BBM 413 Fundamentals of Image Processing

Nov. 6, 2012

Erkut Erdem
Dept. of Computer Engineering
Hacettepe University

Frequency Domain Techniques

Review - Point Operations

- Smallest possible neighborhood is of size 1x1
- Process each point independently of the others
- Output image *g* depends only on the value of *f* at a single point (x,y)
- Transformation function *T* remaps the sample's value:

$$s = T(r)$$

where

- *r* is the value at the point in question
- *s* is the new value in the processed result
- *T* is a *intensity transformation* function

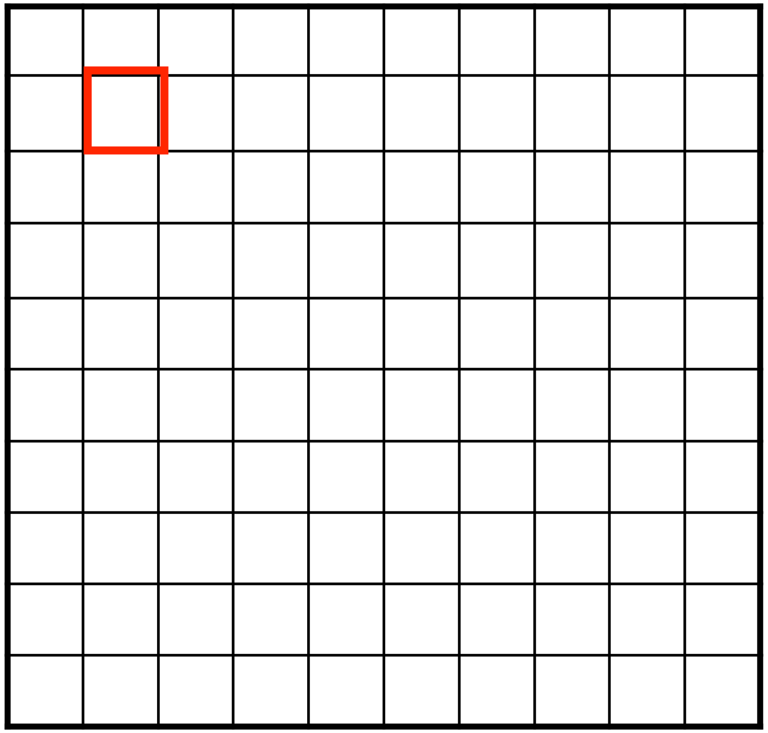
Review – Spatial Filtering

$$g[\cdot\,,\cdot\,]\ \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$



$$h[m,n]=\sum_{k,l} g[k,l]\, f[m+k,n+l]$$

Slide credit: S. Seitz

Review – Spatial Filtering

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10							

$$h[m,n] = \sum_{k,l} g[k,l]\, f[m+k, n+l]$$

Slide credit: S. Seitz

Review – Spatial Filtering

$g[\cdot\,,\cdot\,]\ \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10	20						

$$h[m,n] = \sum_{k,l} g[k,l]\, f[m+k,n+l]$$

Slide credit: S. Seitz

Review – Spatial Filtering

$$g[\cdot\,,\cdot\,]\ \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10	20	30					

$$h[m,n]=\sum_{k,l} g[k,l]\, f[m+k,n+l]$$

Slide credit: S. Seitz

Review – Spatial Filtering

$$g[\cdot,\cdot]\ \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10	20	30	30				

$$h[m,n] = \sum_{k,l} g[k,l]\, f[m+k,n+l]$$

Slide credit: S. Seitz

Review – Spatial Filtering

$$g[\cdot\,,\cdot\,]\ \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$$f[.,.]$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

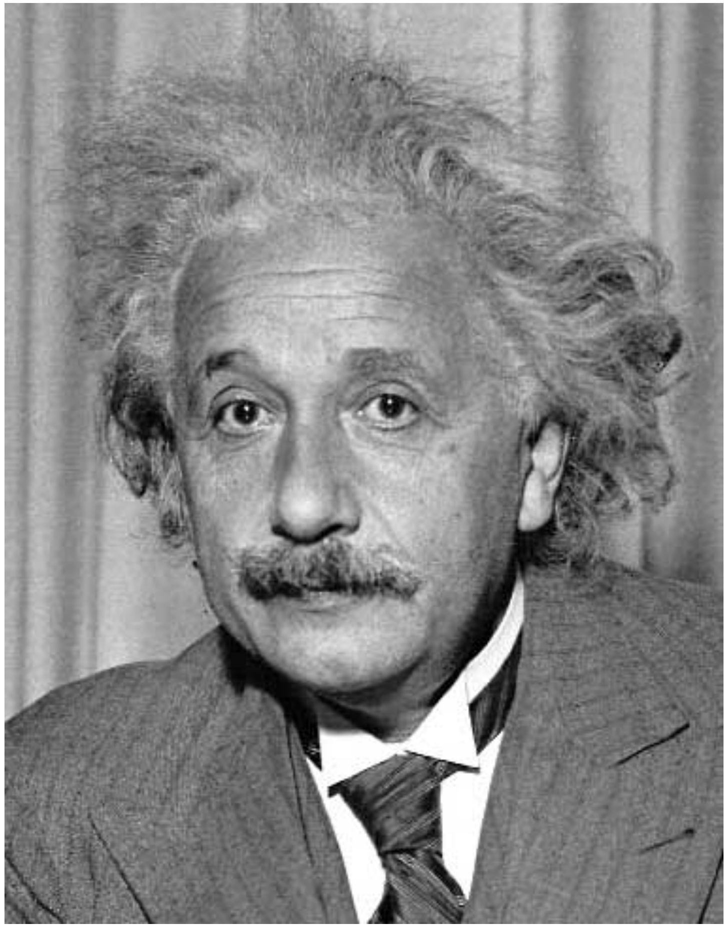
$$h[.,.]$$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

$$h[m,n] = \sum_{k,l} g[k,l]\, f[m+k, n+l]$$

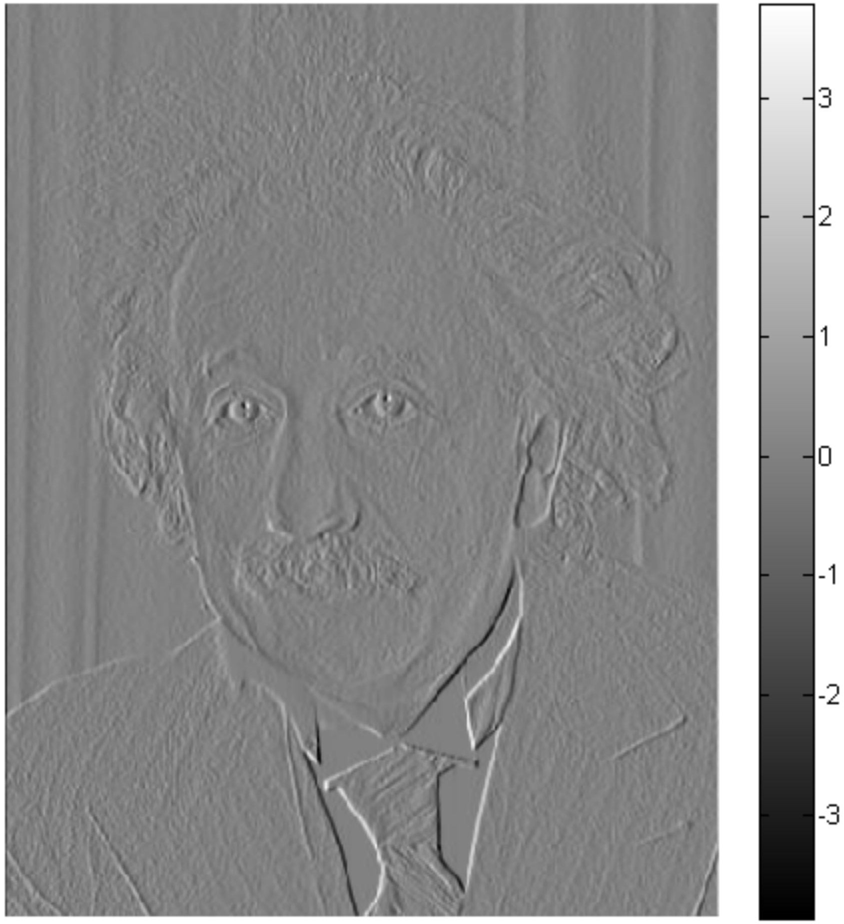
Slide credit: S. Seitz

Review – Spatial Filtering



1	0	-1
2	0	-2
1	0	-1

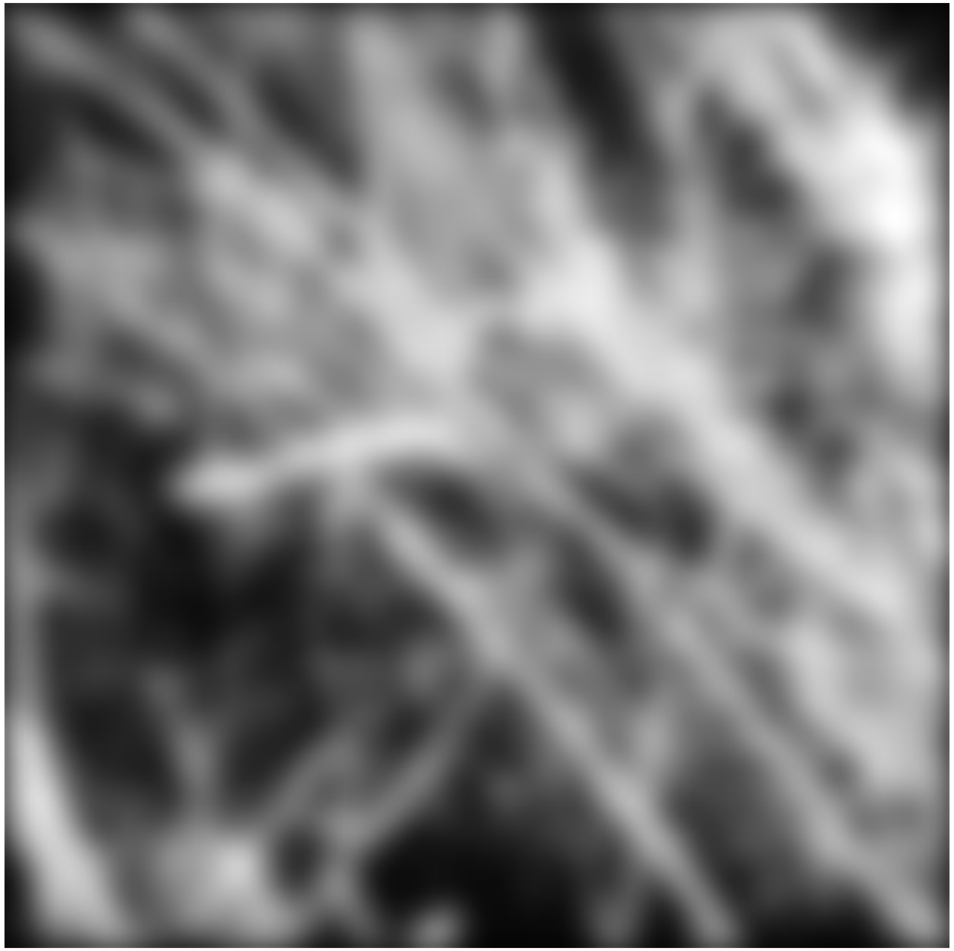
Sobel



Slide credit: J. Hays

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

Gaussian 



Box filter 

Why does a lower resolution image still make sense to us? What do we lose?

Image: http://www.flickr.com/photos/igorms/136916757/

Slide credit: D. Hoiem

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.*

- 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
  - there are some subtle restrictions

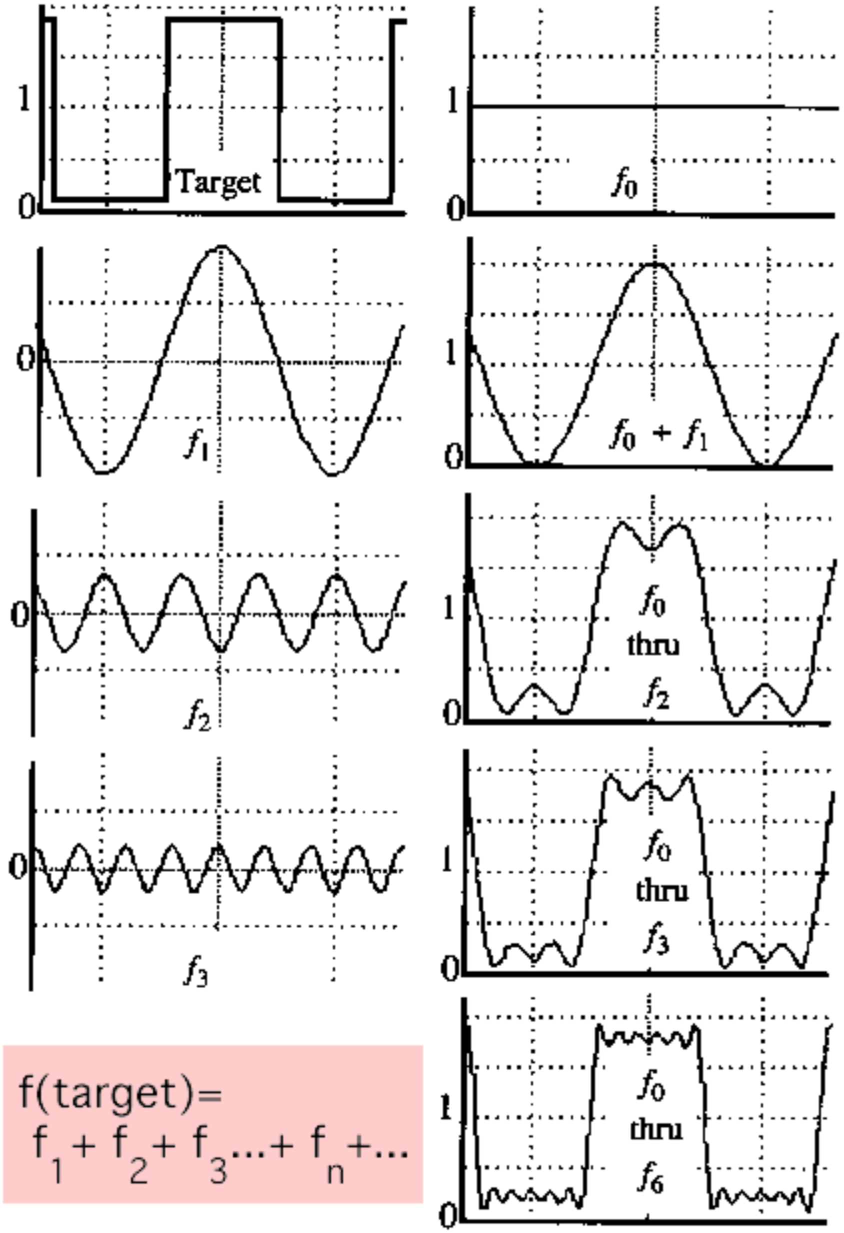
Slide credit: A. Efros

A sum of sines

Our building block:

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

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



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

Slide credit: A. Efros

Fourier Transform

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

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

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

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

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

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

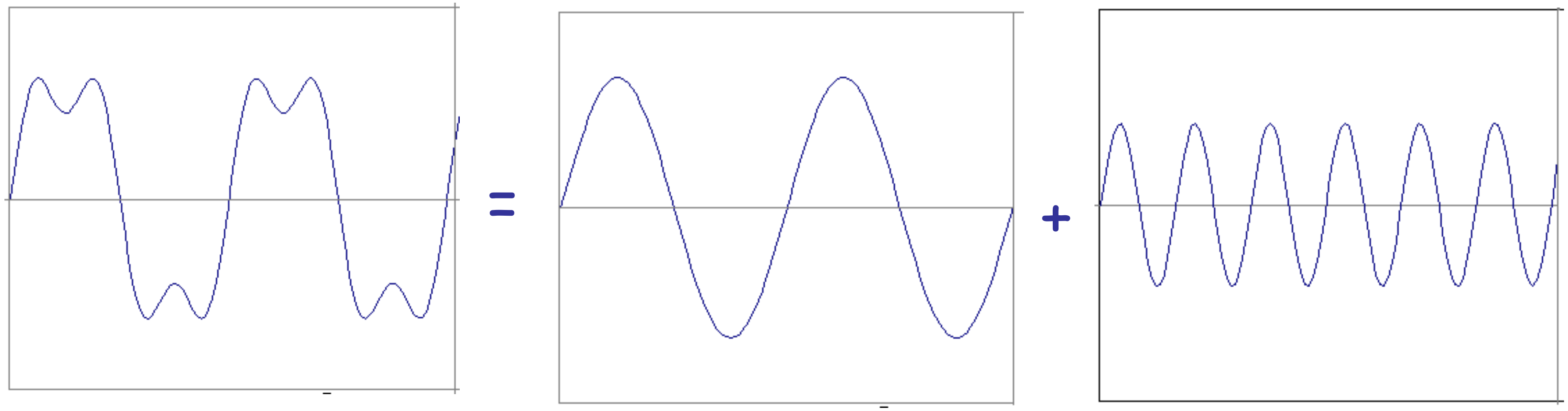
We can always go back:

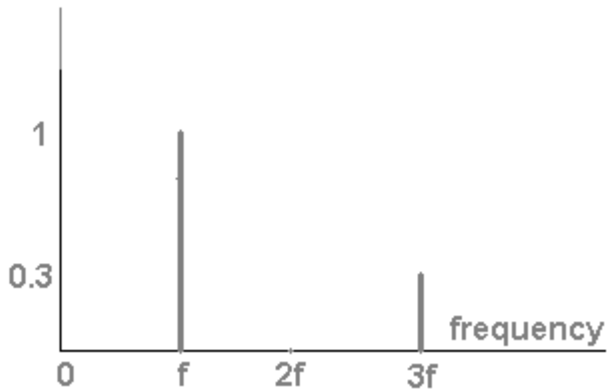
$$F(w) \longrightarrow \boxed{\text{Inverse Fourier Transform}} \longrightarrow f(x)$$

Slide credit: A. Efros

Frequency Spectra

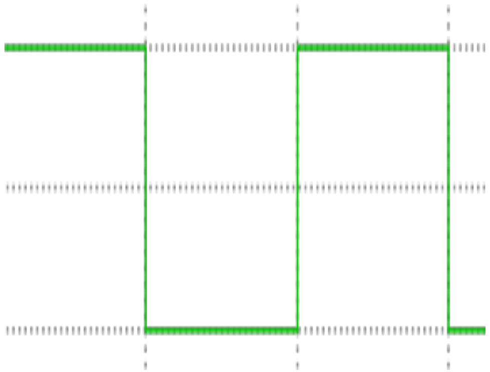
- example: $g(t) = \sin(2\pi f\, t) + (1/3)\sin(2\pi(3f)\, t)$





