Image Filtering (Frequency Domain)

Dr. Sander Ali Khowaja,
Assistant Professor, Department of Telecommunication Engineering
Faculty of Engineering and Technology, University of Sindh, Pakistan
Senior Member, IEEE – Member, ACM
https://sander-ali.github.io

Computer Vision & Image Processing

Some AI related news

Augmented reality headset enables users to see hidden objects

The device could help workers locate objects for fulfilling e-commerce orders or identify parts for assembling products.

Watch Video

Adam Zewe | MIT News Office
February 27, 2023

Previous Class

- Linear filtering is sum of dot product at each position
  - Can smooth, sharpen, translate (among many other uses)
- Gaussian filters
  - Low pass filters, separability, variance
- Attend to details:
  - filter size, extrapolation, cropping
- Applications:
  - Noise models and nonlinear image filters
  - Texture representation

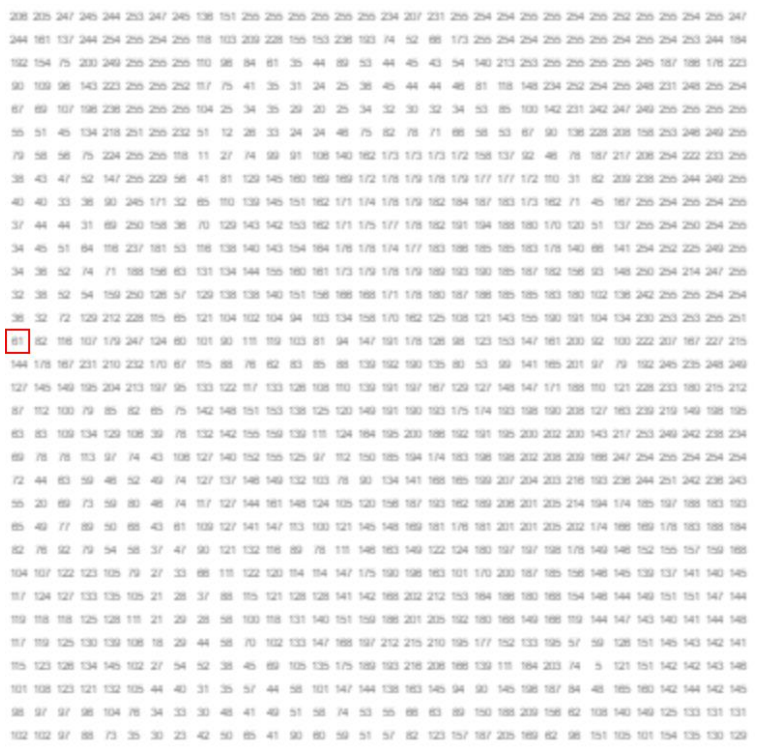
$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

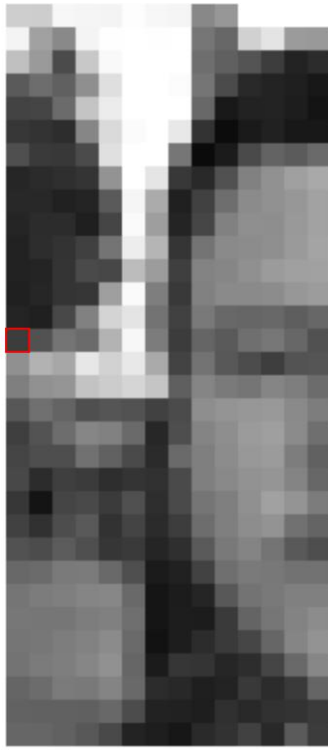
Today's class

- Review of image filtering in spatial domain
- Fourier transform and frequency domain
- Frequency view of filtering
- Image downsizing and interpolation
- Goals:
  - Understand 2D Fourier transform
  - Understand how to implement filtering in Fourier domain
  - Understand aliasing and how to prevent aliasing

Application: Photometric Stereo

- http://setosa.io/ev/image-kernels/



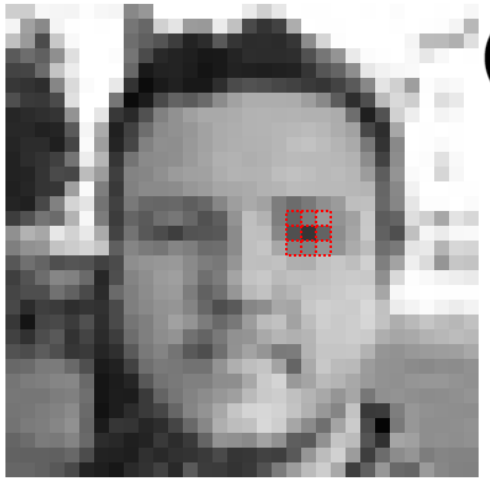


Let's walk through applying the following 3x3 **sharpen** kernel to the image of a face from above.

sharpen

$$\begin{pmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{pmatrix}$$

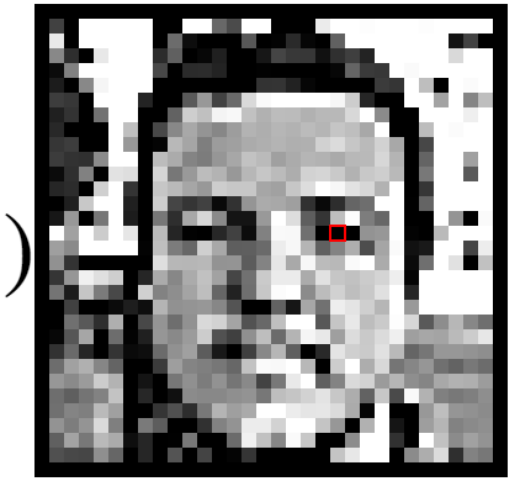
Below, for each 3x3 block of pixels in the image on the left, we multiply each pixel by the corresponding entry of the kernel and then take the sum. That sum becomes a new pixel in the image on the right. Hover over a pixel on either image to see how its value is computed.



input image

(98 × 0 + 123 × -1 + 153 × 0
+ 80 × -1 + 53 × 5 + 99 × -1
+ 129 × 0 + 127 × -1 + 148 × 0)
= -164

kernel: sharpen



output image

Review : Questions

Filtering Operator

Fill in the blanks:

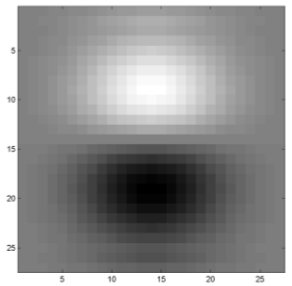
a) $\_ = D * B$

b) $\bar{A} = \_ * \_$

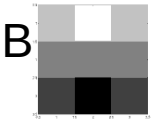
c) $F = \bar{D} * \_$

d) $\_ = D * \bar{D}$

A

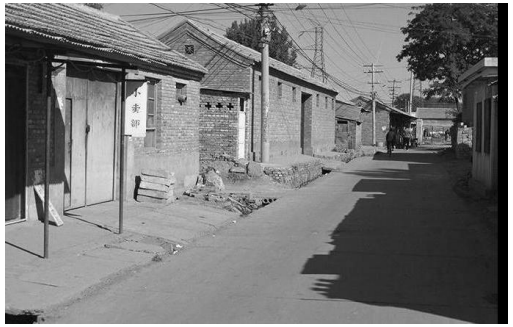


B

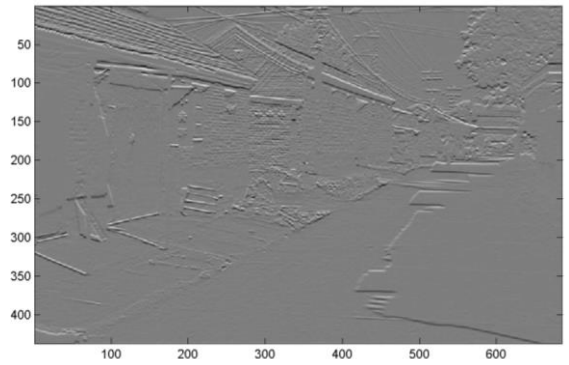


E

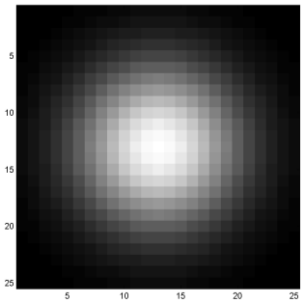
F



G

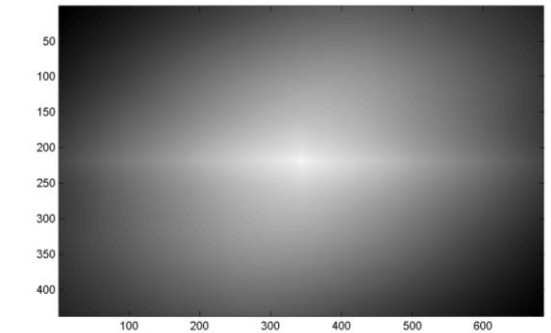


C

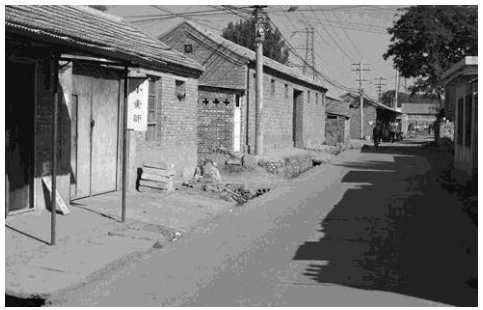


H

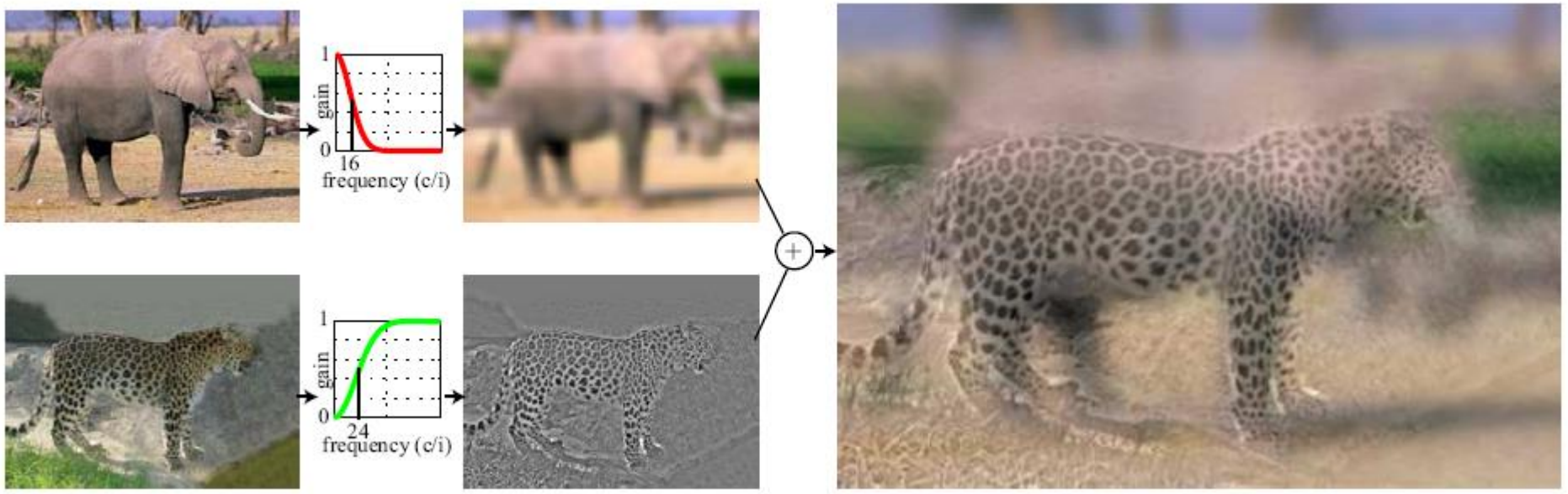
I



D



Hybrid Images

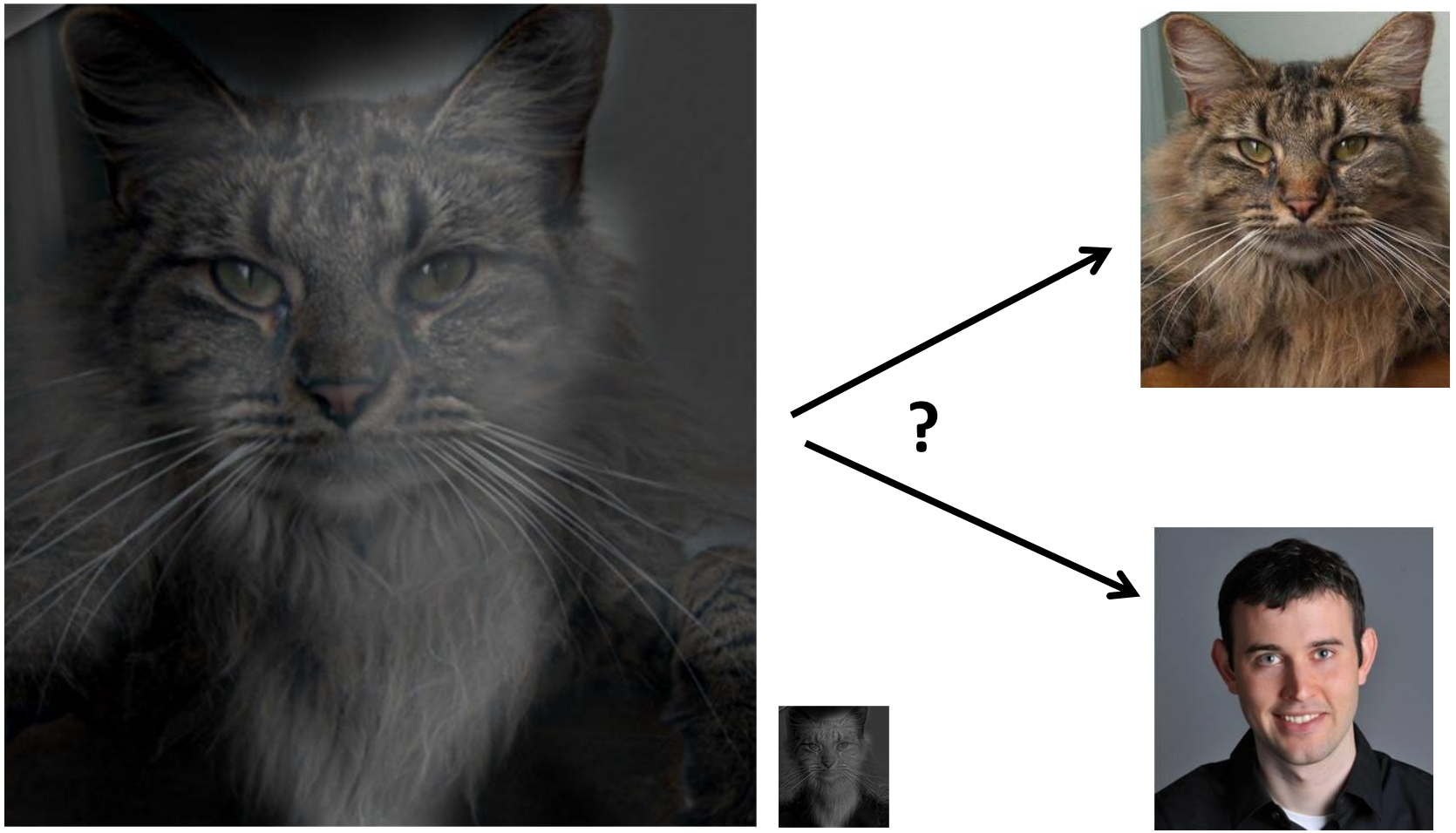


- A. Oliva, A. Torralba, P.G. Schyns, "Hybrid Images," SIGGRAPH 2006

Slide credit: Derek Hoiem

Hybrid Images

Why do we get different, distance-dependent interpretations of hybrid images?



Slide credit: Derek Hoiem

Why?

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

Slide credit: Derek Hoiem

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

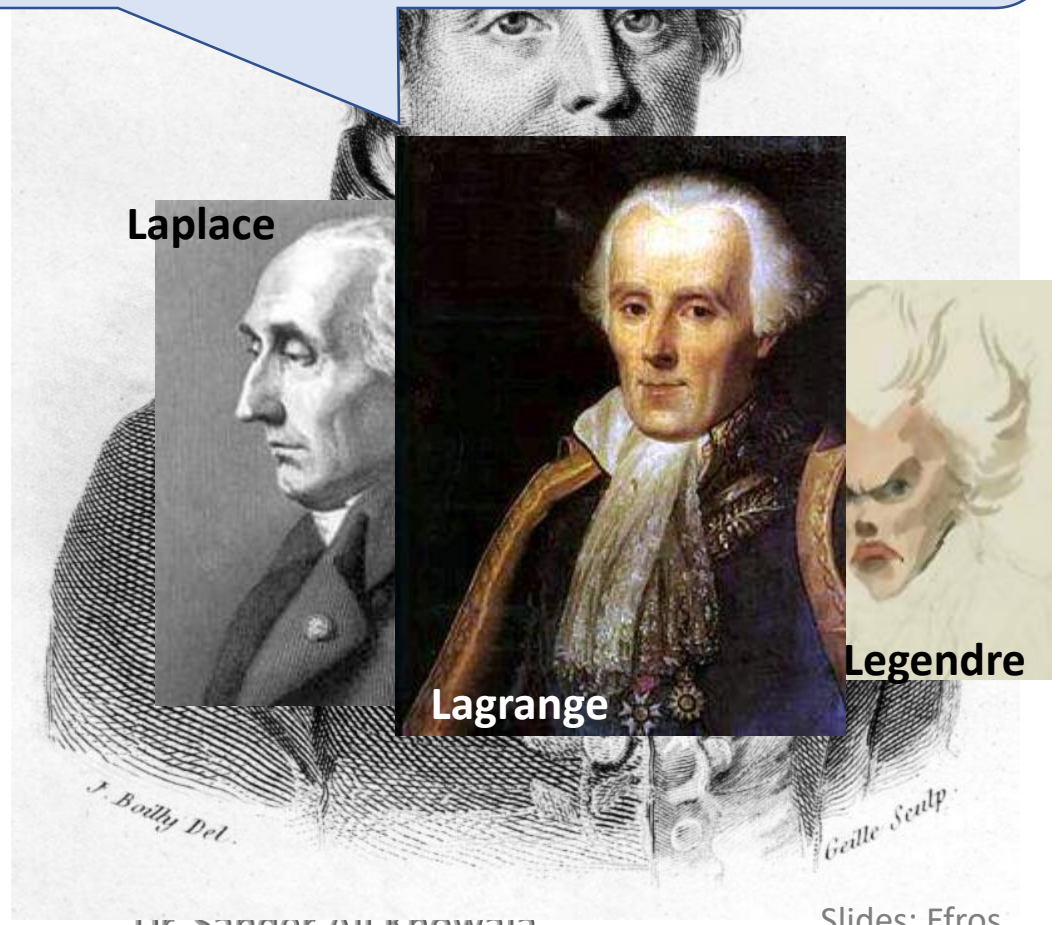
Jean Baptiste Joseph Fourier (1768-1830)

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

> *...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.*



Fourier, Joseph

Fourier, Joseph (1768-1830)

French mathematician who discovered that any periodic motion can be written as a superposition of sinusoidal and cosinusoidal vibrations. He developed a mathematical theory of heat in *Théorie Analytique de la Chaleur (Analytic Theory of Heat)*, (1822), discussing it in terms of differential equations.

Fourier was a friend and advisor of Napoleon. Fourier believed that his health would be improved by wrapping himself up in blankets, and in this state he tripped down the stairs in his house and killed himself. The paper of Galois which he had taken home to read shortly before his death was never recovered.

SEE ALSO: Galois

Add

How would math have changed if the Slanket or Snuggie had been invented?

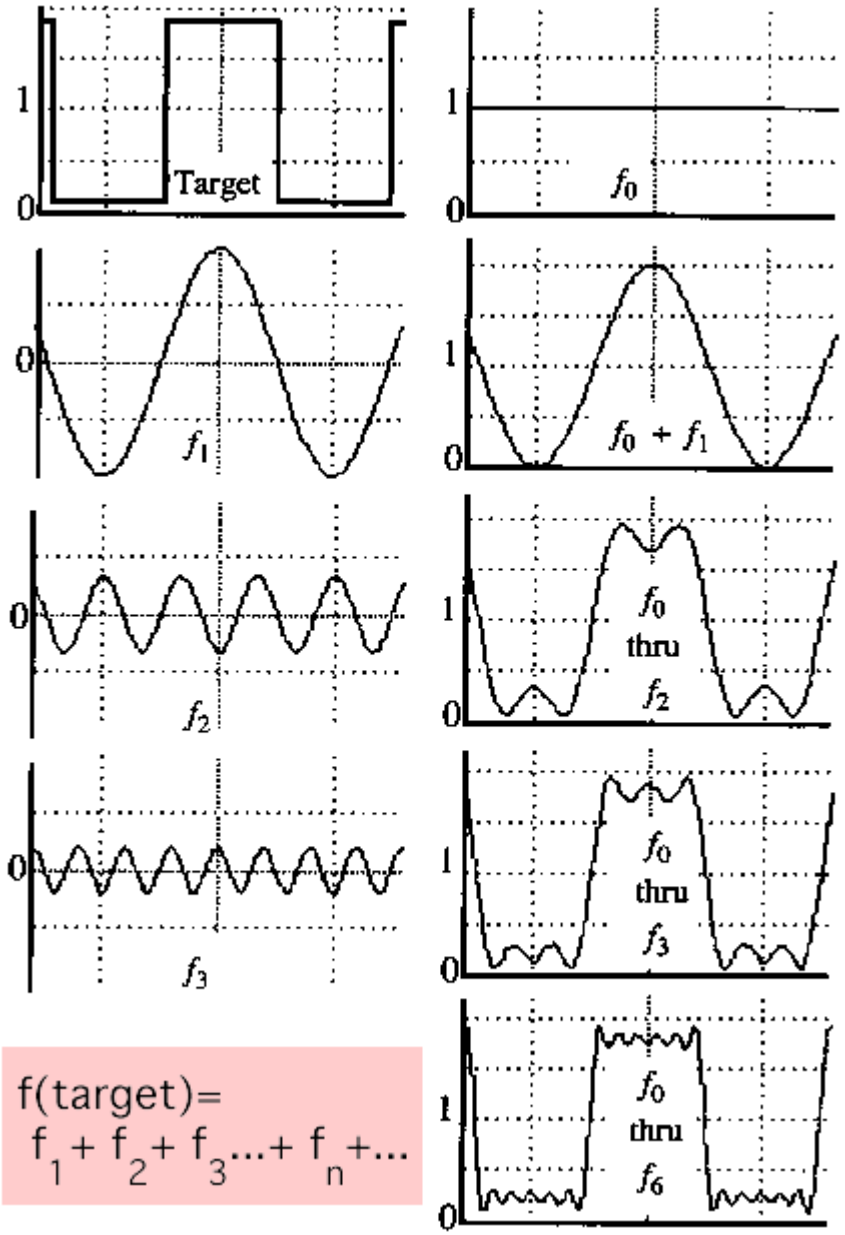
Slide credit: James Hays

A Sum of Sines

Our building block:

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

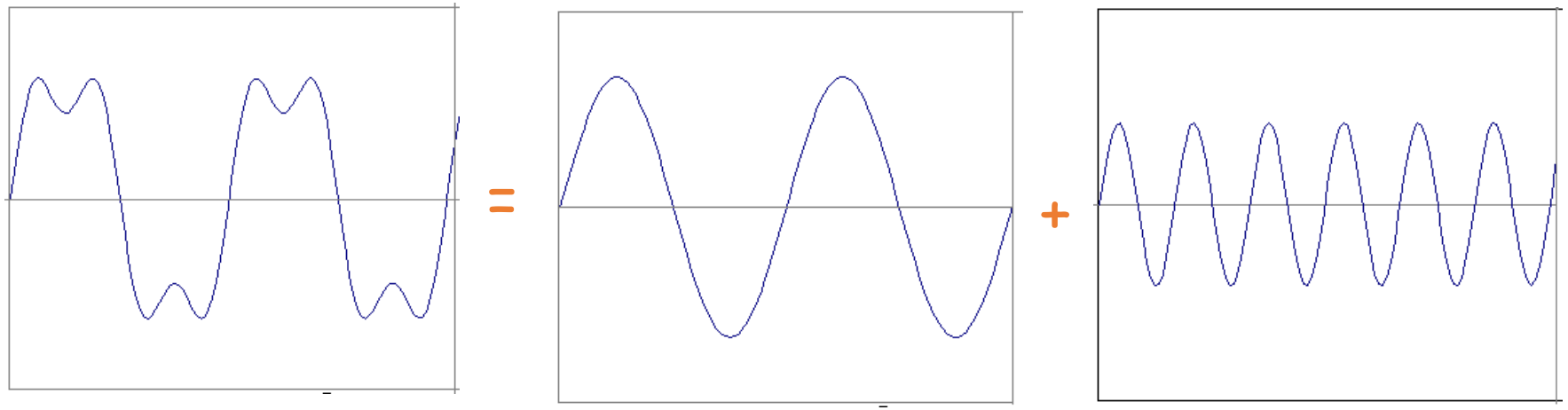
Add enough of them to get any signal *f(x)* you want!

