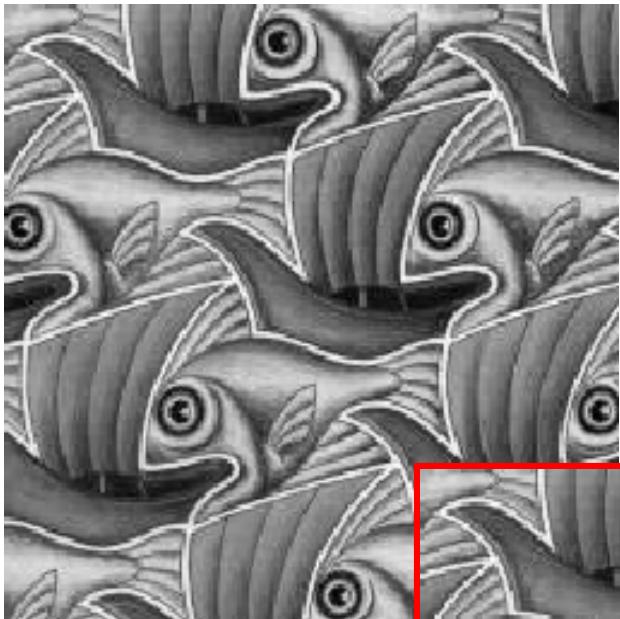


Laplacian Pyramid



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

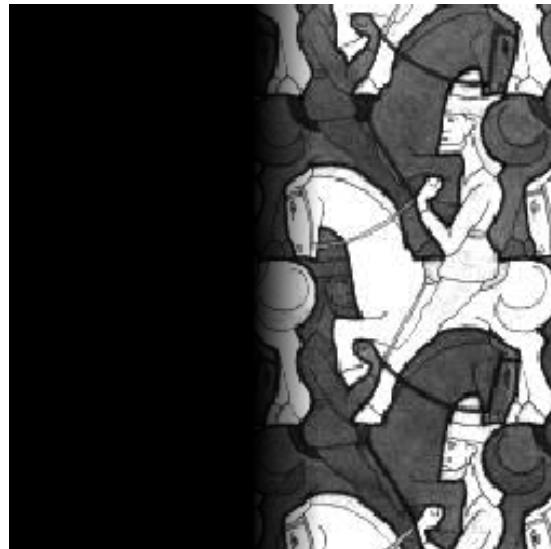
Blending



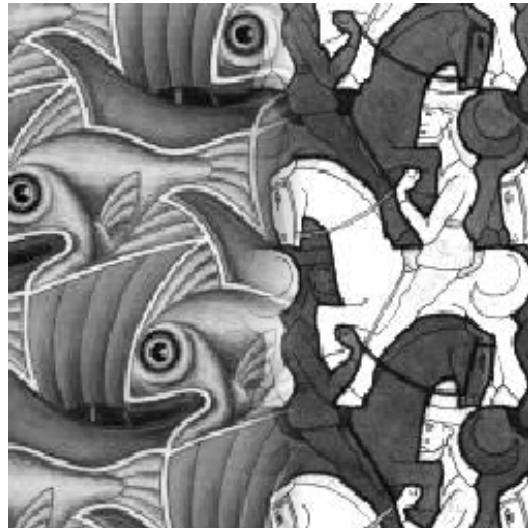
Alpha Blending / Feathering



+

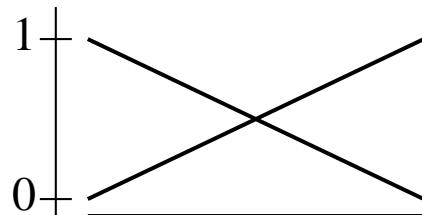
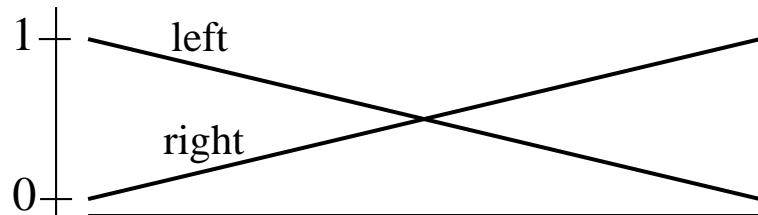
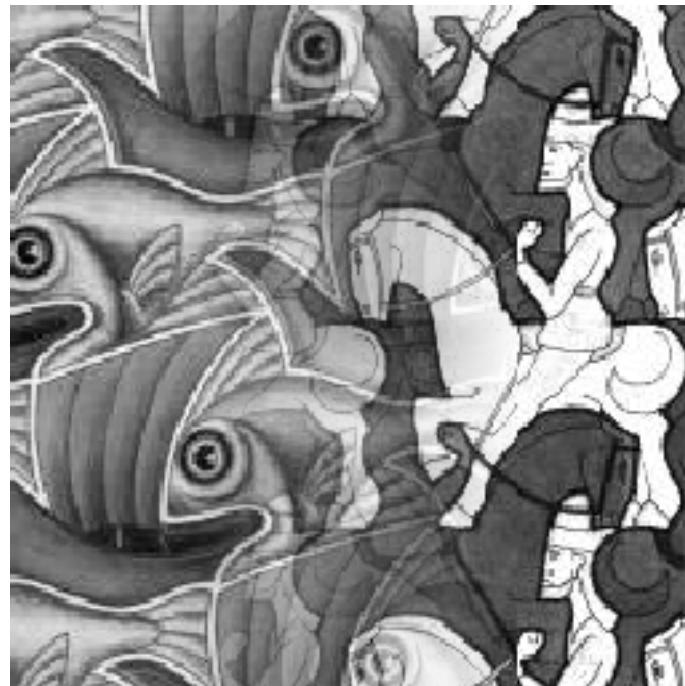