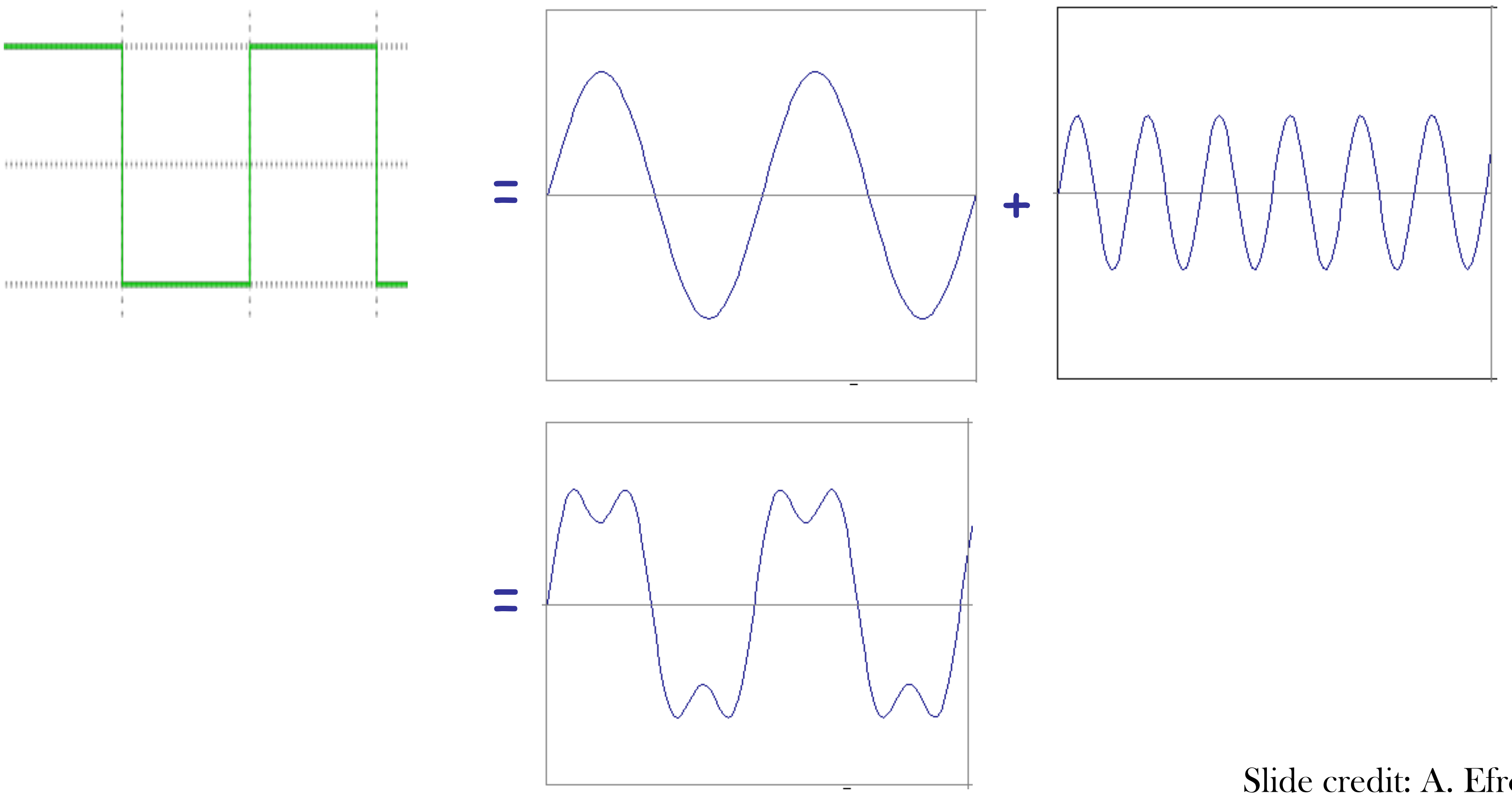
Slide credit: A. Efros

Frequency Spectra



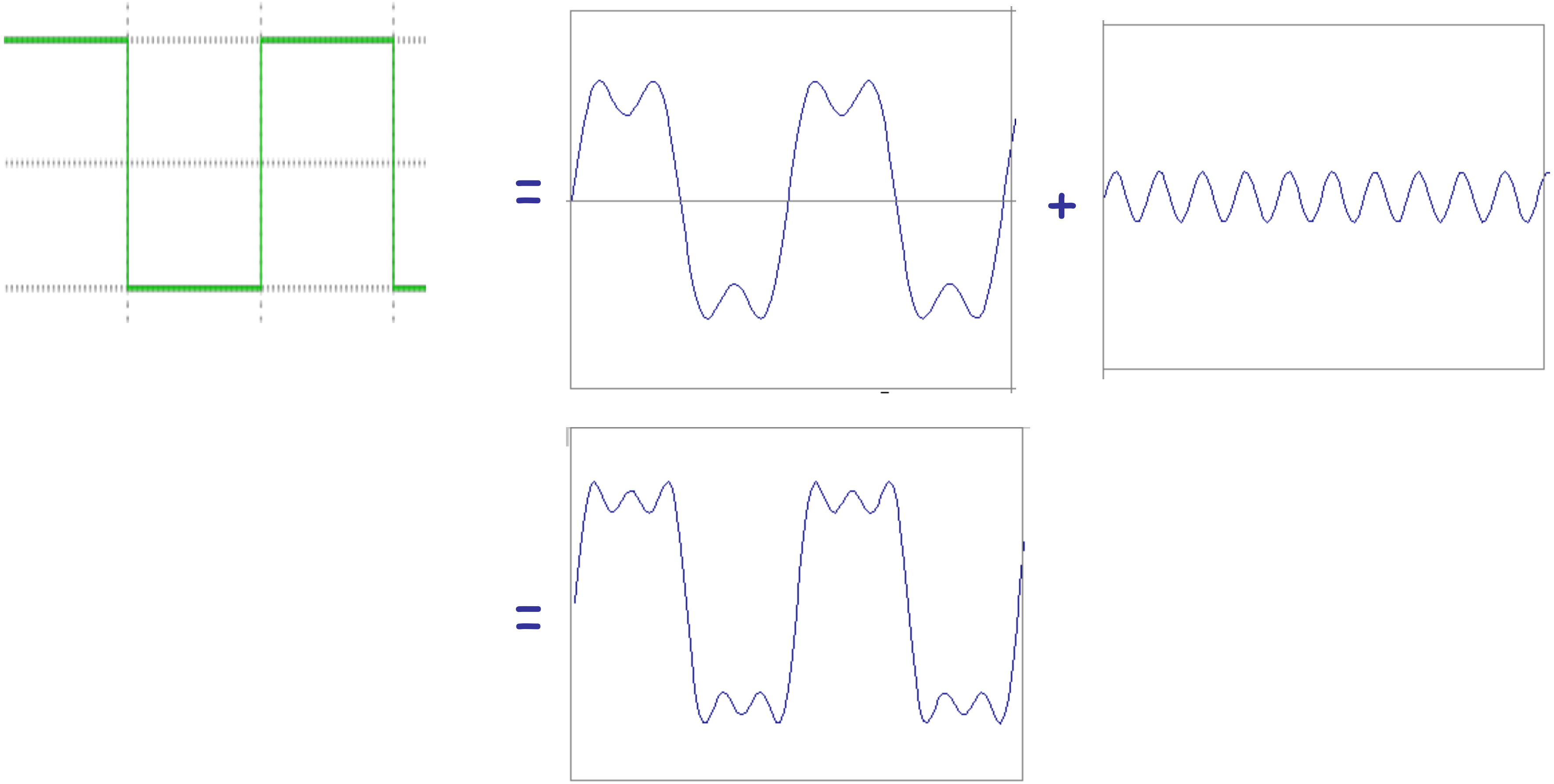
Slide credit: A. Efros

Frequency Spectra



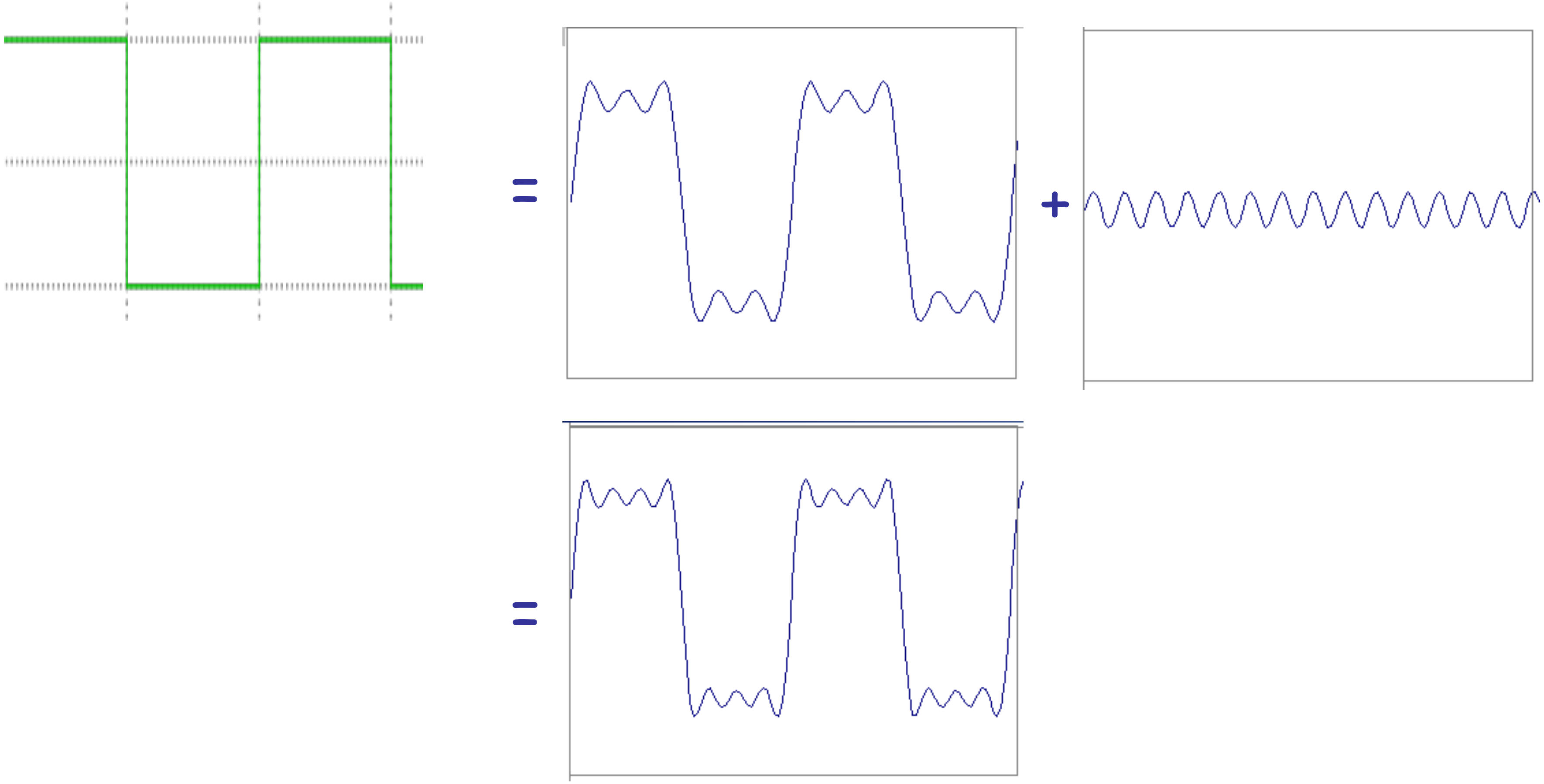
Slide credit: A. Efros

Frequency Spectra



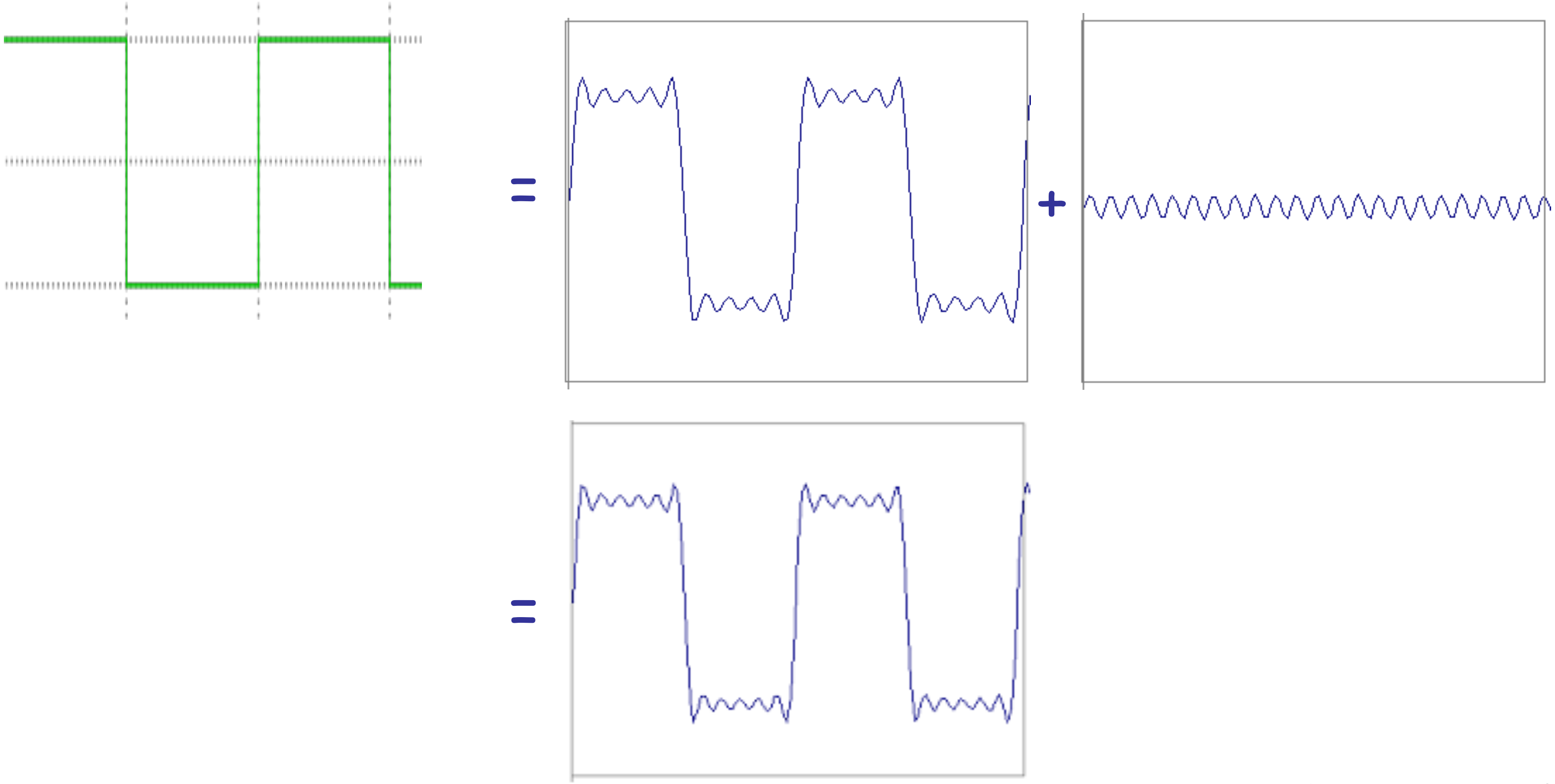
Slide credit: A. Efros

Frequency Spectra



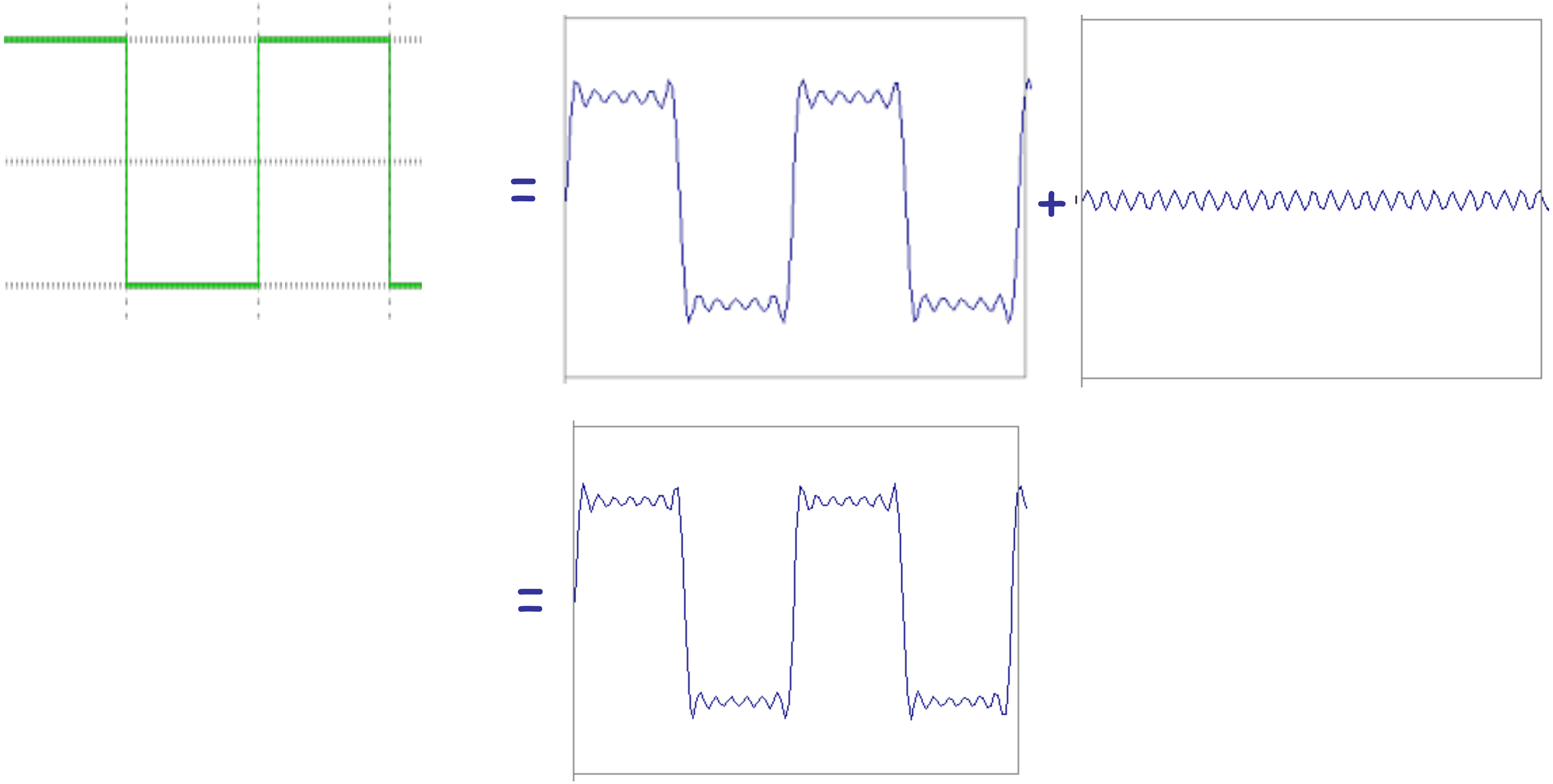
Slide credit: A. Efros

Frequency Spectra



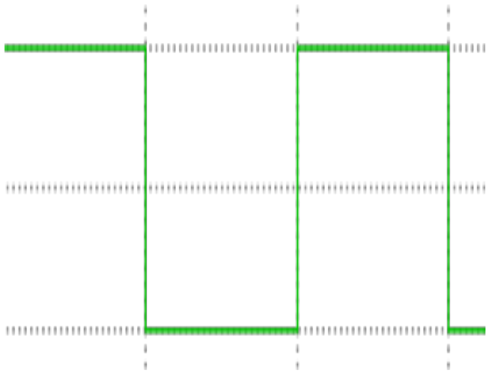
Slide credit: A. Efros

Frequency Spectra

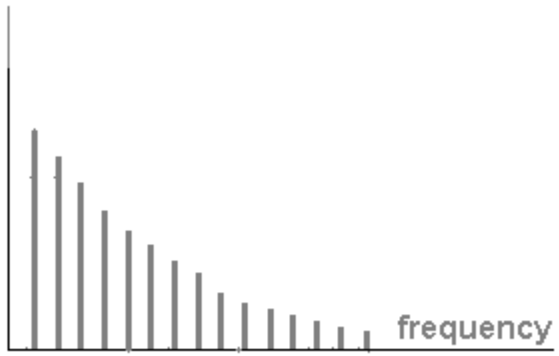


Slide credit: A. Efros

Frequency Spectra

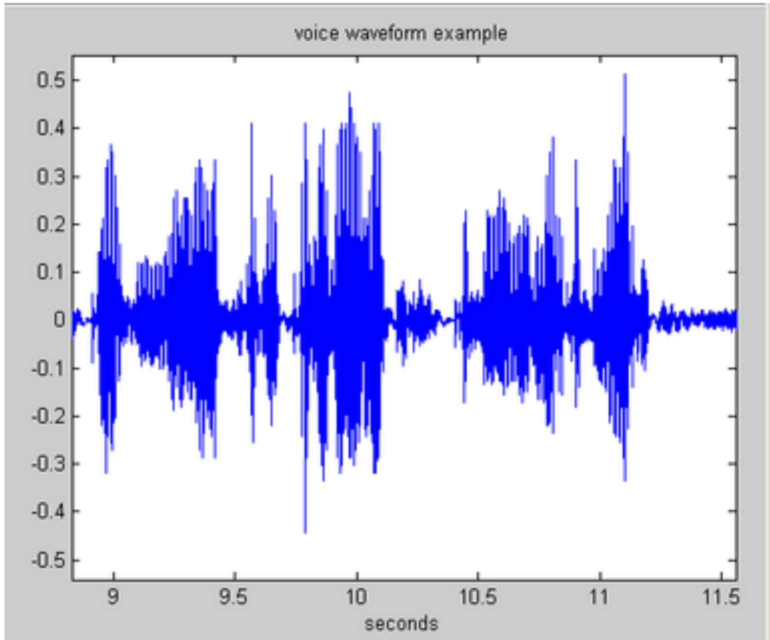


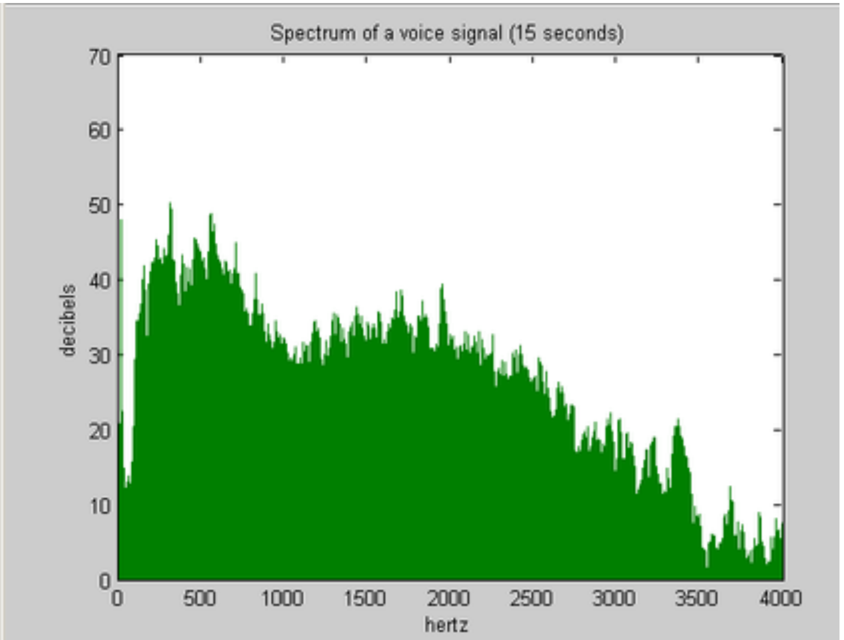
$$= \boxed{A\sum_{k=1}^{\infty}\frac{1}{k}\sin(2\pi kt)}$$



Example: Music

- We think of music in terms of frequencies at different magnitudes





Slide credit: D . Hoeim

The Discrete Fourier transform

- Forward transform

$$F[m,n] = \sum_{k=0}^{M-1}\sum_{l=0}^{N-1} f[k,l]\, e^{-\pi i\left(\frac{km}{M}+\frac{\mathrm{ln}}{N}\right)}$$

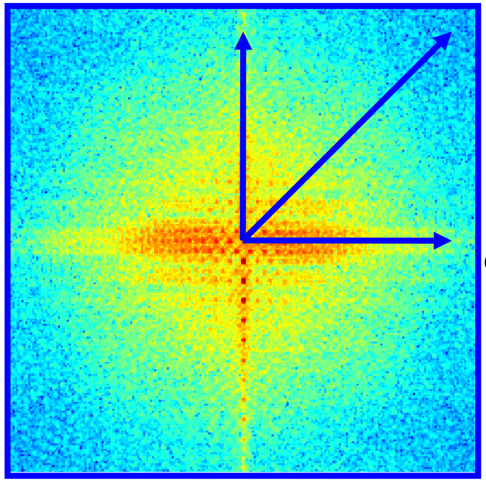
- Inverse transform

$$f[k,l] = \frac{1}{MN}\sum_{k=0}^{M-1}\sum_{l=0}^{N-1} F[m,n]\, e^{+\pi i\left(\frac{km}{M}+\frac{\mathrm{ln}}{N}\right)}$$

Slide credit: B. Freeman and A. Torralba

How to interpret 2D Fourier Spectrum

Vertical orientation



45 deg.

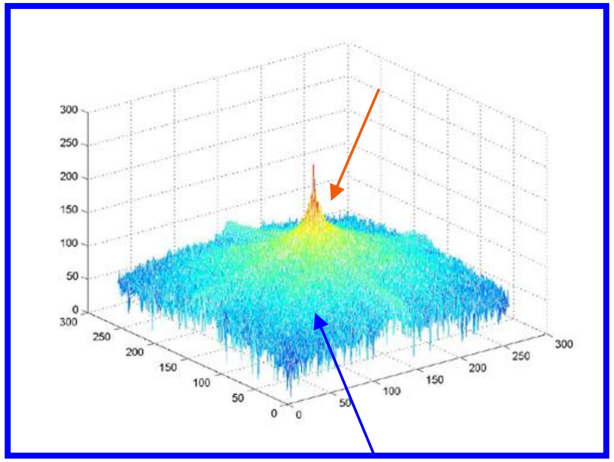
Horizontal orientation

0

0 f_{max}

fx in cycles/image

Low spatial frequencies

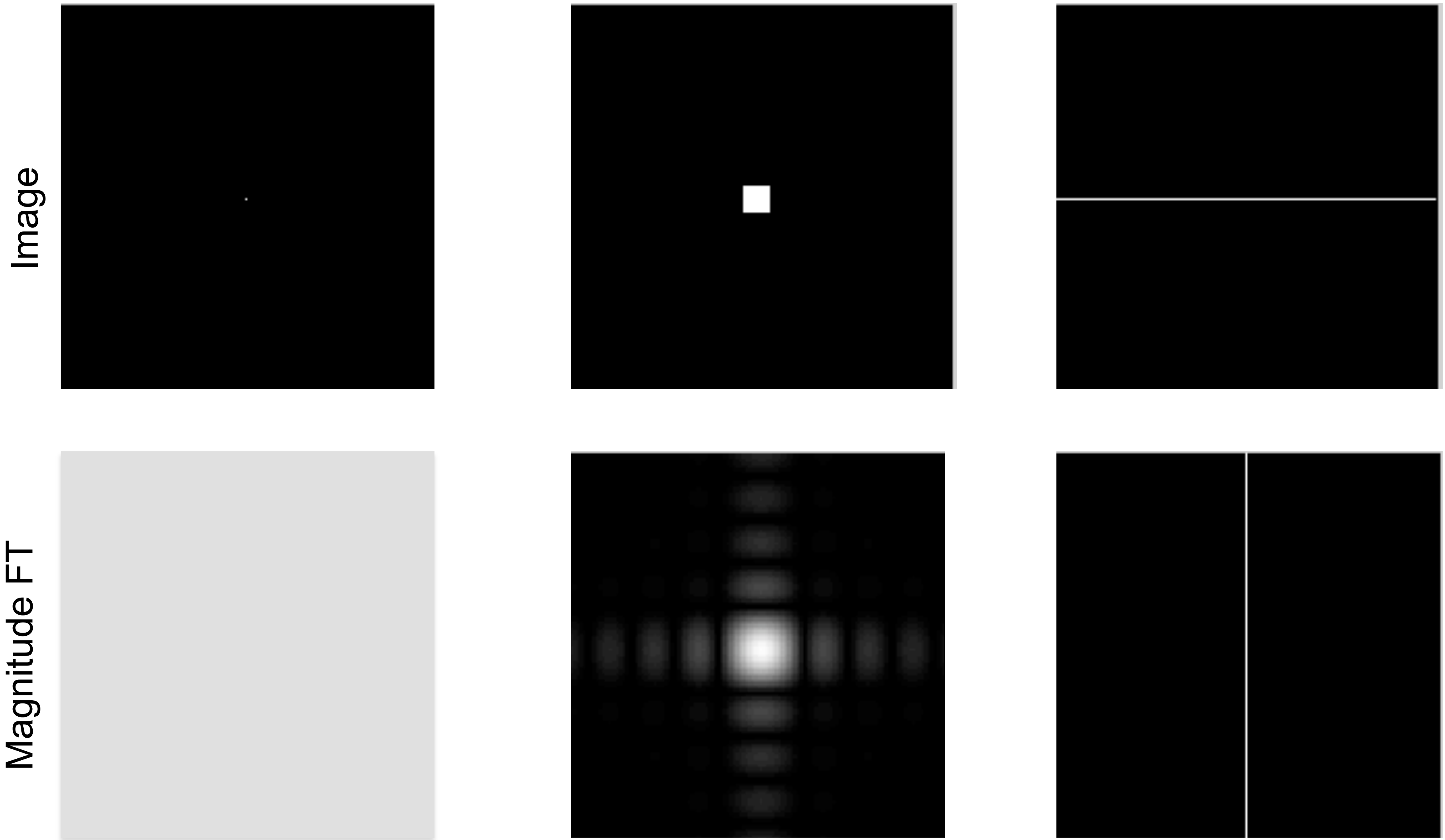


High spatial frequencies

Log power spectrum

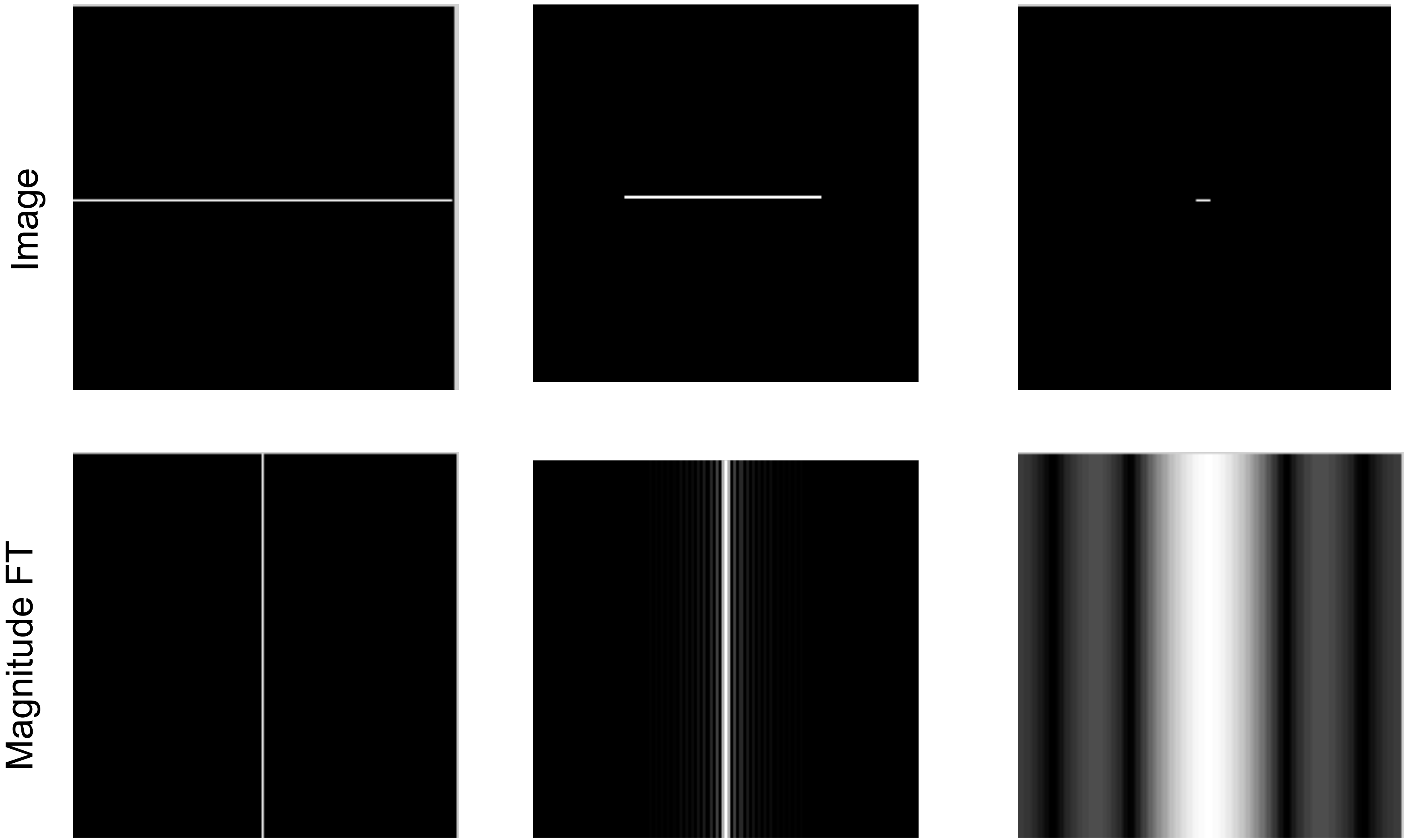
Slide credit: B. Freeman and A. Torralba