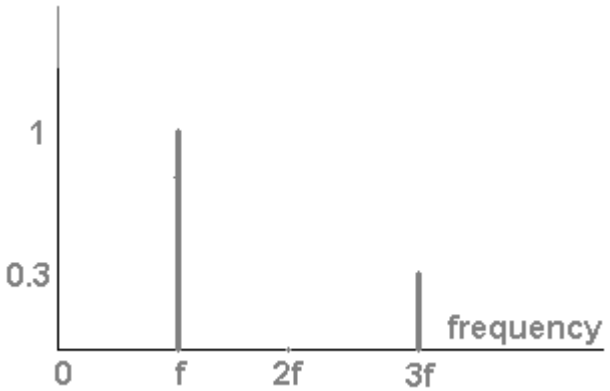


f(target)=
$f_1 + f_2 + f_3 \ldots + f_n + \ldots$

Frequency Spectra

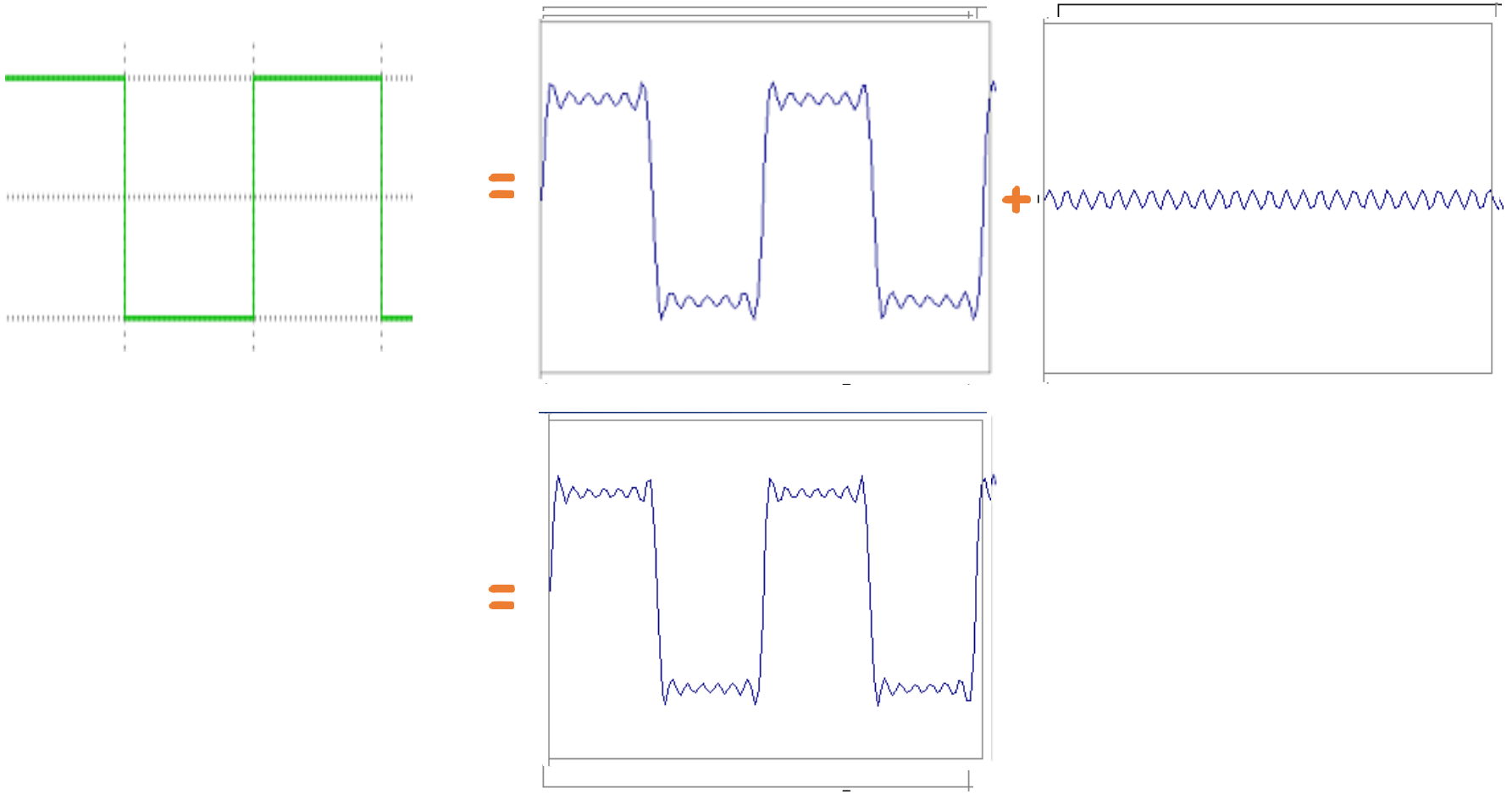
- example : $g(t) = \sin(2\pi f\, t) + (1/3)\sin(2\pi(3f)\, t)$



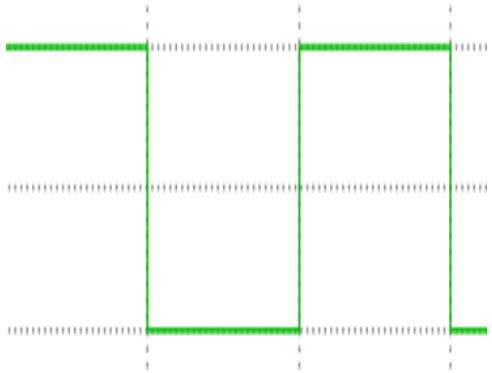


Slides: Efros

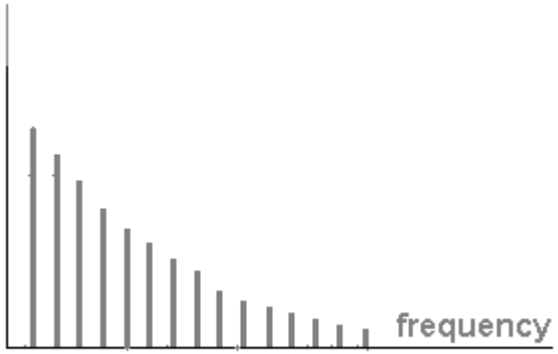
Frequency Spectra



Frequency Spectra

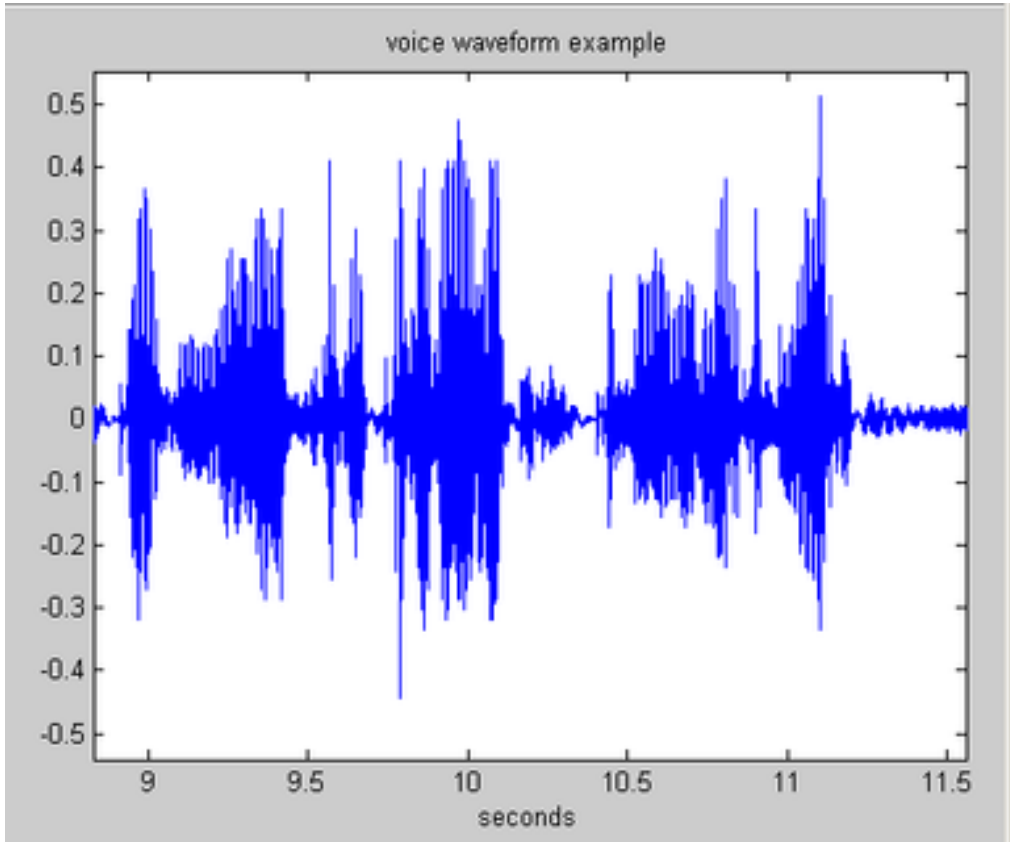


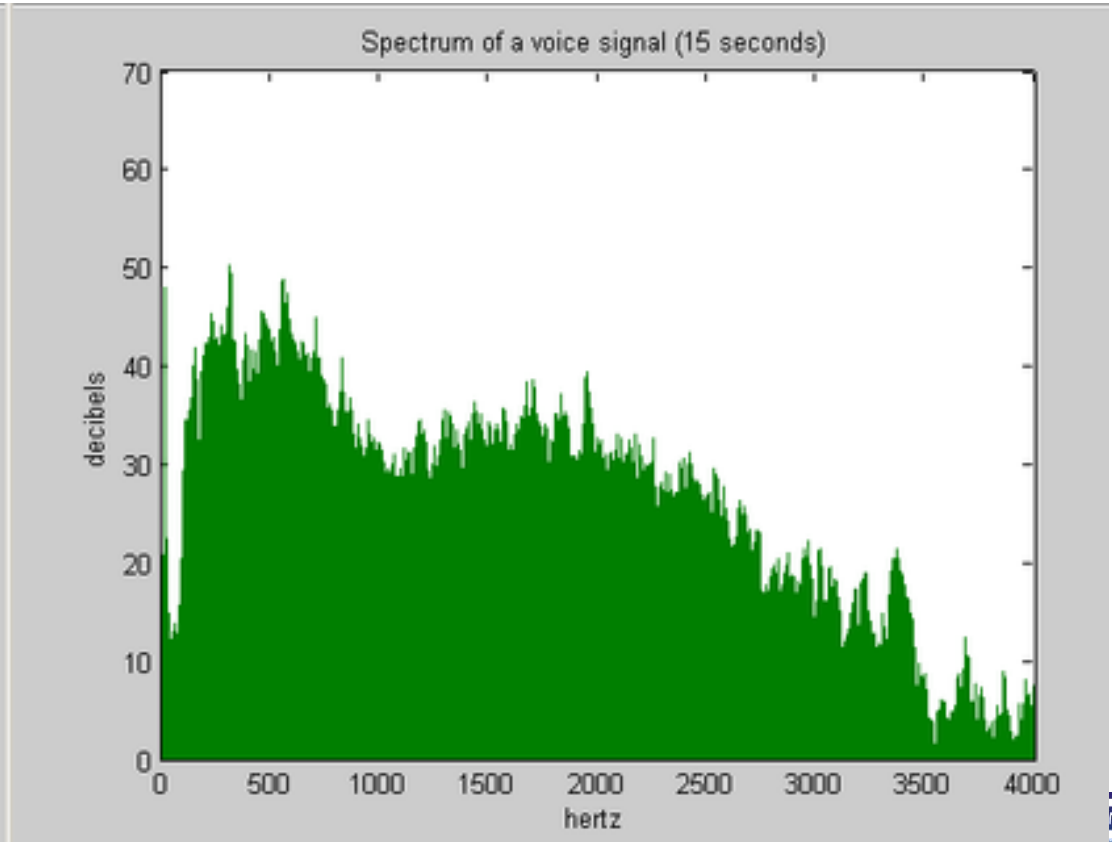
$$= A\sum_{k=1}^{\infty}\frac{1}{k}\sin(2\pi kt)$$



Example: Music

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

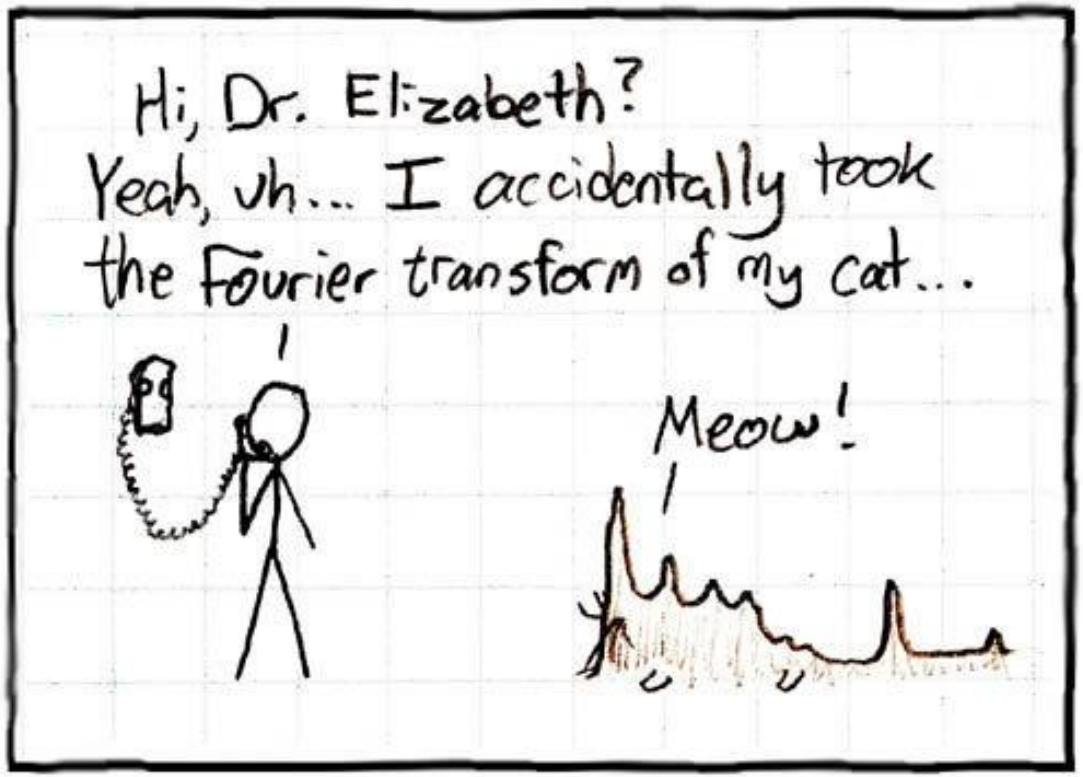




Other Signals

- We can also think of all kinds of other signals the same way

Cats(?)

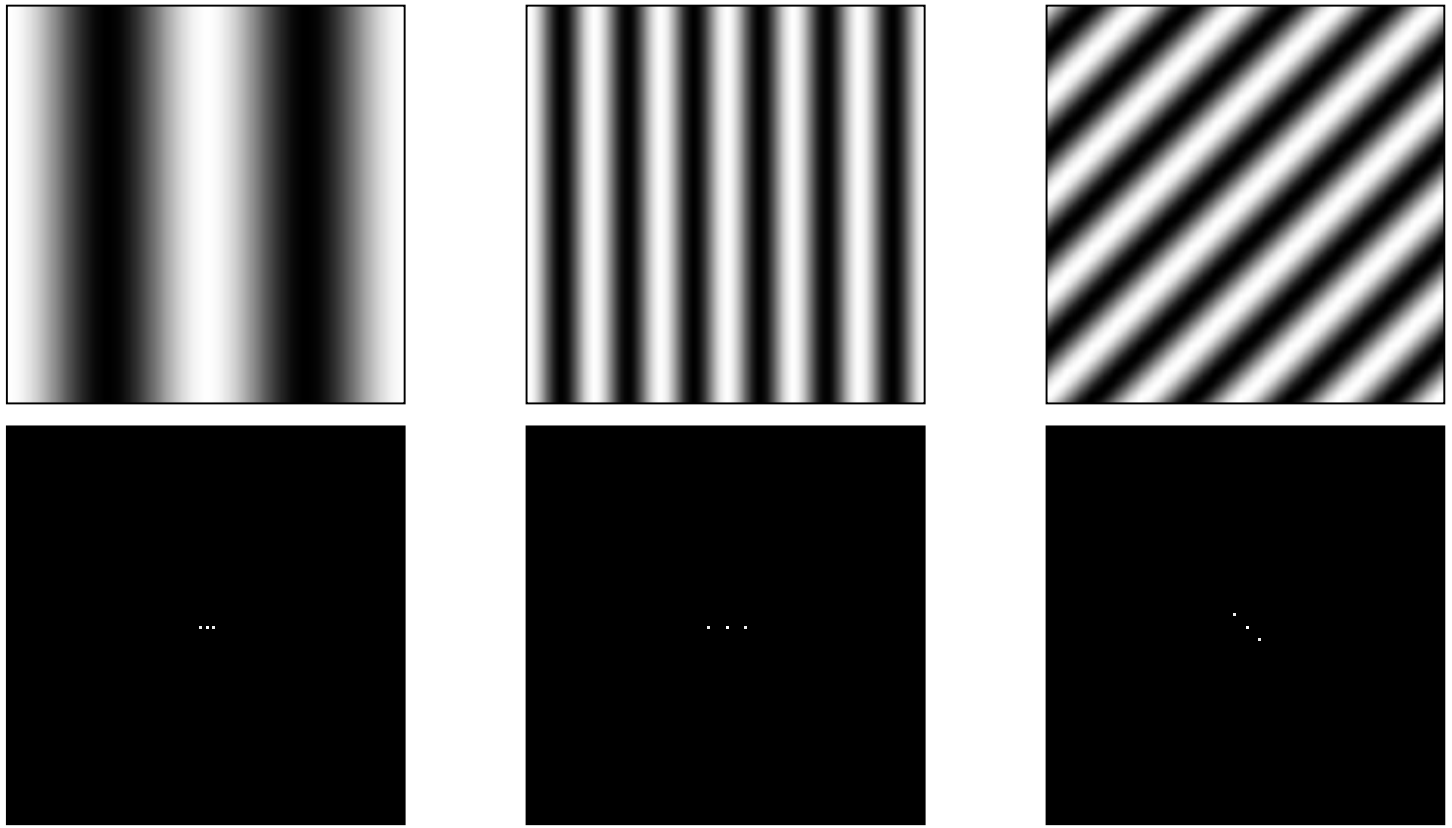


xkcd.com

Fourier Analysis in Images

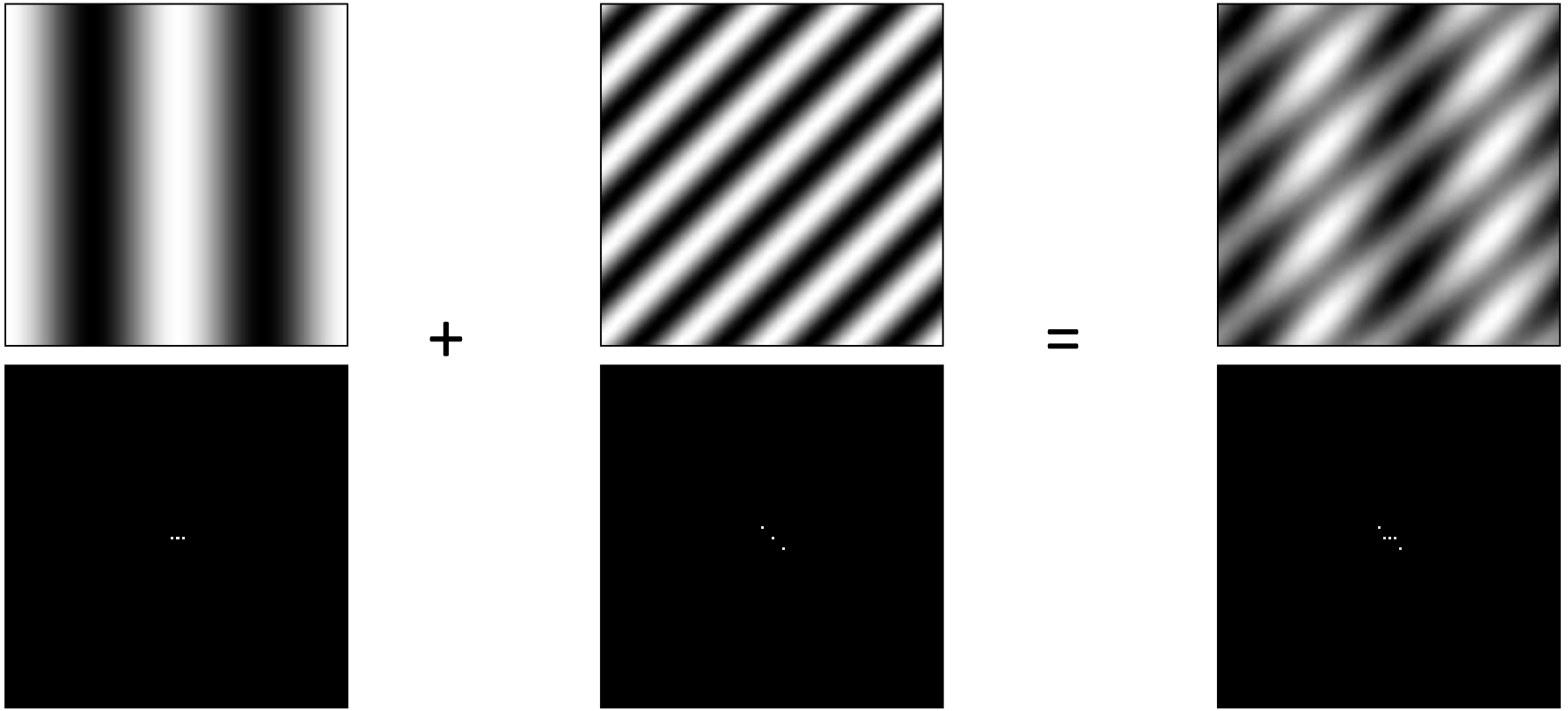
Intensity Image

Fourier Image



http://sharp.bu.edu/~slehar/fourier/fourier.html#filtering

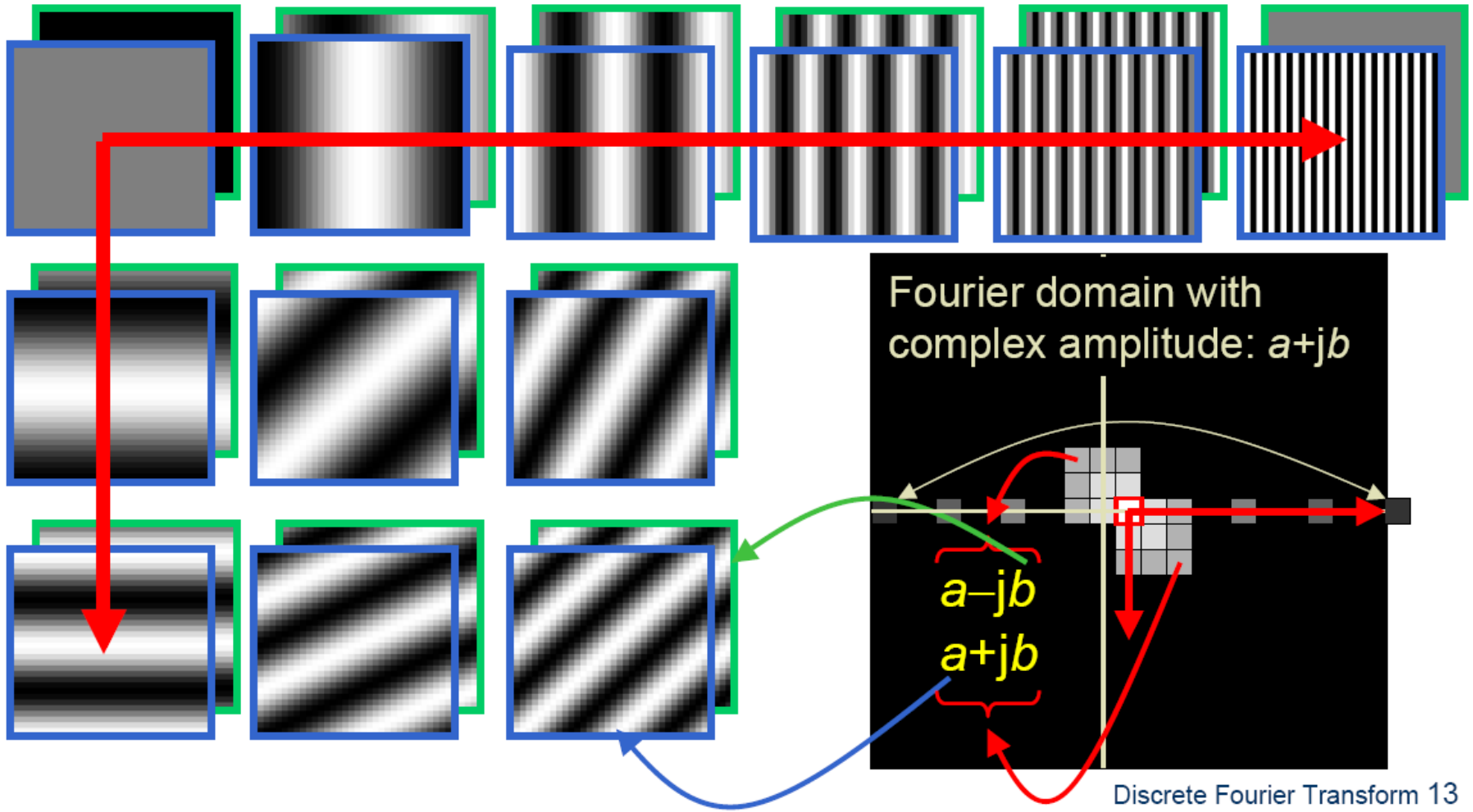
Signals can be composed



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

Fourier Transform

Fourier analysis is a method by which any two-dimensional luminance image can be analyzed into the sum of a set of sinusoidal gratings that differ in spatial frequency, orientation, amplitude and phase.



