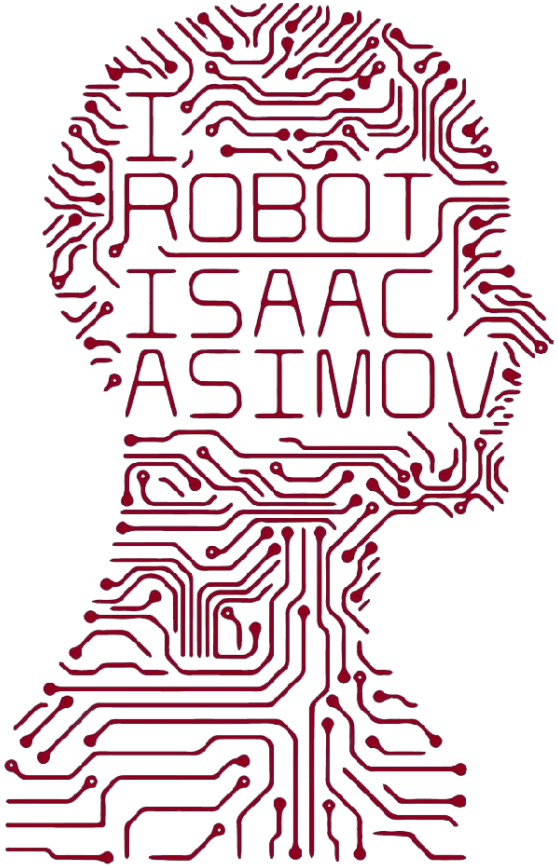
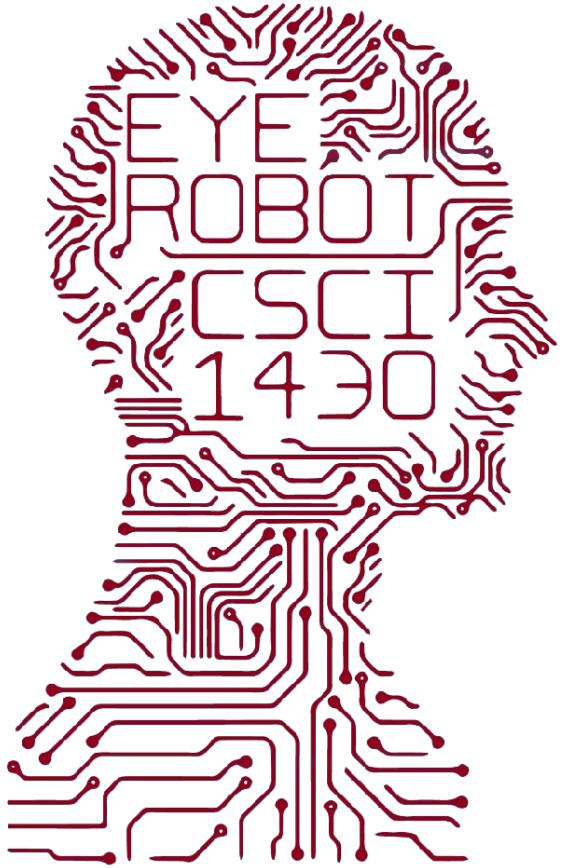