Some important Fourier Transforms



Slide credit: B. Freeman and A. Torralba

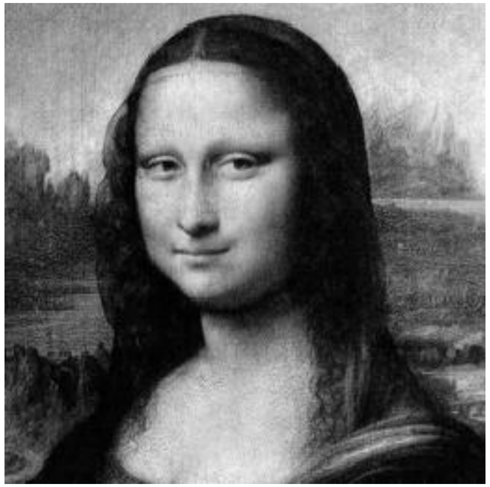
Some important Fourier Transforms



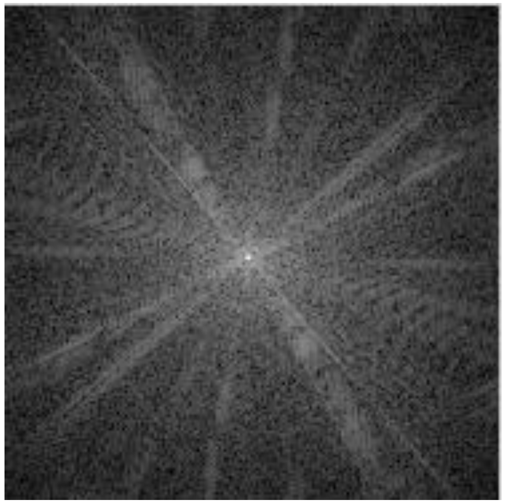
Slide credit: B. Freeman and A. Torralba

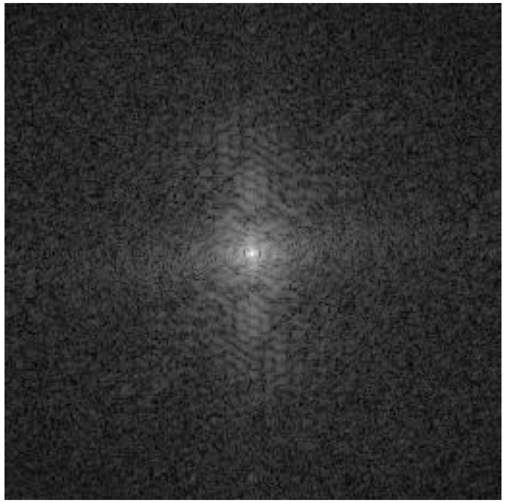
The Fourier Transform of some important images

Image



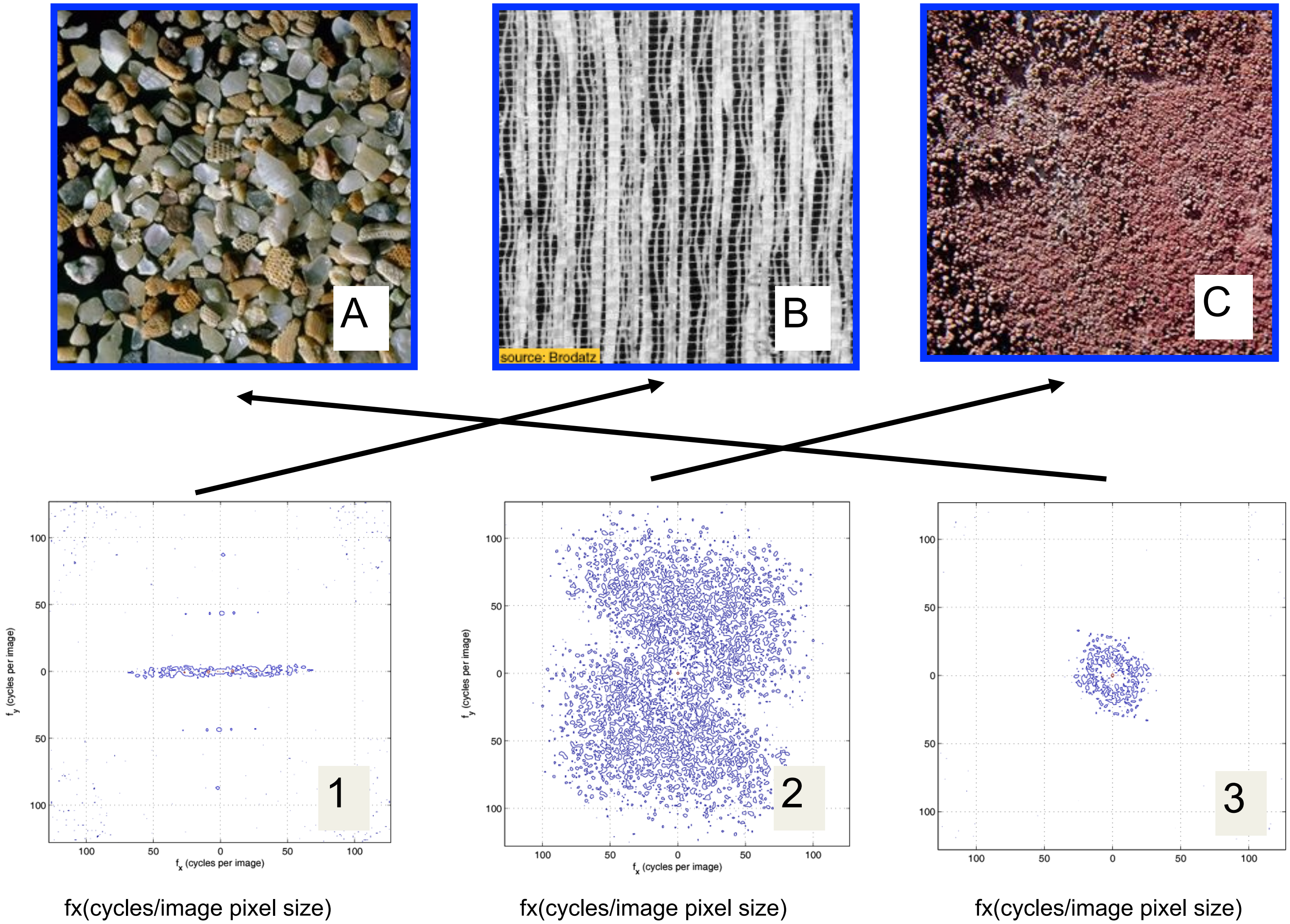
Log(1+Magnitude FT)





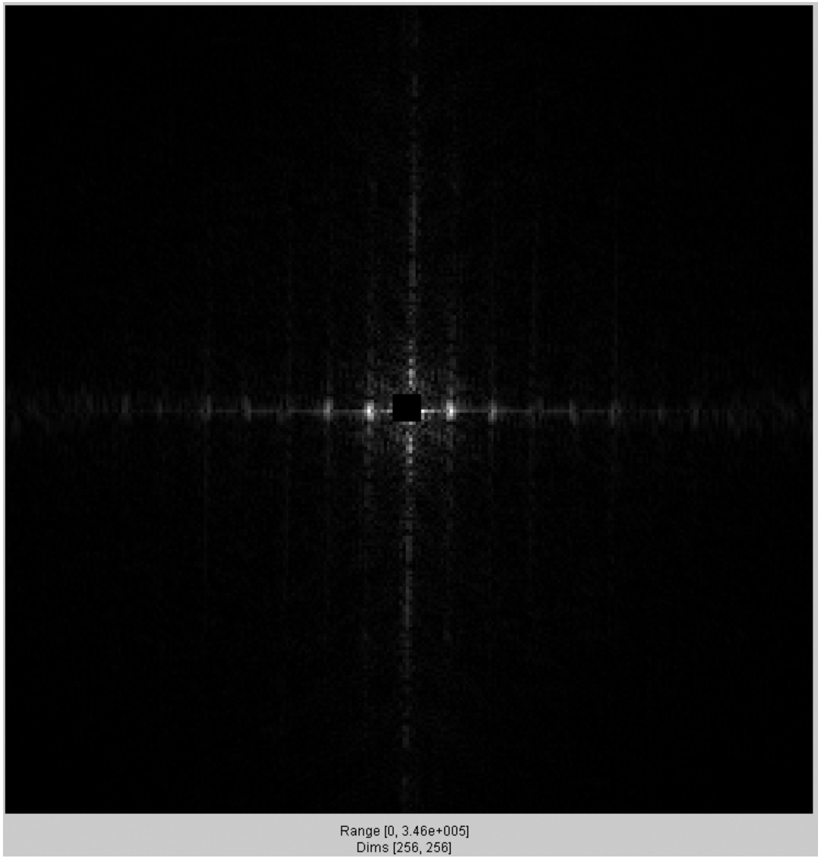
Slide credit: B. Freeman and A. Torralba

Fourier Amplitude Spectrum



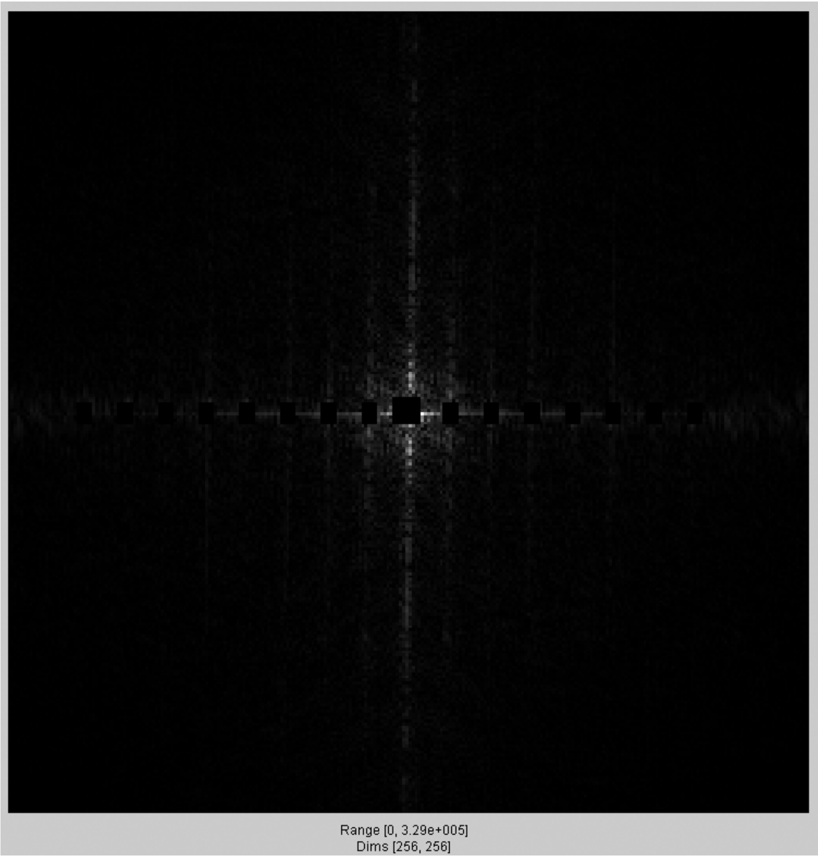
Slide credit: B. Freeman and A. Torralba

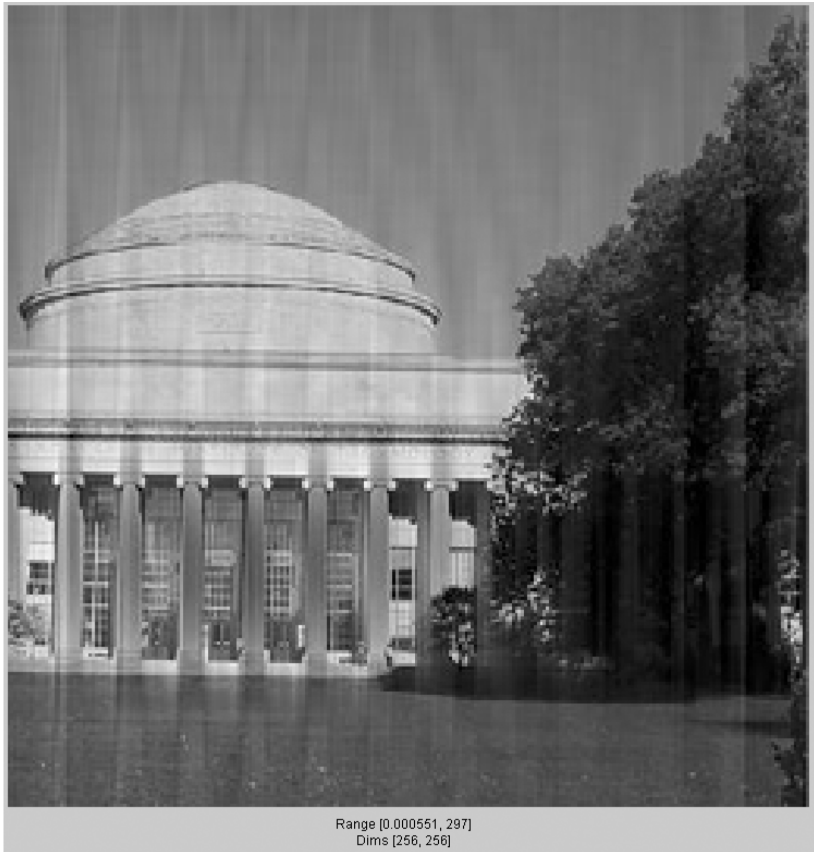
Fourier transform magnitude



Slide credit: B. Freeman and A. Torralba

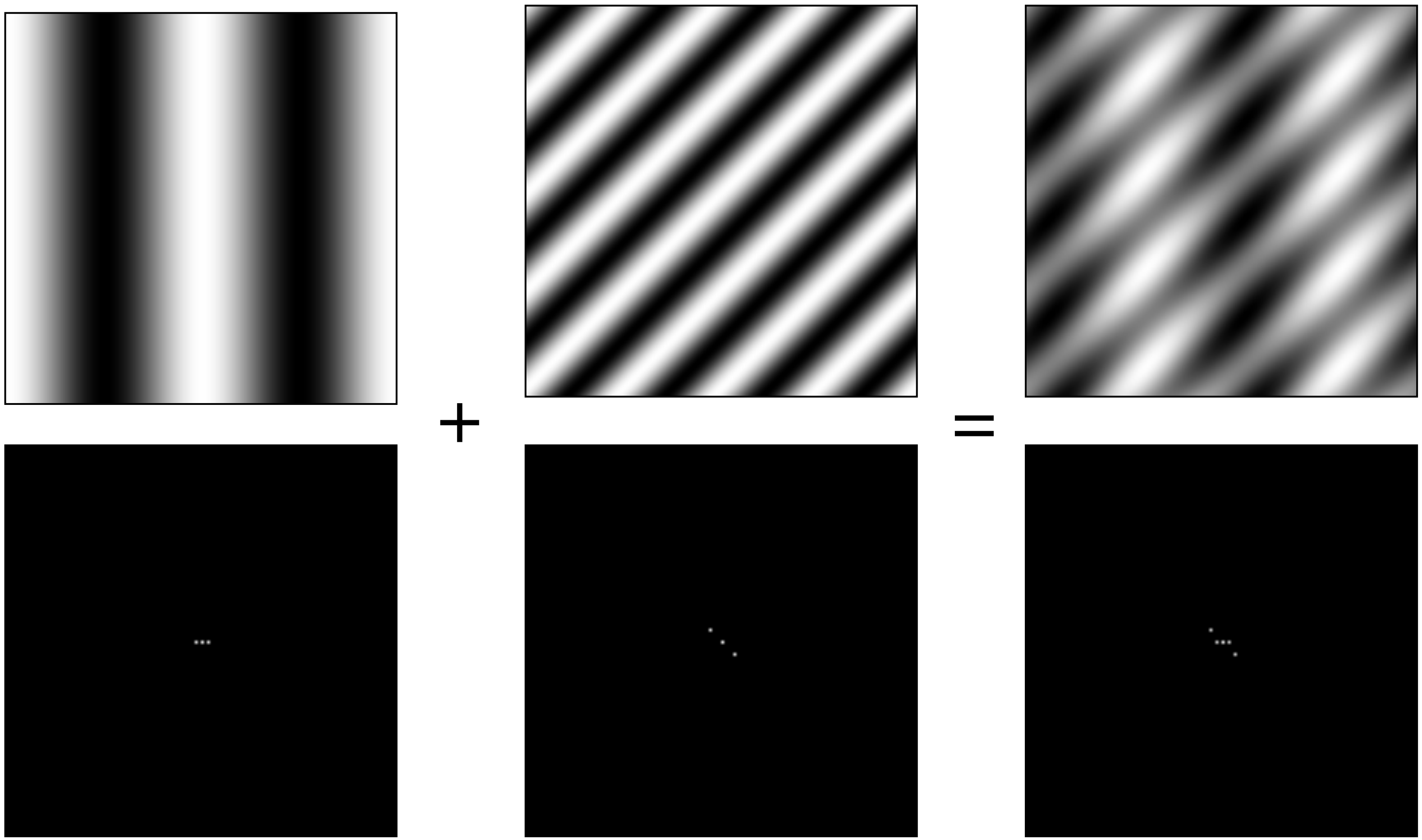
Masking out the fundamental and harmonics from periodic pillars





Slide credit: B. Freeman and A. Torralba

Signals can be composed



http://sharp.bu.edu/~slehar/fourier/fourier.html#filtering
More: http://www.cs.unm.edu/~brayer/vision/fourier.html

Slide credit: A. Efros

The Convolution Theorem

- 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

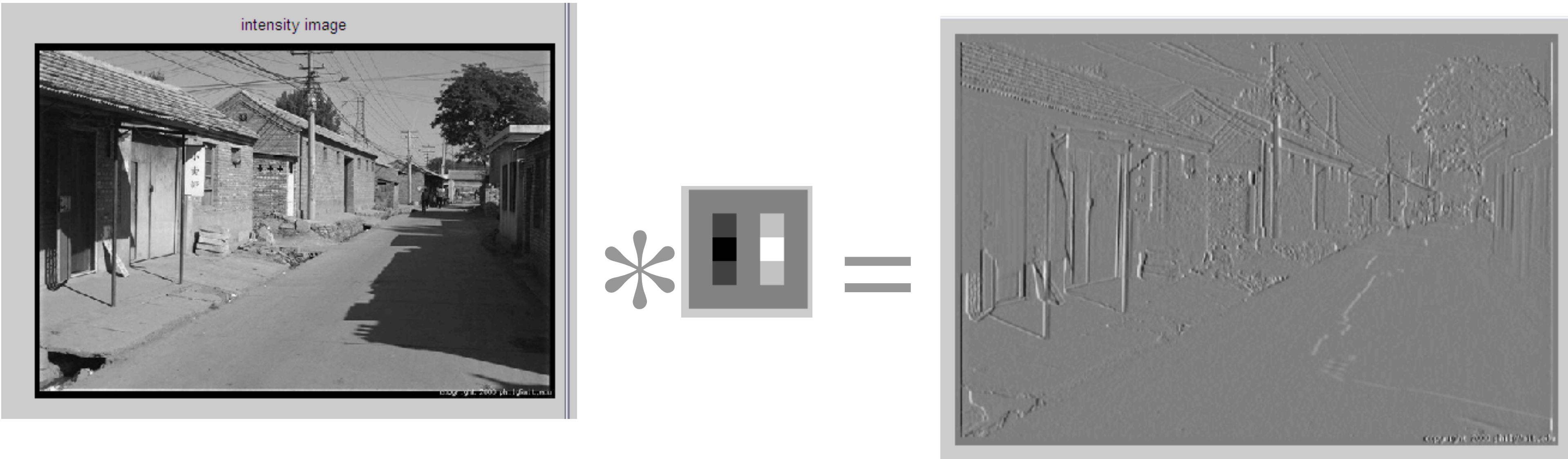
$$\mathrm{F}^{-1}[gh] = \mathrm{F}^{-1}[g] * \mathrm{F}^{-1}[h]$$