—

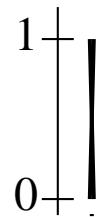
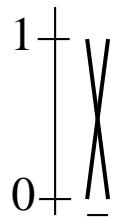
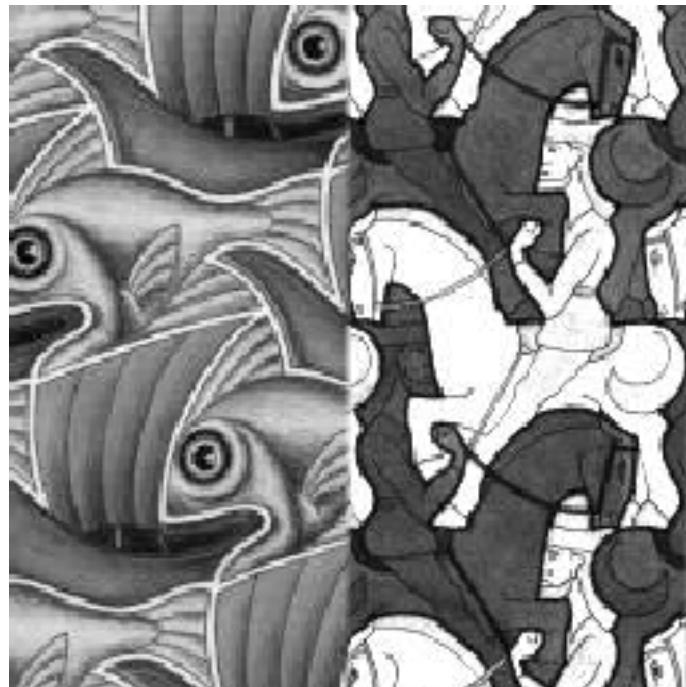
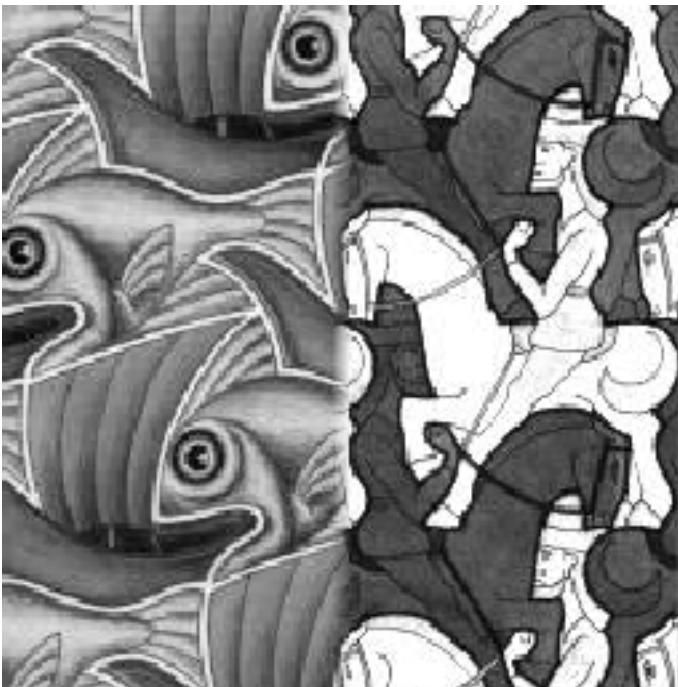