FUTURE VISION



COMPUTER VISION

Review of Last Lectures

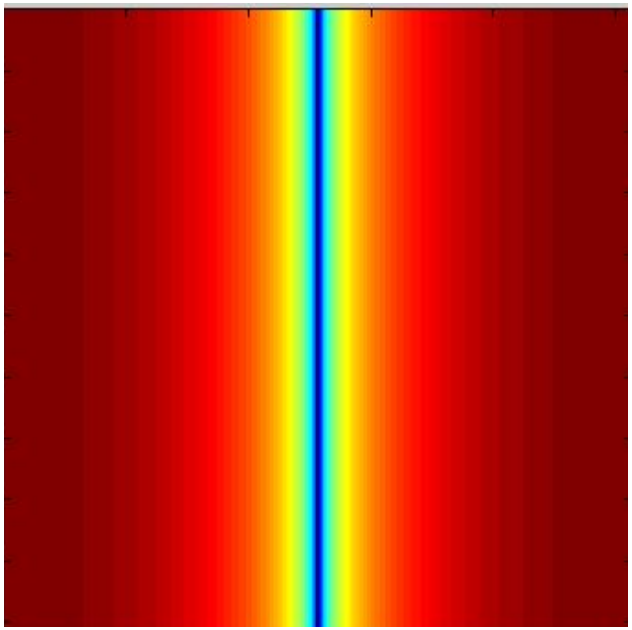
- Applications of filters
  - Template matching (SSD or Normxcorr2)
    - SSD can be done with linear filters, is sensitive to overall intensity
  - Gaussian pyramid
    - Coarse-to-fine search, multi-scale detection
  - Laplacian pyramid
    - Teases apart different frequency bands while keeping spatial information
    - Can be used for compositing in graphics
  - Downsampling
    - Need to sufficiently low-pass before downsampling

Review of Filtering

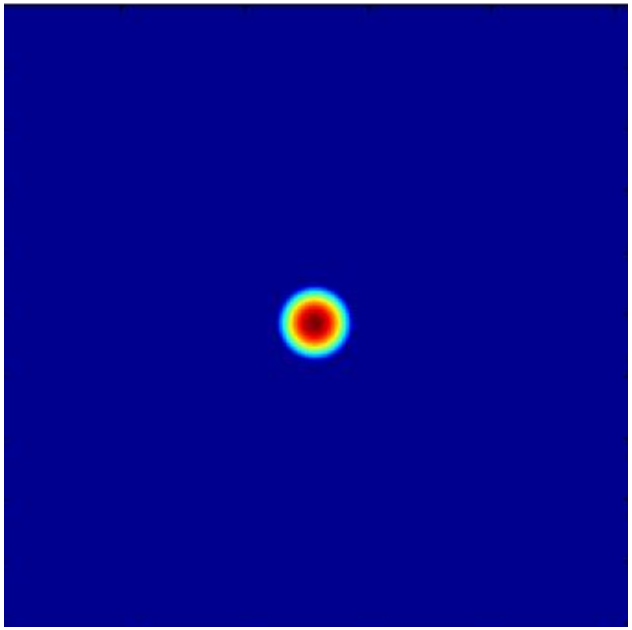
Linear filters for basic processing

- Edge filter (high-pass)
- Gaussian filter (low-pass)

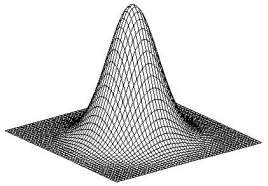
[-1 1]