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

Slide credit: A. Efros

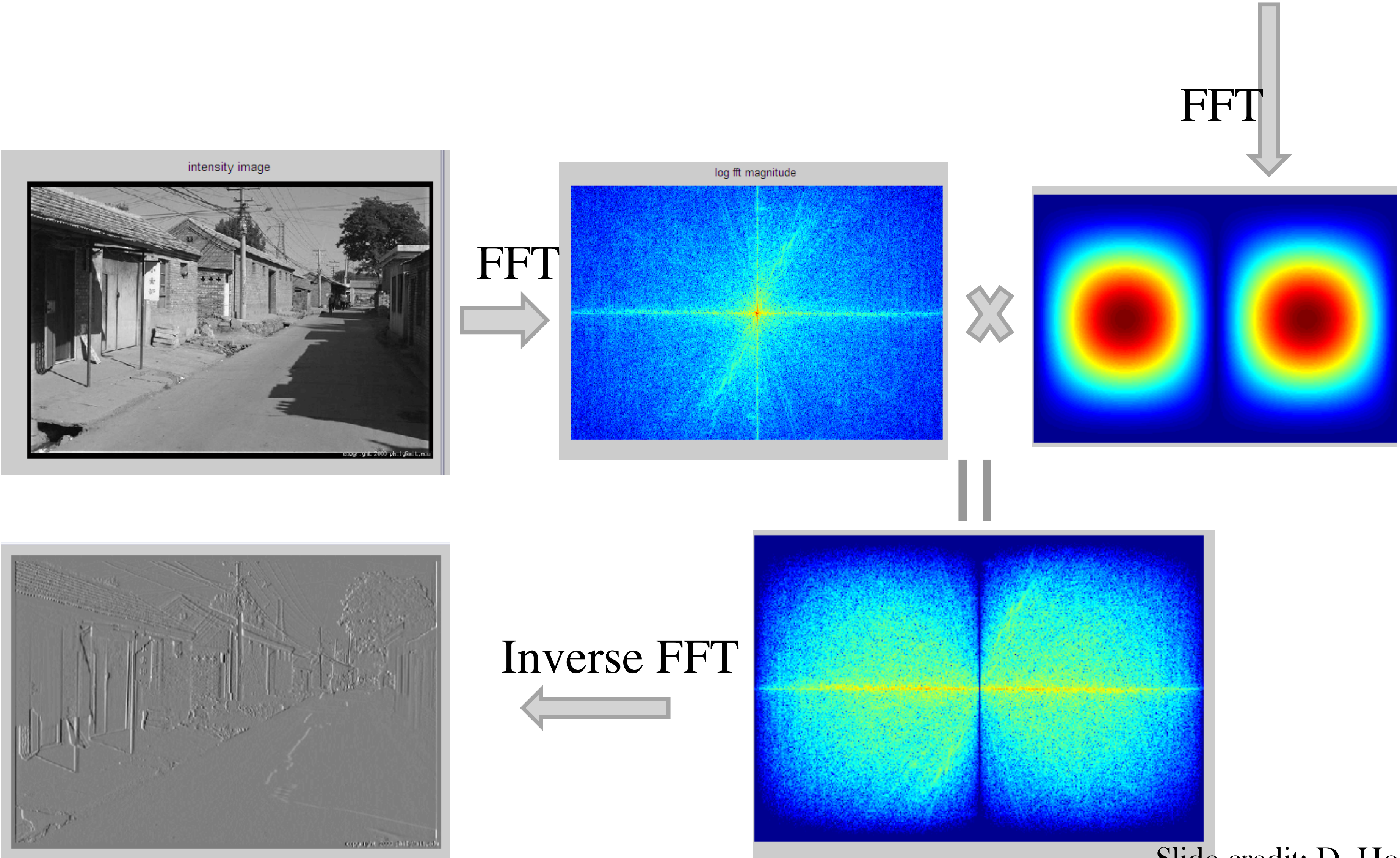
Filtering in spatial domain

1	0	-1
2	0	-2
1	0	-1



Slide credit: D. Hoiem

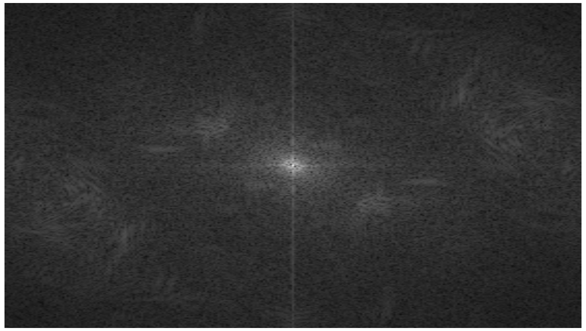
Filtering in frequency domain



Slide credit: D. Hoiem

2D convolution theorem example

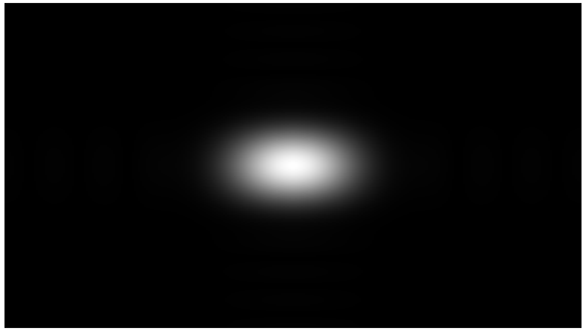
$f(x,y)$



$|F(s_x,s_y)|$

$*$ $\times$

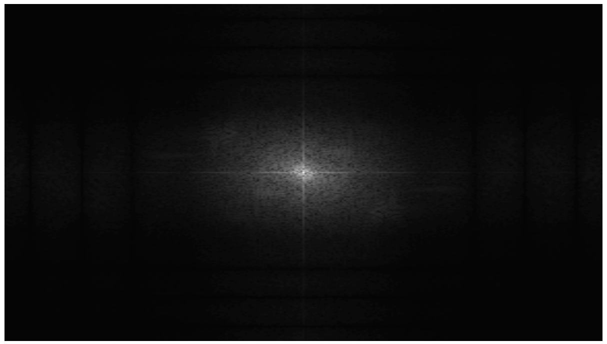
$h(x,y)$



$|H(s_x,s_y)|$

$\Downarrow$ $\Downarrow$

$g(x,y)$



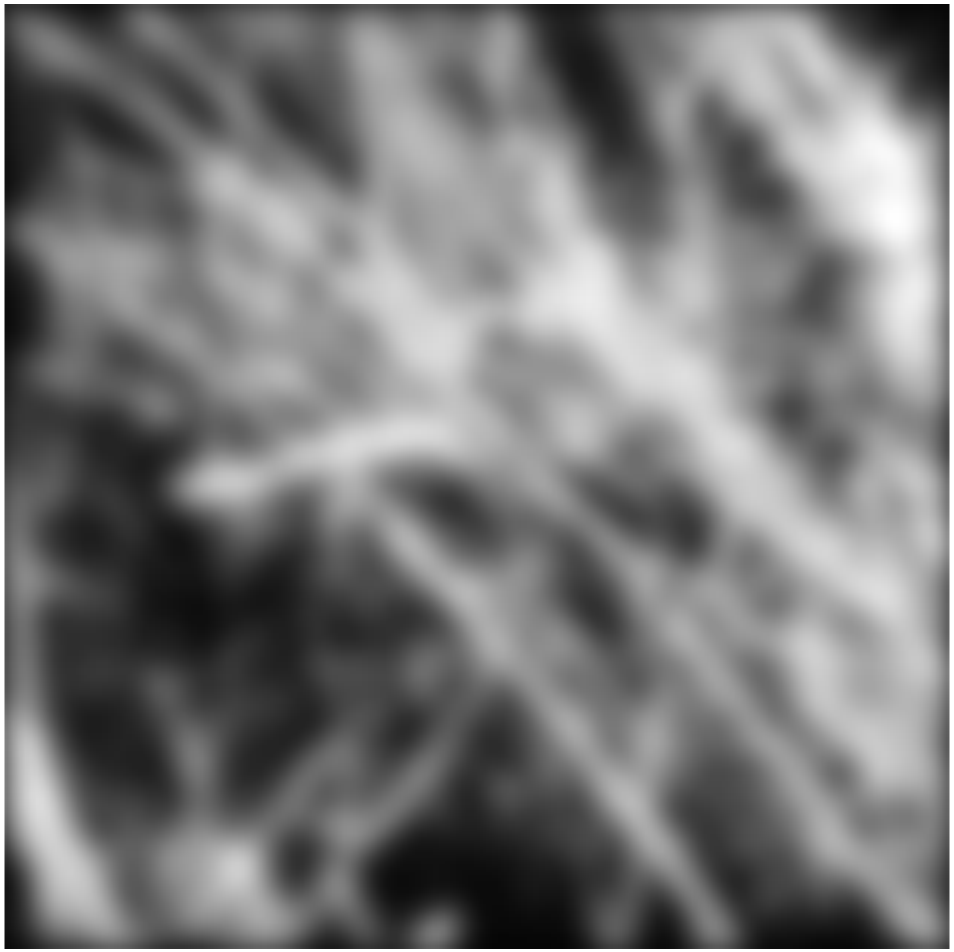
$|G(s_x,s_y)|$

Slide credit: A. Efros

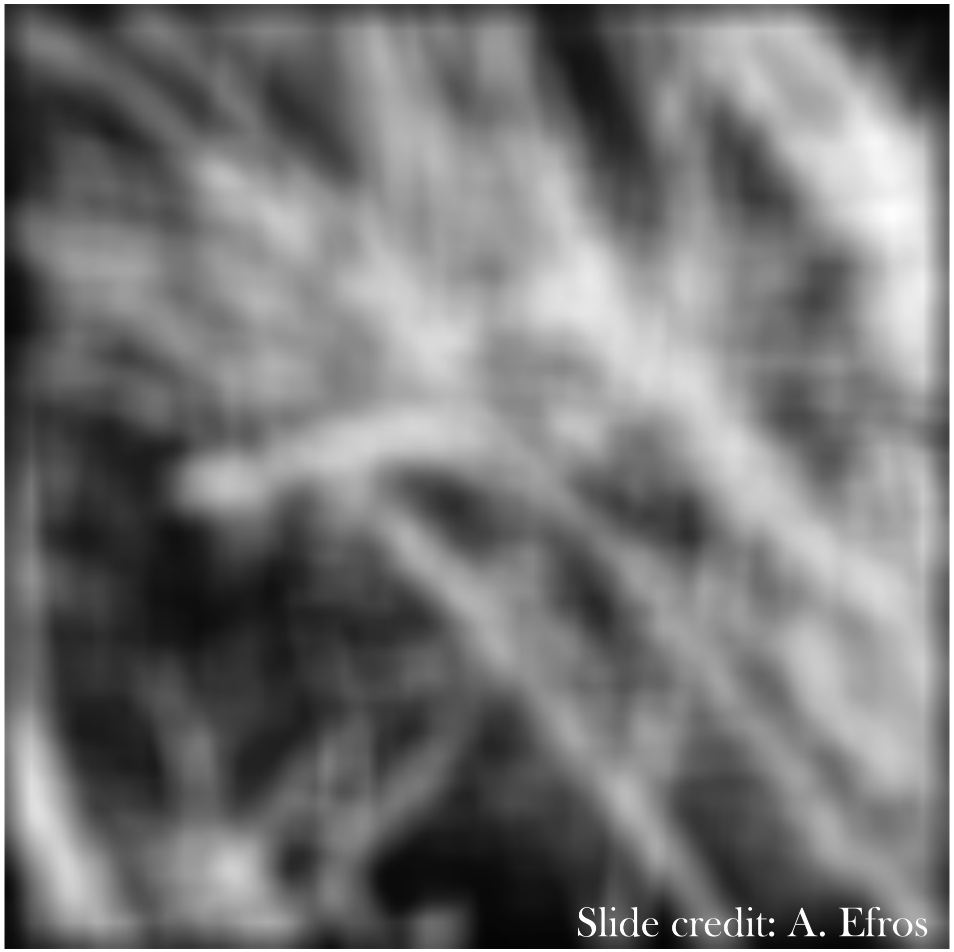
Filtering

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

Gaussian

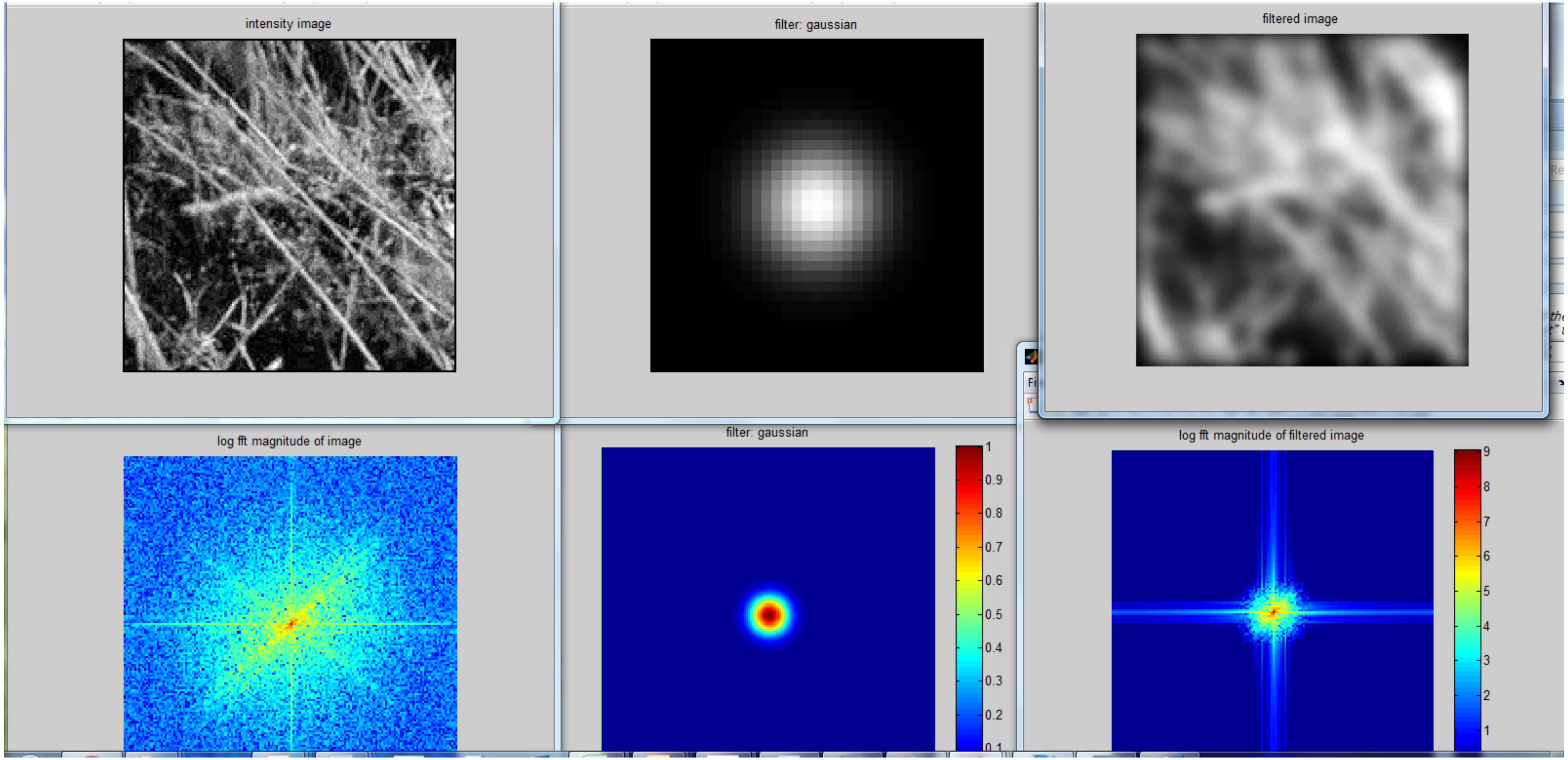


Box filter



Filtering

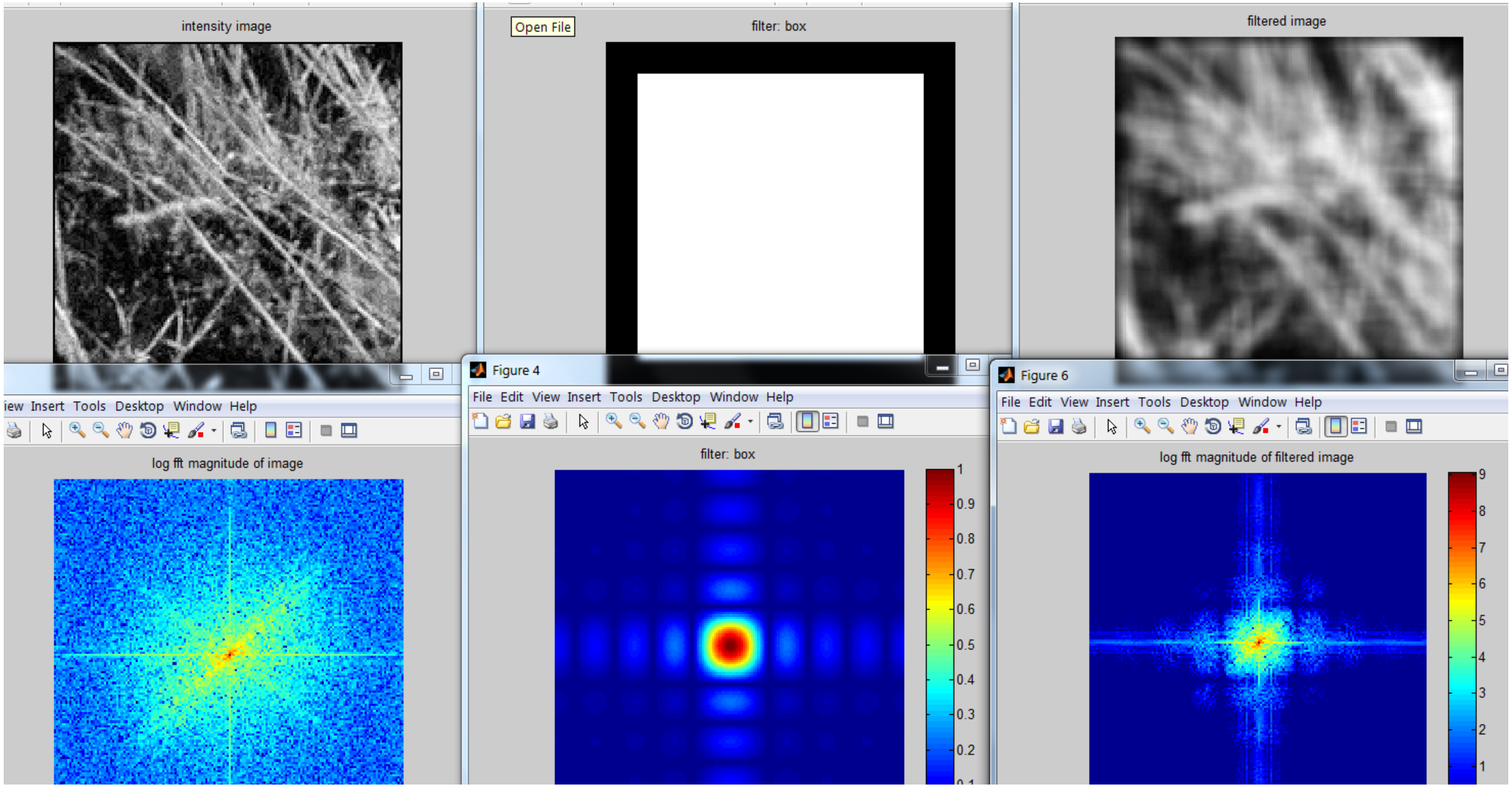
Gaussian



Slide credit: A. Efros

Filtering

Box Filter



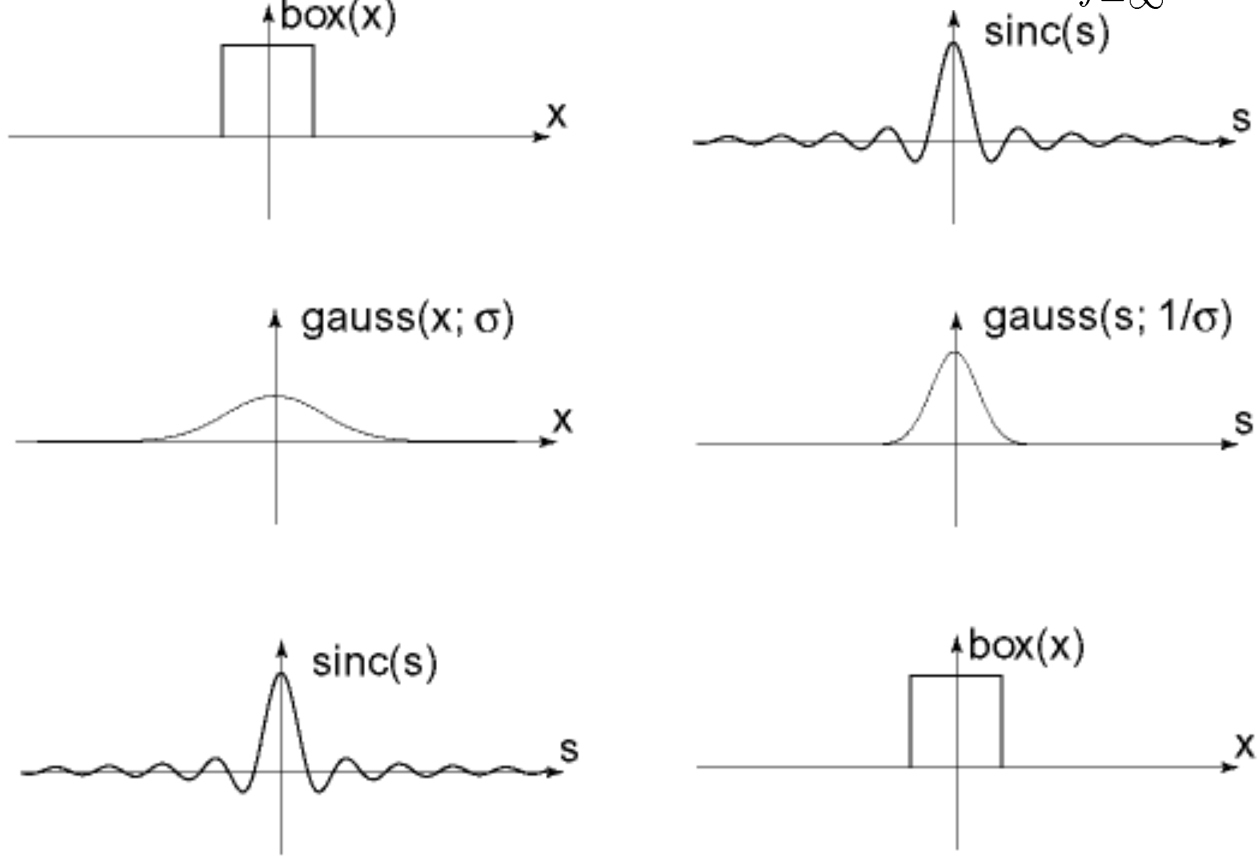
Slide credit: A. Efros

Fourier Transform pairs

Spatial domain

Frequency domain

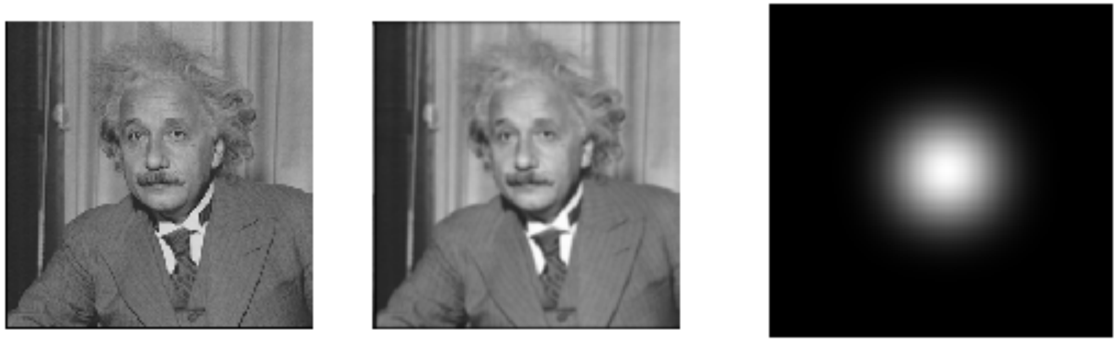
$$F(s) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi sx} dx$$



Slide credit: A. Efros

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

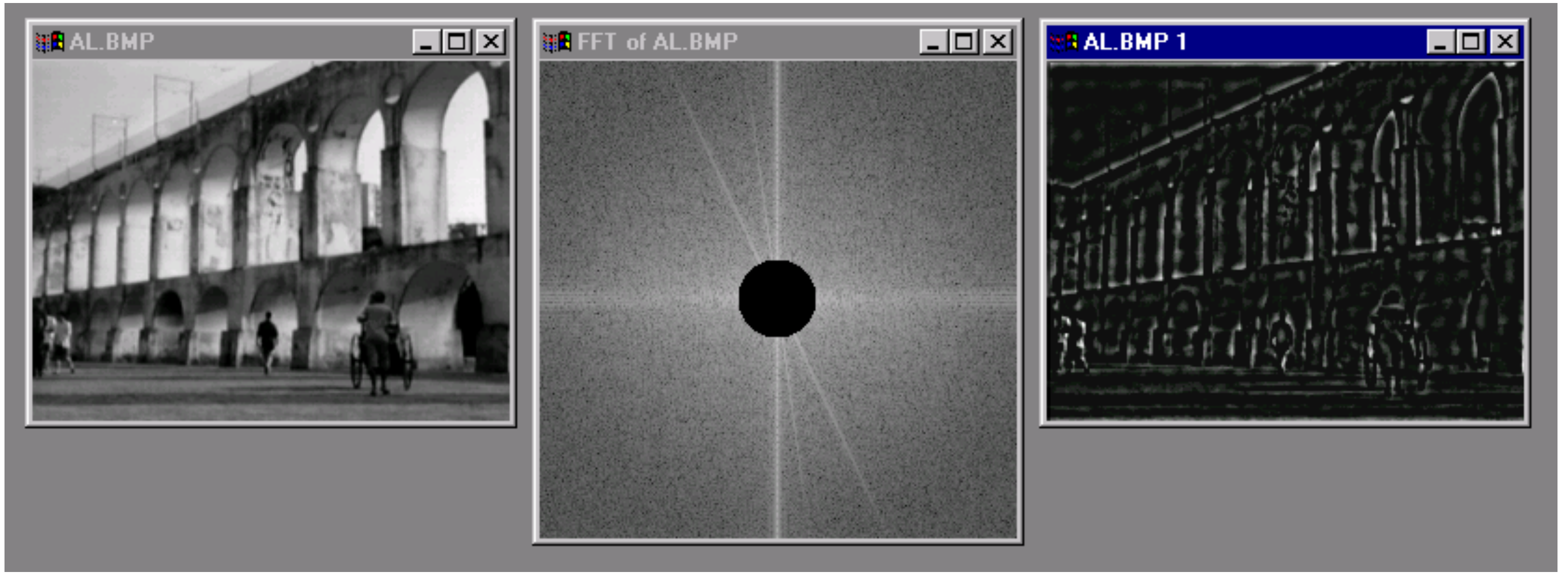
low-pass:



High-pass / band-pass:

Slide credit: A. Efros

Edges in images



Slide credit: A. Efros

FFT in Matlab

- Filtering with fft

```
im = ... % “im” should be a gray-scale floating point image
[imh, imw] = size(im);
fftsize = 1024; % should be order of 2 (for speed) and include padding
im_fft = fft2(im, fftsize, fftsize); % 1) fft im with padding
hs = 50; % filter half-size
fil = fspecial('gaussian', hs*2+1, 10);
fil_fft = fft2(fil, fftsize, fftsize); % 2) fft fil, pad to same size as
image
im_fil_fft = im_fft .* fil_fft; % 3) multiply fft images
im_fil = ifft2(im_fil_fft); % 4) inverse fft2
im_fil = im_fil(1+hs:size(im,1)+hs, 1+hs:size(im, 2)+hs); % 5) remove padding
```

- Displaying with fft

```
figure(1), imagesc(log(abs(fftshift(im_fft)))), axis image, colormap jet
```

Slide credit: D. Hoiem

Phase and Magnitude

- Curious fact
  - all natural images have about the same magnitude transform
  - hence, phase seems to matter, but magnitude largely doesn't

- Demonstration
  - Take two pictures, swap the phase transforms, compute the inverse - what does the result look like?