$$I_{\text{blend}} = \alpha I_{\text{left}} + (1-\alpha) I_{\text{right}}$$

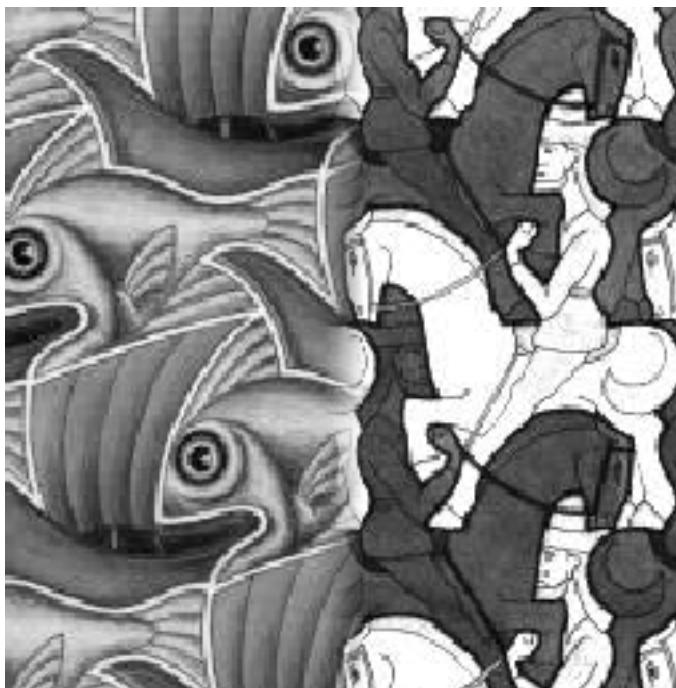
Affect of Window Size



Affect of Window Size



Good Window Size



“Optimal” Window: smooth but not ghosted

What is the Optimal Window?

To avoid seams

- window = size of largest prominent feature

To avoid ghosting

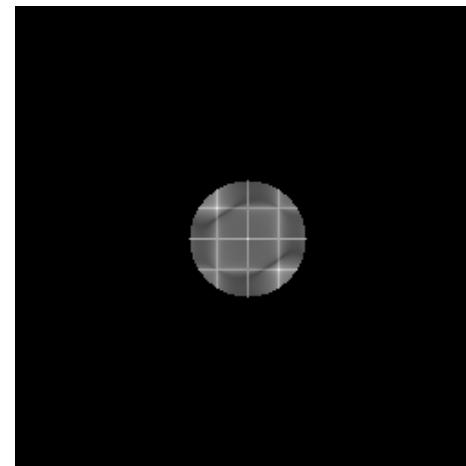
- window $\leq 2 \times$ size of smallest prominent feature

Natural to cast this in the *Fourier domain*

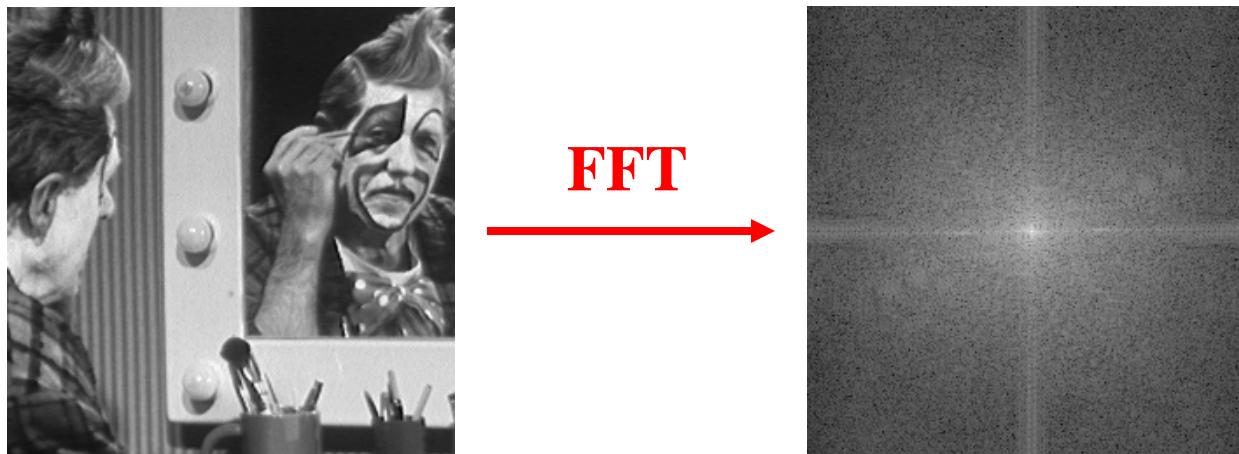
- largest frequency $\leq 2 \times$ size of smallest frequency
- image frequency content should occupy one “octave” (power of two)