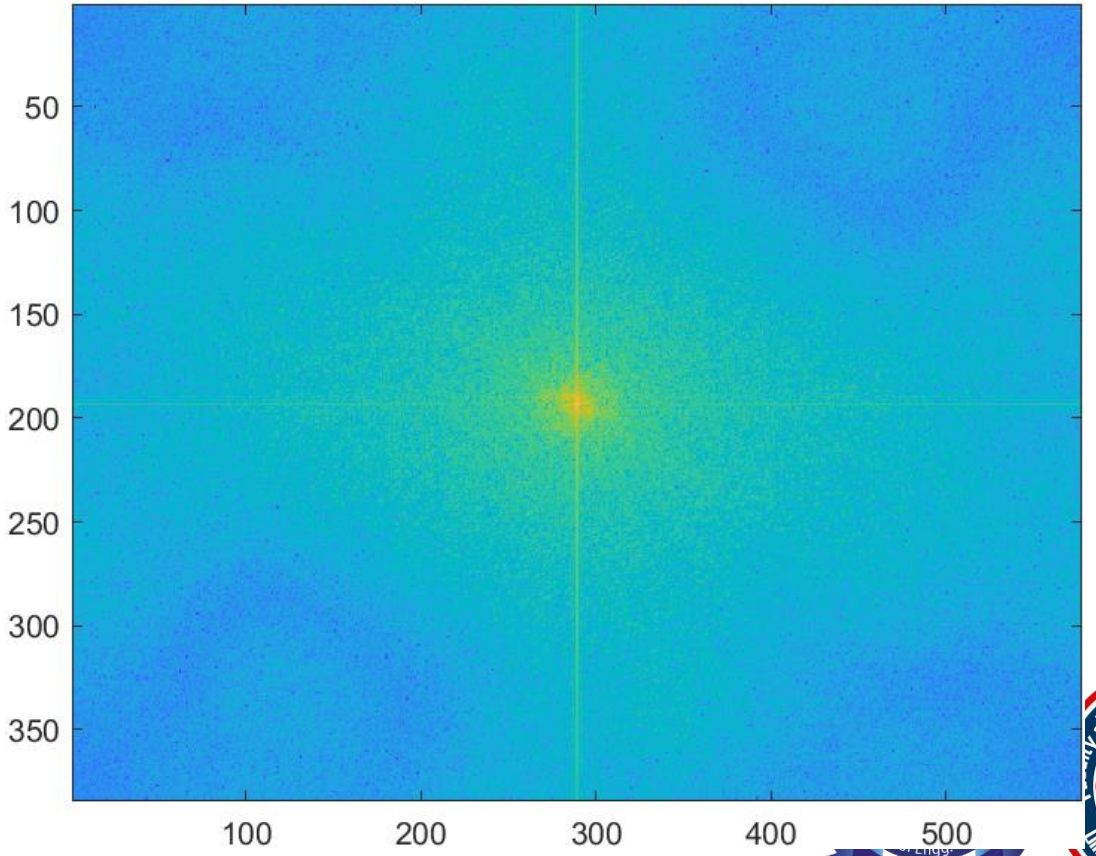
Teases away fast vs. slow changes in the image.

Image as a sum of basis images

Slide credit: A Efros

Example

- im = imread('0901.png’);
- im = rgb2gray(im);
- imagesc(log(abs(fftshift(fft2(im)))));



2D Fourier Transform

Fourier Transform:

$$F(u,v) = \iint_{-\infty}^{\infty} f(x,y) e^{-i2\pi(ux+vy)} dxdy$$

u and v are frequencies along x and y, respectively

Inverse Fourier Transform:

$$f(x,y) = \iint_{-\infty}^{\infty} F(u,v) e^{i2\pi(xu+yv)} dudv$$

Discrete Fourier Transform (DFT):

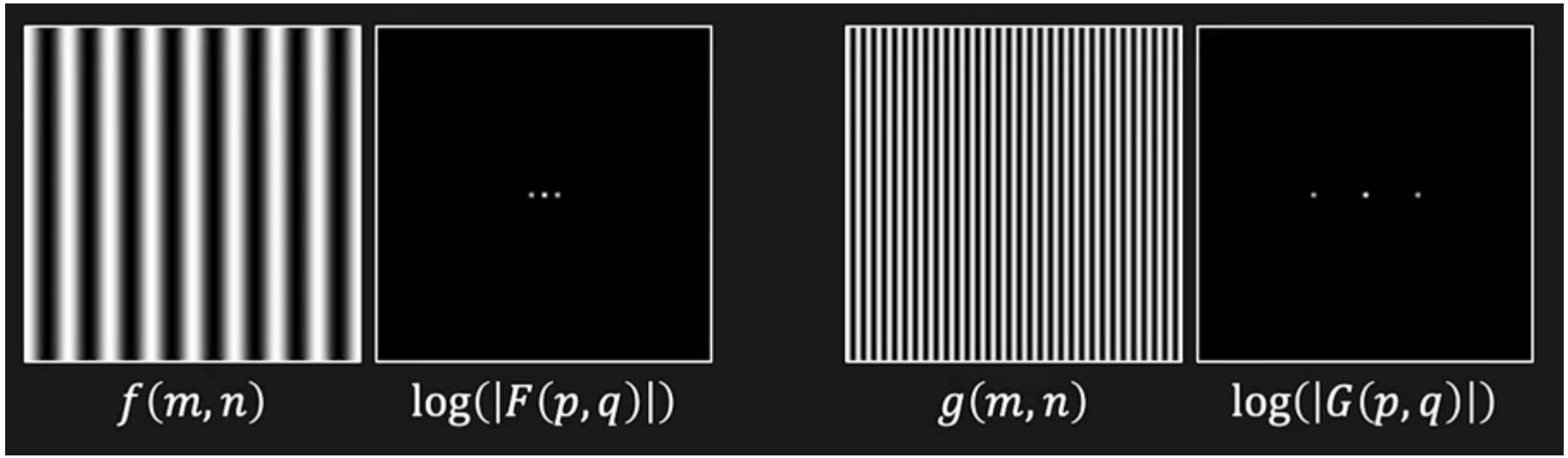
$$F[p,q] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f[m,n] e^{-i2\pi pm/M} e^{-i2\pi qn/N}$$

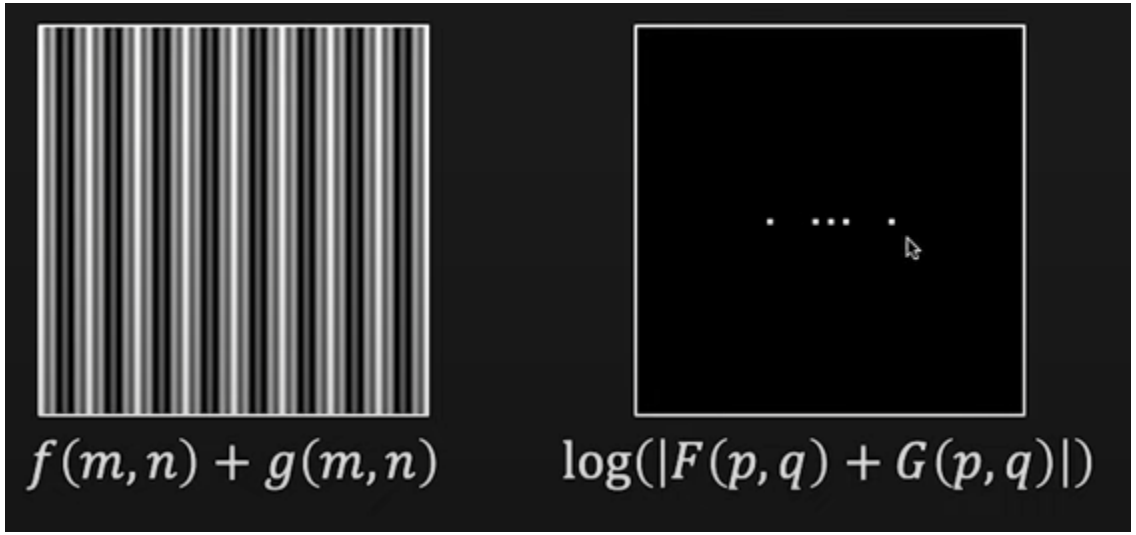
p and q are frequencies along m and n, respectively

Inverse Discrete Fourier Transform (IDFT):

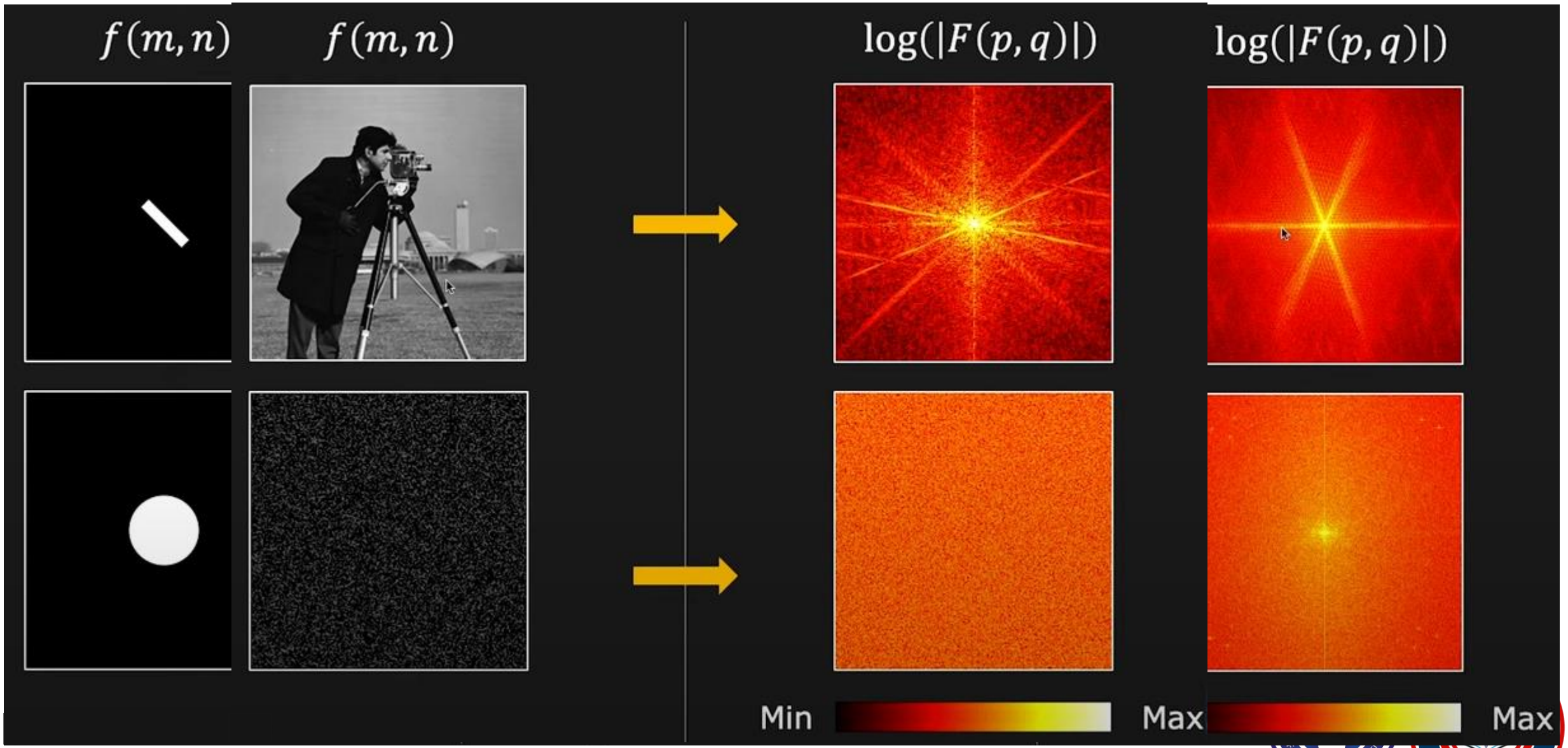
$$f[m,n] = \frac{1}{MN} \sum_{p=0}^{M-1} \sum_{q=0}^{N-1} F[p,q] e^{i2\pi pm/M} e^{i2\pi qn/N}$$

2D Fourier Transform : Example 1





2D Fourier Transform : Example 1



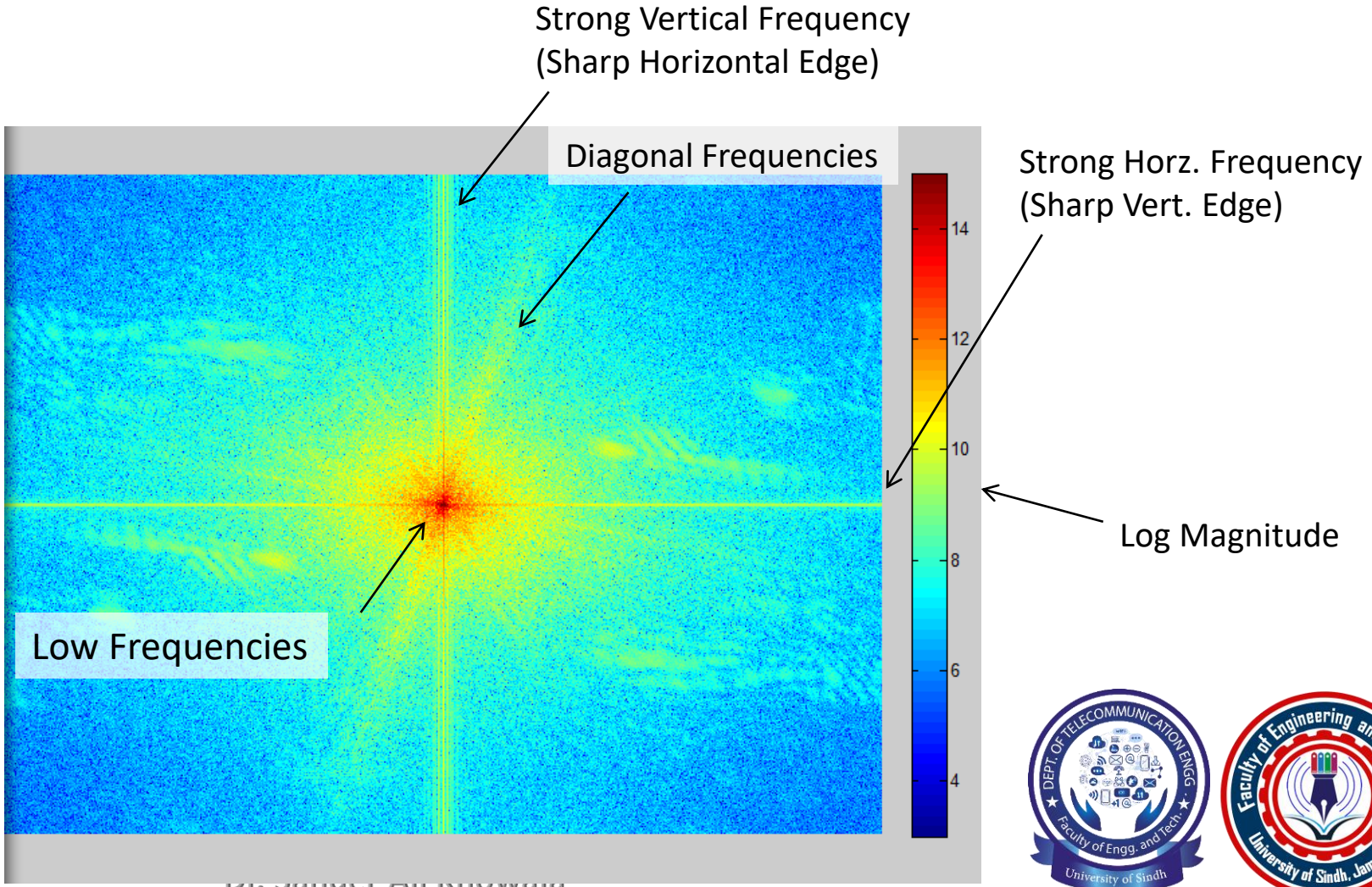
Fourier Transform

- Fourier transform stores the magnitude and phase at each frequency
  - Magnitude encodes how much signal there is at a particular frequency
  - Phase encodes spatial information (indirectly)
  - For mathematical convenience, this is often notated in terms of real and complex numbers

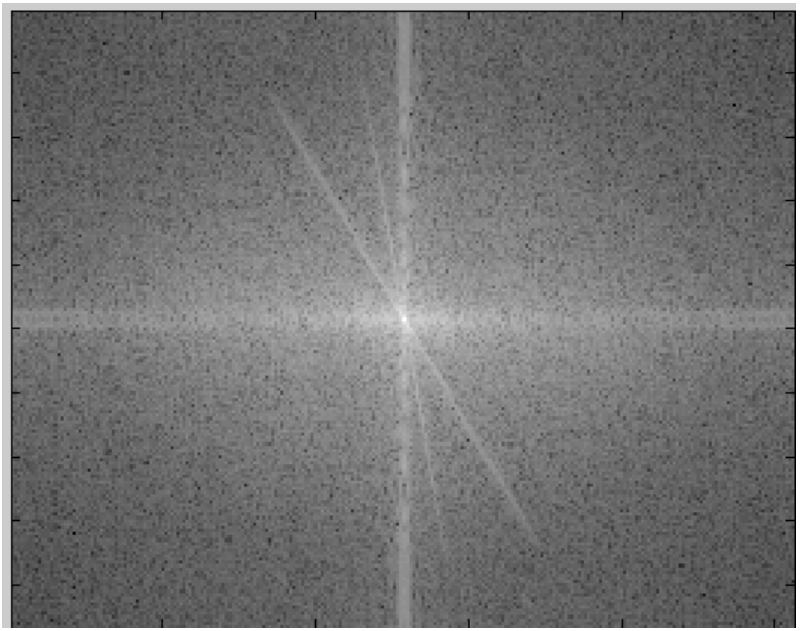
Amplitude: $A = \pm\sqrt{R(\omega)^2 + I(\omega)^2}$ Phase: $\phi = \tan^{-1}\frac{I(\omega)}{R(\omega)}$

Euler's formula: $e^{inx} = \cos(nx) + i\sin(nx)$

Interpreting Magnitude Images

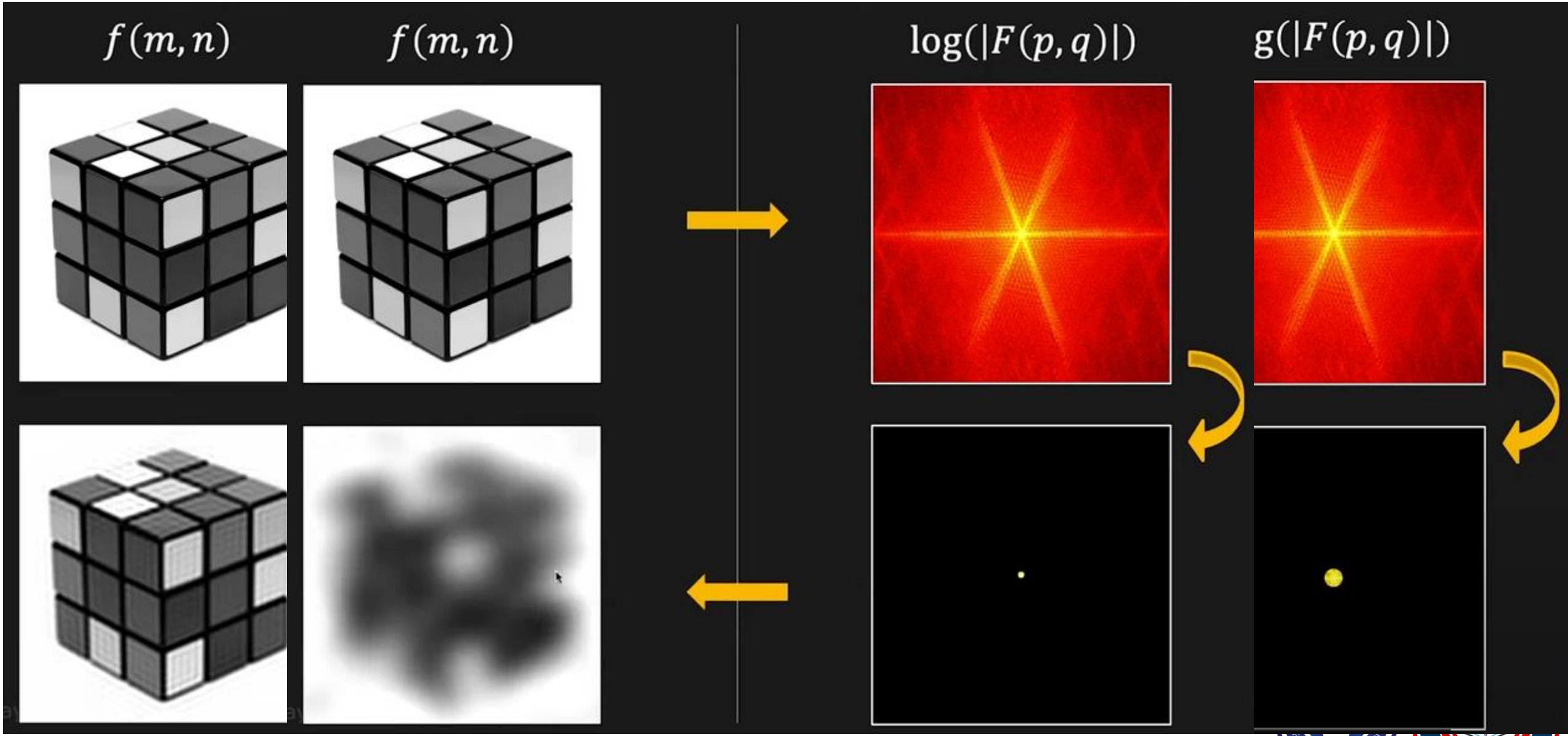


Man-mad Scene

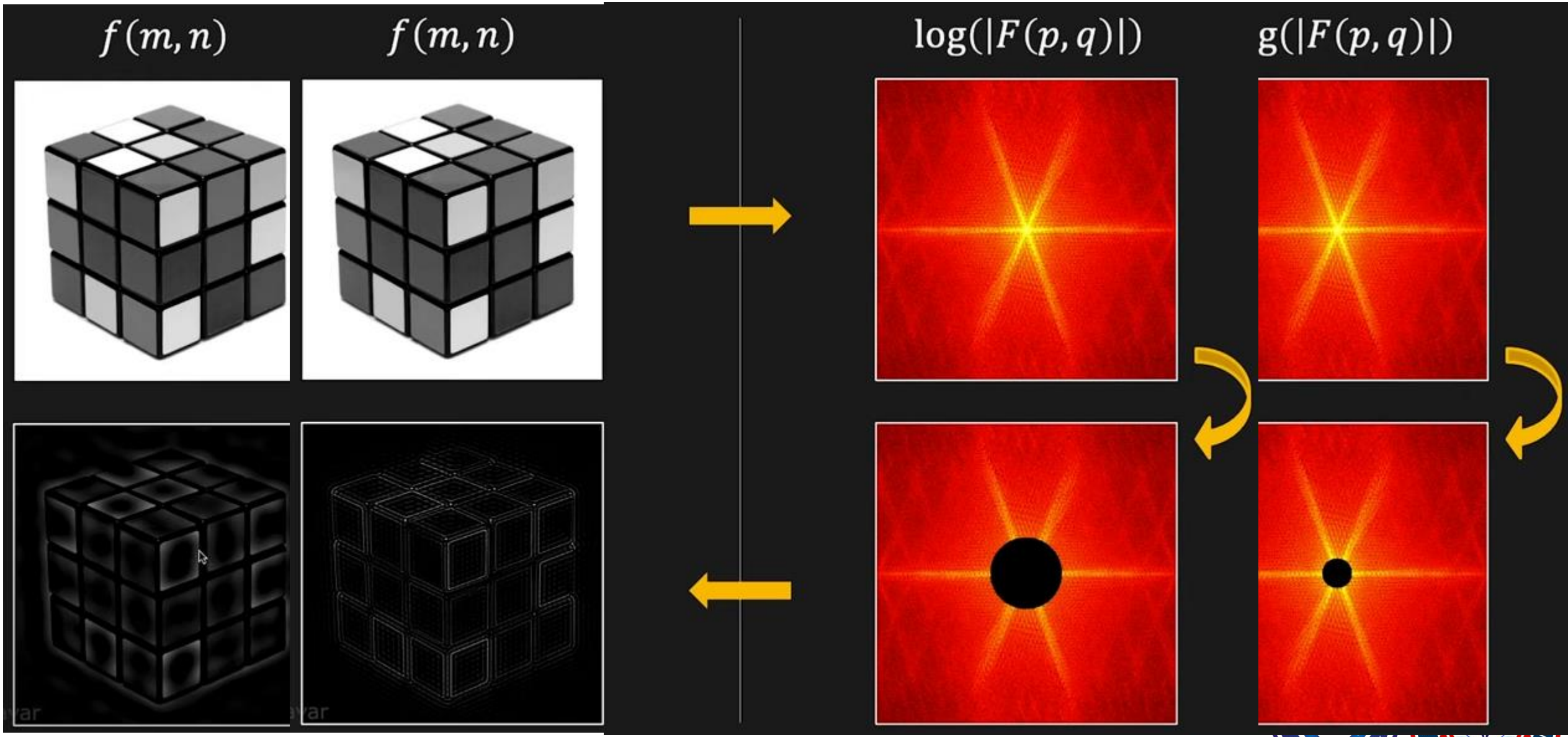


Chan Change Spectrum, then reconstruct

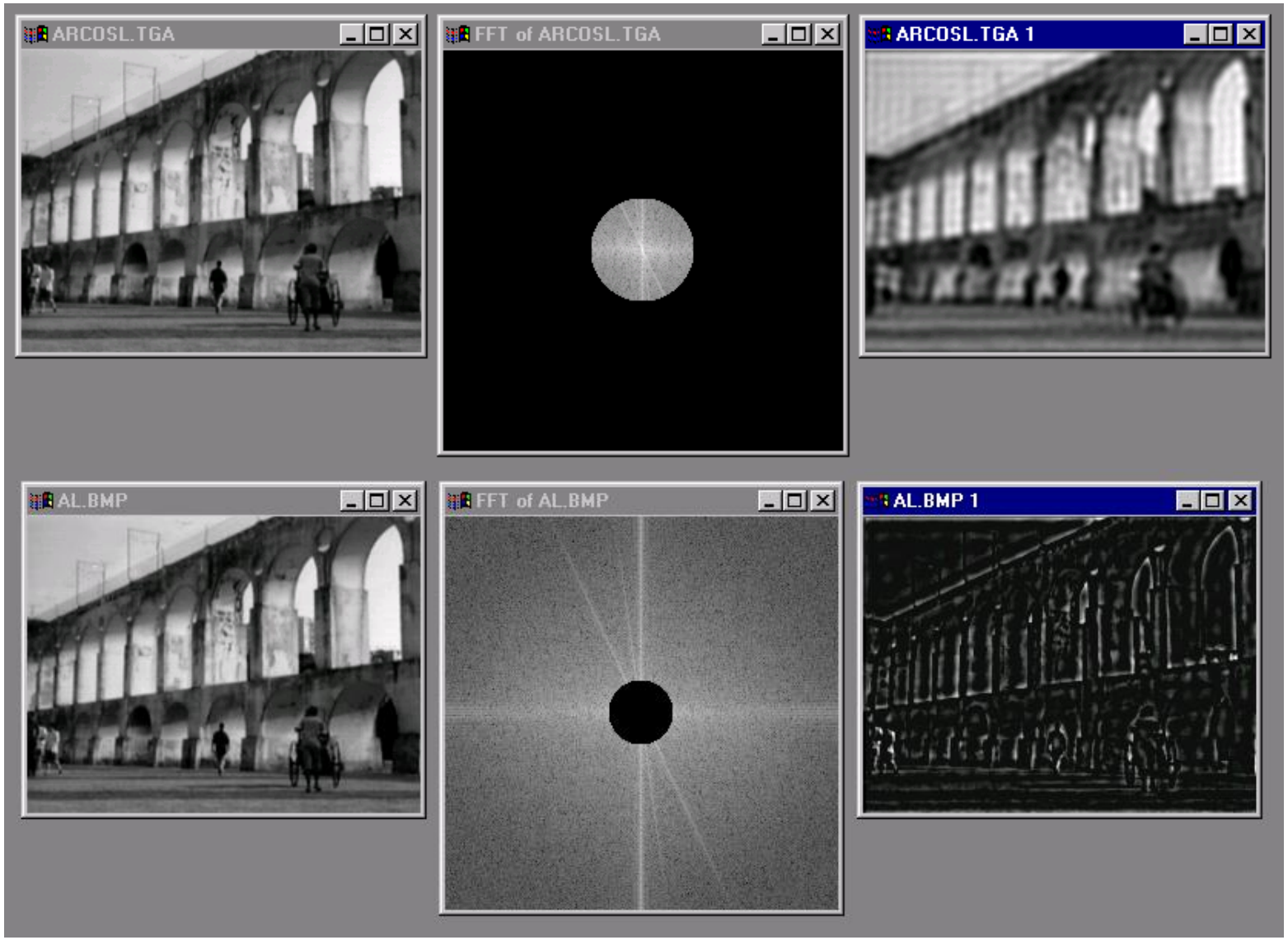
Low Pass Filtering



High Pass Filtering



Low and High Pass Filters



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!