Image with cheetah phase (and zebra magnitude)

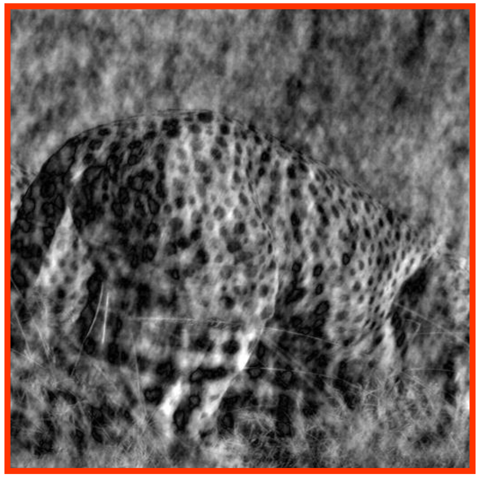
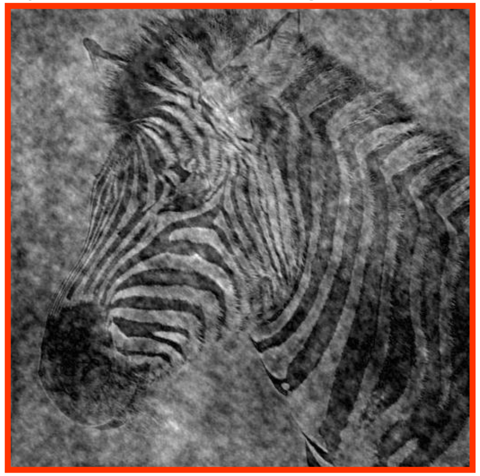


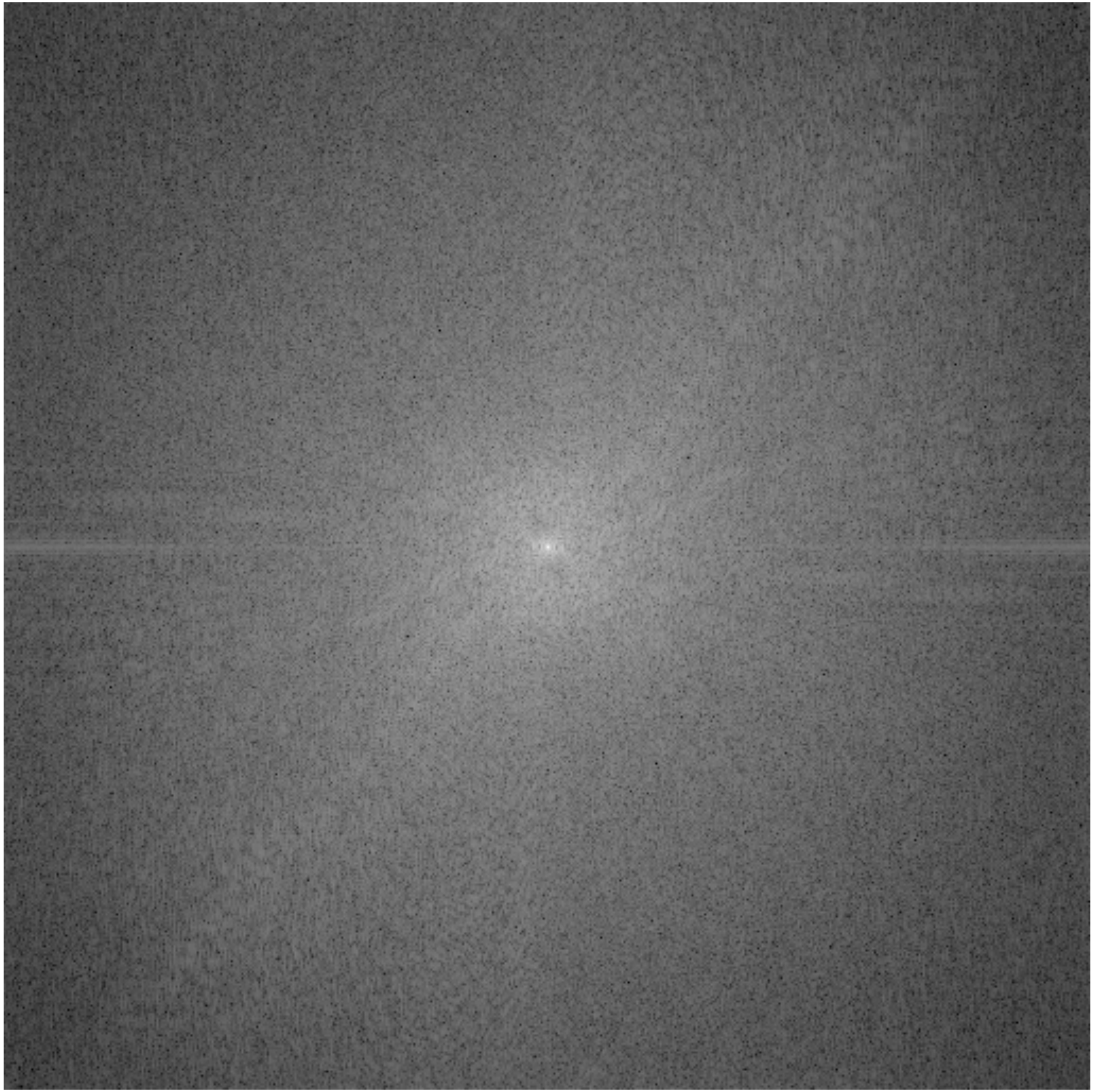
Image with zebra phase (and cheetah magnitude)



Slide credit: B. Freeman and A. Torralba

Slide credit: B. Freeman and A. Torralba

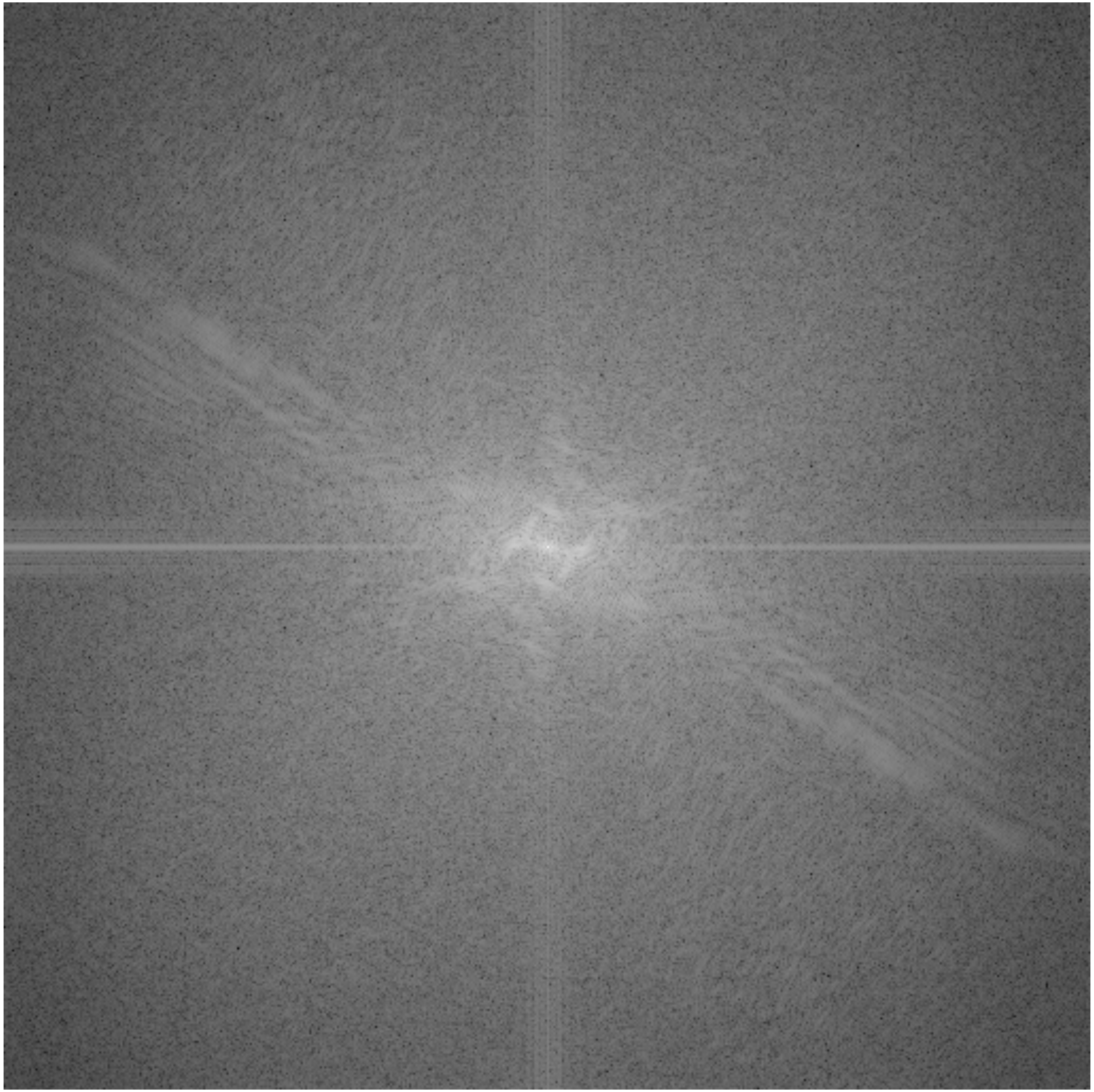
This is the magnitude transform of the cheetah picture



Slide credit: B. Freeman and A. Torralba

Slide credit: B. Freeman and A. Torralba

This is the magnitude transform of the zebra picture

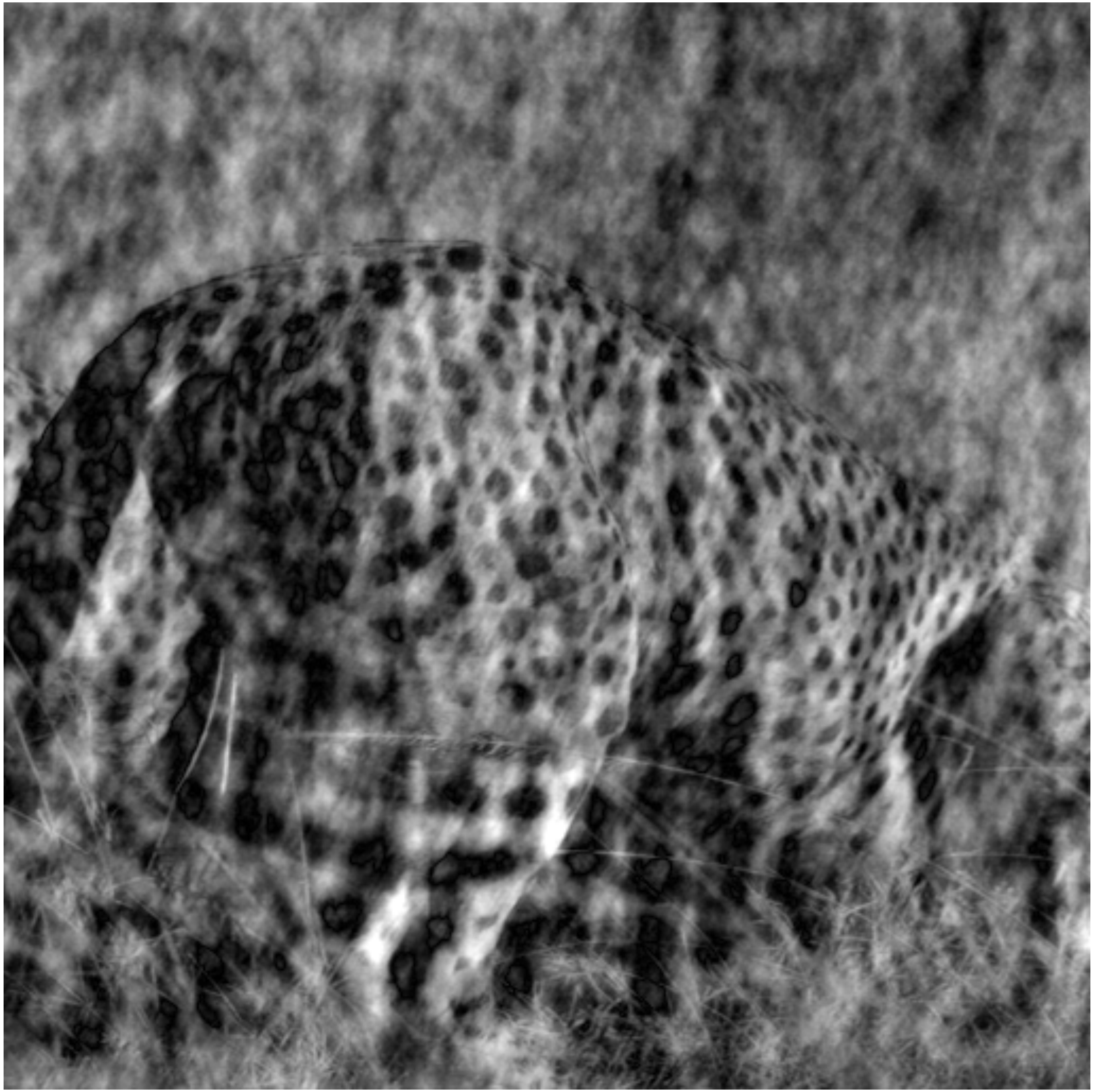


Slide credit: B. Freeman and A. Torralba

Reconstruction with zebra phase, cheetah magnitude

Slide credit: B. Freeman and A. Torralba

Reconstruction with cheetah phase, zebra magnitude



Slide credit: B. Freeman and A. Torralba

What is a good representation for image analysis?

- Fourier transform domain tells you "what" (textural properties), but not "where".
- Pixel domain representation tells you "where" (pixel location), but not "what".
- Want an image representation that gives you a local description of image events—what is happening where.

Slide credit: B. Freeman and A. Torralba

Sampling

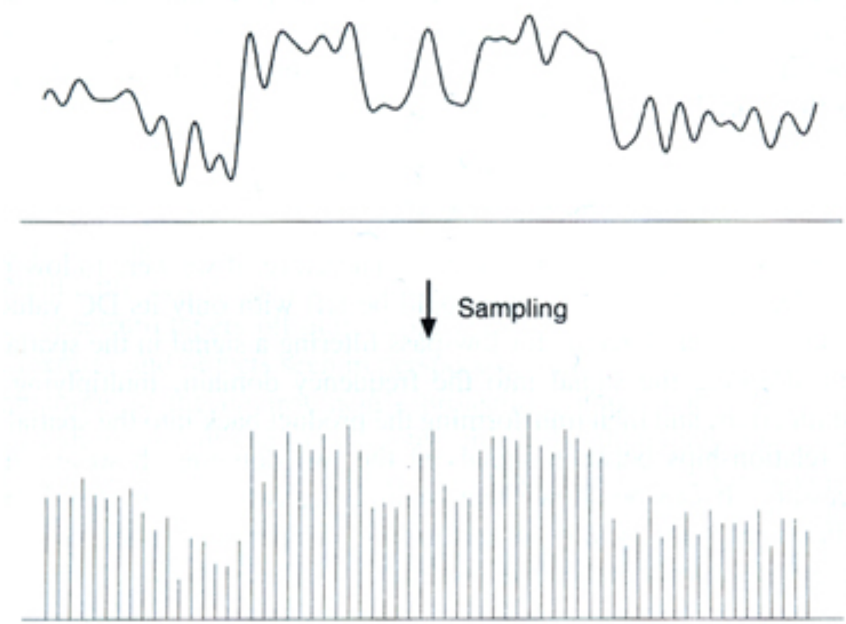
Why does a lower resolution image still make sense to us?
What do we lose?

Image: http://www.flickr.com/photos/igorms/136916757/

Slide credit: D. Hoiem

Sampled representations

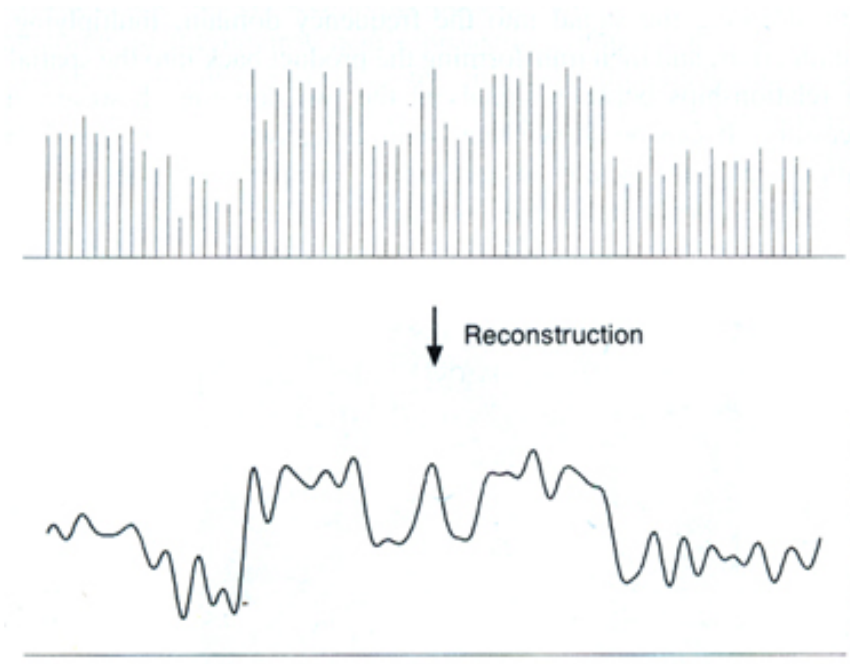
- How to store and compute with continuous functions?
- Common scheme for representation: samples
  - write down the function's values at many points



Slide credit: S. Marschner

Reconstruction

- Making samples back into a continuous function
  - for output (need realizable method)
  - for analysis or processing (need mathematical method)
  - amounts to "guessing" what the function did in between

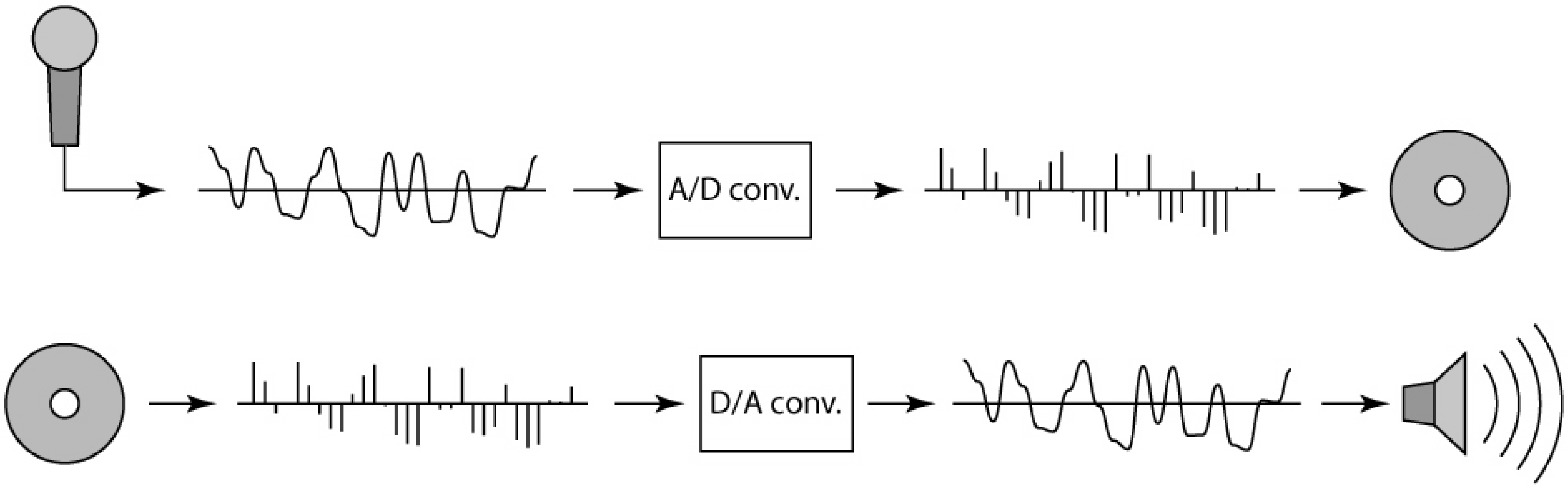


Slide credit: S. Marschner

Sampling in digital audio

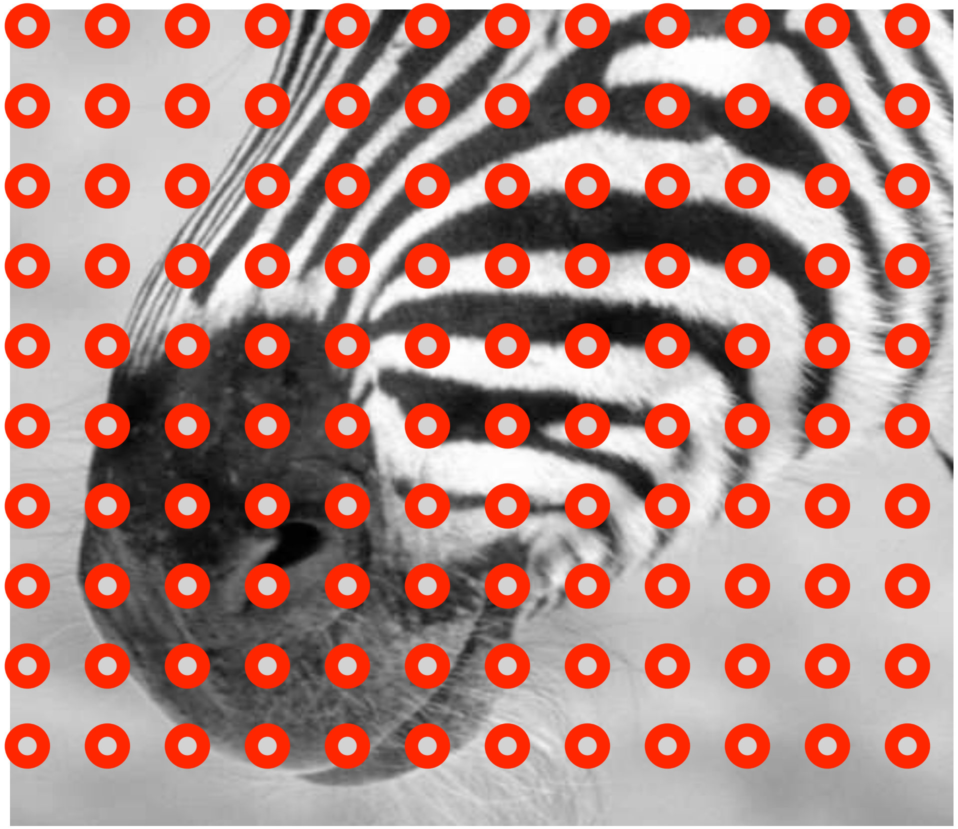
- Recording: sound to analog to samples to disc
- Playback: disc to samples to analog to sound again

– how can we be sure we are filling in the gaps correctly?