FFT
→



What if the Frequency Spread is Wide



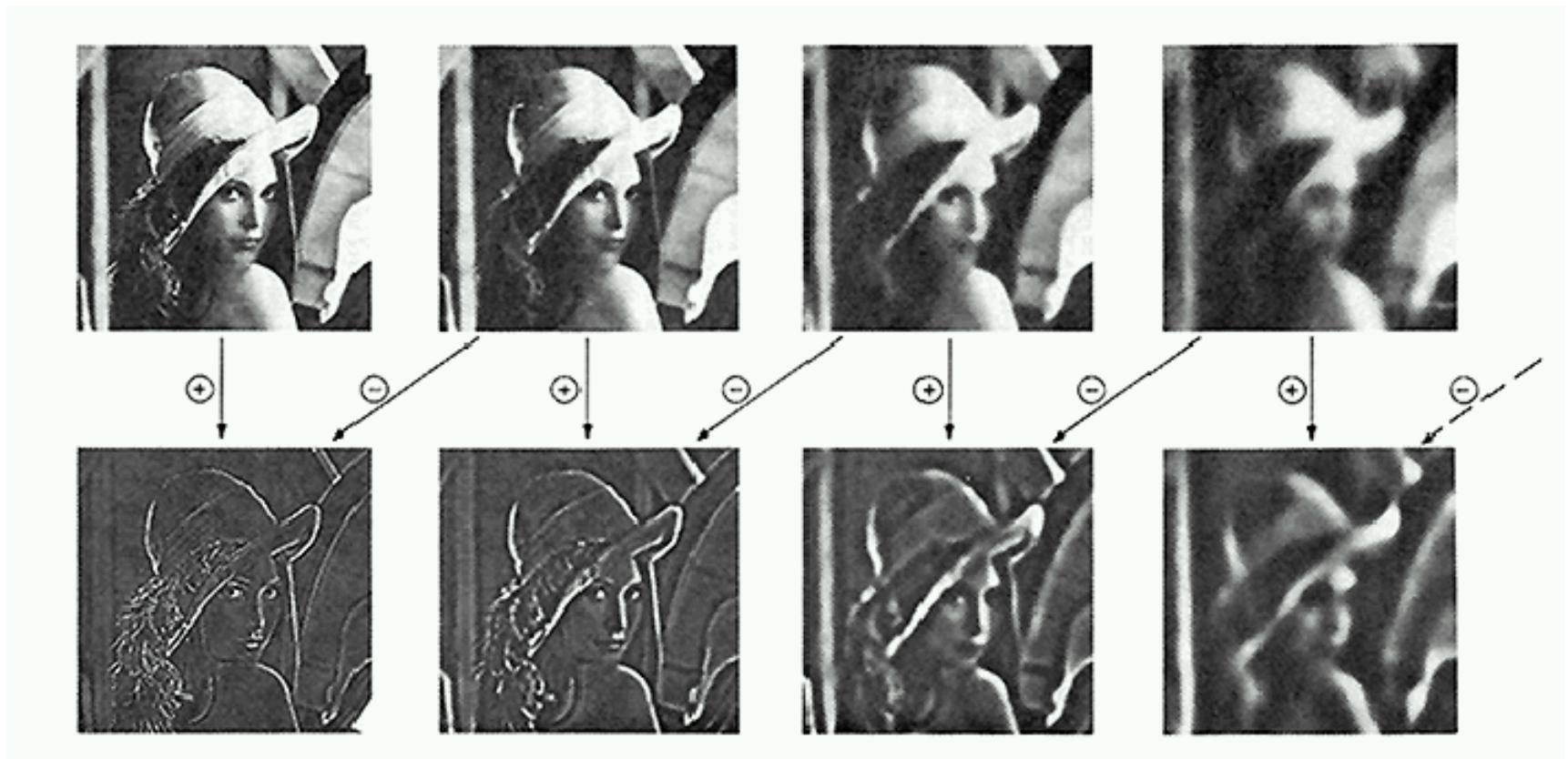
Idea (Burt and Adelson)

- Compute $F_{\text{left}} = \text{FFT}(I_{\text{left}})$, $F_{\text{right}} = \text{FFT}(I_{\text{right}})$
- Decompose Fourier image into octaves (bands)
 - $F_{\text{left}} = F_{\text{left}}^1 + F_{\text{left}}^2 + \dots$
- Feather corresponding octaves F_{left}^i with F_{right}^i
 - Can compute inverse FFT and feather in spatial domain
- Sum feathered octave images in frequency domain