Properties of Fourier Transform

- Linearity $$\mathcal{F}[ax(t) + by(t)] = a\mathcal{F}[x(t)] + b\mathcal{F}[y(t)]$$

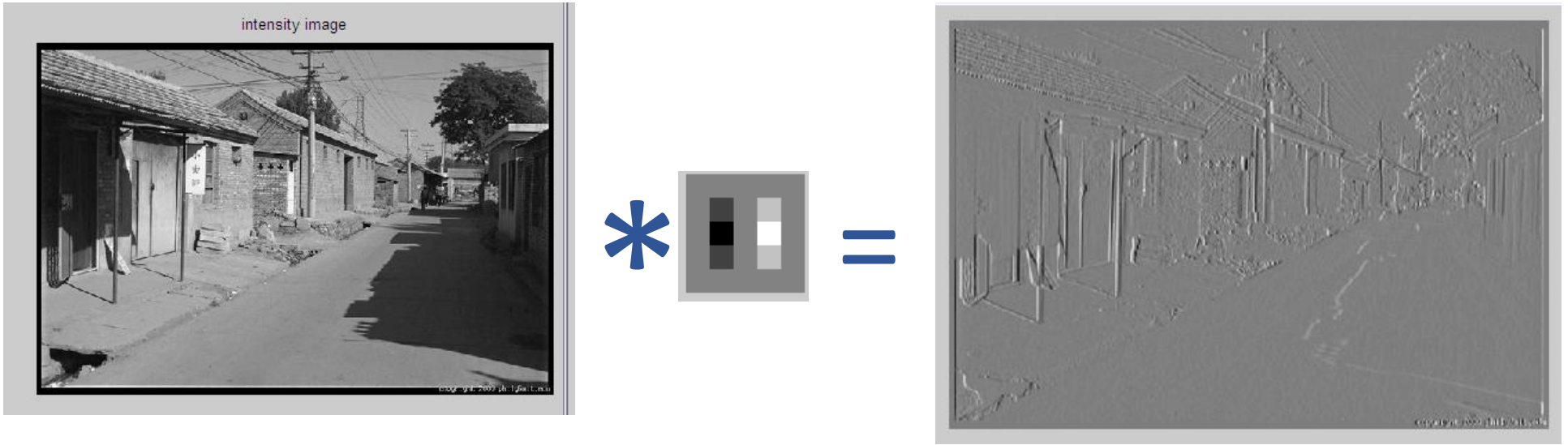
- Fourier transform of a real signal is symmetric about the origin

- The energy of the signal is the same as the energy of its Fourier transform

See Szeliski Book (3.4)

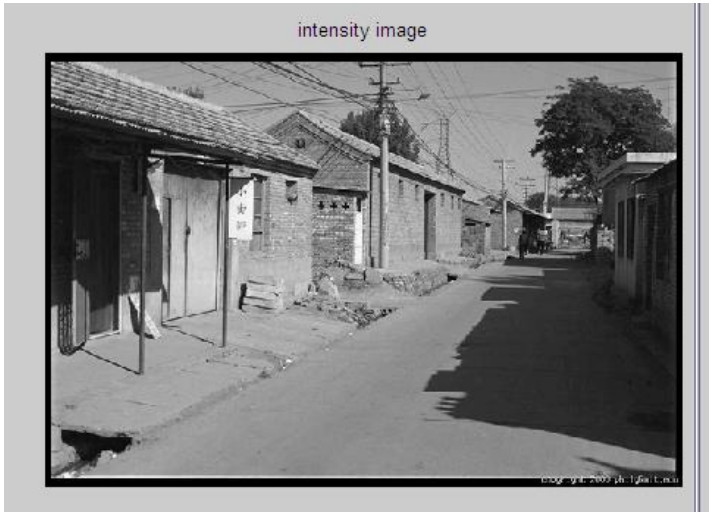
Filtering in Spatial Domain

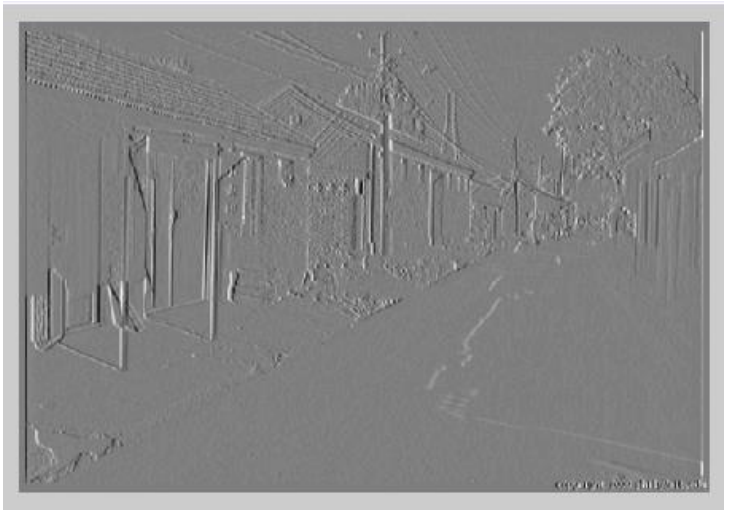
1	0	-1
2	0	-2
1	0	-1



Filtering in Fourier Domain

Spatial domain

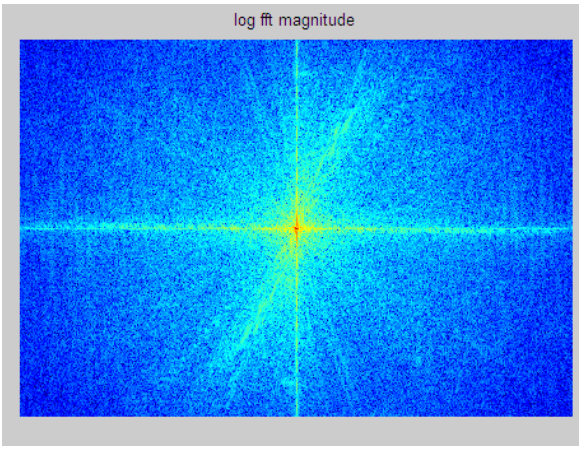


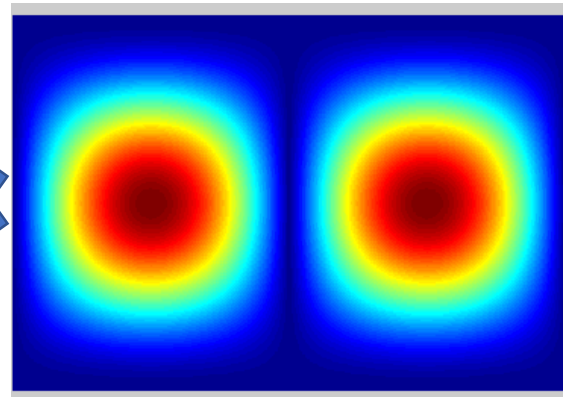


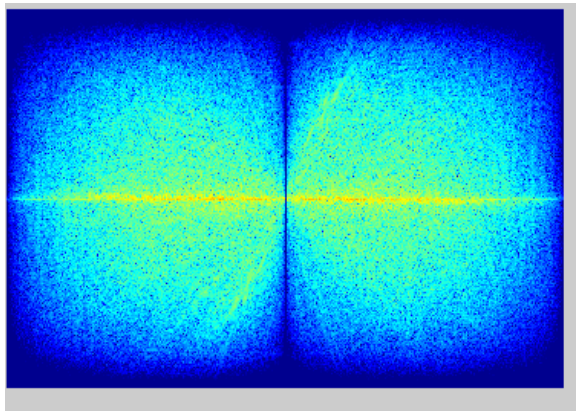
FFT

FFT

Inverse FFT

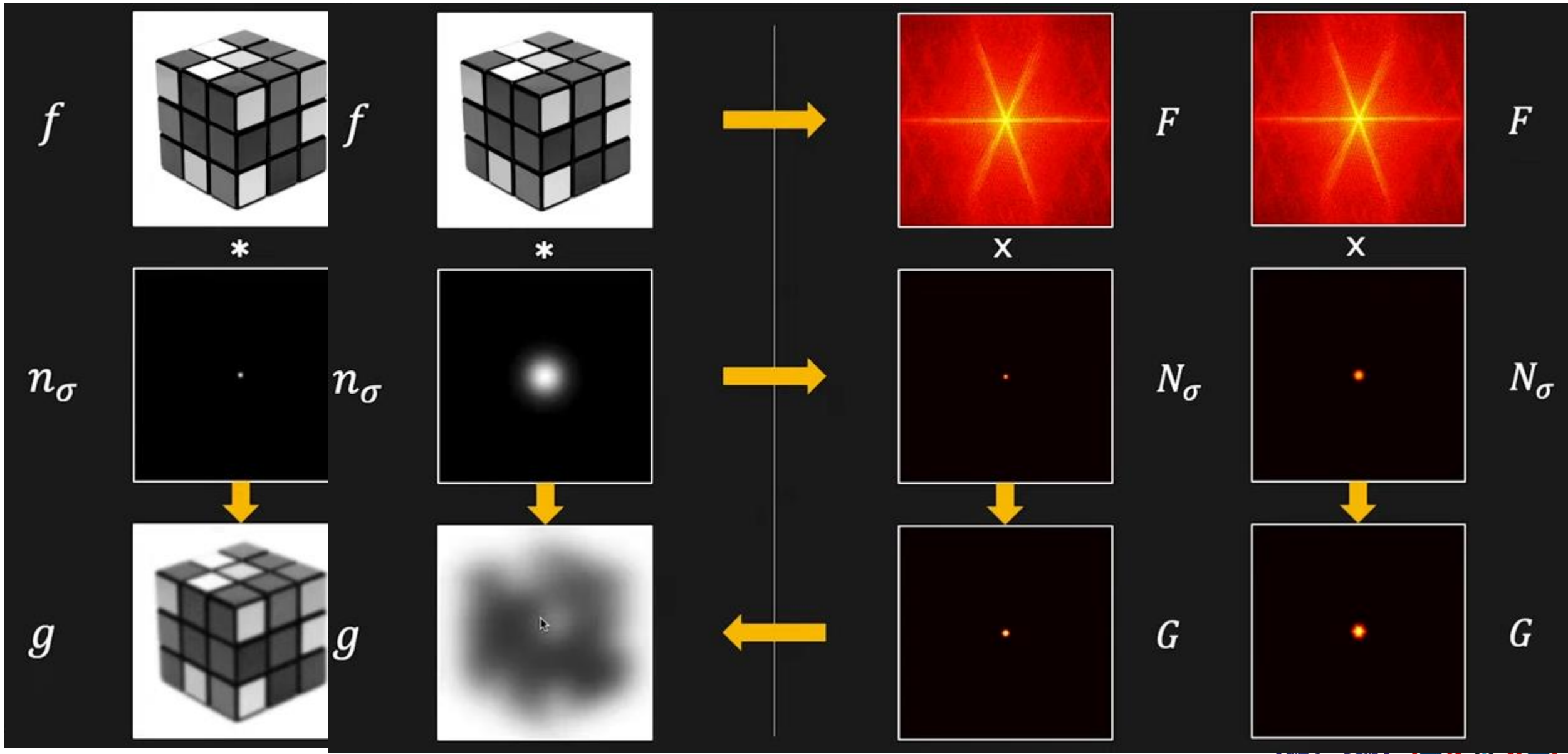






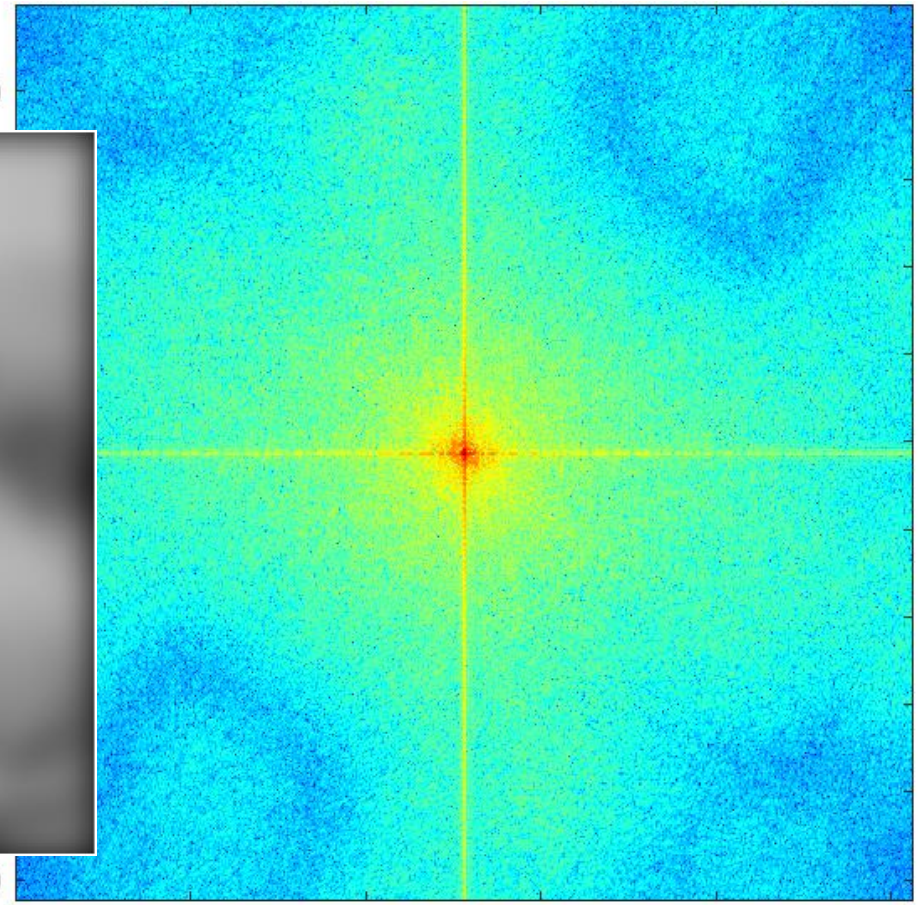
Frequency domain

Gaussian Smoothing in Frequency Domain



FFT based Gaussian Filtering Example

```
1   im = rgb2gray(imread('0901.png'));
2   im = im2double(im);
3   [imh, imw] = size(im);
4   fftsize = 1024;
5   hs = 50;
6   fil = fspecial('gaussian',hs*2+1,10);
7   im_fft = fft2(im,fftsize,fftsize);
8   fil_fft = fft2(fil,fftsize,fftsize);
9   im_fil_fft = im_fft.*fil_fft;
10  im_fil = ifft2(im_fil_fft);
11  im_fil = im_fil(1+hs:size(im,1)+hs, 1+hs:size(im,2)+hs);
12  imagesc(log(abs(fftshift(im_fft)))), axis image, colormap jet
```

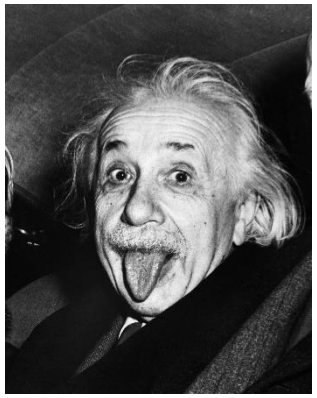


Questions

Which has more information, the phase or the magnitude?

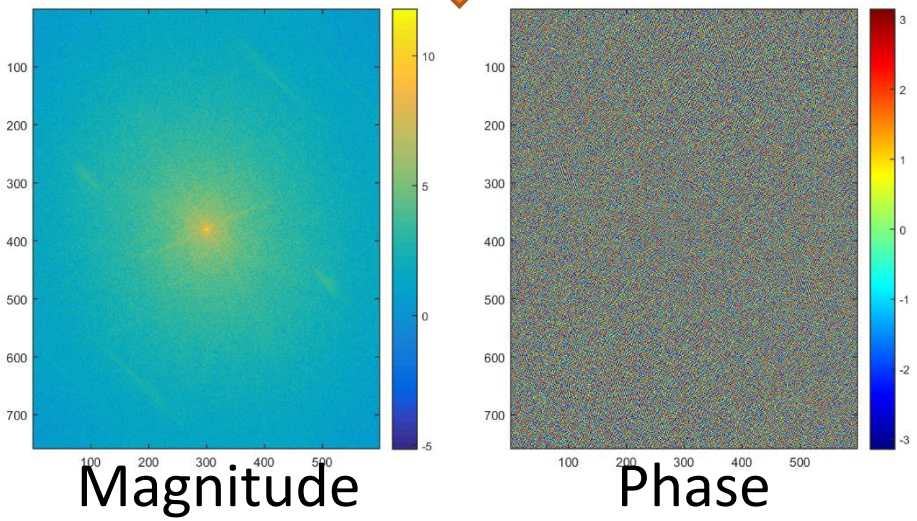
What happens if you take the phase from one image and combine it with the magnitude from another image?

Phase vs Magnitude



Intensity image

FFT

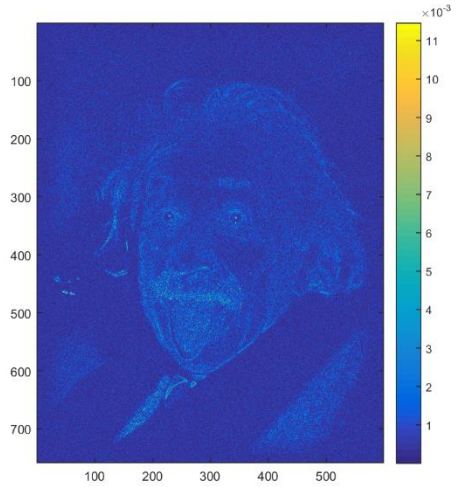


Magnitude

Phase

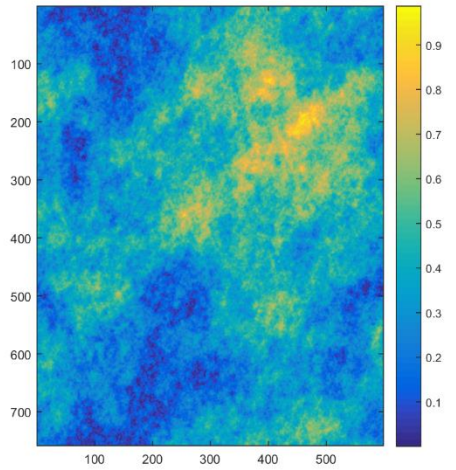
Use random magnitude

Inverse FFT

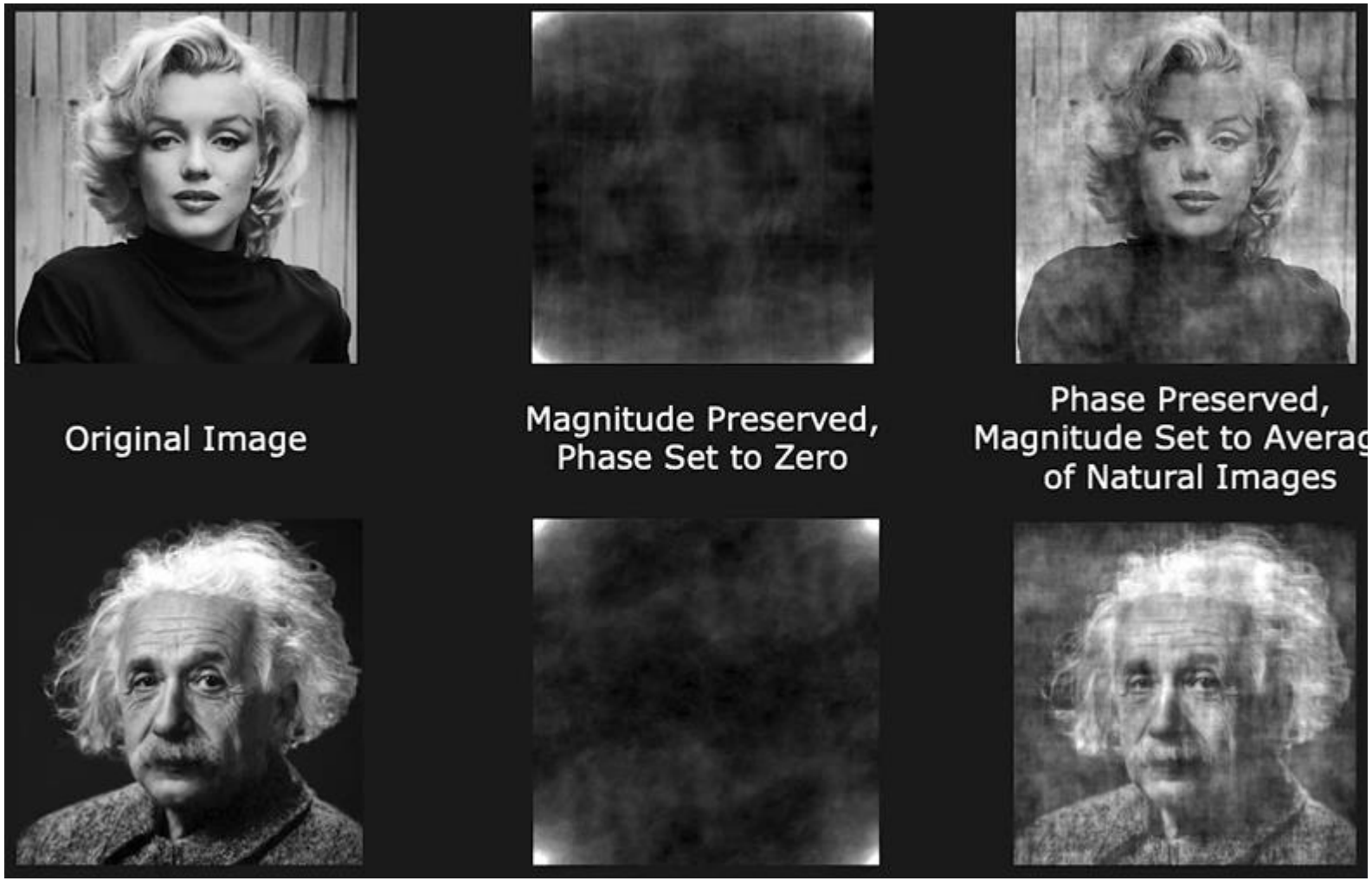


Use random phase

Inverse FFT



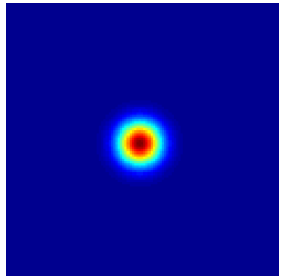
Importance of Phase



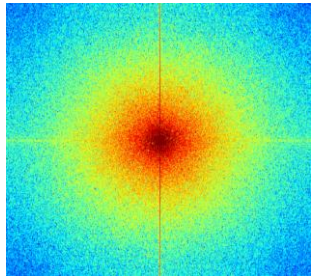
Question

Match the spatial domain image to the Fourier magnitude image

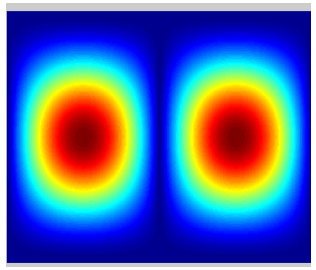
1



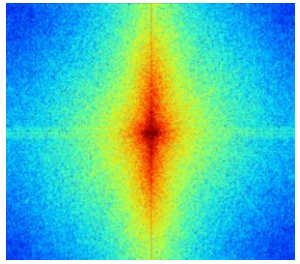
2



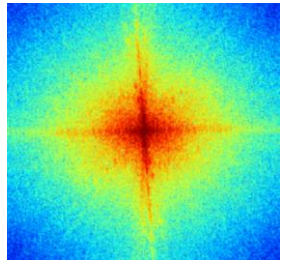
3



4



5

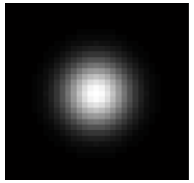


A

B

C

D



E

Image Half-sizing

This image is too big to fit on the screen. How can we reduce it?

How to generate a half-sized version?