Slide credit: S. Marschner

Subsampling by a factor of 2

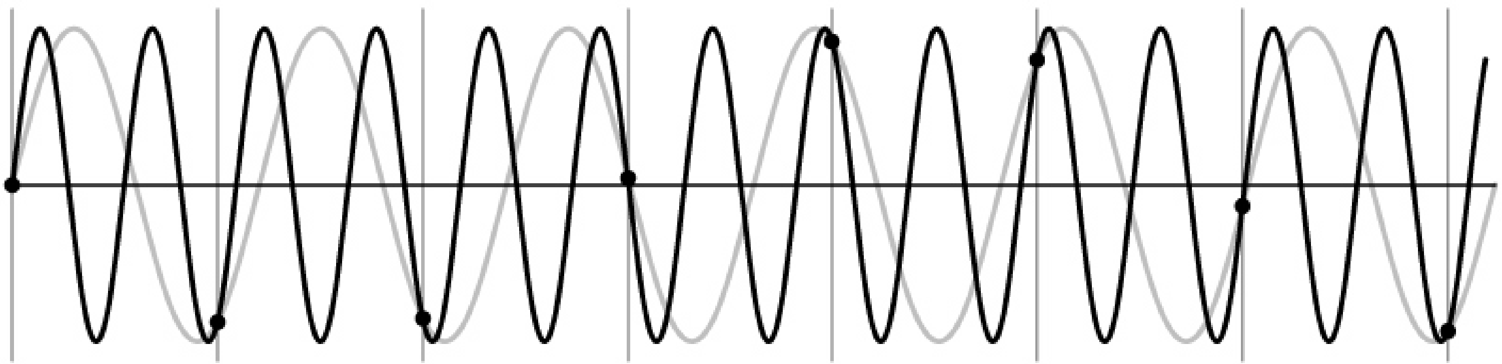


Throw away every other row and column to create a 1/2 size image

Slide credit: D. Hoiem

Undersampling

- What if we "missed" things between the samples?
- Simple example: undersampling a sine wave
  - unsurprising result: information is lost
  - surprising result: indistinguishable from lower frequency
  - also was always indistinguishable from higher frequencies
  - *aliasing*: signals "traveling in disguise" as other frequencies

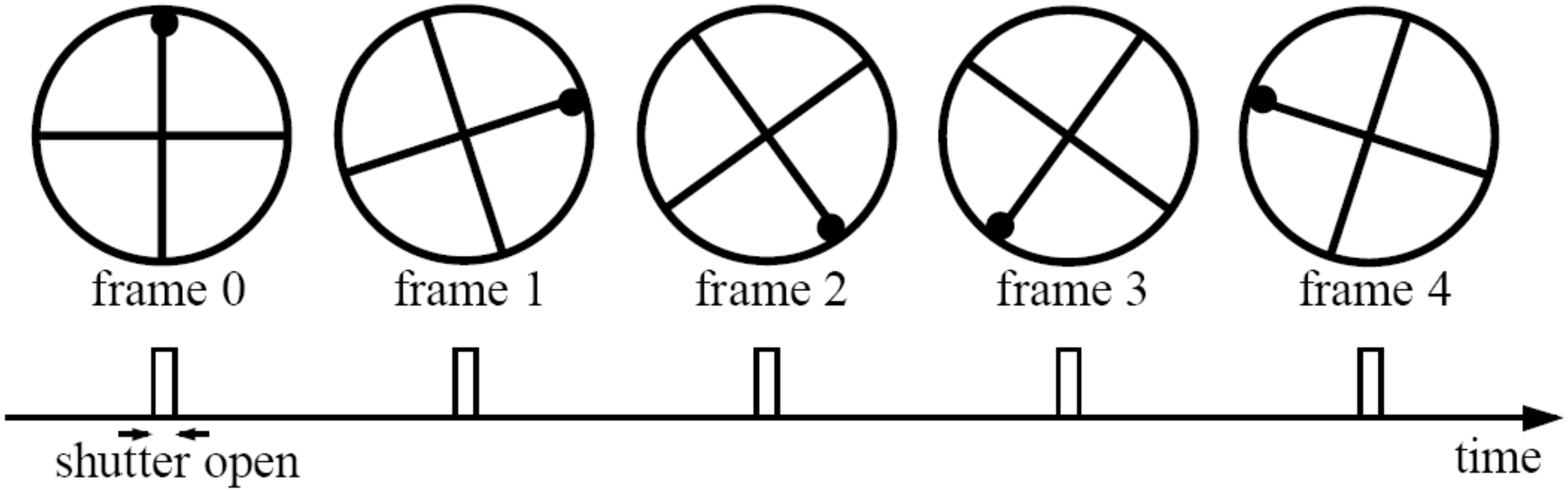


Slide credit: S. Marschner

Aliasing in video

Imagine a spoked wheel moving to the right (rotating clockwise). Mark wheel with dot so we can see what's happening.

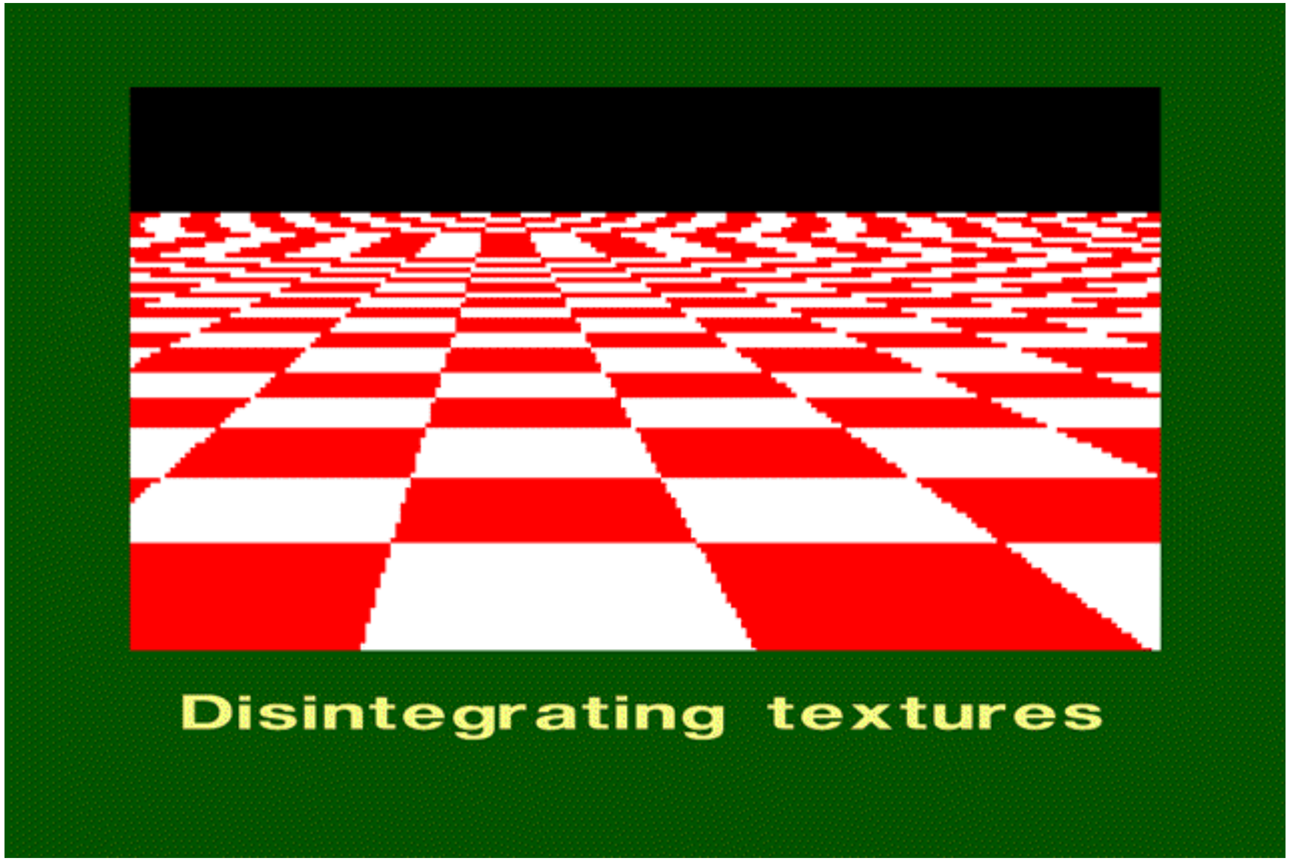
If camera shutter is only open for a fraction of a frame time (frame time = 1/30 sec. for video, 1/24 sec. for film):



Without dot, wheel appears to be rotating slowly backwards! (counterclockwise)

Slide credit: S. Seitz

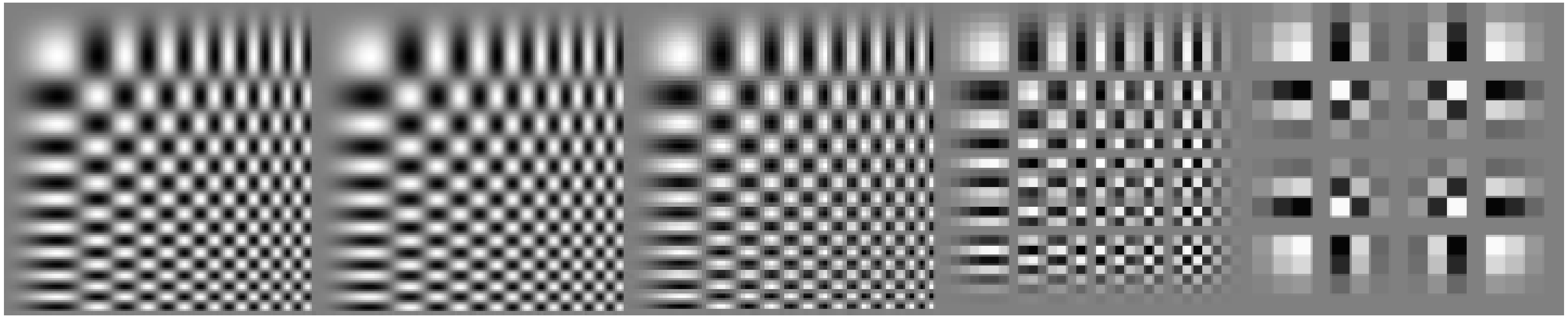
Aliasing in graphics



Slide credit: A. Efros

Sampling and aliasing

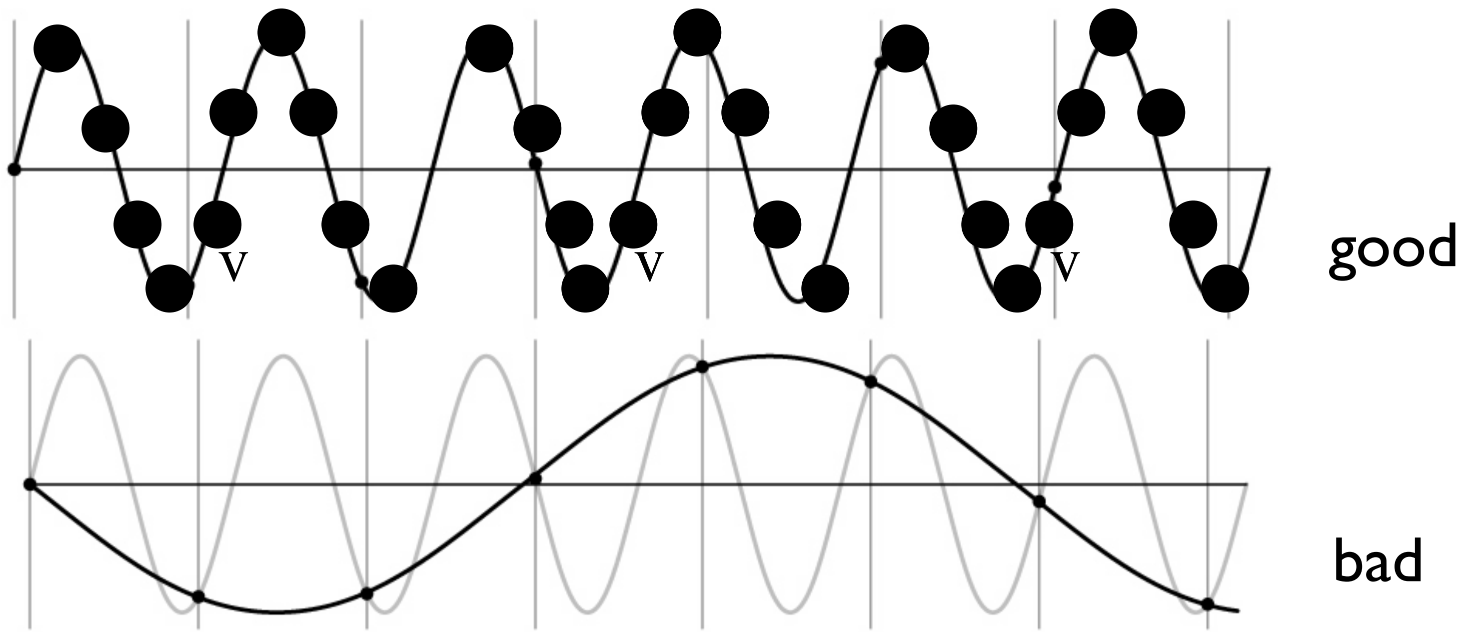
256x256 128x128 64x64 32x32 16x16



Slide credit: D. Hoiem

Nyquist-Shannon Sampling Theorem

- When sampling a signal at discrete intervals, the sampling frequency must be $\geq 2 \times f_{max}$
- f_{max} = max frequency of the input signal
- This will allows to reconstruct the original perfectly from the sampled version



Slide credit: D. Hoiem

Anti-aliasing

Solutions:

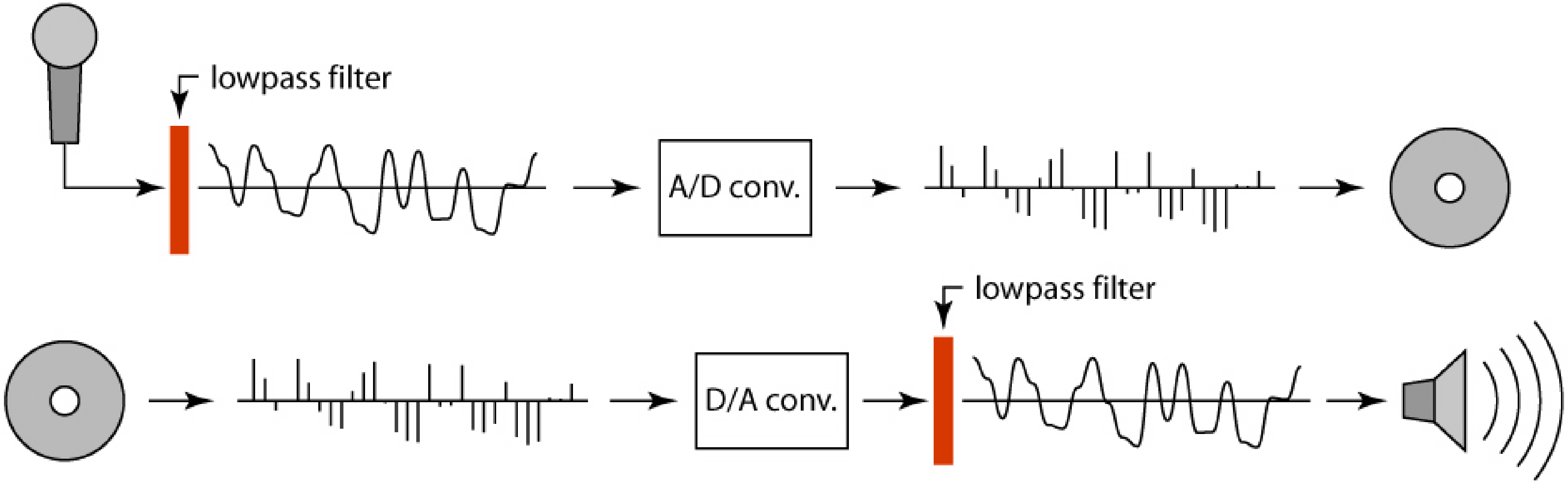
- Sample more often

- Get rid of all frequencies that are greater than half the new sampling frequency
  - Will lose information
  - But it's better than aliasing
  - Apply a smoothing filter

Slide credit: D. Hoiem

Preventing aliasing

- Introduce lowpass filters:
  - remove high frequencies leaving only safe, low frequencies
  - choose lowest frequency in reconstruction (disambiguate)



Slide credit: S. Marschner

Algorithm for downsampling by factor of 2

1. Start with image(h, w)
2. Apply low-pass filter

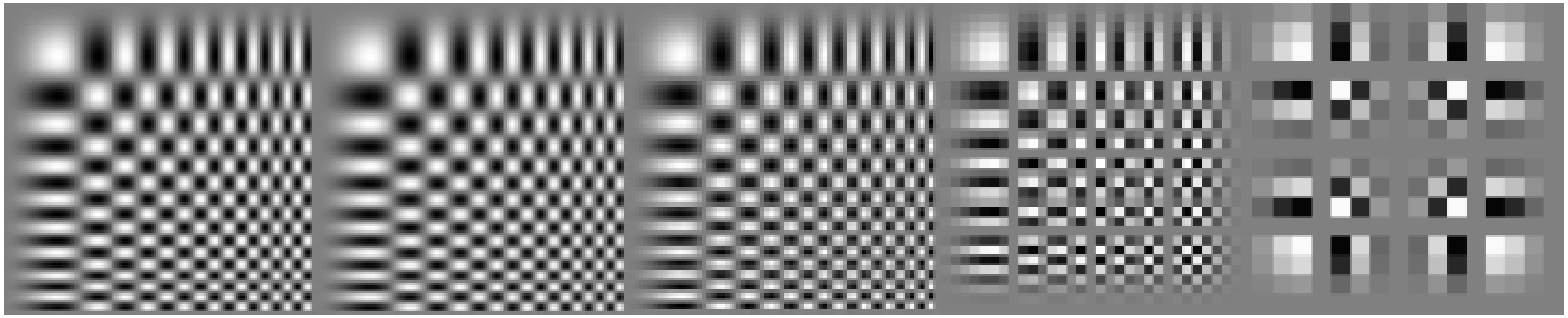
   im_blur = imfilter(image, fspecial('gaussian', 7, 1))
3. Sample every other pixel

   im_small = im_blur(1:2:end, 1:2:end);

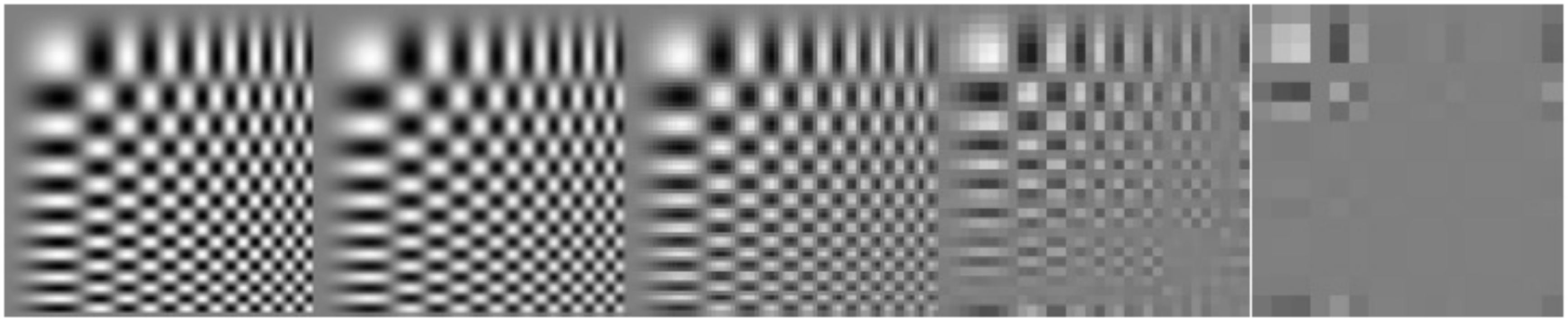
Slide credit: D. Hoiem

Anti-aliasing

256x256 128x128 64x64 32x32 16x16

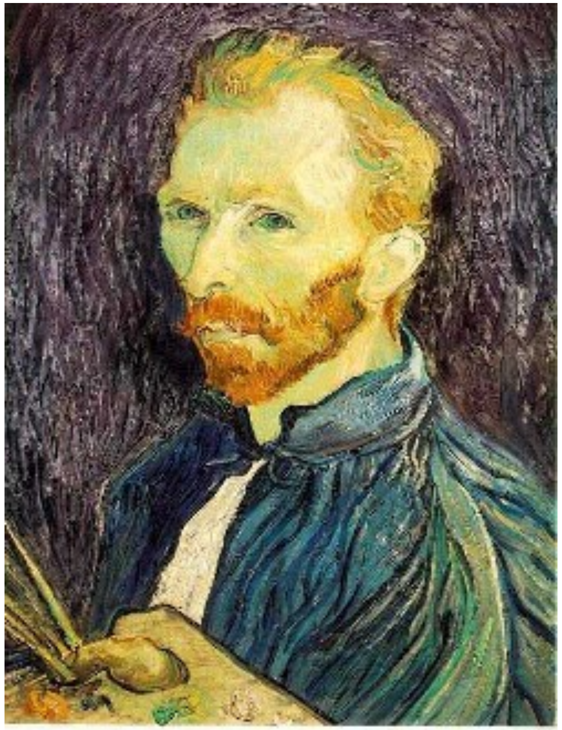


256x256 128x128 64x64 32x32 16x16



Slide credit: Forsyth and Ponce

Subsampling without pre-filtering



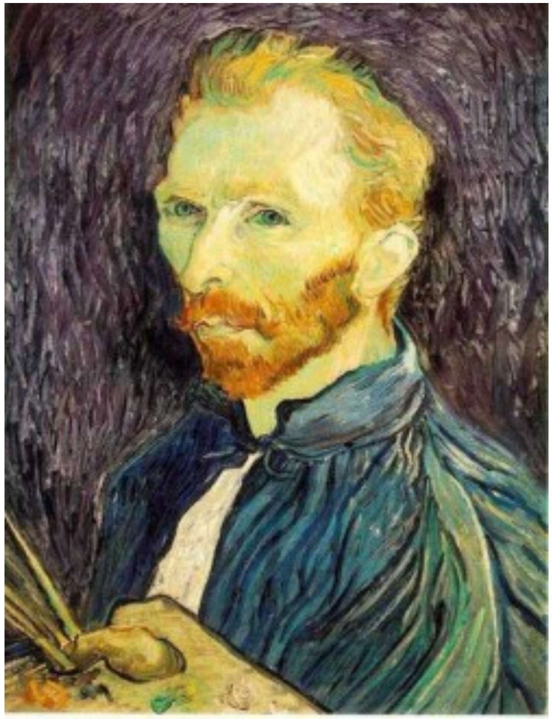
1/2

1/4 (2x zoom)

1/8 (4x zoom)

Slide credit: S. Seitz

Subsampling with Gaussian pre-filtering



Gaussian 1/2

G 1/4

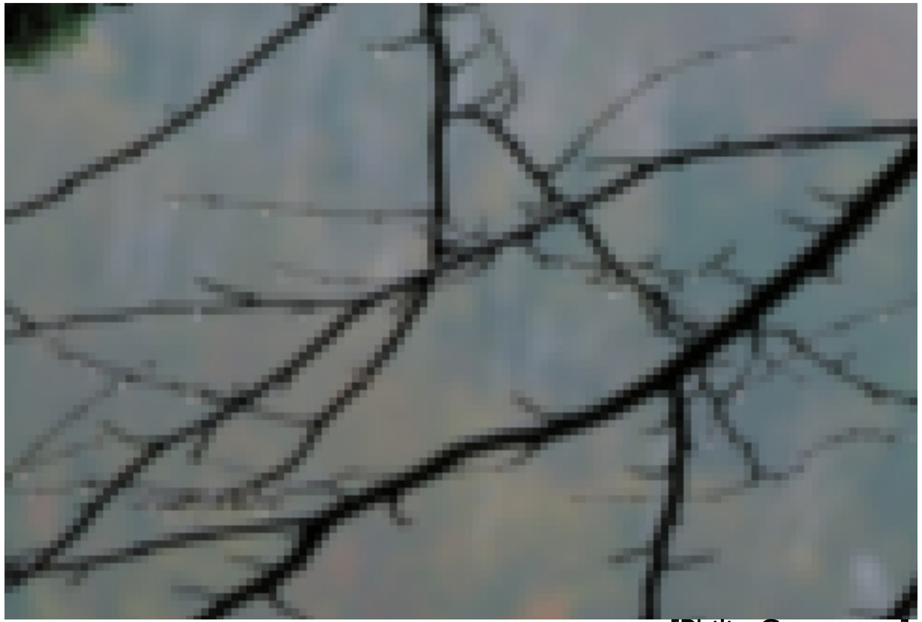
G 1/8

Slide credit: S. Seitz

[Philip Greenspun]

1000 pixel width

Slide credit: S. Marschner



[Philip Greenspun]

by dropping pixels

gaussian filter

250 pixel width

Slide credit: S. Marschner

Analyzing local image structures

Too much

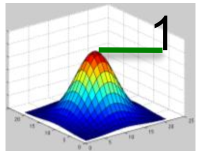
Too little

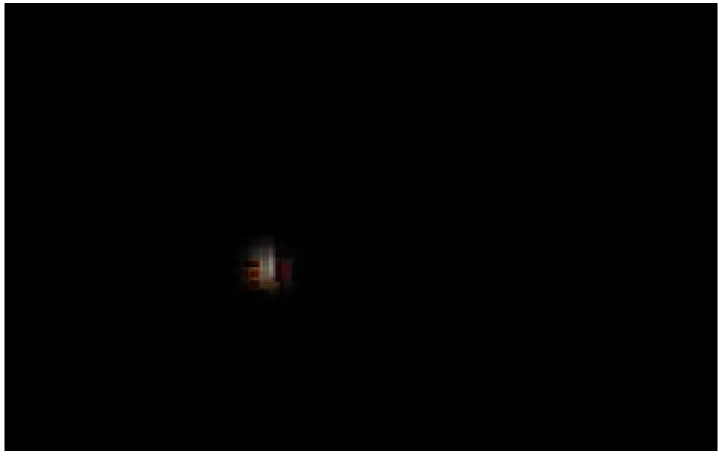
Slide credit: B. Freeman and A. Torralba