Better implemented in *spatial domain*

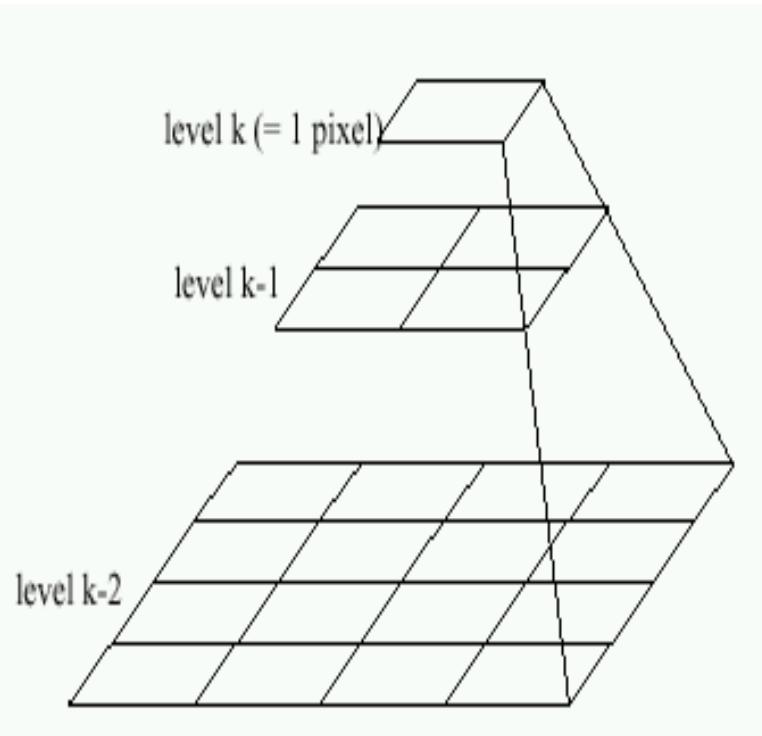
Octaves in the Spatial Domain

Lowpass Images

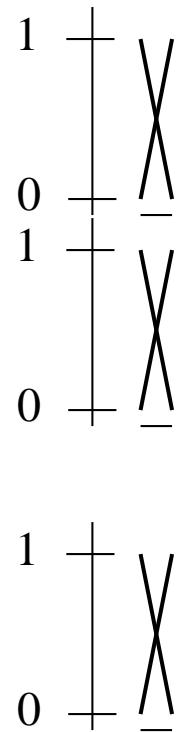


Bandpass Images

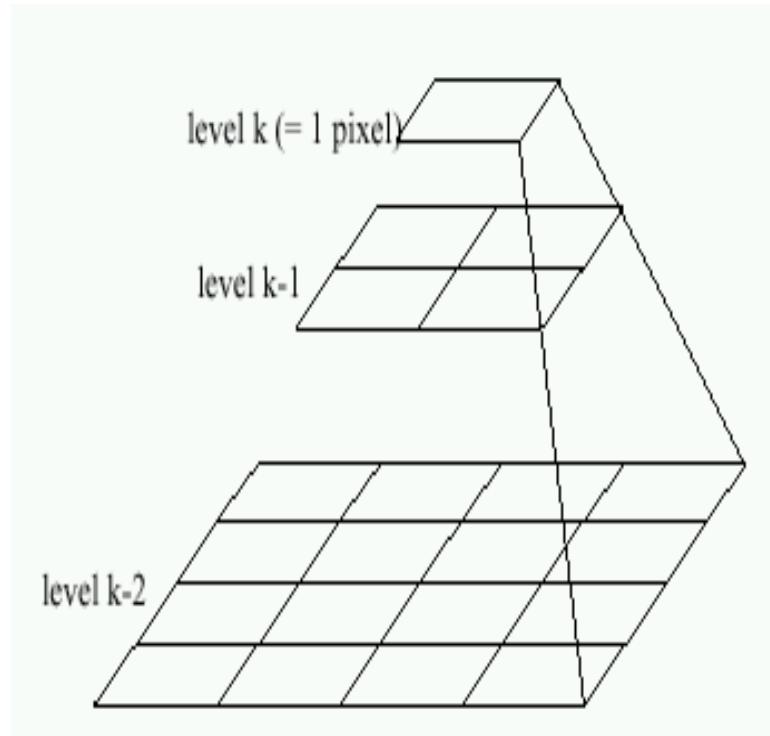
Pyramid Blending



Left pyramid

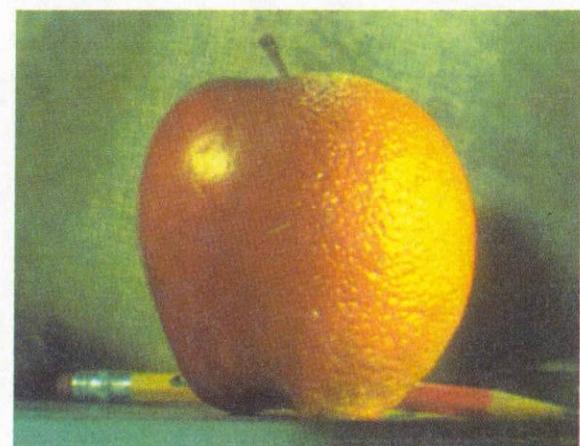
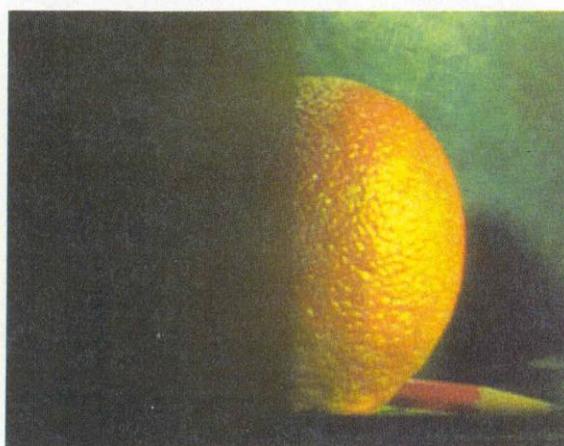