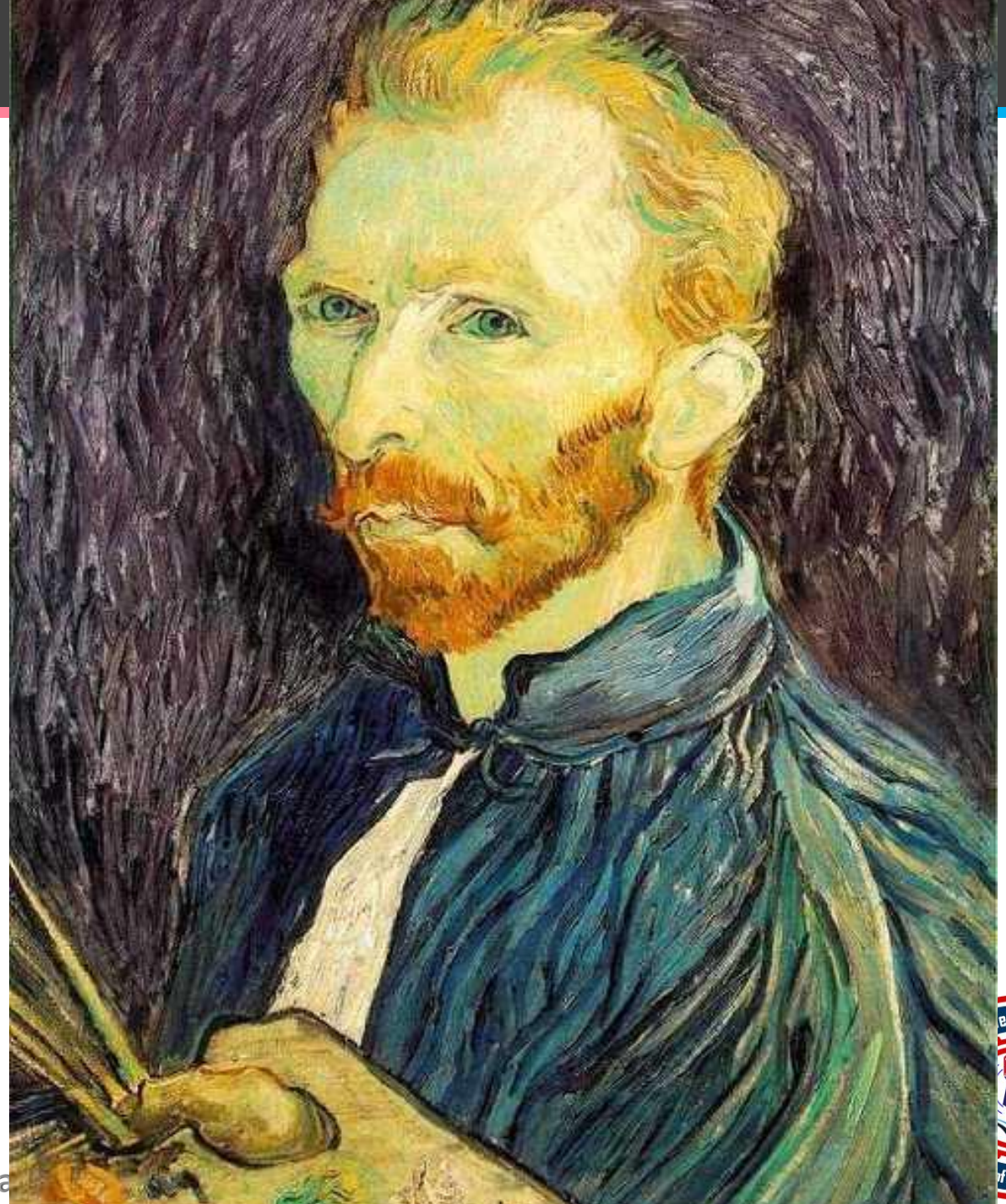
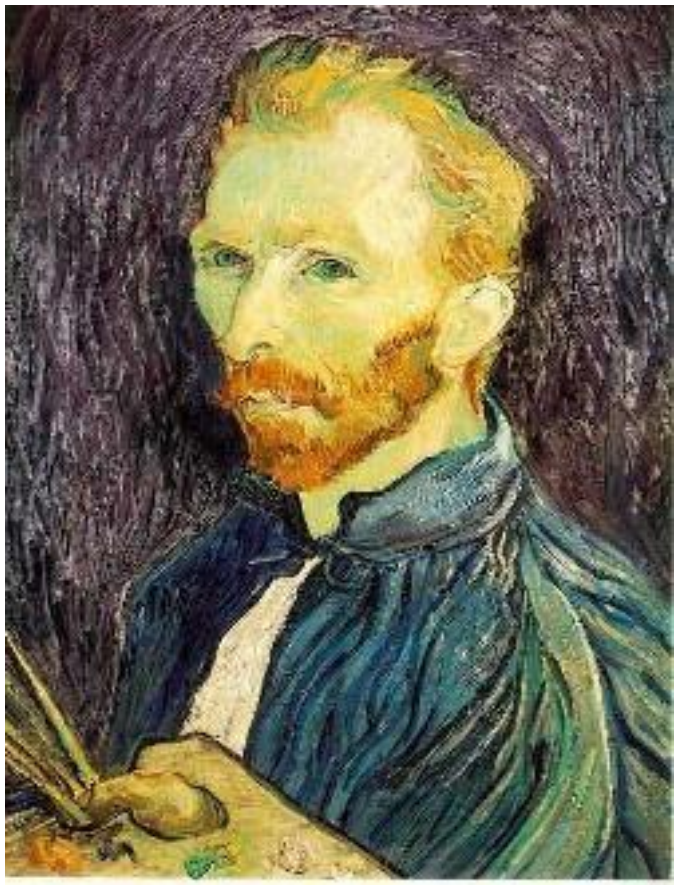


Image Sub-Sampling



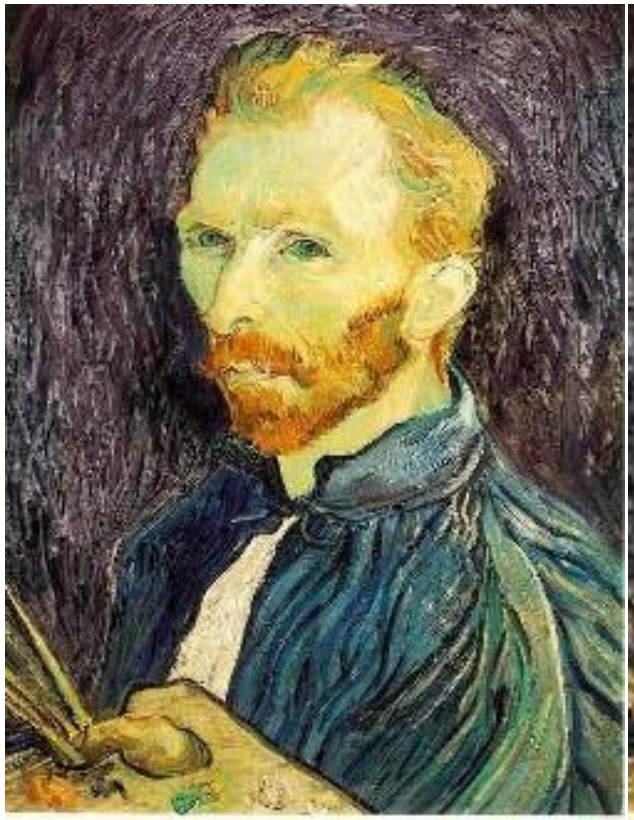
1/4

1/8

Throw away every other row and column to create a 1/2 size image - called *image sub-sampling*

Slide by Steve Seitz

Image Sub-Sampling



1/2

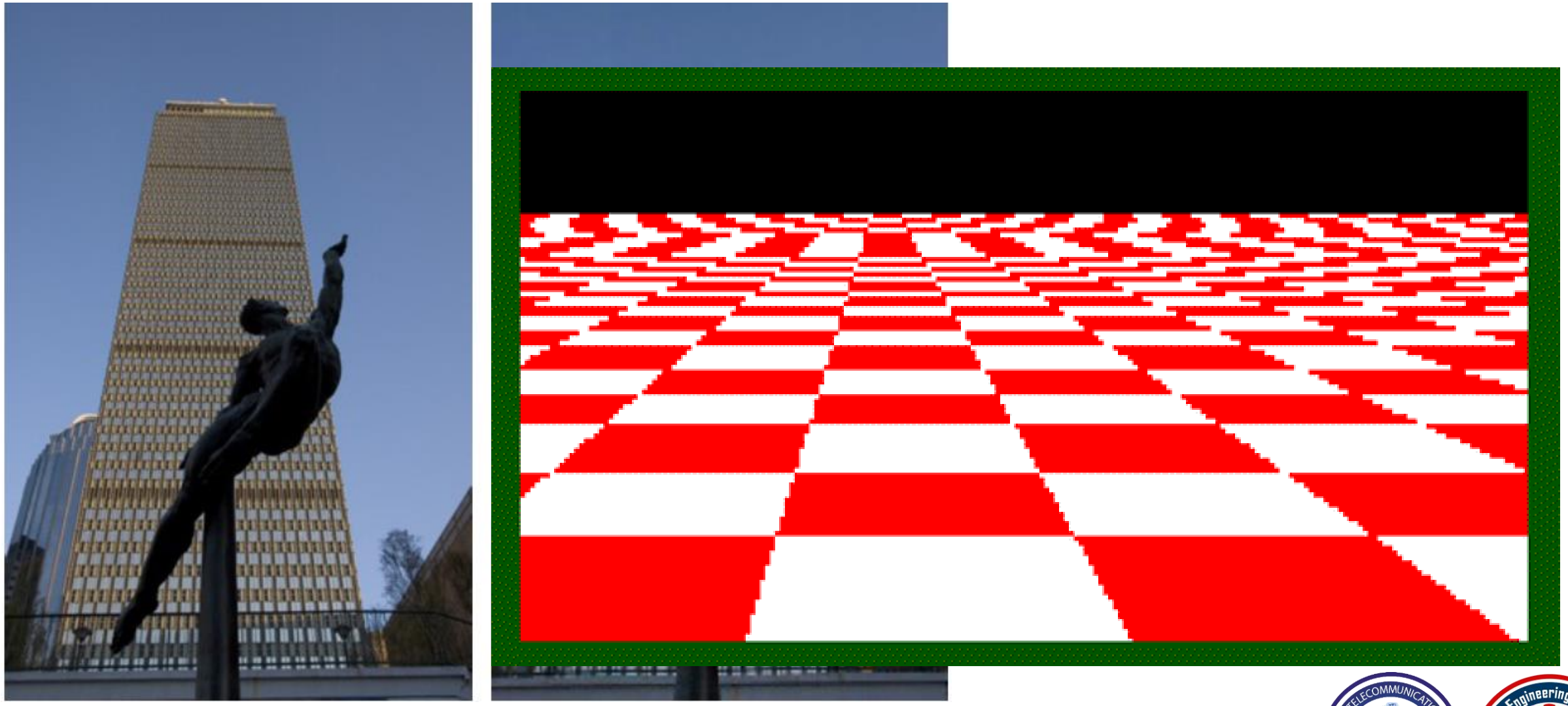
1/4 (2x zoom)

1/8 (4x zoom)

Why does this look so crufty?
Aliasing! What do we do?

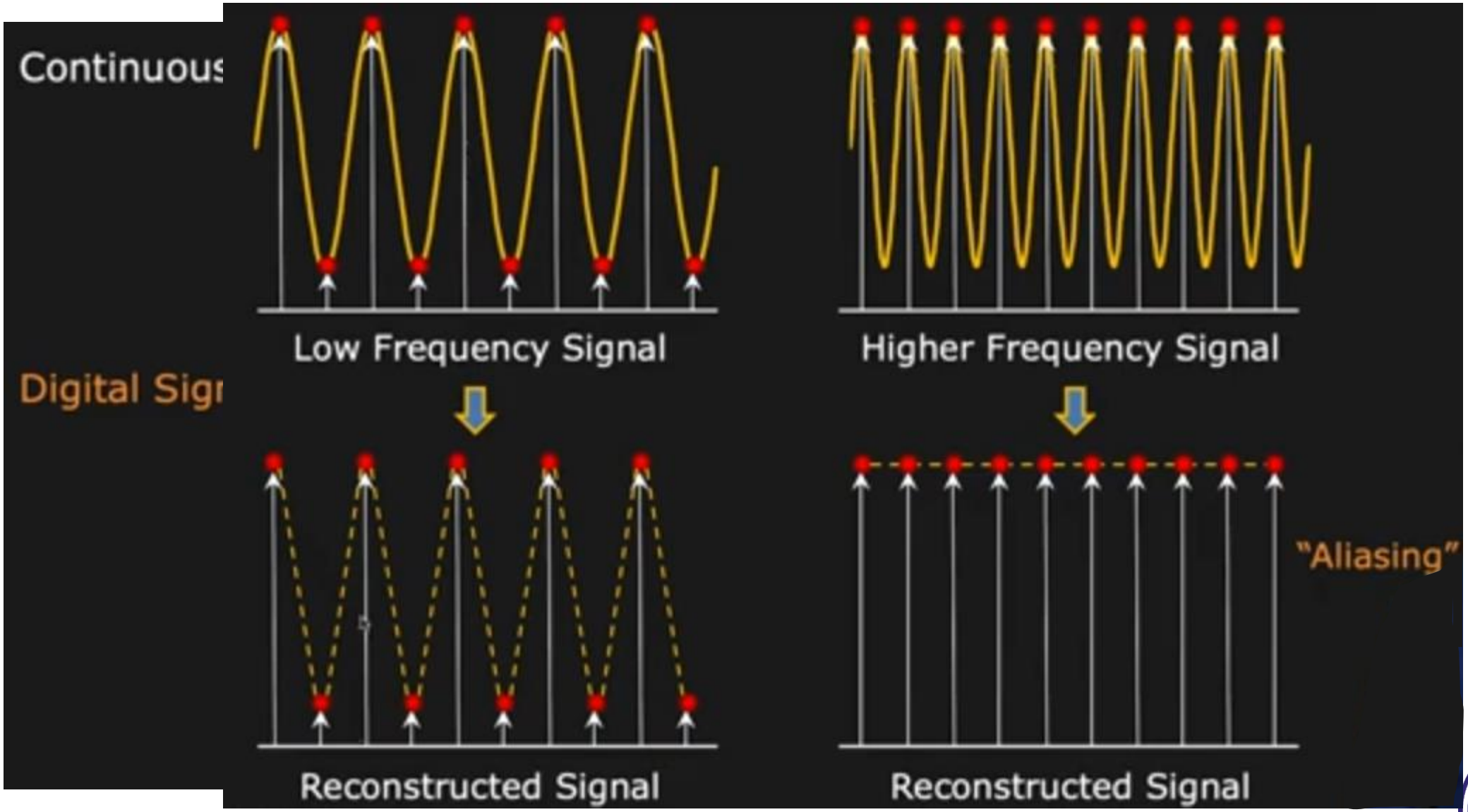
Slide by Steve Seitz

Image Sub-Sampling



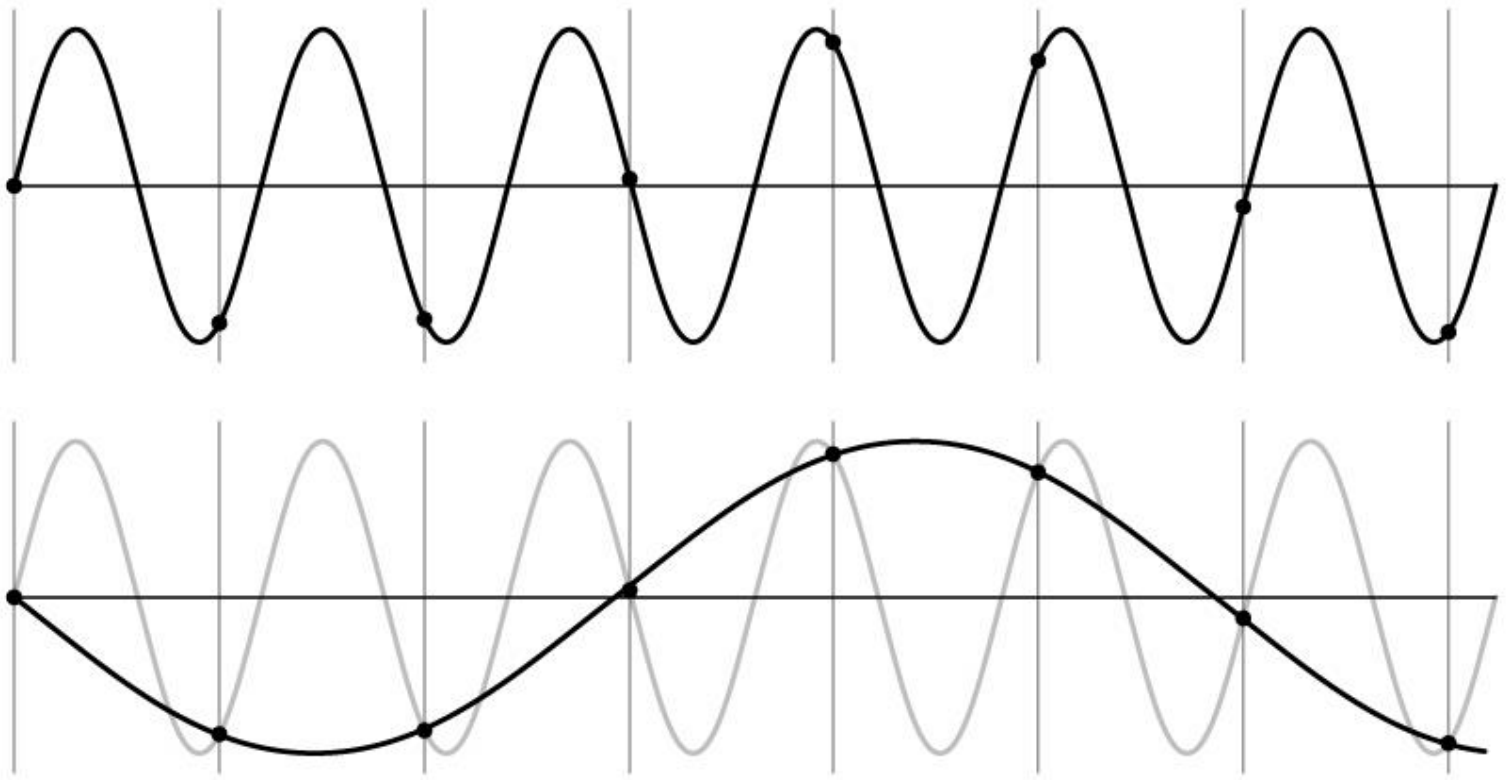
Source: F. Durand

From Continuous to Digital Image



Aliasing Problem

- 1D example (sinewave):



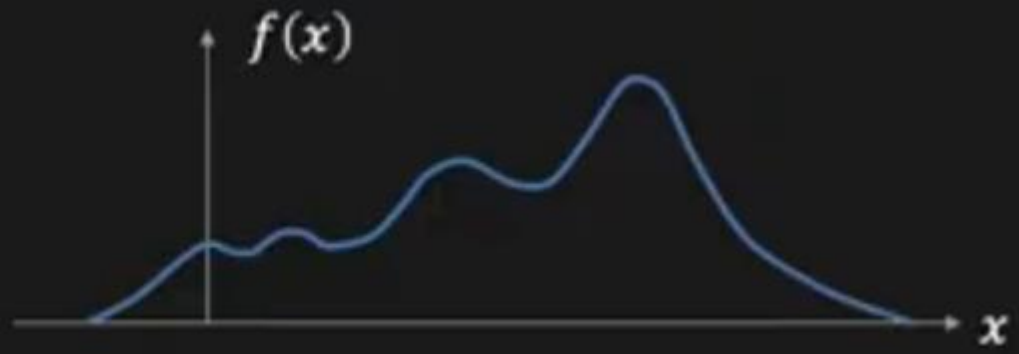
Source: S. Marschner

Aliasing Problem

- Sub-sampling may be dangerous….
- Characteristic errors may appear:
  - “Wagon wheels rolling the wrong way in movies”
  - “Checkerboards disintegrate in ray tracing”
  - “Striped shirts look funny on color television”

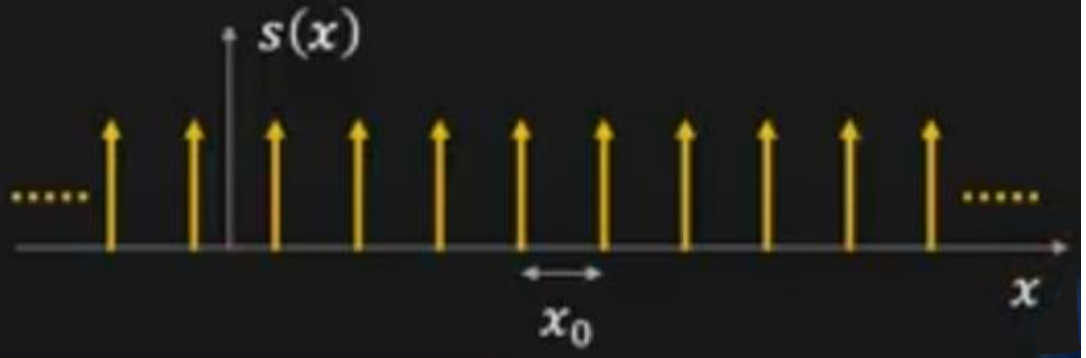
Sampling Theory

Continuous Signal:



Shah Function (Impulse Train):

$$s(x) = \sum_{n=-\infty}^{\infty} \delta(x - nx_0)$$

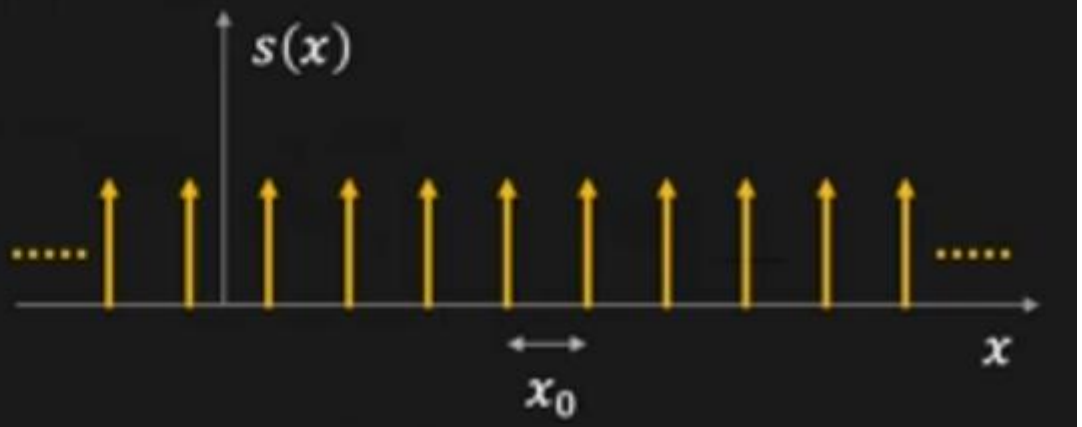


Sampled Function: $f_s(x) = f(x)s(x)$

Shah Function (Impulse Train)

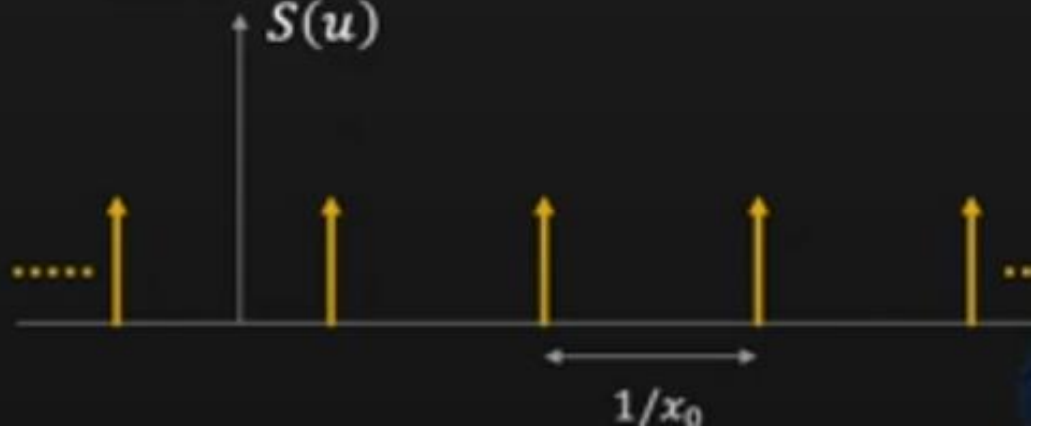
Shah Function (Spatial Domain):

$$s(x) = \sum_{n=-\infty}^{\infty} \delta(x - nx_0)$$



Shah Function (Fourier Domain):

$$S(u) = \frac{1}{x_0} \sum_{n=-\infty}^{\infty} \delta\left(x - \frac{n}{x_0}\right)$$



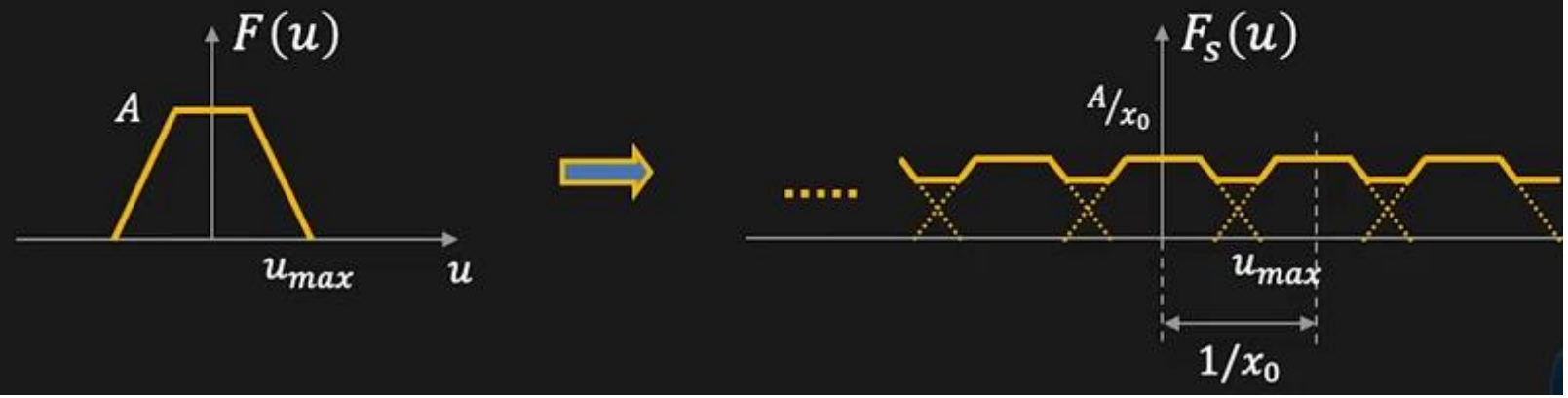
Fourier Analysis of Sampled Signals

Sampled Signal:

$$f_s(x) = f(x)s(x) = f(x)\sum \delta(x - nx_0)$$

$$F_s(u) = F(u) * S(u) = F(u) * \frac{1}{x_0}\sum \delta(u - n/x_0)$$

If $u_{max} > \frac{1}{2x_0}$ Aliasing



Nyquist Theorem

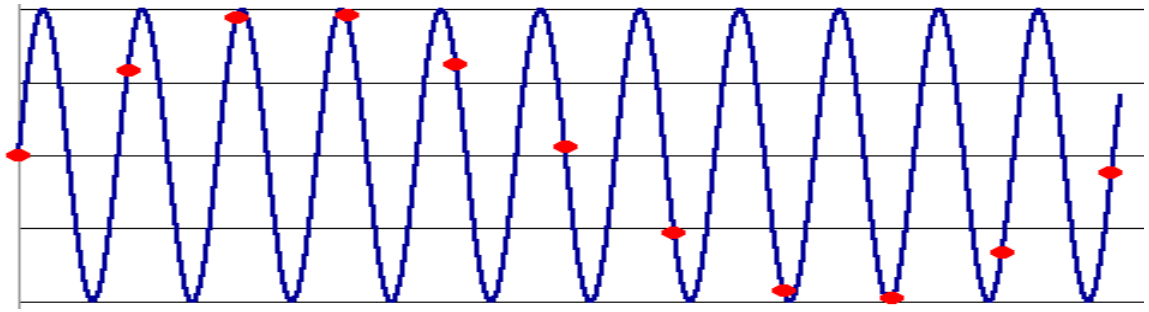
Can we recover $f(x)$ from $f_s(x)$? In other words, can we recover $F(u)$ from $F_s(u)$?