The image through the Gaussian window

Too much

$$h(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$



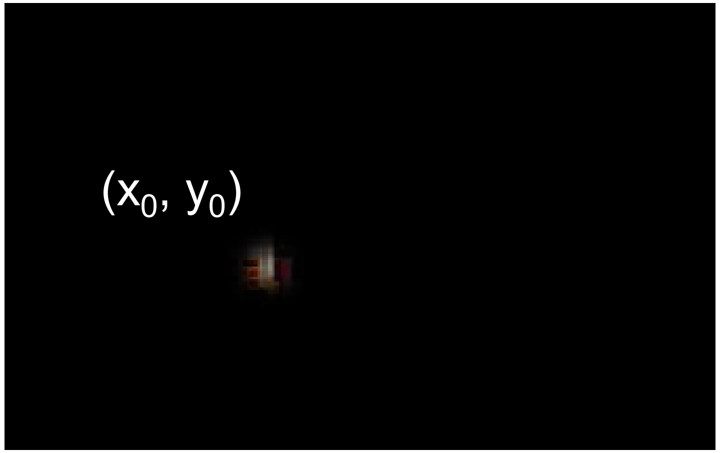


Too little

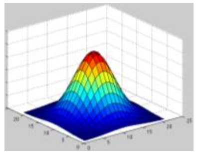
Probably still too little…
…but hard enough for now

Slide credit: B. Freeman and A. Torralba

Analysis of local frequency



$$h(x,y;x_0,y_0) = e^{-\frac{(x-x_o)^2+(y-y_o)^2}{2\sigma^2}}$$



Fourier basis:

$$e^{j2\pi u_0 x}$$

Gabor wavelet:

$$\psi(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} e^{j2\pi u_0 x}$$

We can look at the real and imaginary parts:

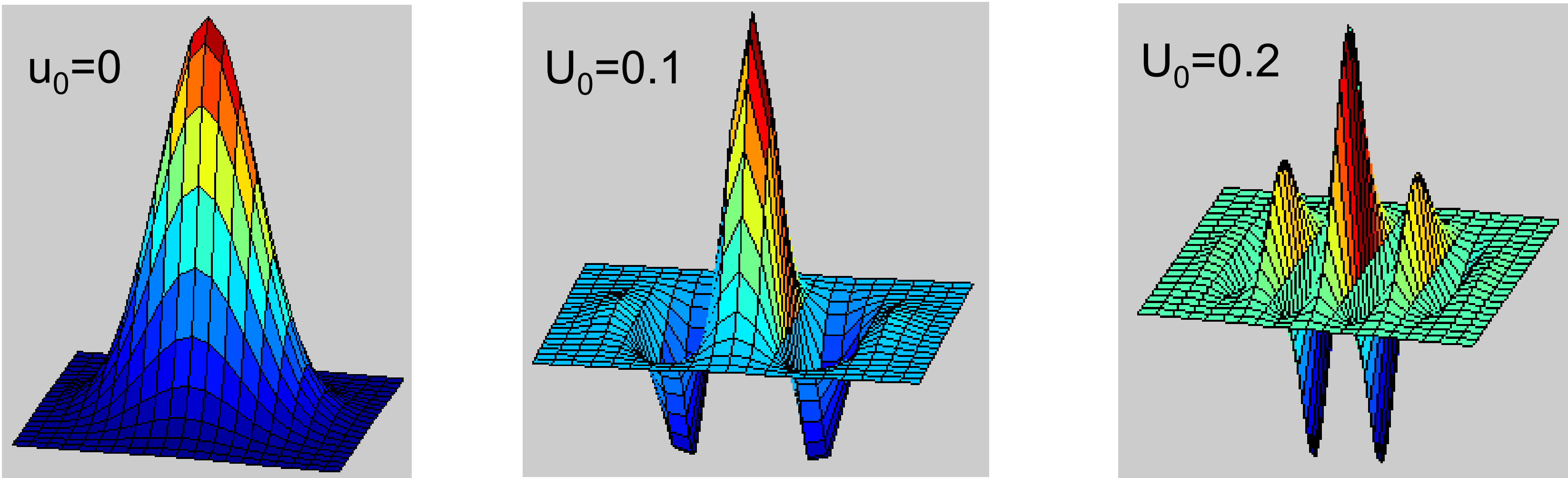
$$\psi_c(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} \cos(2\pi u_0 x)$$

$$\psi_s(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} \sin(2\pi u_0 x)$$

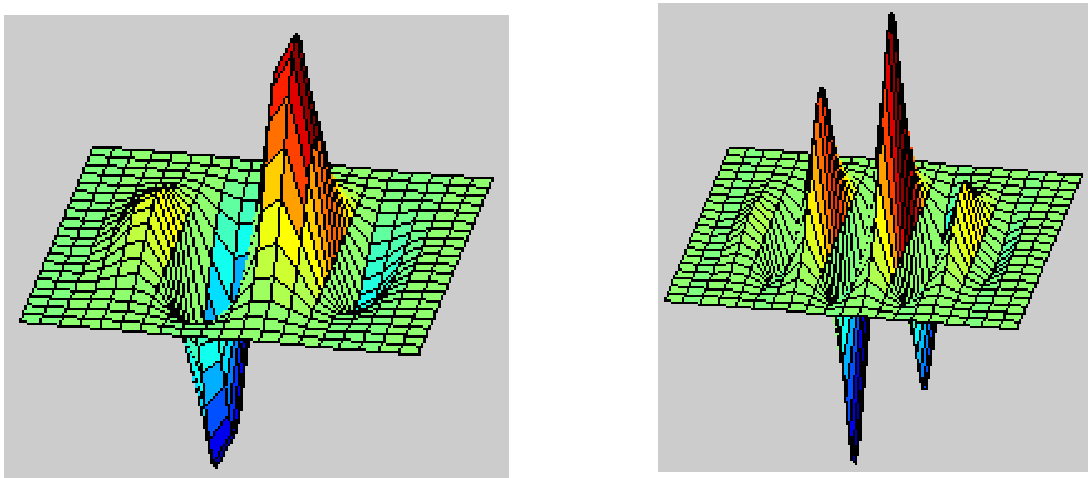
Slide credit: B. Freeman and A. Torralba

Gabor wavelets

$$\psi_c(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} \cos(2\pi u_0 x)$$

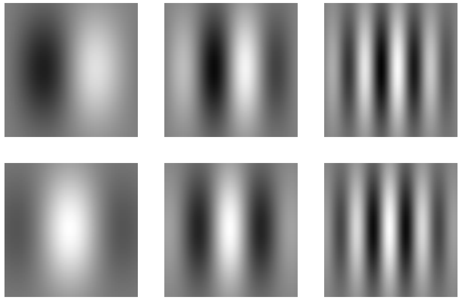


$$\psi_s(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} \sin(2\pi u_0 x)$$

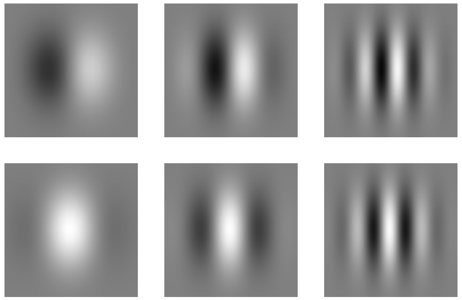


Slide credit: B. Freeman and A. Torralba

Gabor filters



Gabor filters at different scales and spatial frequencies



Top row shows anti-symmetric (or odd) filters; these are good for detecting odd-phase structures like edges.
Bottom row shows the symmetric (or even) filters, good for detecting line phase contours.

Slide credit: B. Freeman and A. Torralba

John Daugman, 1988

IEEE TRANSACTIONS ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING, VOL. 36, NO. 7, JULY 1988

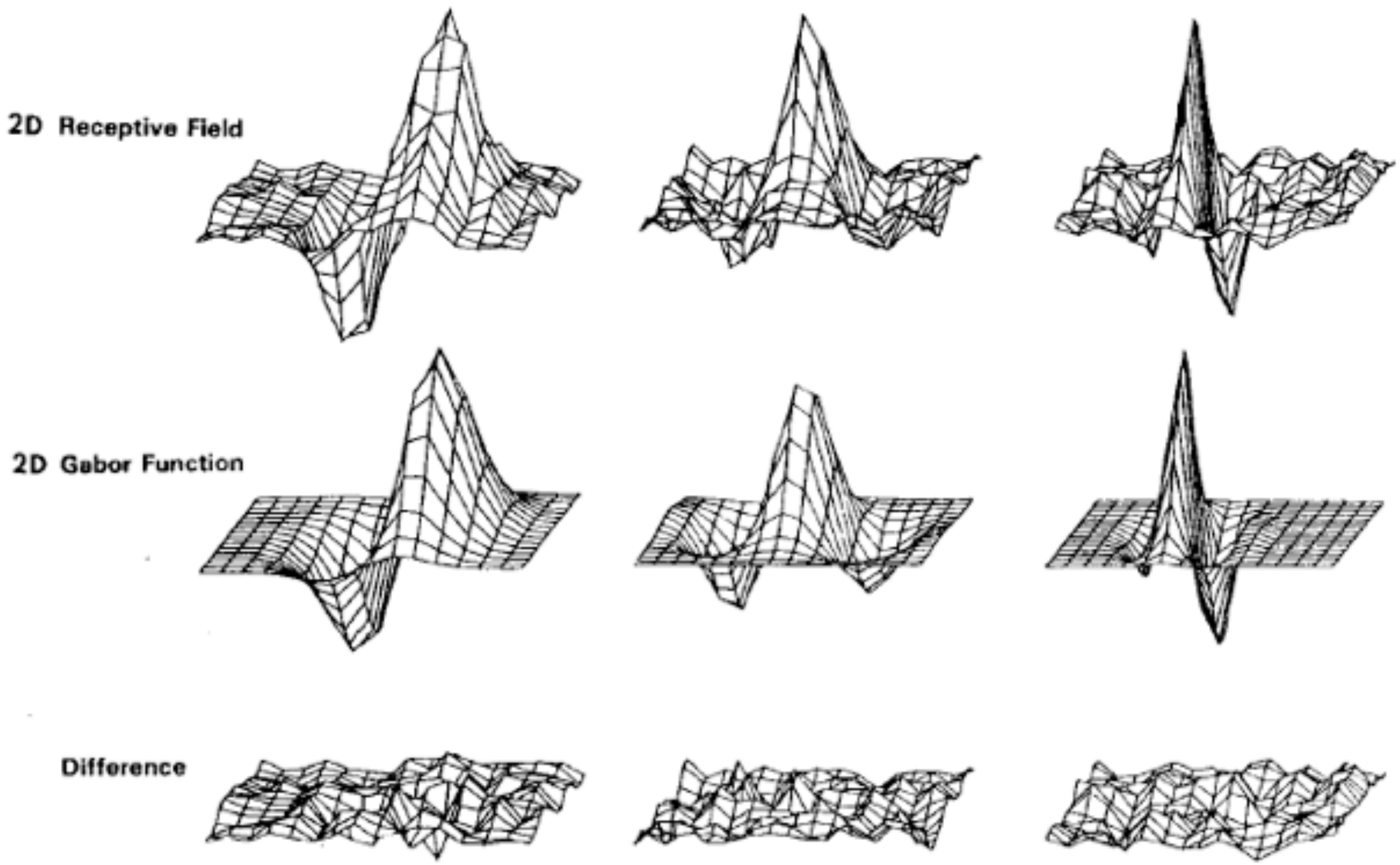
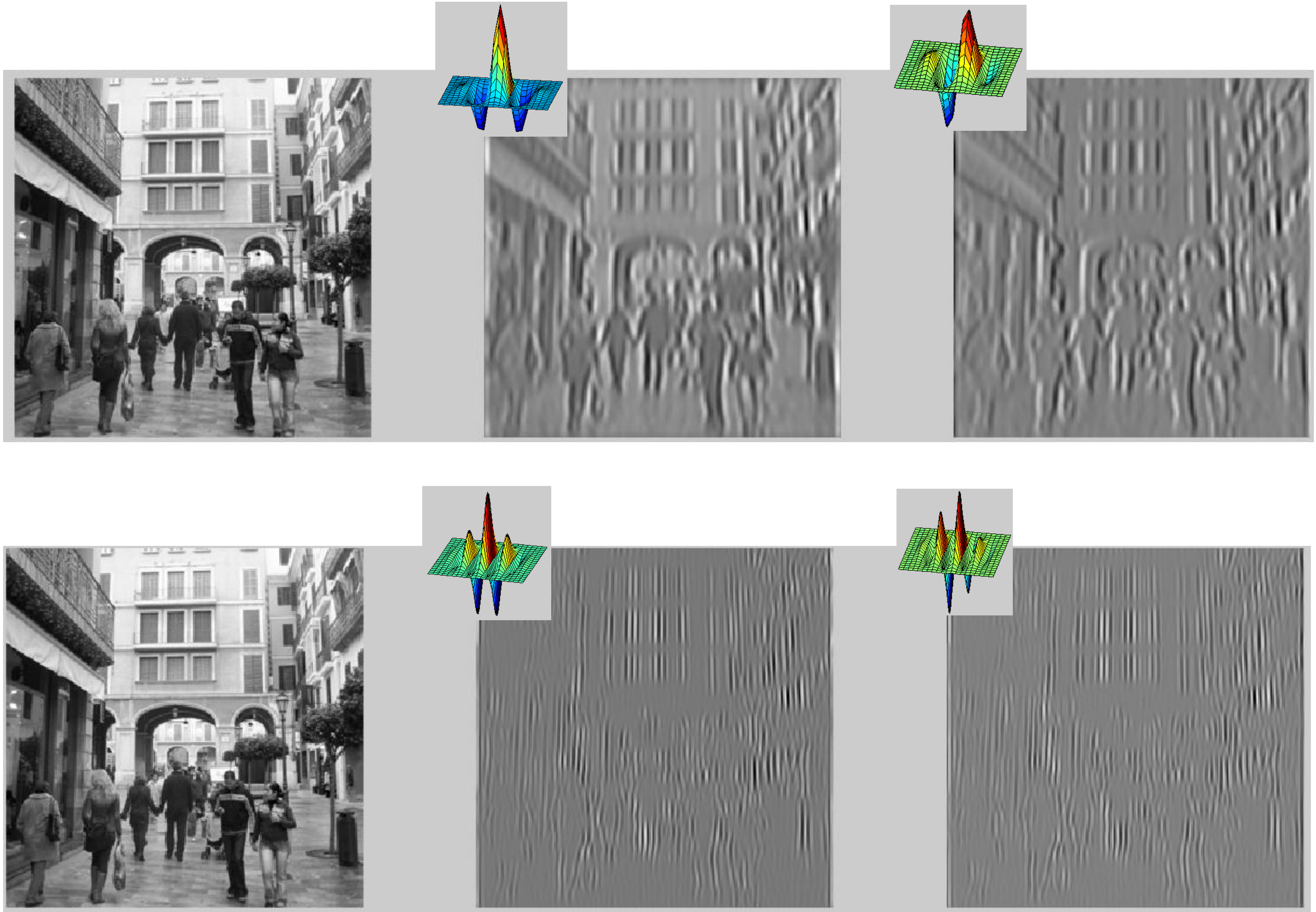


Fig. 5. Top row: illustrations of empirical 2-D receptive field profiles measured by J. P. Jones and L. A. Palmer (personal communication) in simple cells of the cat visual cortex. Middle row: best-fitting 2-D Gabor elementary function for each neuron, described by (10). Bottom row: residual error of the fit, indistinguishable from random error in the Chi-squared sense for 97 percent of the cells studied.

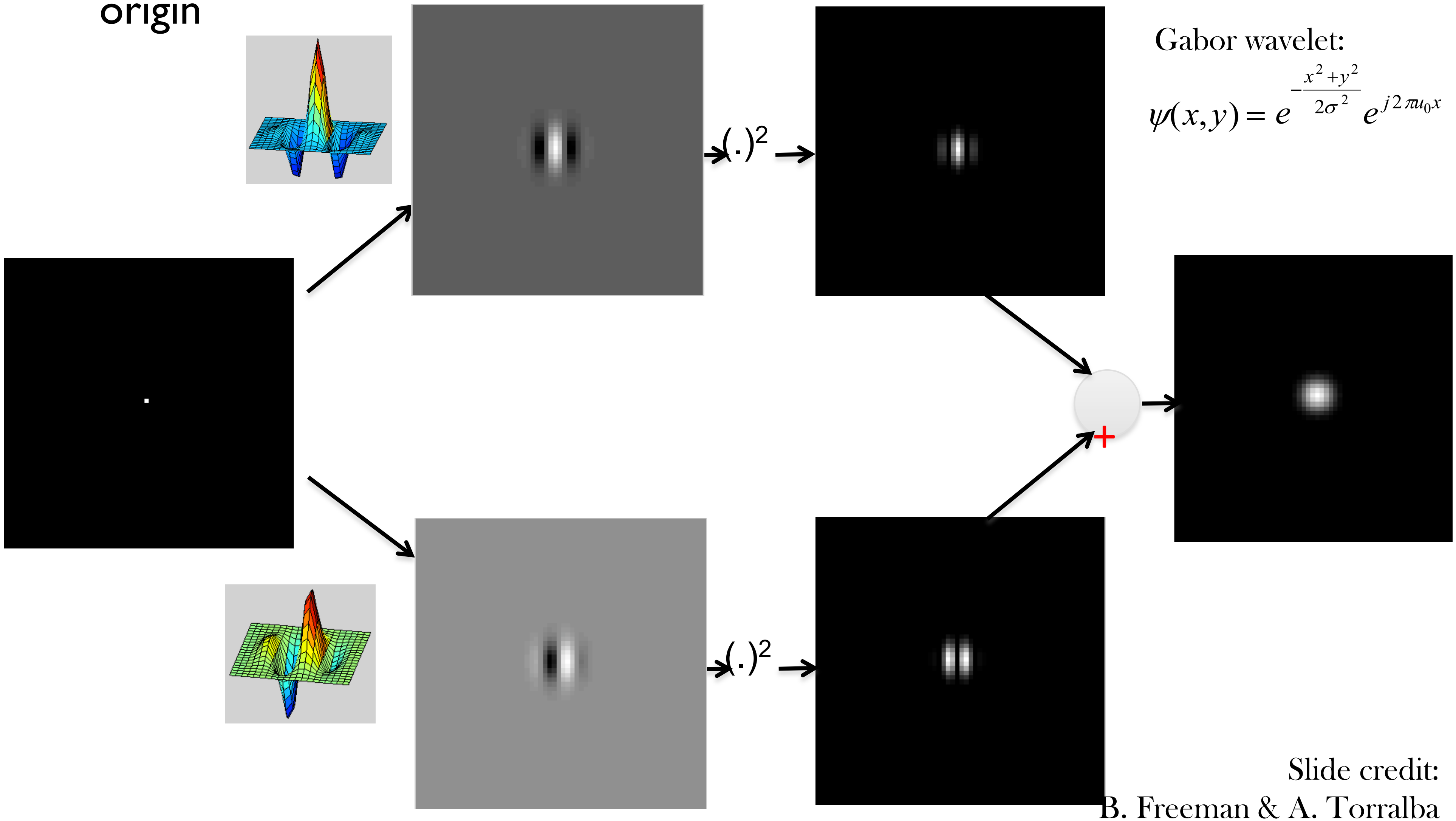
Slide credit: B. Freeman and A. Torralba



Slide credit: B. Freeman and A. Torralba

Quadrature filter pairs

- A quadrature filter is a complex filter whose real part is related to its imaginary part via a Hilbert transform along a particular axis through the origin

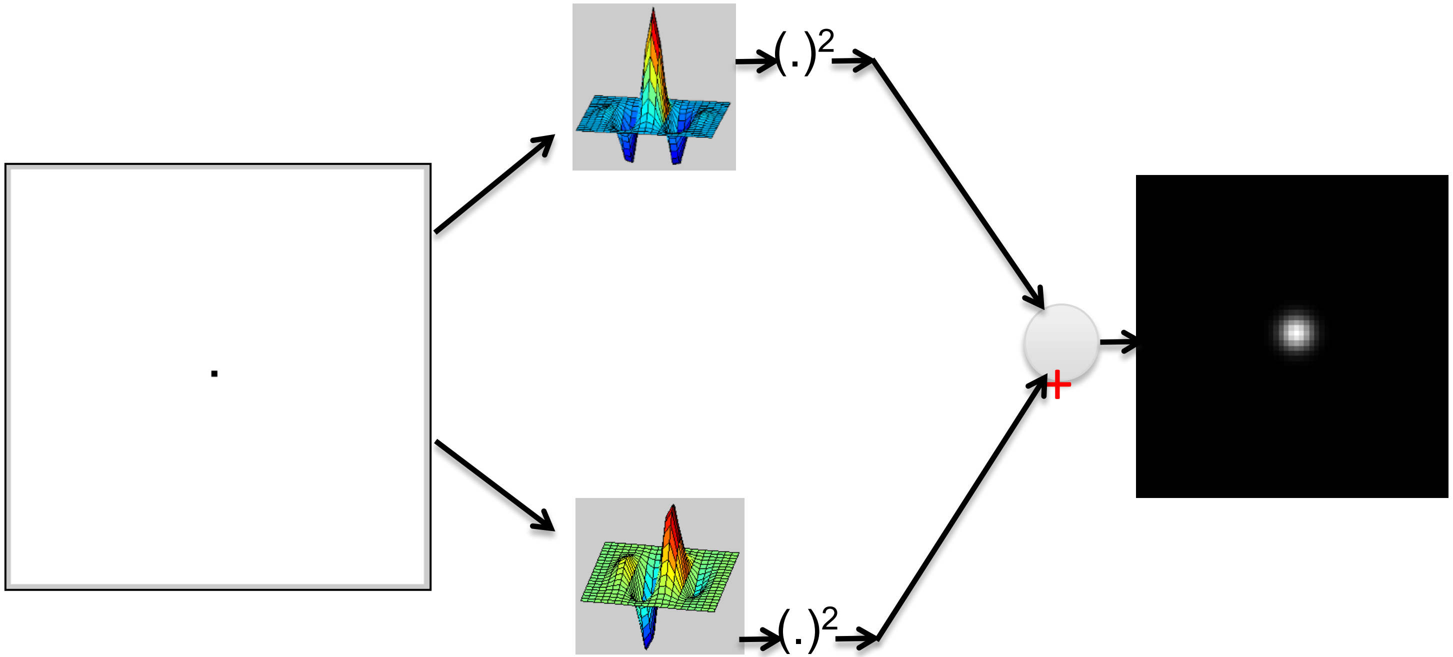


Gabor wavelet:

$$\psi(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} e^{j2\pi u_0 x}$$

Slide credit:
B. Freeman & A. Torralba

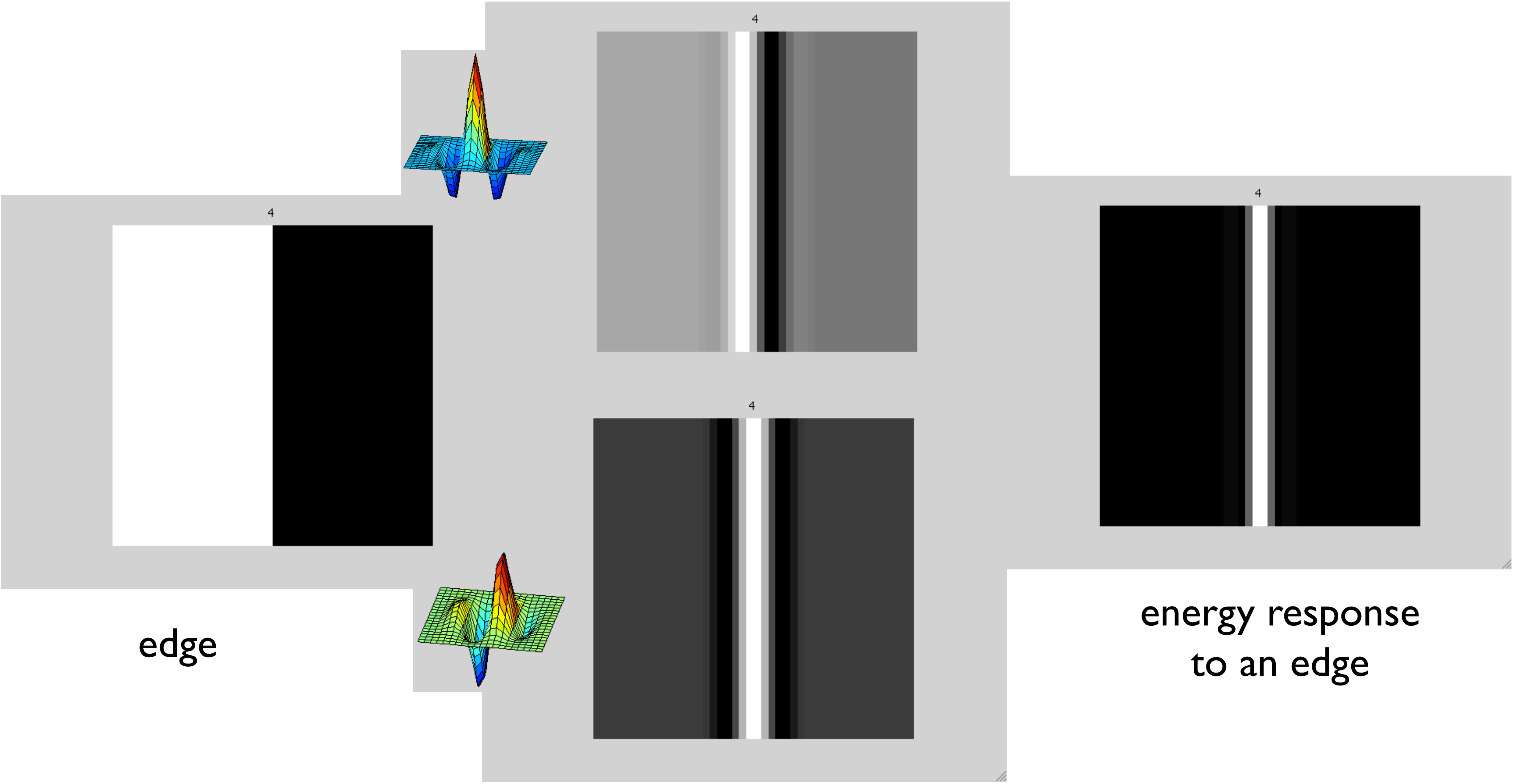
Quadrature filter pairs



Contrast invariance! (same energy response for white dot on black background as for a black dot on a white background).