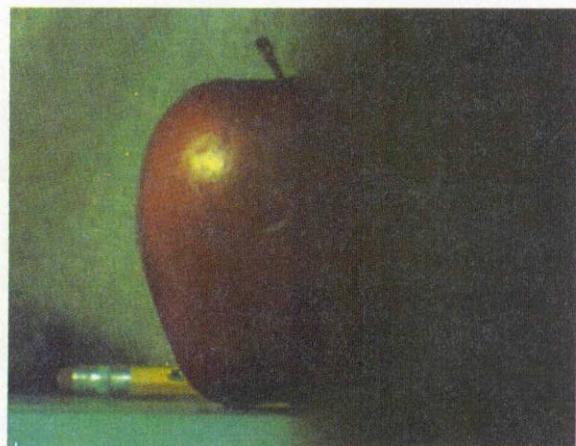
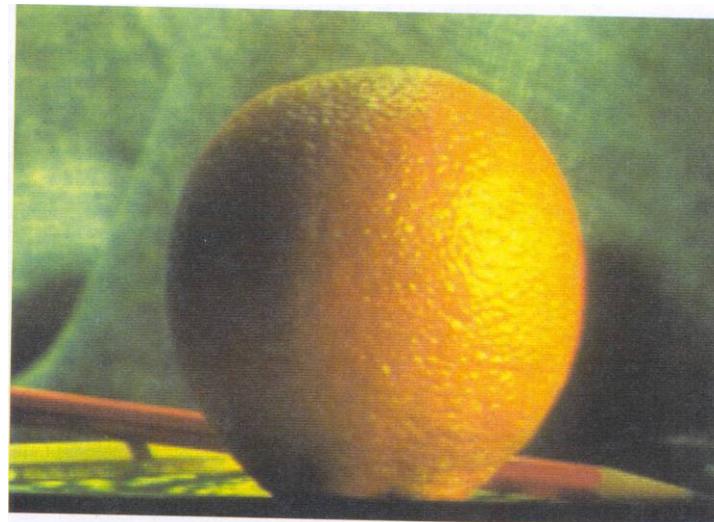
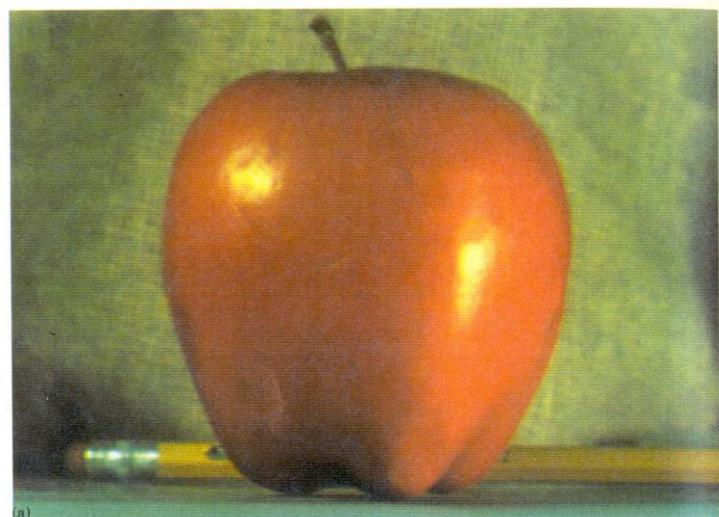


blend



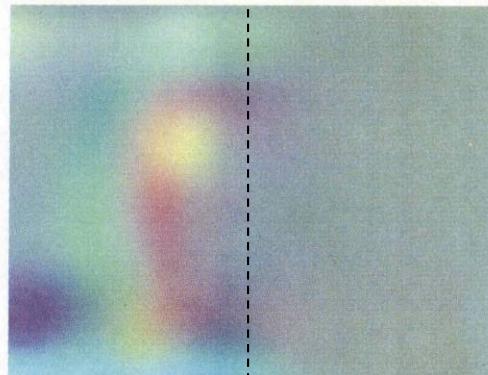
Right pyramid

Pyramid Blending

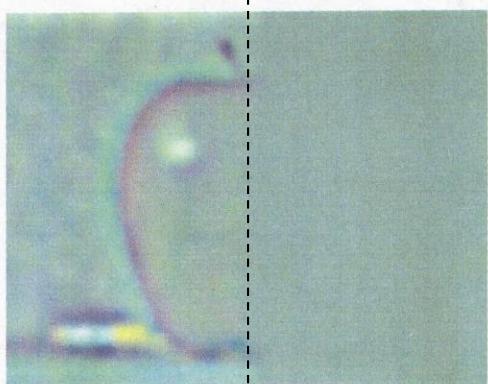


laplacian
level

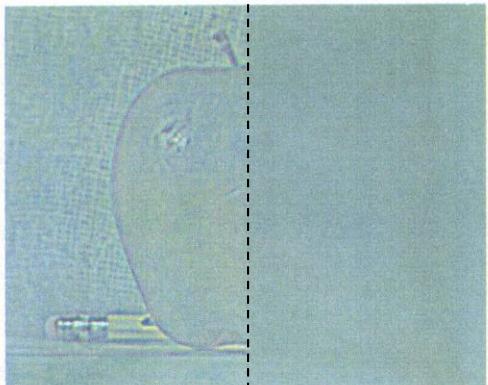
4



(c)