Only if $u_{max} \leq \frac{1}{2x_0}$ (Nyquist Frequency)

$$F(u) = F_s(u)C(u)$$

$$C(u) = \begin{cases} x_0, & |u| < 1/2x_0 \\ 0, & Otherwise \end{cases}$$

Aliasing



- Occurs when your sampling rate is not high enough to capture the amount of detail in your image
- Can give you the wrong signal/image—an *alias*

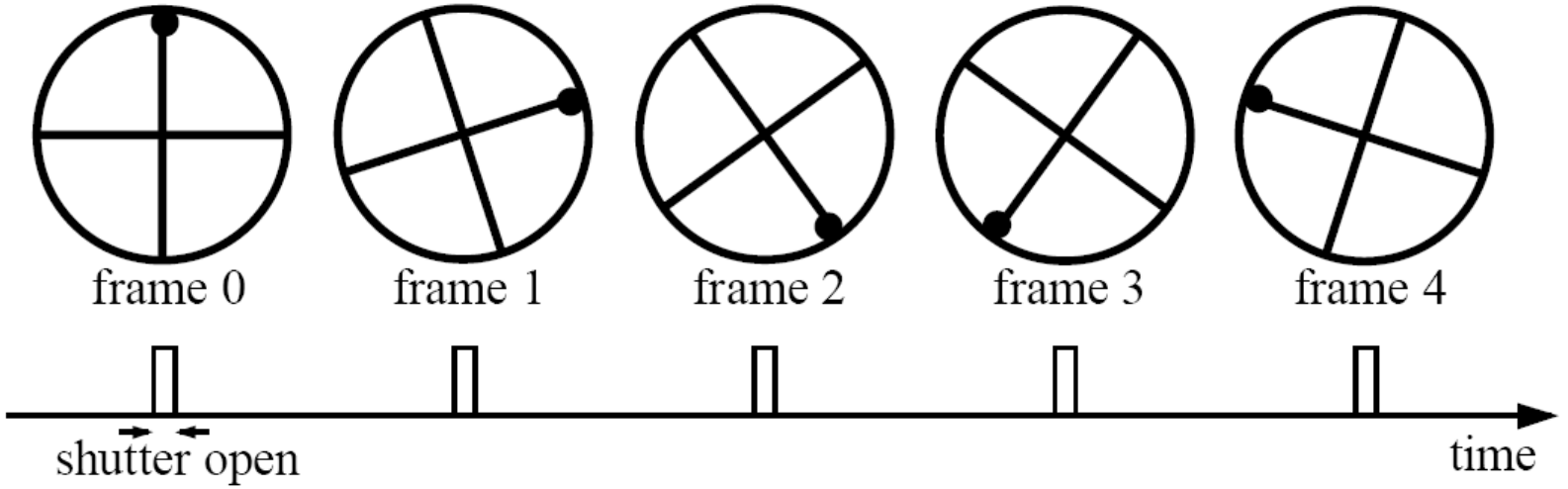
- To do sampling right, need to understand the structure of your signal/image
- To avoid aliasing:
  - sampling rate ≥ 2 * max frequency in the image
    - said another way: ≥ two samples per cycle
  - This minimum sampling rate is called the **Nyquist rate**

Source: L. Zhang

Wagon Wheel Effect

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)

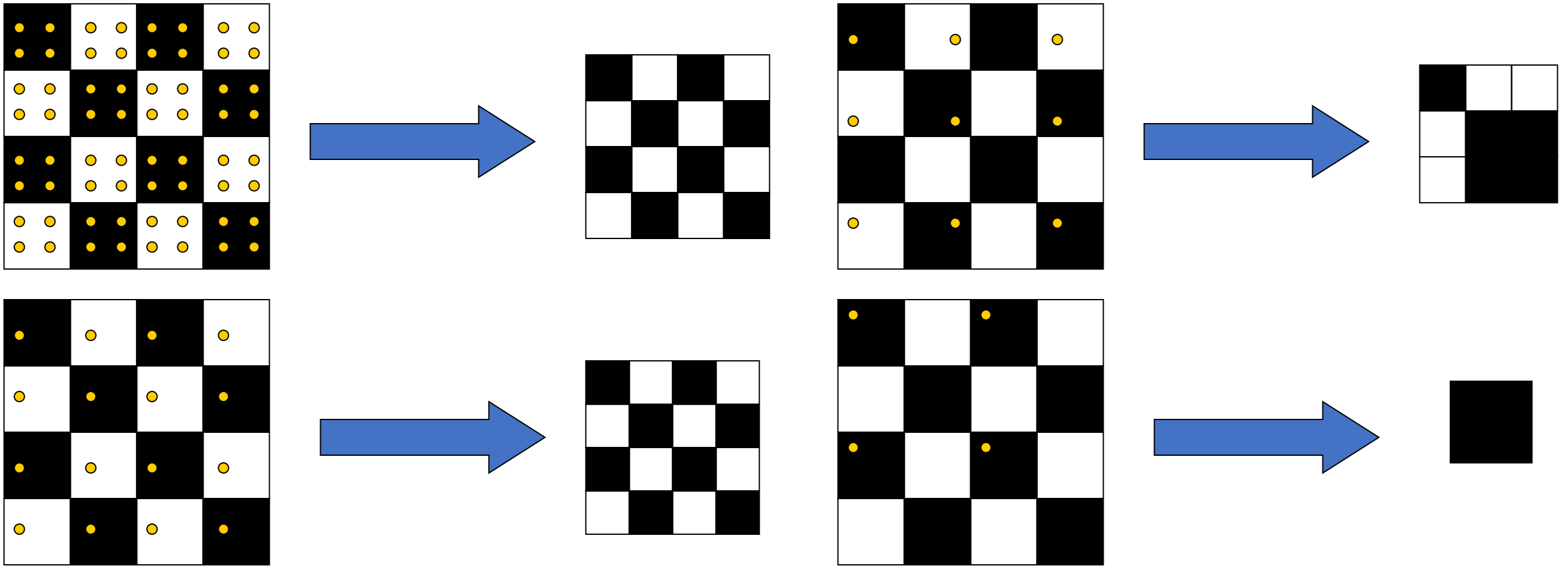
(See http://www.michaelbach.de/ot/mot_wagonWheel/index.html)

Source: L. Zhang

Wagon-Wheel Effect

https://www.youtube.com/watch?v=QOwzkND_ooU

Sampling an Image



Examples of GOOD sampling

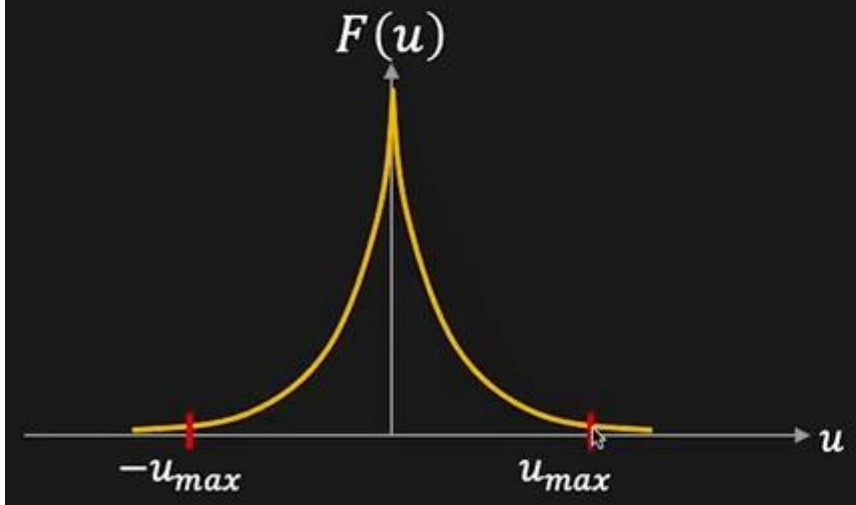
Examples of BAD sampling -> Aliasing

Example


"Well sampled" image
"Under sampled" image
(visible aliasing artifacts)

Aliasing in Digital Imaging

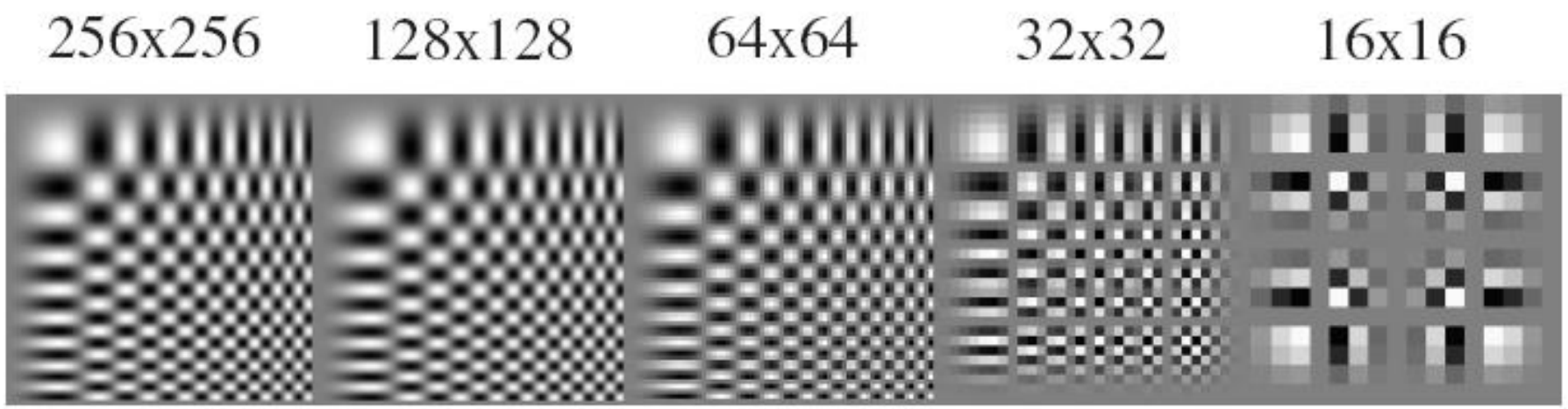
Aliasing occurs when imaging a scene (signal) that has frequencies above the image sensor's Nyquist Frequency

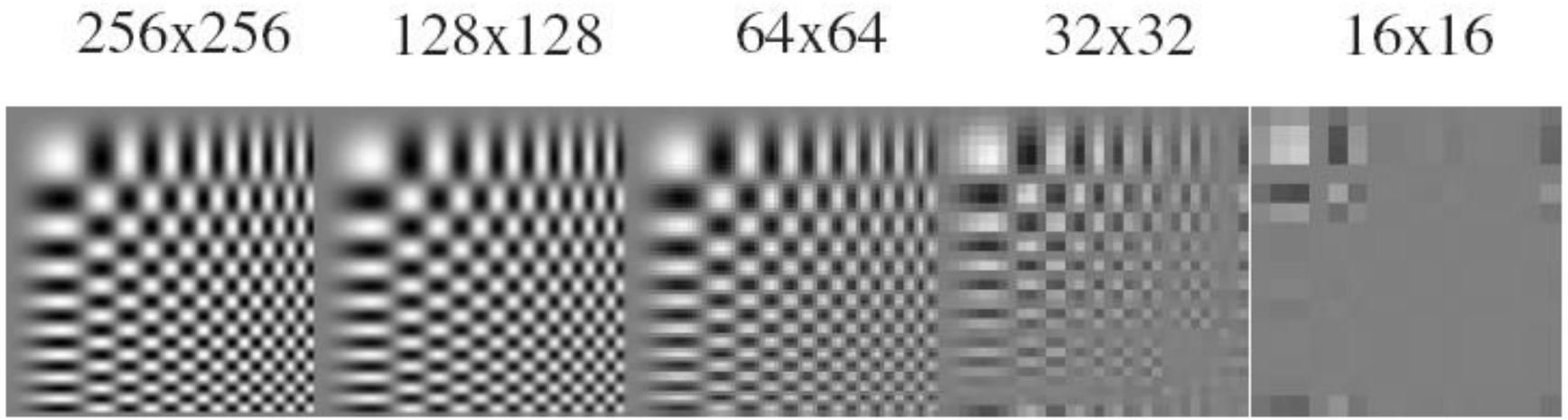


Typical Power Spectrum of Natural Scenes

Aliasing artifacts usually occur in the form of Moiré patterns

Anti-Aliasing





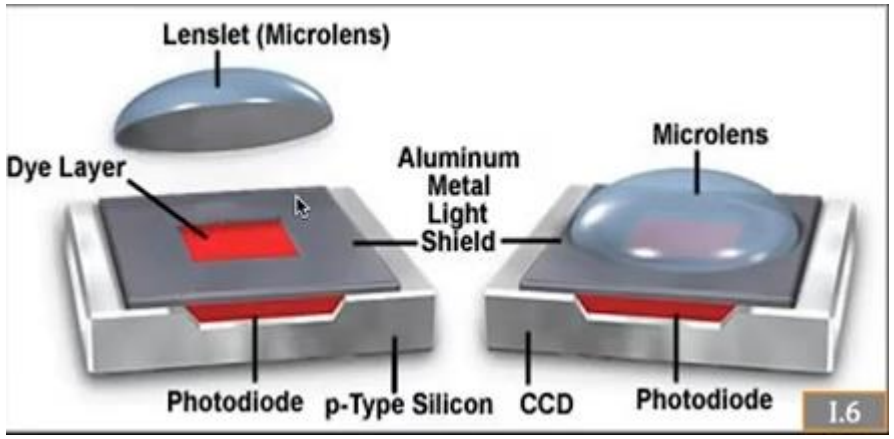
Forsyth and Ponce 2002

Minimizing the Effects of Aliasing

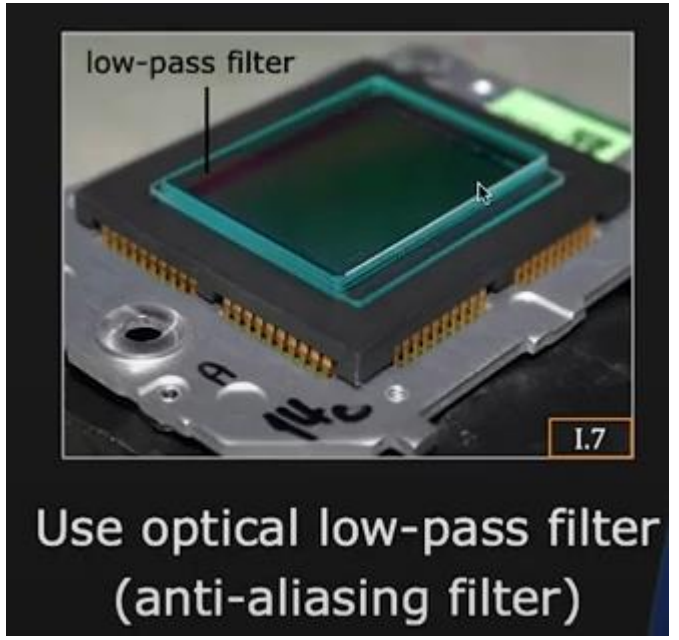
Band Limit: Clip the signal above the Nyquist Frequency.

Effectively, “blur” the scene before sampling.

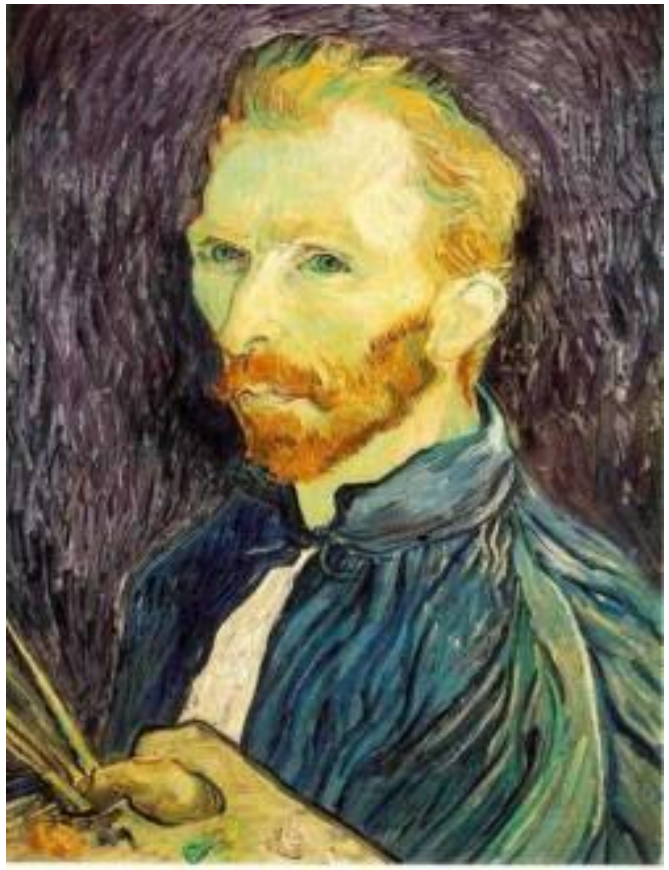
Sensors use two strategies



Pixels are area-samplers
(box-averaging filter)



Gaussian (Low-Pass) Pre-Filtering



Gaussian 1/2

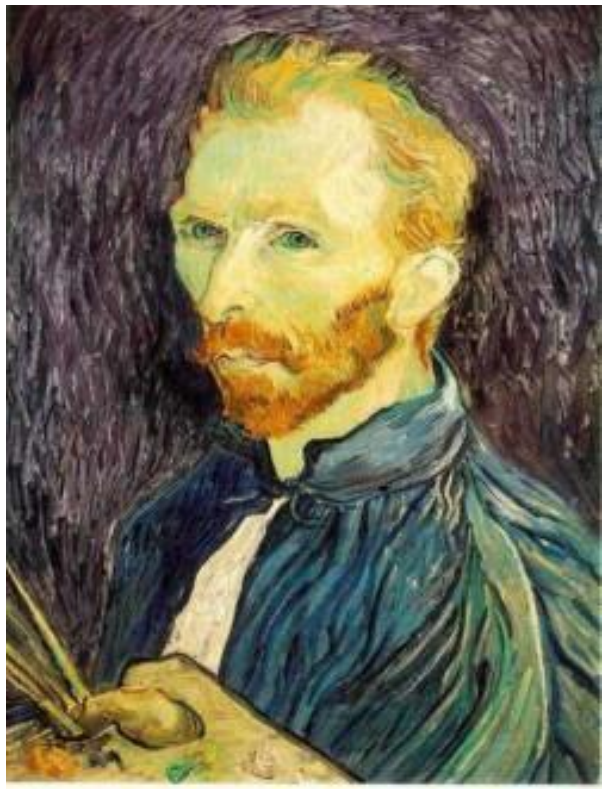
G 1/4

G 1/8

- Solution: filter the image, *then* subsample

Source: S. Seitz

Subsampling with Gaussian Pre-Filtering



Gaussian 1/2

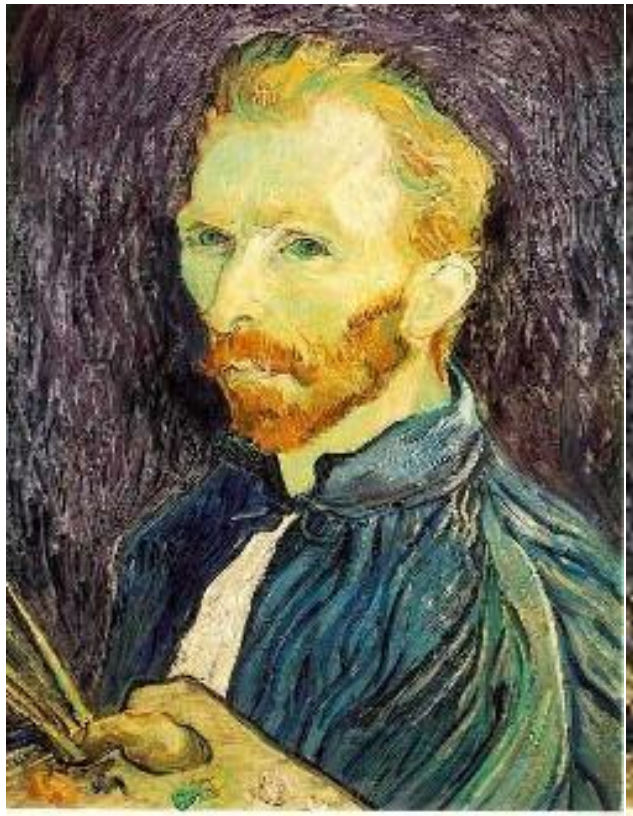
G 1/4

G 1/8

- Solution: filter the image, *then* subsample

Source: S. Seitz

Compare With



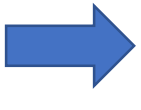
1/2

1/4 (2x zoom)

1/8 (4x zoom)

Source: S. Seitz

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



Human Vision: Fovea and Periphery

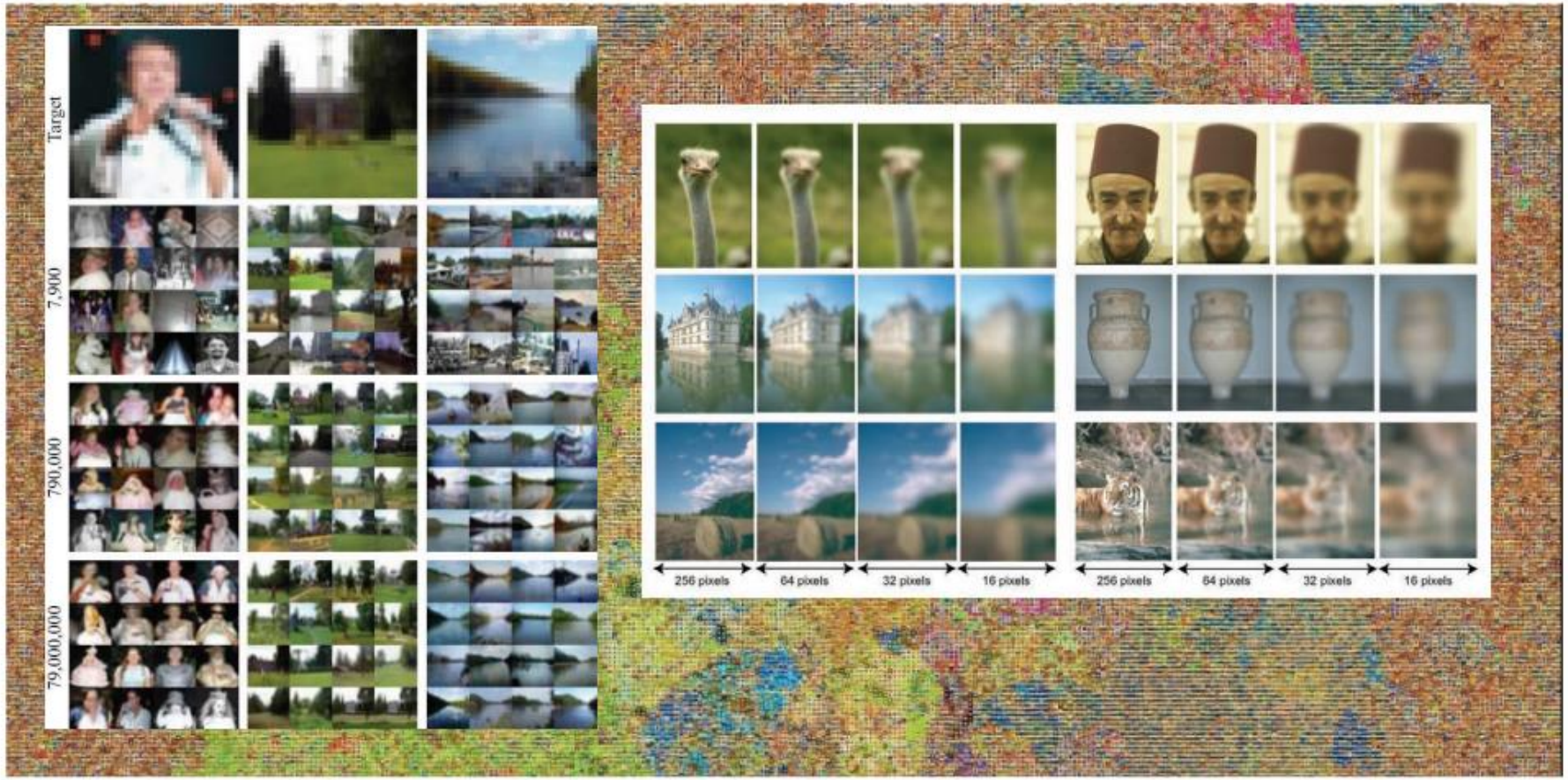
Some properties of image encoding like blurring, color representation set the stage for what is available to the neural system

Camera

Human: Acuity decreases with eccentricity

Human Vision: Receptive Field size scale with eccentricity

80 millions tiny images

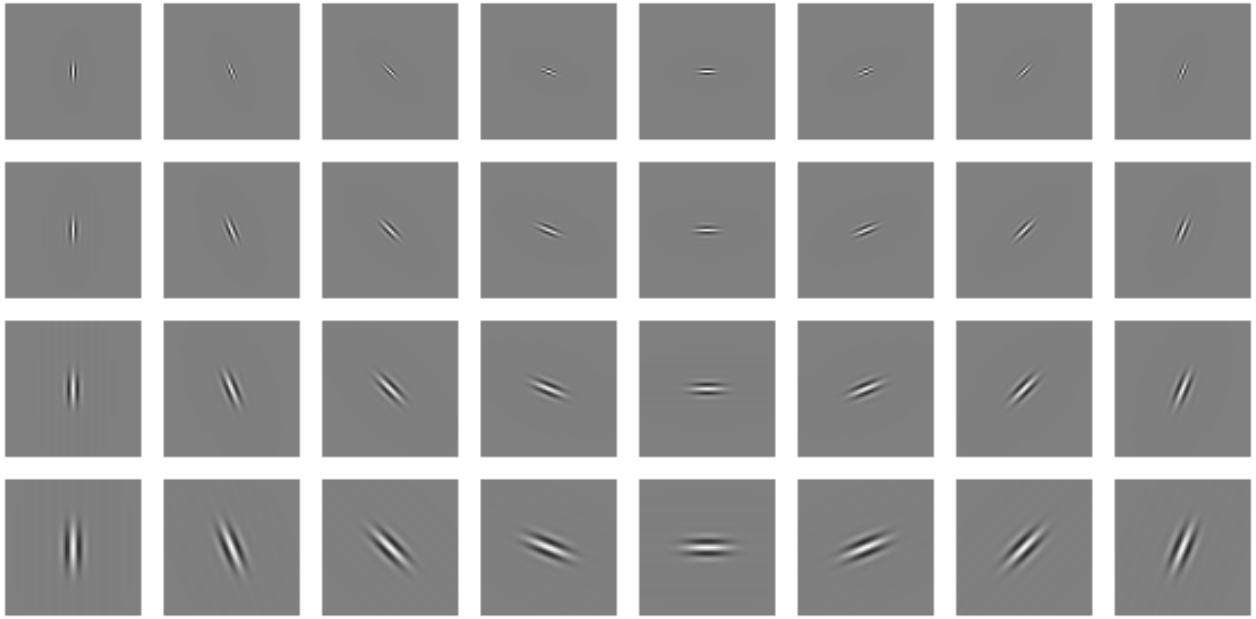


A. Torralba, R. Fergus, W.T. Freeman. PAMI 2008

Torralba (2009). How many pixels make an image?