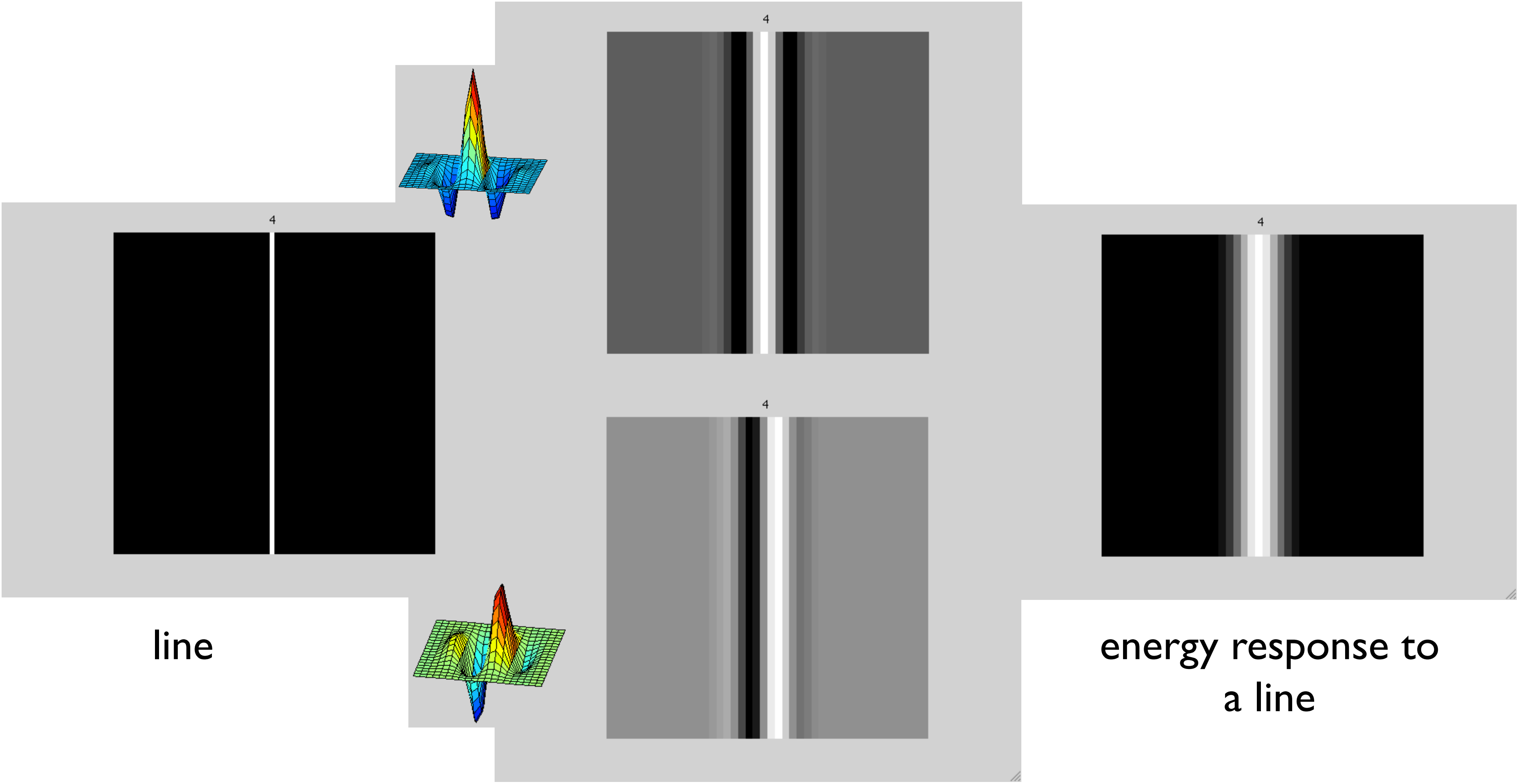
Slide credit: B. Freeman and A. Torralba

Quadrature filter pairs



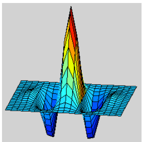
Slide credit: B. Freeman and A. Torralba

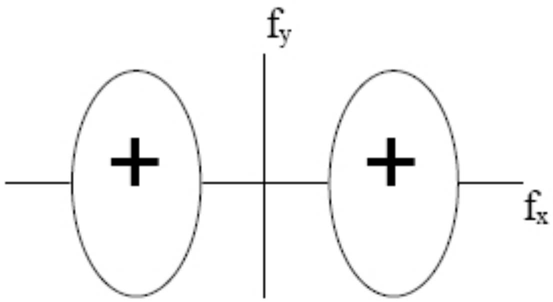
Quadrature filter pairs



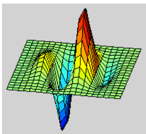
Slide credit: B. Freeman and A. Torralba

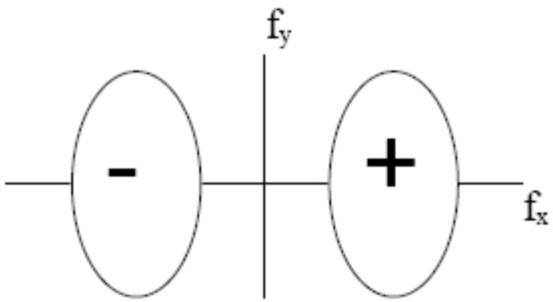
How quadrature pair filters work





(a) Frequency response of even filter, G
(real)



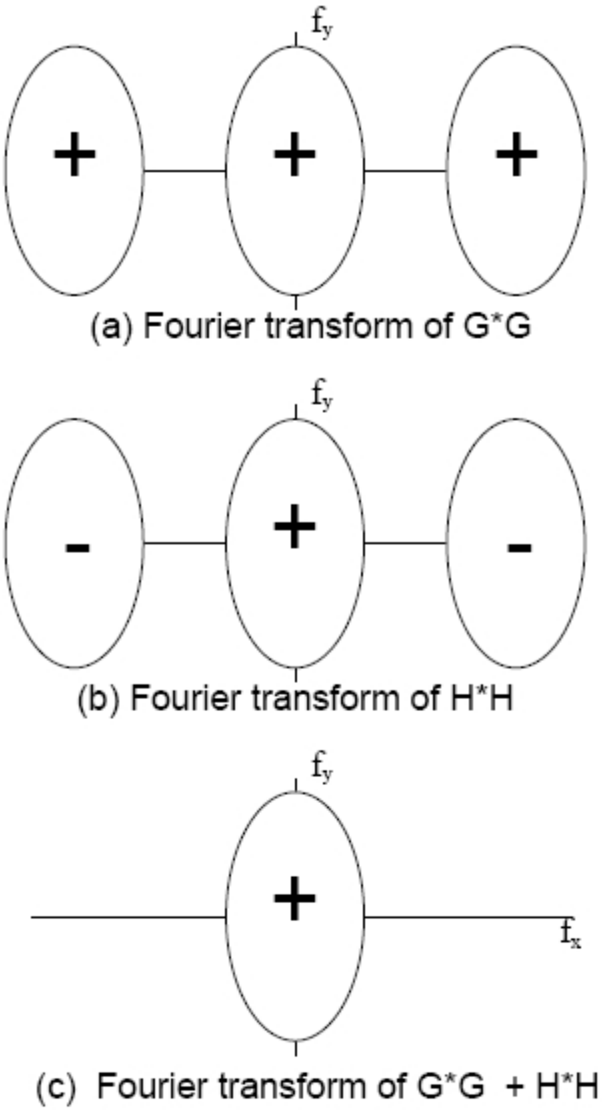


(b) Frequency response of odd filter, H
(imaginary)

Figure 3-5: Frequency content of two bandpass filters in quadrature. (a) even phase filter, called G in text, and (b) odd phase filter, H. Plus and minus sign illustrate relative sign of regions in the frequency domain. See Fig. 3-6 for calculation of the frequency content of the energy measure derived from these two filters.

Slide credit: B. Freeman and A. Torralba

How quadrature pair filters work



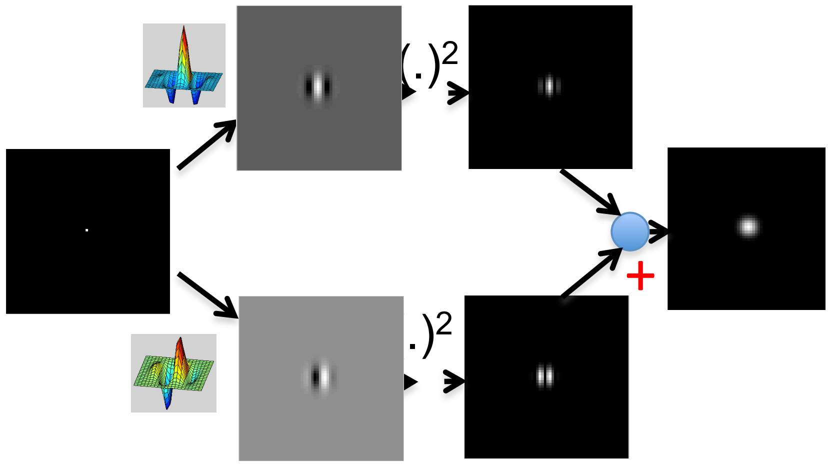


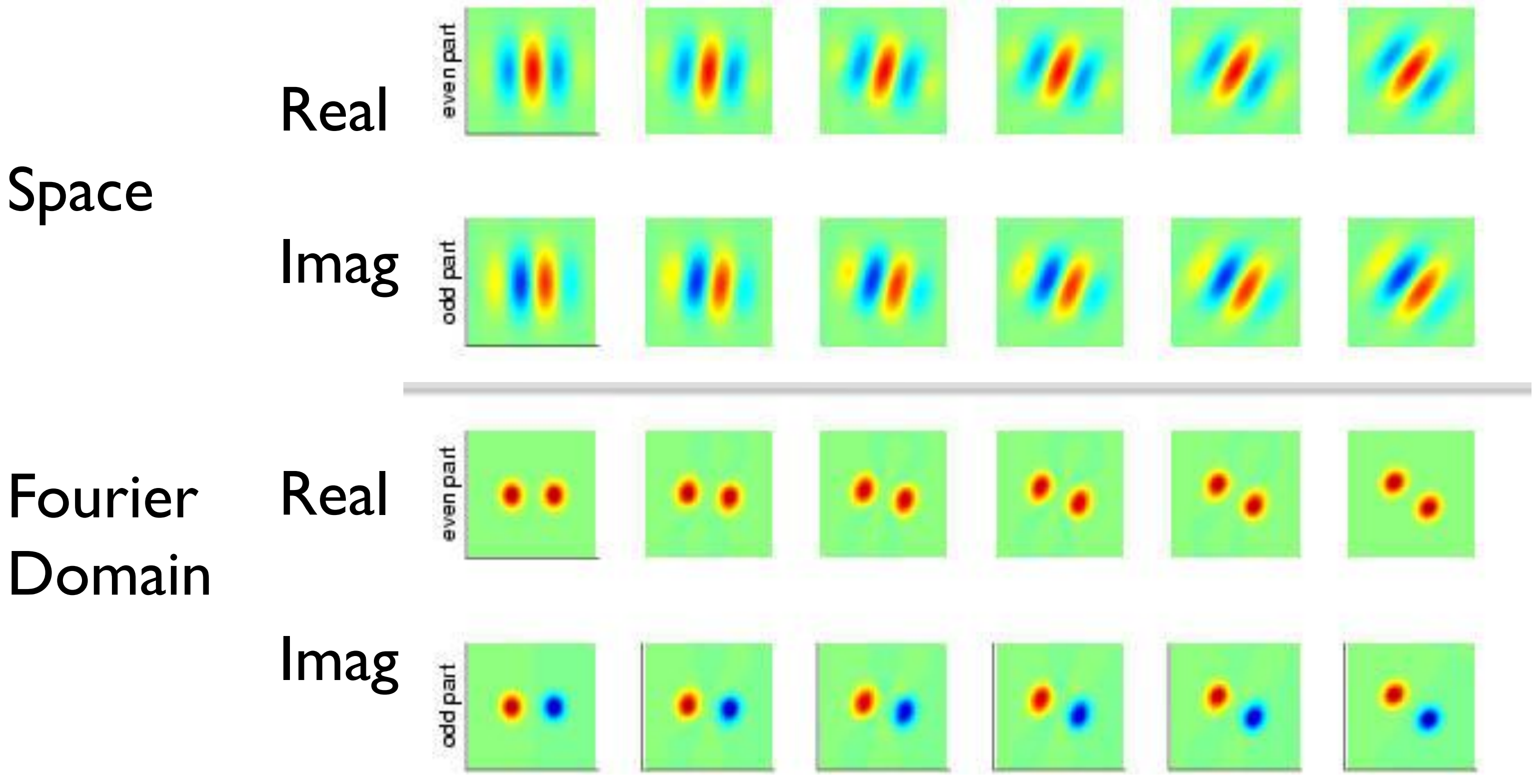
Figure 3-6: Derivation of energy measure frequency content for the filters of Fig. 3-5. (a) Fourier transform of $G * G$. (b) Fourier transform of $H * H$. Each squared response has 3 lobes in the frequency domain, arising from convolution of the frequency domain responses. The center lobe is modulated down in frequency while the two outer lobes are modulated up. (There are two sign changes which combine to give the signs shown in (b). To convolve H with itself, we flip it in f_x and f_y, which interchanges the $+$ and $-$ lobes of Fig. 3-5 (b). Then we slide it over an unflipped version of itself, and integrate the product of the two. That operation will give positive outer lobes, and a negative inner lobe. However, H has an imaginary frequency response, so multiplying it by itself gives an extra factor of -1, which yields the signs shown in (b)). (c) Fourier transform of the energy measure, $G * G + H * H$. The high frequency lobes cancel, leaving only the baseband spectrum, which has been demodulated in frequency from the original bandpass response. This spectrum is proportional to the sum of the auto-correlation functions of either lobe of Fig. 3-5 (a) and either lobe of Fig. 3-5 (b).

Slide credit: B. Freeman and A. Torralba

Oriented Filters

- Gabor wavelet: $\psi(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} e^{j2\pi u_0 x}$

- Tuning filter orientation: $$x' = \cos(\alpha)x + \sin(\alpha)y$$ $$y' = -\sin(\alpha)x + \cos(\alpha)y$$

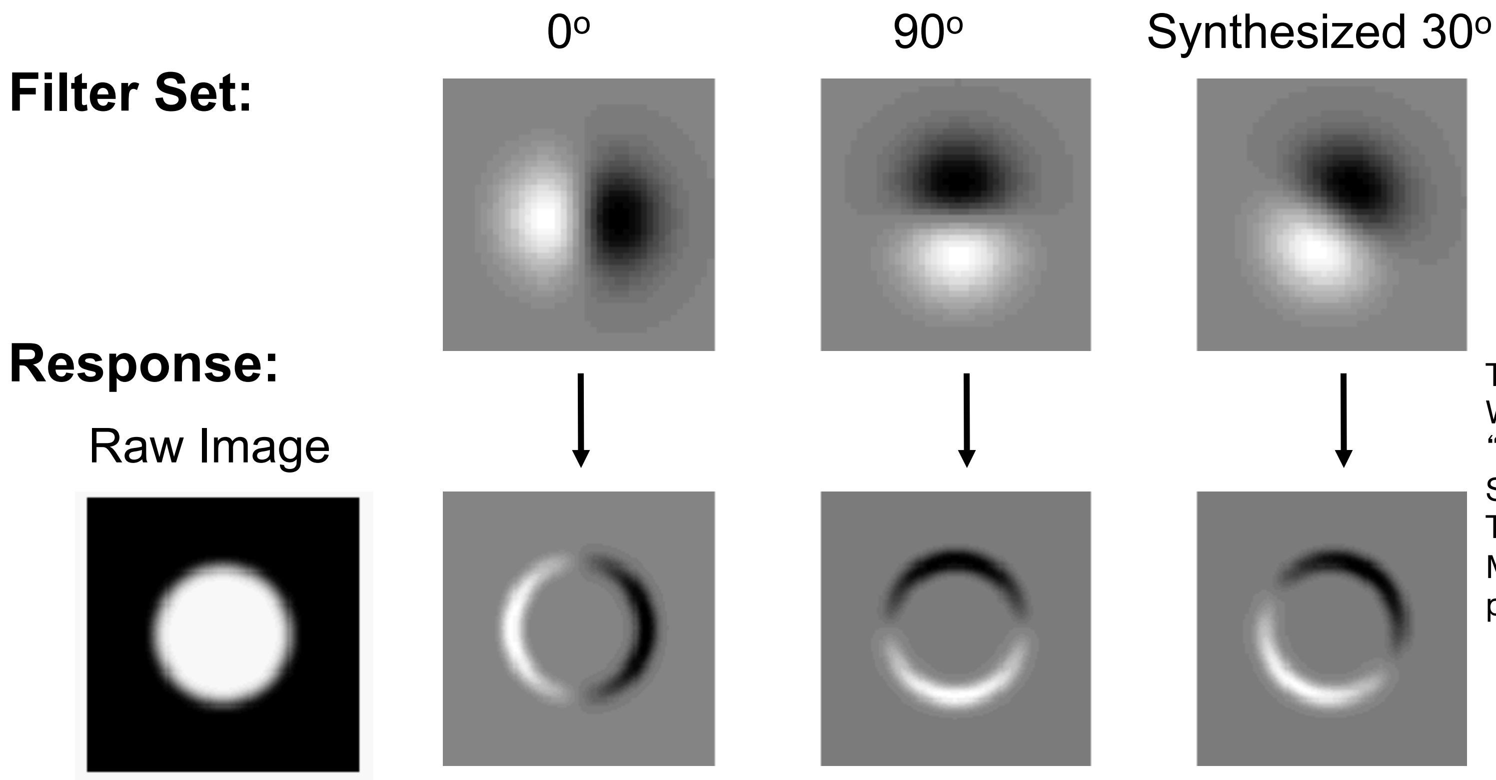


Slide credit: B. Freeman and A. Torralba

Simple example

"Steerability"-- the ability to synthesize a filter of any orientation from a linear combination of filters at fixed orientations.

$$G_\theta^1 = \cos(\theta) G_0^1 + \sin(\theta) G_{90}^1$$



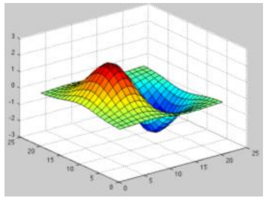
Taken from:
W. Freeman, T. Adelson, "The Design and Use of Sterrable Filters", IEEE Trans. Patt, Anal. and Machine Intell., vol 13, #9, pp 891-900, Sept 1991

Slide credit: B. Freeman and A. Torralba

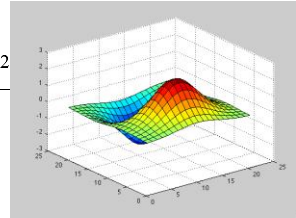
Steerable filters

Derivatives of a Gaussian:

$$h_x(x,y) = \frac{\partial h(x,y)}{\partial x} = \frac{-x}{2\pi\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}}$$



$$h_y(x,y) = \frac{\partial h(x,y)}{\partial y} = \frac{-y}{2\pi\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}}$$



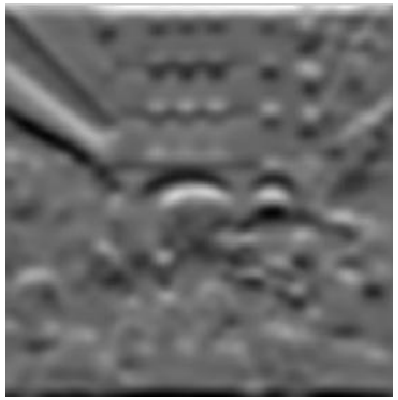
An arbitrary orientation can be computed as a linear combination of those two basis functions:

$$h_\alpha(x,y) = \cos(\alpha) h_x(x,y) + \sin(\alpha) h_y(x,y)$$

The representation is "shiftable" on orientation: We can interpolate any other orientation from a finite set of basis functions.

cos(〈)

+sin(〈)



=

Freeman & Adelson, 1992

Slide credit: B. Freeman and A. Torralba

Steerable filters

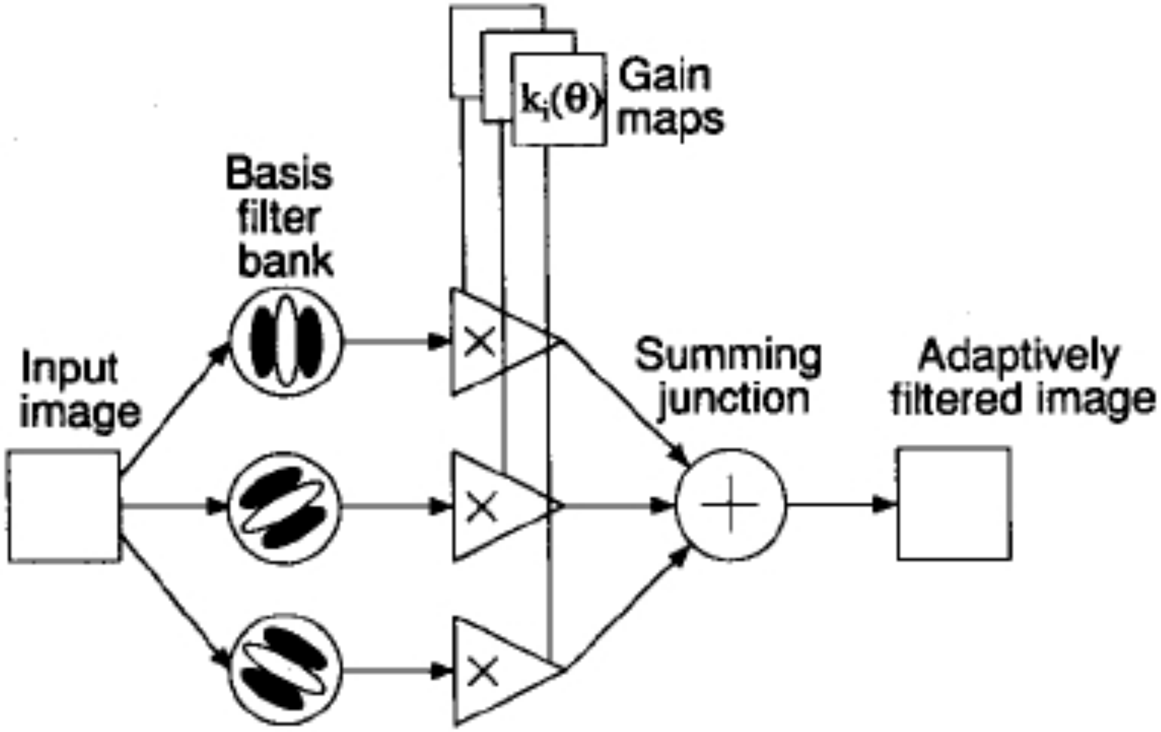


Fig. 3. Steerable filter system block diagram. A bank of dedicated filters process the image. Their outputs are multiplied by a set of gain maps that adaptively control the orientation of the synthesized filter.

Slide credit: B. Freeman and A. Torralba