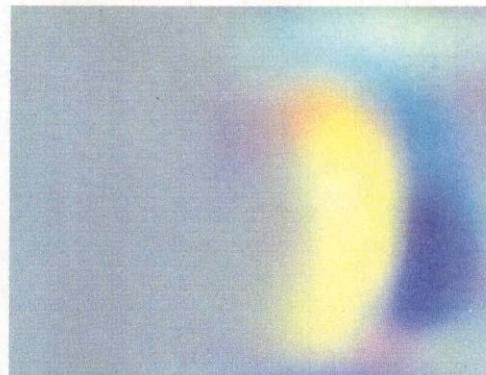


(b)

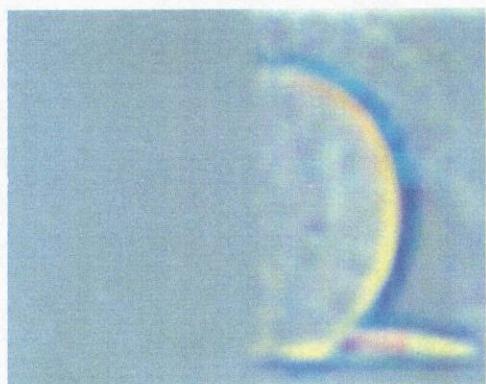


(a)

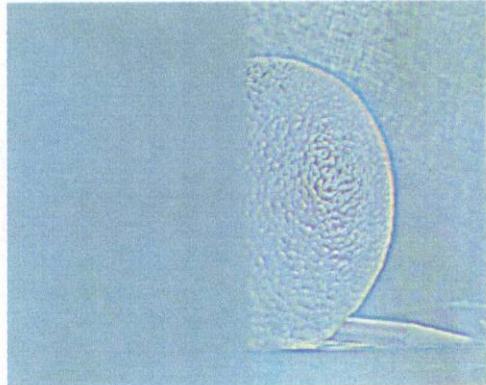
left pyramid



(g)

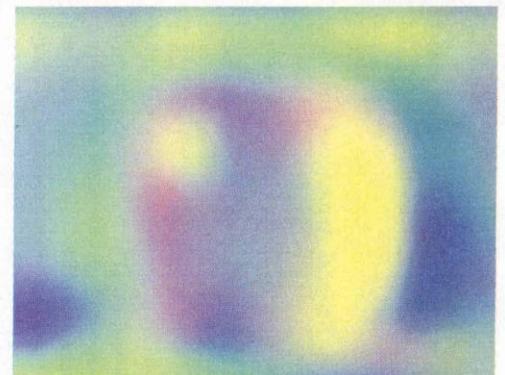


(f)

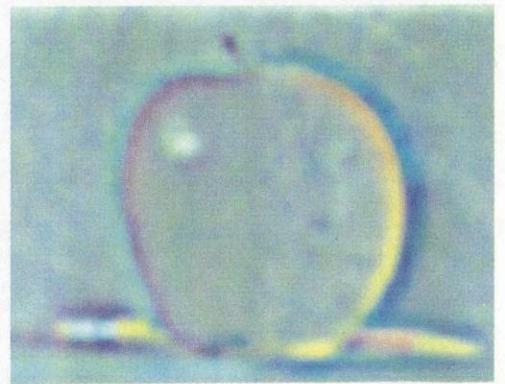


(e)

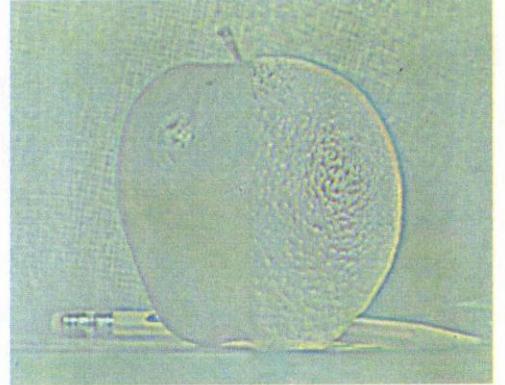
right pyramid



(k)



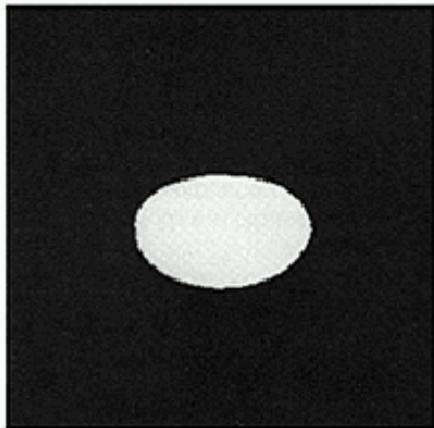
(j)