Clues from Human Perception

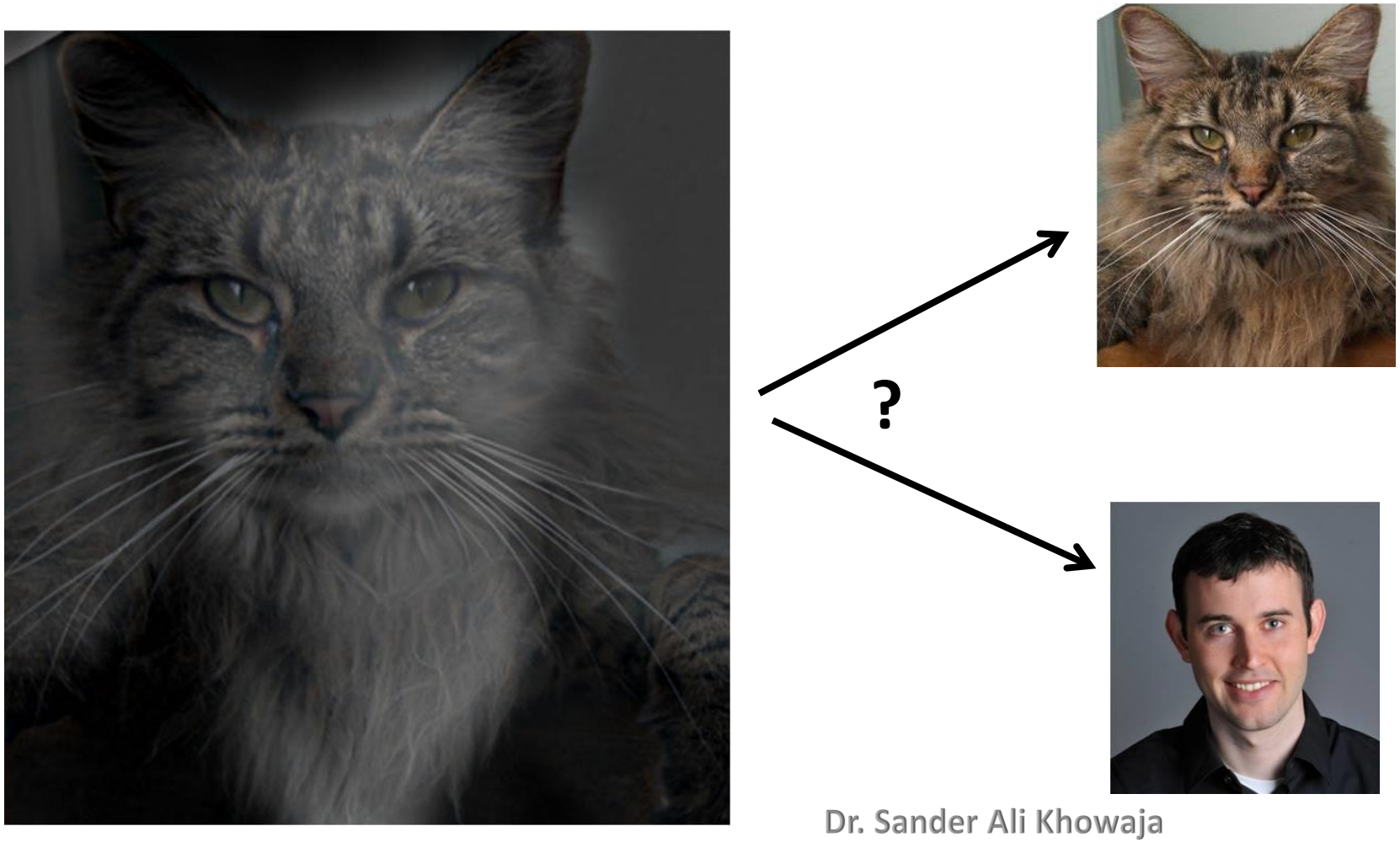
- Early processing in humans filters for various orientations and scales of frequency
- Perceptual cues in the mid-high frequencies dominate perception
- When we see an image from far away, we are effectively subsampling it



Early Visual Processing: Multi-scale edge and blob filters

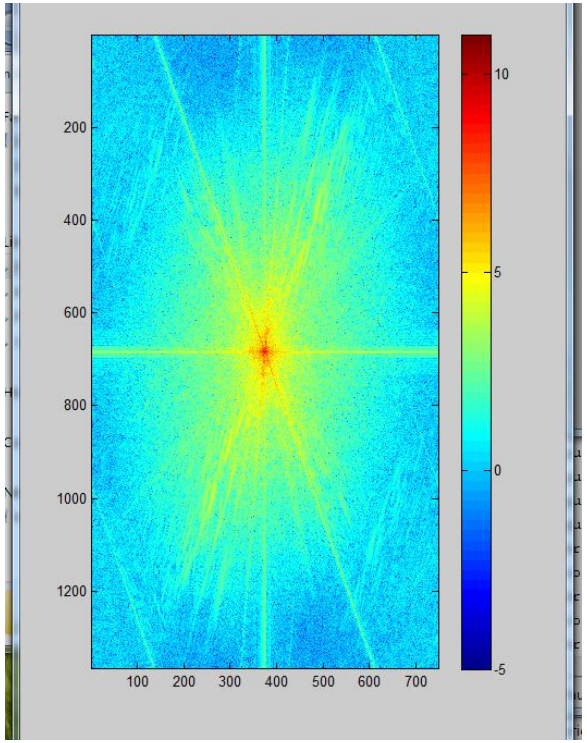
Hybrid Image

Why do we get different, distance-dependent interpretations of hybrid images?



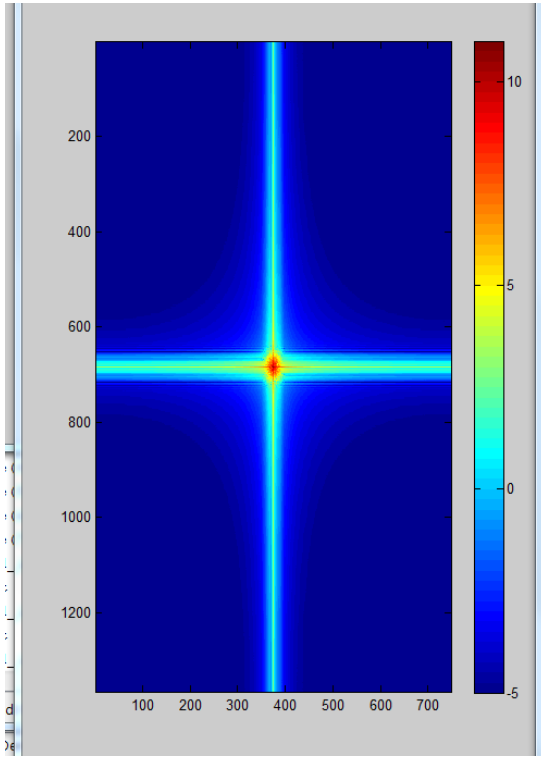
Hybrid Image in FFT

Hybrid Image



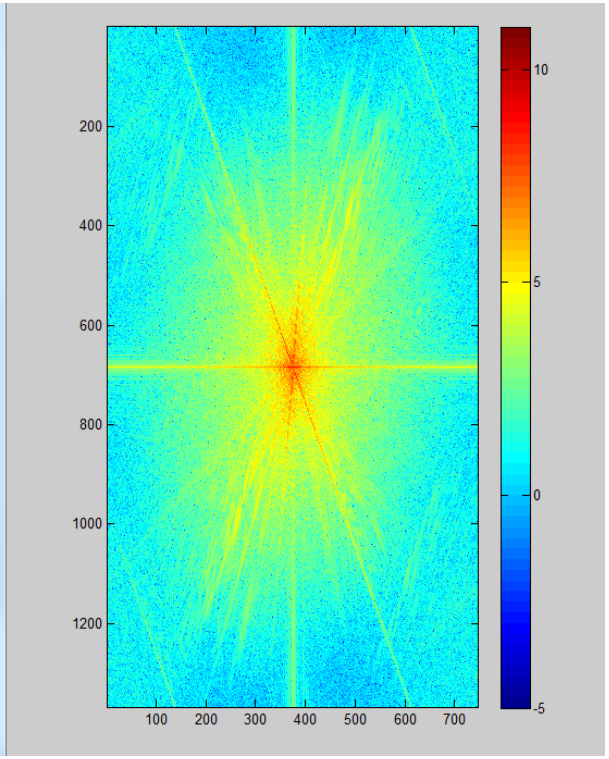
=

Low-passed Image

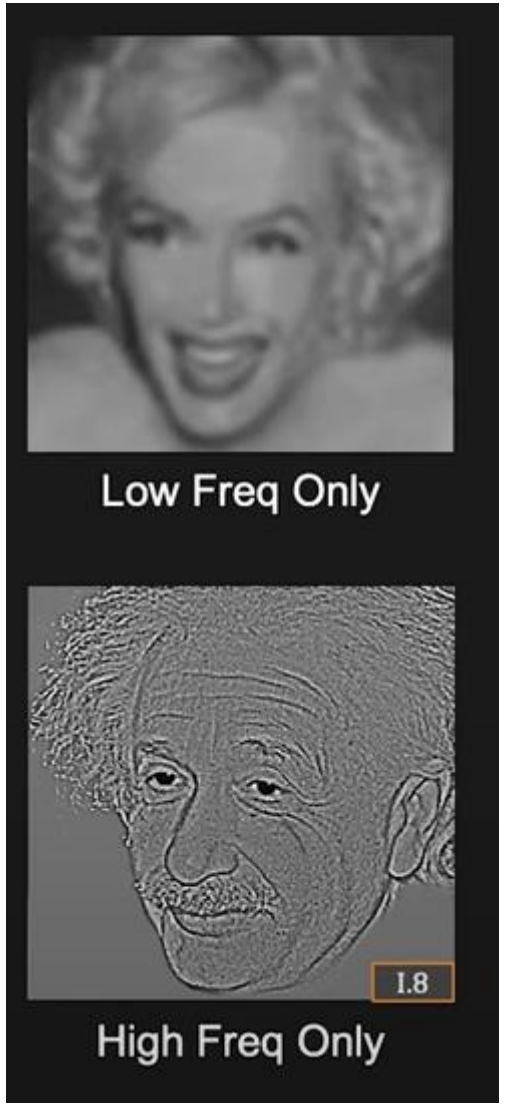


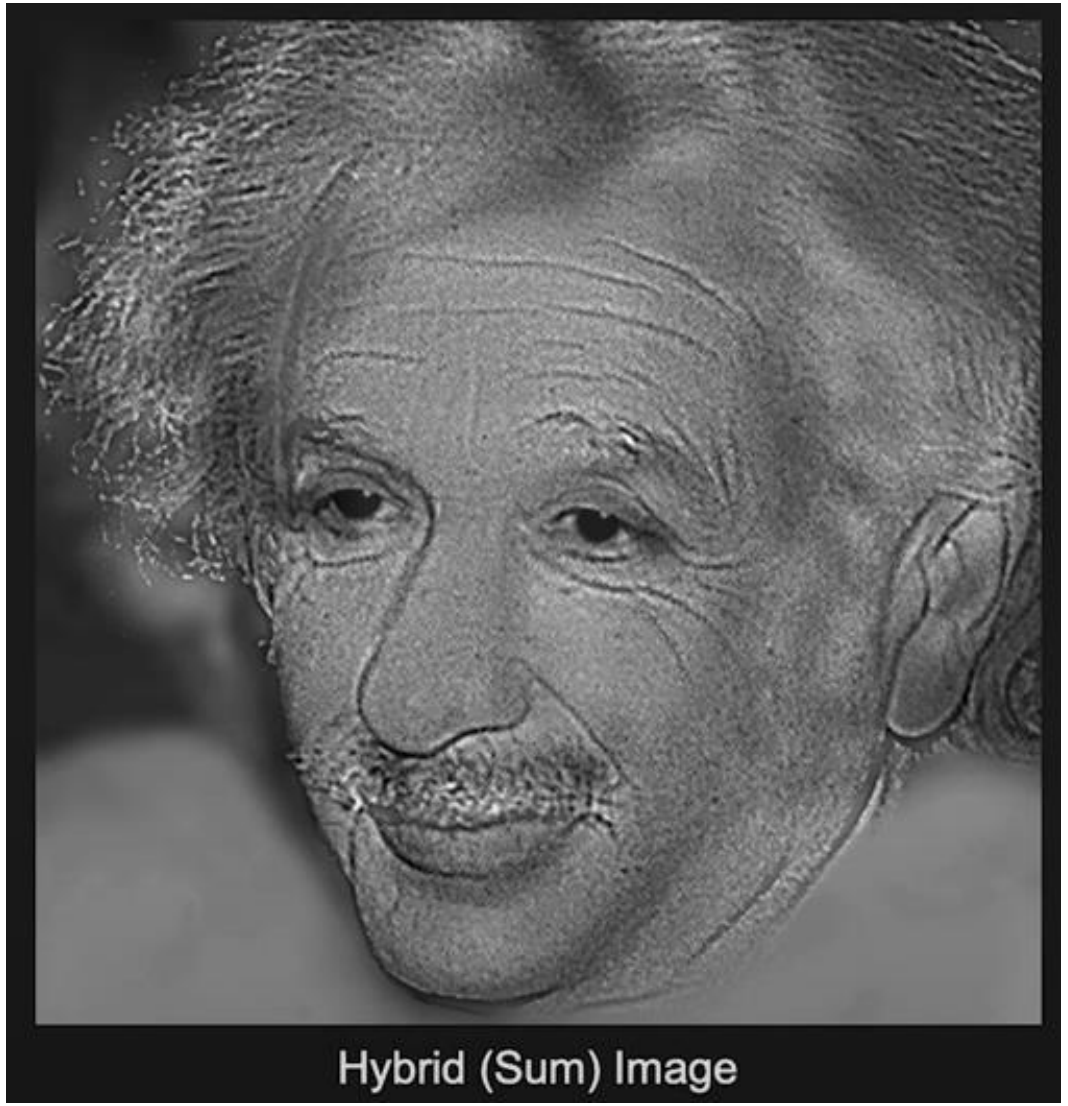
+

High-passed Image

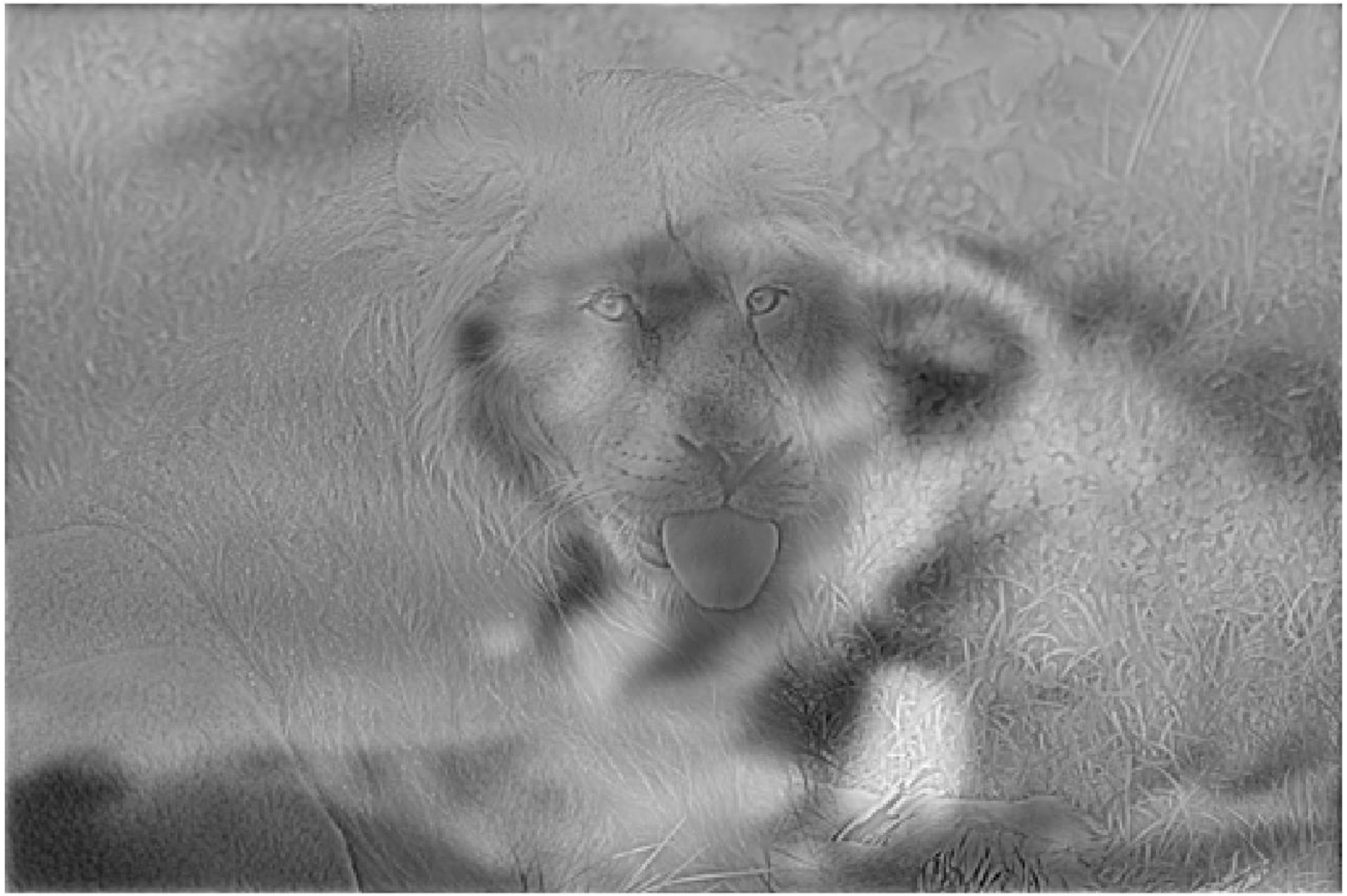


Hybrid Images

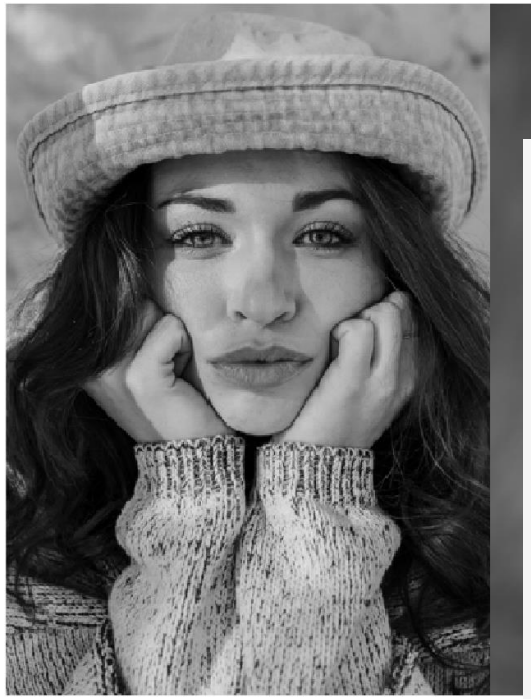




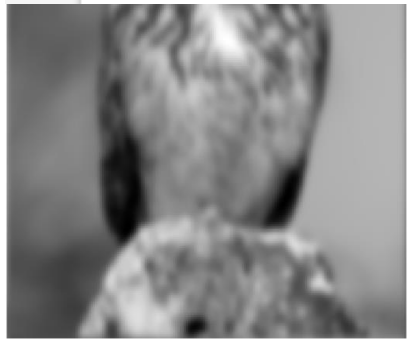
Hybrid Image Example



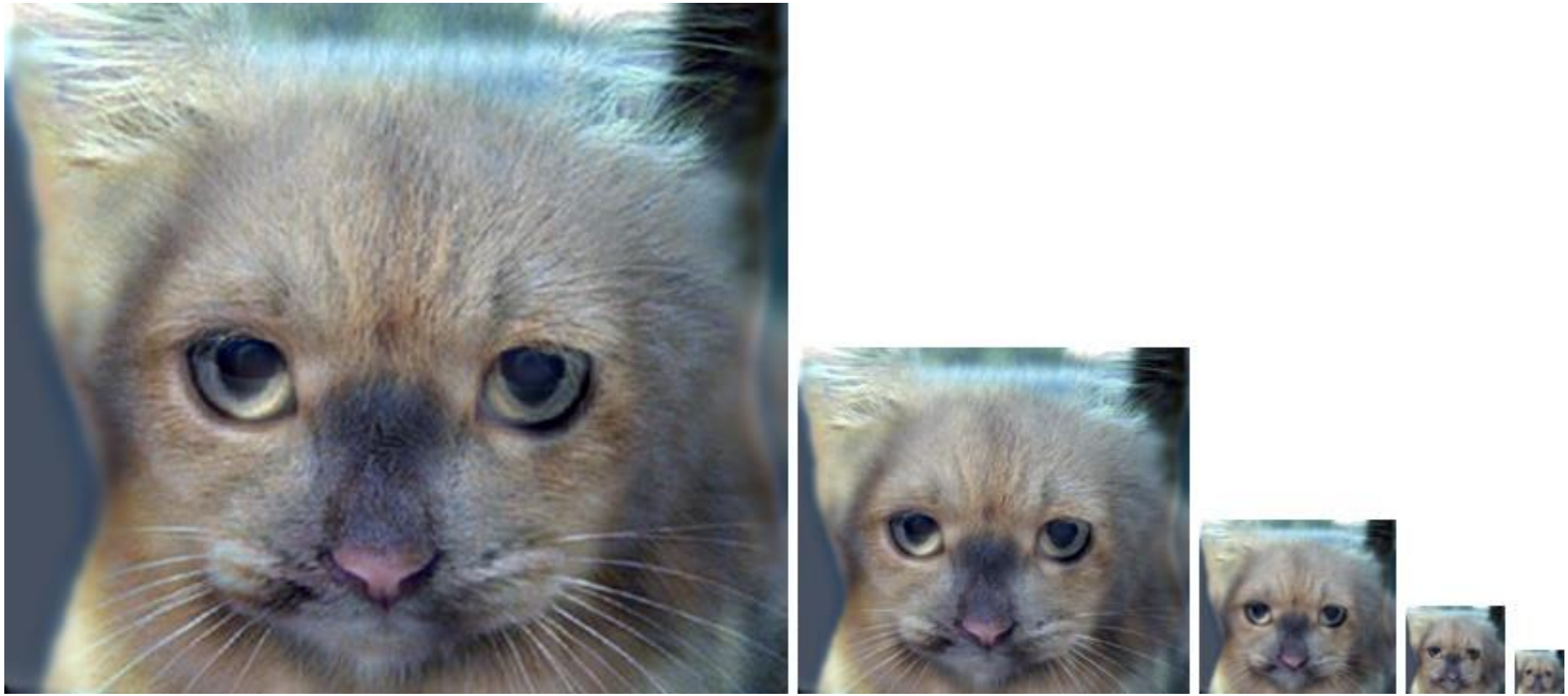
Hybrid Image Example



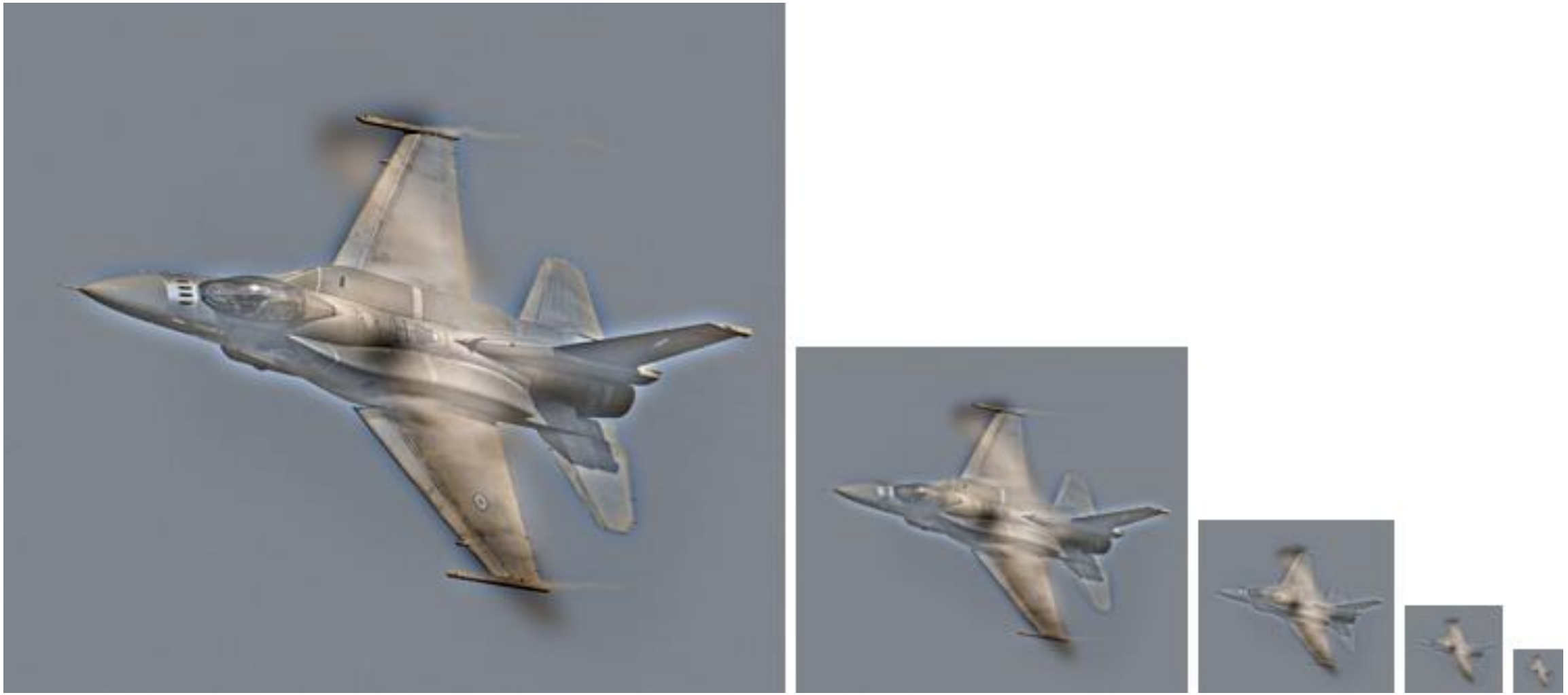
```
im1 = imread('mm1.jpg');
im2 = imread('ae1.jpg');
im1 = imresize(im1,[347 317]);
im11 = histeq(im2double(rgb2gray(im1)));
im12 = im2double(rgb2gray(im2));

sigma = 3;
gaussian_dimension = 3*sigma*2+1;

im11_hpf = highPassFilter(im11,gaussian_dimension);

imshow(im11_hpf,[])

sigma = 2;
gaussian_dimension=3*sigma*2+1;
im12_lpf = lowPassFilter(im12,gaussian_dimension);

figure, imshow(im12_lpf,[])

himage = im11_hpf + im12_lpf;

figure, imshow(himage,[])
```