(i)

blended pyramid

Blending Regions

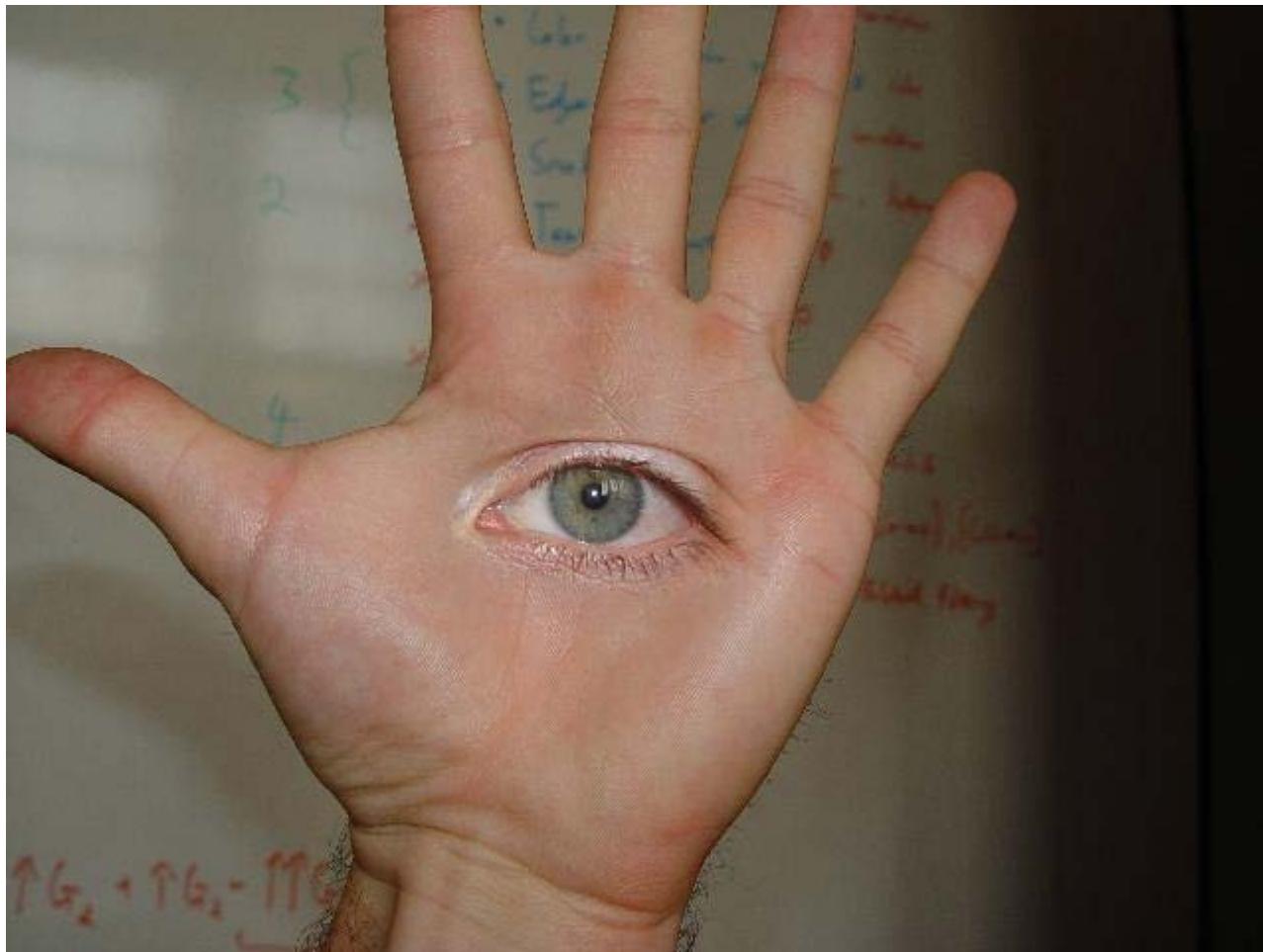


Laplacian Pyramid: Blending

General Approach:

1. Build Laplacian pyramids LA and LB from images A and B
2. Build a Gaussian pyramid GR from selected region R
3. Form a combined pyramid LS from LA and LB using nodes of GR as weights:
 - $LS(i,j) = GR(i,j) * LA(i,j) + (1 - GR(i,j)) * LB(i,j)$
4. Collapse the LS pyramid to get the final blended image

Horror Photo



© david dmartin (Boston College)

Results from this class (fall 2005)



© Chris Cameron

Simplification: Two-band Blending

Brown & Lowe, 2003

- Only use two bands: high freq. and low freq.
- Blends low freq. smoothly
- Blend high freq. with no smoothing: use binary alpha



2-band “Laplacian Stack” Blending



Low frequency ($\lambda > 2$ pixels)



High frequency ($\lambda < 2$ pixels)

Linear Blending



2-band Blending