Some Examples of Scale-based Hybrid Images



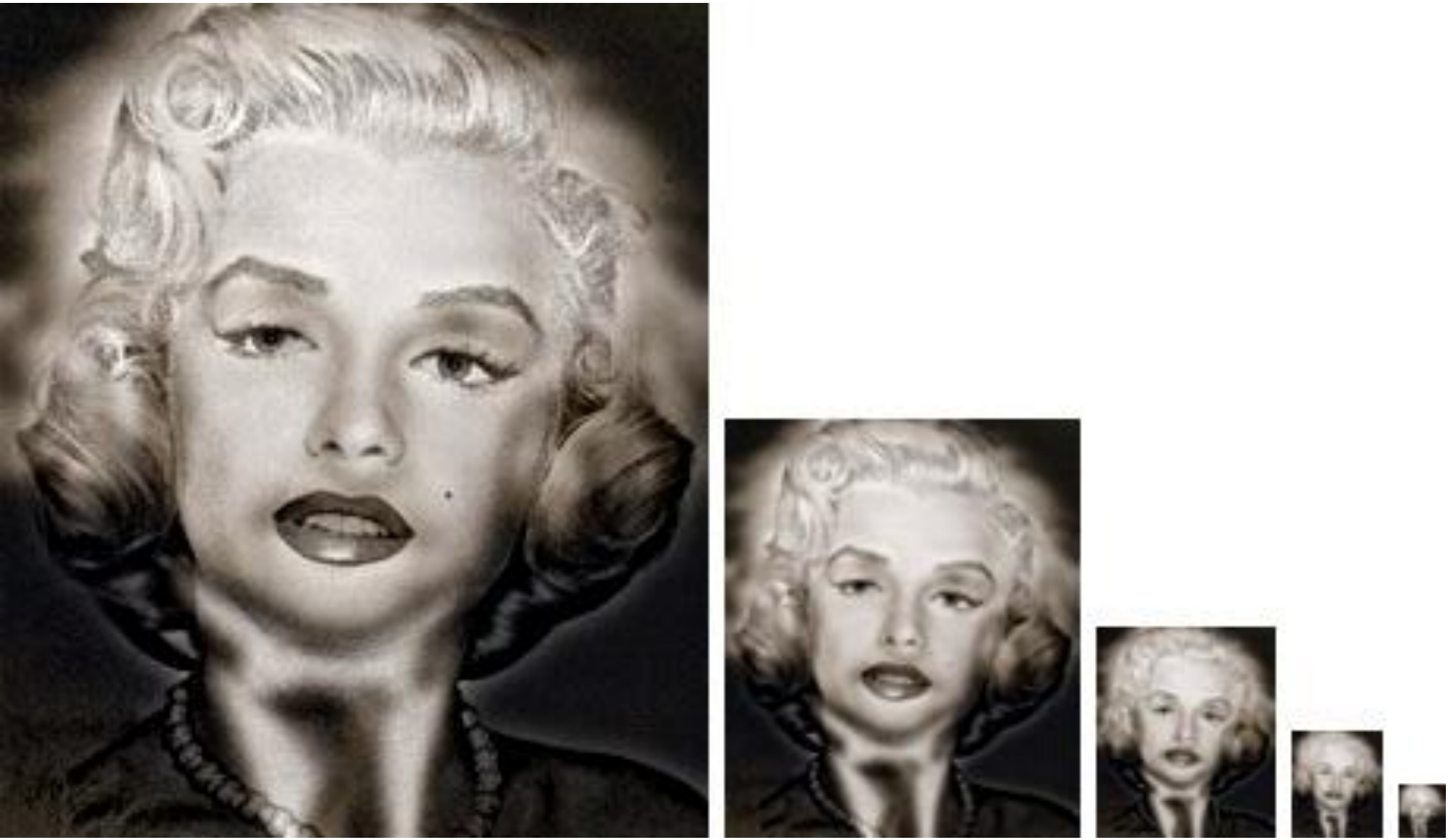
Some Examples of Scale-based Hybrid Images



Some Examples of Scale-based Hybrid Images

Some Examples of Scale-based Hybrid Image

Some Examples of Scale-based Hybrid Image



Upsampling

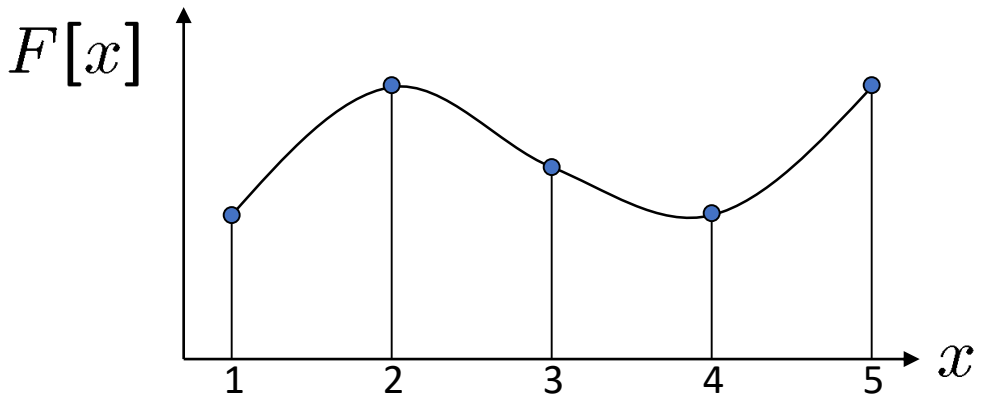
- This image is too small for this screen: 
- How can we make it 10 times as big?
- Simplest approach:

  repeat each row

  and column 10 times
- (“Nearest neighbor

  interpolation”)

Image Interpolation



d = 1 in this example

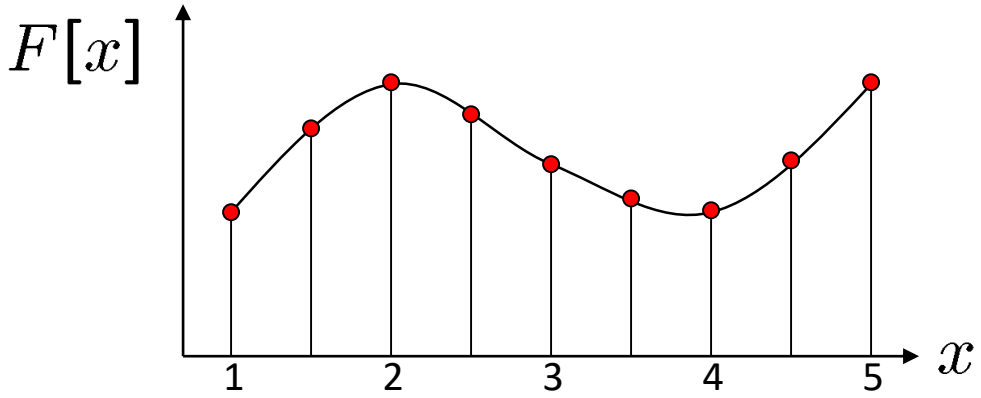
Recall how a digital image is formed

$$F[x, y] = \text{quantize}\{f(xd, yd)\}$$

- It is a discrete point-sampling of a continuous function
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale

Adapted from: S. Seitz

Image Interpolation



d = 1 in this example

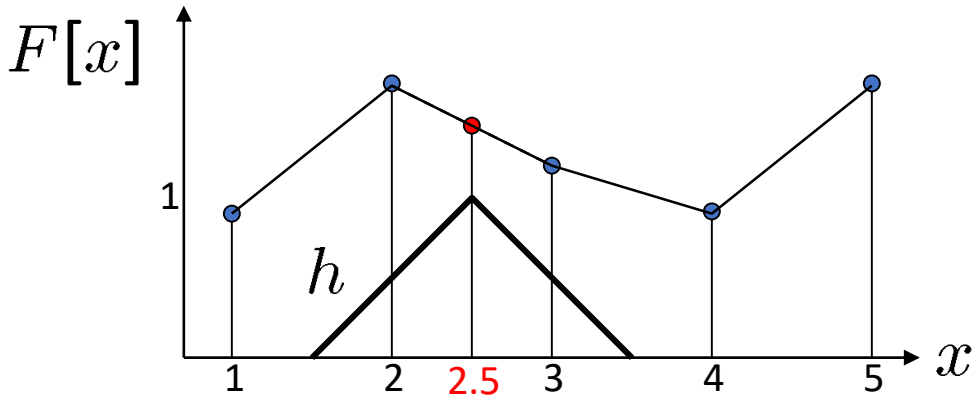
Recall how a digital image is formed

$$F[x, y] = \text{quantize}\{f(xd, yd)\}$$

- It is a discrete point-sampling of a continuous function
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale

Adapted from: S. Seitz

Image Interpolation



d = 1 in this example

- What if we don't know f ?
  - Guess an approximation: $\tilde{f}$
  - Can be done in a principled way: filtering
  - Convert F to a continuous function:

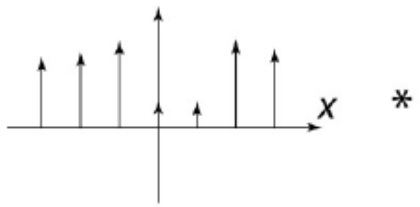
$$f_F(x) = F(\tfrac{x}{d}) \text{ when } \tfrac{x}{d} \text{ is an integer, 0 otherwise}$$

  - Reconstruct by convolution with a *reconstruction filter, h*

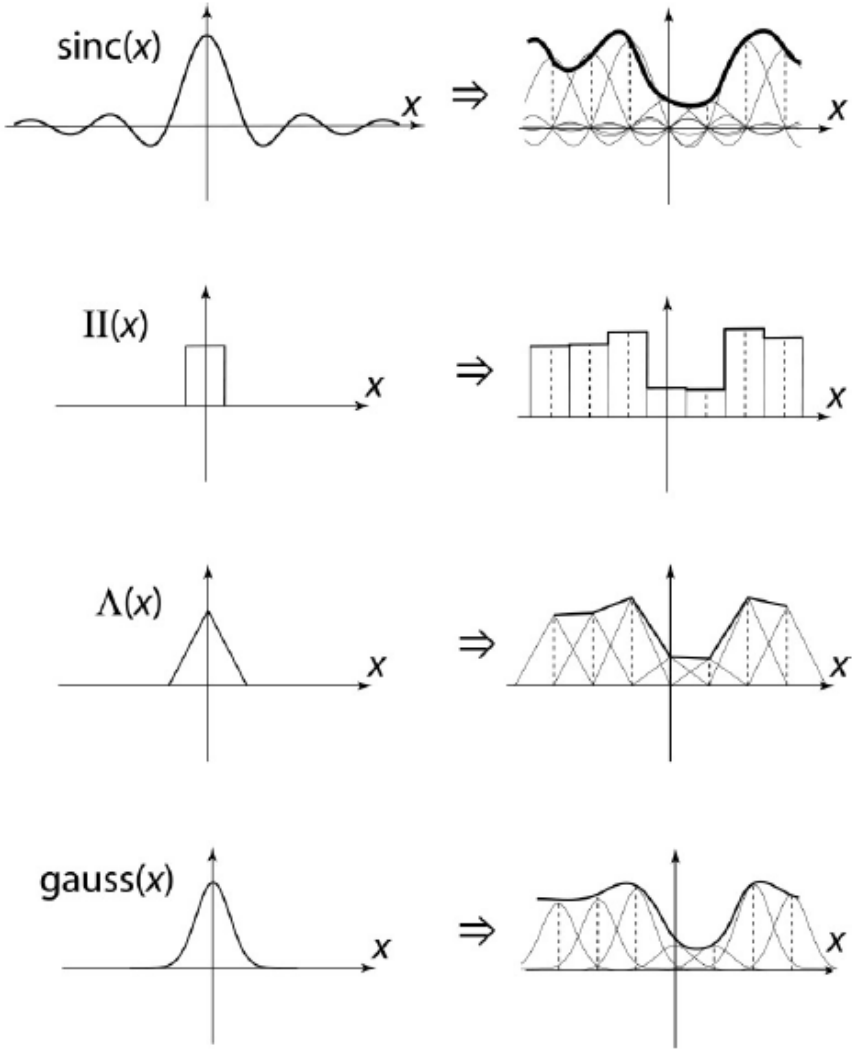
$$\tilde{f} = h * f_F$$

Adapted from: S. Seitz

Image Interpolation



*



"Ideal" reconstruction

Nearest-neighbor interpolation

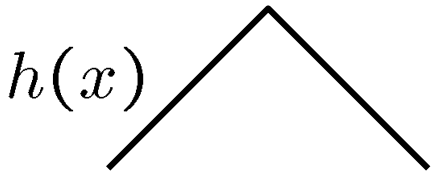
Linear interpolation

Gaussian reconstruction

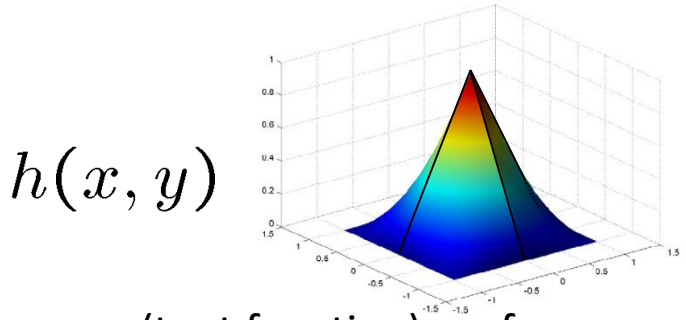
Source: B. Curless

Reconstruction Filters

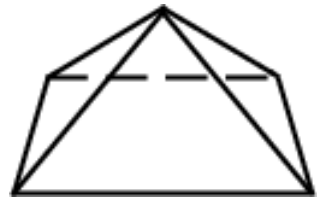
- What does the 2D version of this hat function look like?



performs
linear interpolation



(tent function) performs
bilinear interpolation

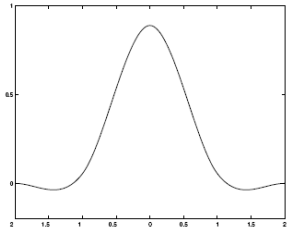


Often implemented without cross-correlation

- E.g., http://en.wikipedia.org/wiki/Bilinear_interpolation

Better filters give better resampled images

- **Bicubic** is common choice



Cubic reconstruction filter

$$r(x)=\frac{1}{6}\begin{cases}(12-9B-6C)|x|^3+(-18+12B+6C)|x|^2+(6-2B) & |x|<1\\ ((-B-6C)|x|^3+(6B+30C)|x|^2+(-12B-48C)|x|+(8B+24C) & 1\le|x|<2\\ 0 & otherwise\end{cases}$$

Image Interpolation

Original image:  x 10

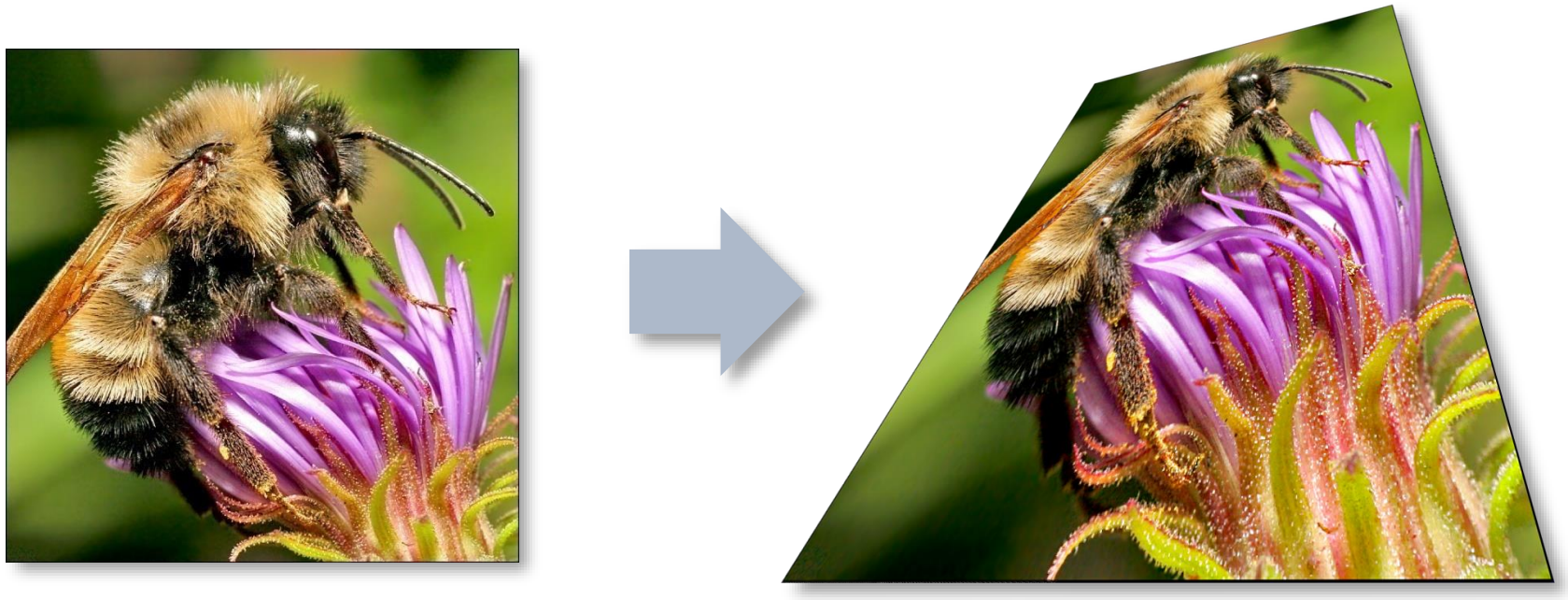
Nearest-neighbor interpolation

Bilinear interpolation

Bicubic interpolation

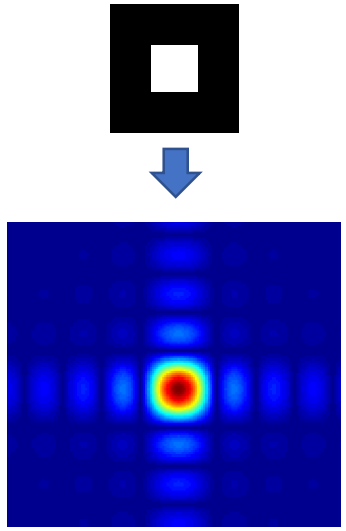
Image Interpolation

Also used for *resampling*

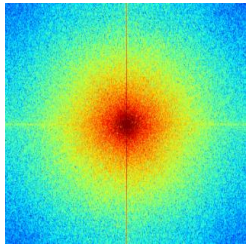


Things to Remember

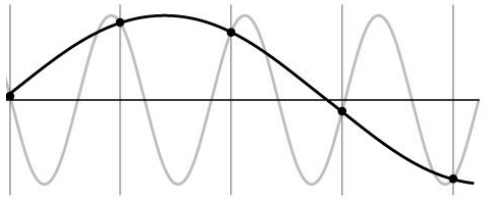
- Sometimes it makes sense to think of images and filtering in the frequency domain
  - Fourier analysis



- Can be faster to filter using FFT for large images (N logN vs. N^2 for auto-correlation)



- Images are mostly smooth
  - Basis for compression
- Remember to low-pass before sampling

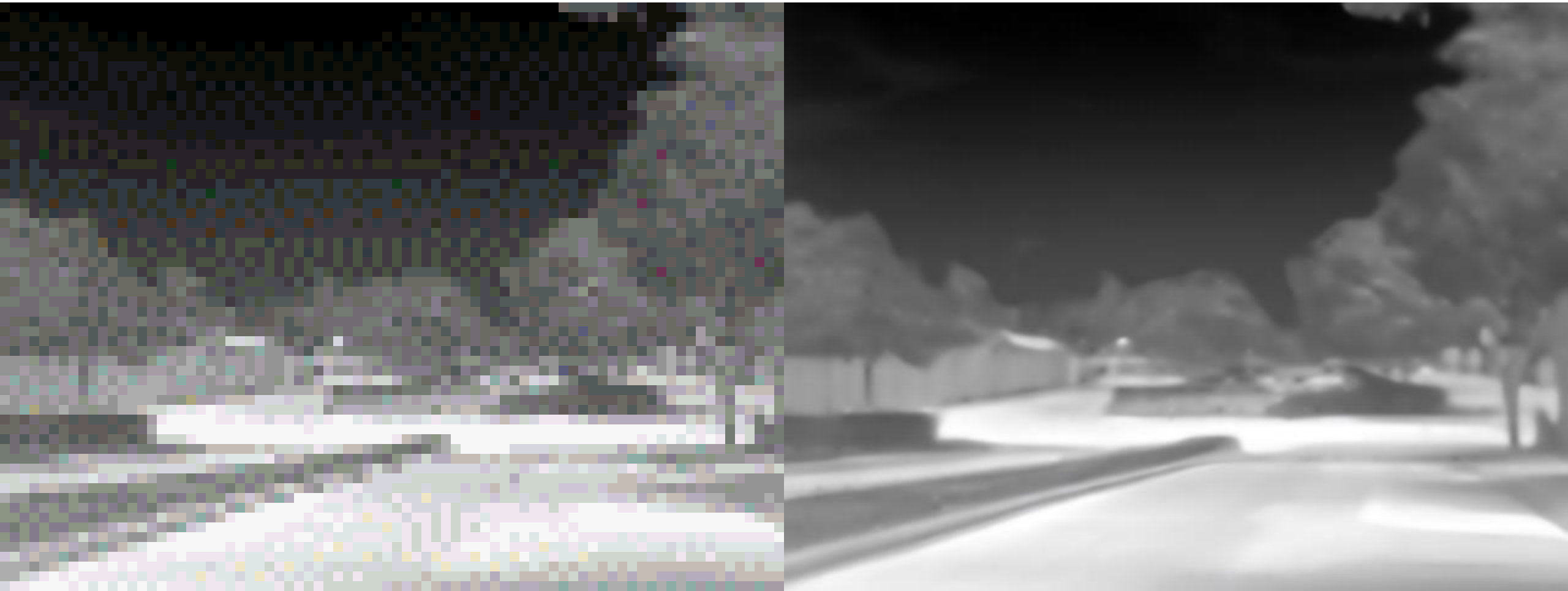


Some Examples

x8

Some Examples

Some Examples



Some Examples

Some Examples

Some Examples

Python code for image super resolution using EDSR

- import cv2
- from cv2 import dnn_superres
- import os
- from os import listdir
- import glob
- sr = dnn_superres.DnnSuperResImpl_create()
- path = "EDSR_x2.pb"
- sr.readModel(path)
- sr.setModel("edsr",2)
- img = cv2.imread(“image path”)
- result = sr.upsample(img)
- cv2.imwrite(filename,result)

MATLAB

- MATLAB provides an example of super resolution using deep learning method VDSR

- Follow the steps in the link below

- https://www.mathworks.com/help/images/single-image-super-resolution-using-deep-learning.html