FFT of Gradient Filter

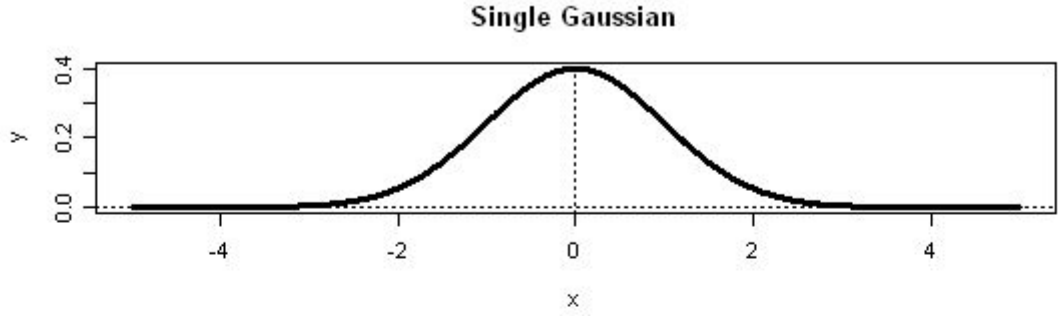


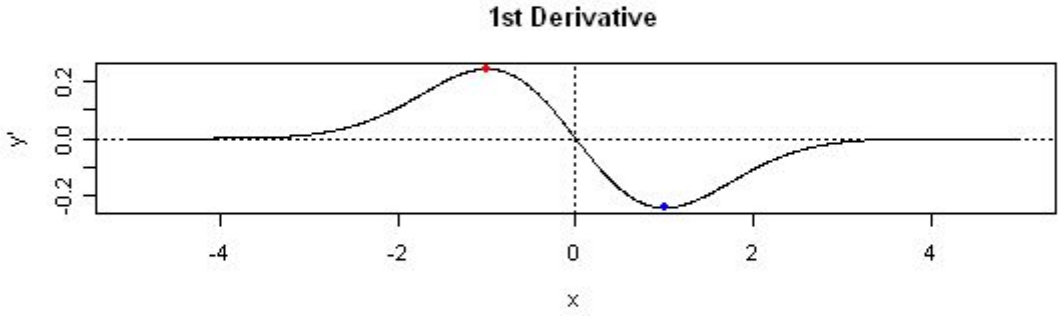
FFT of Gaussian

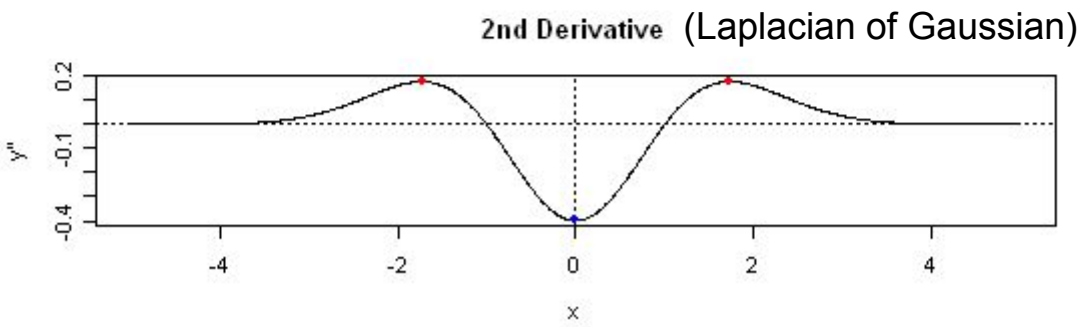


Gaussian

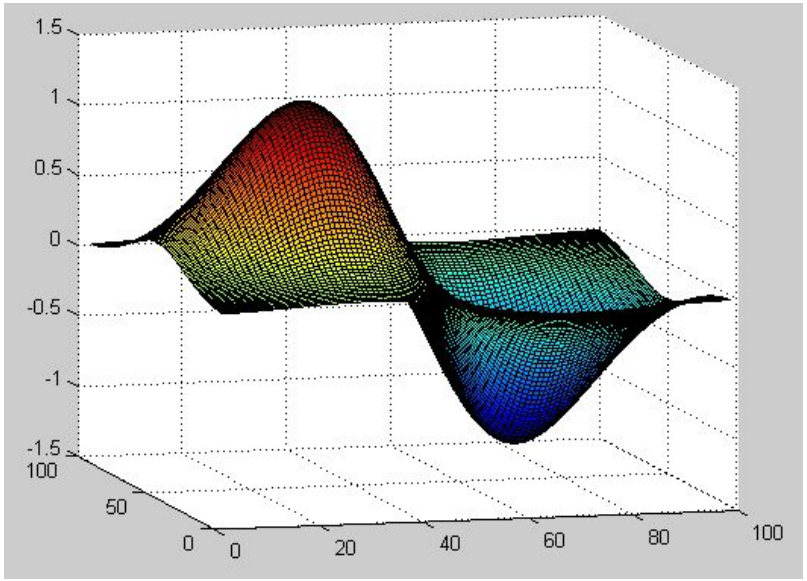
More Useful Filters



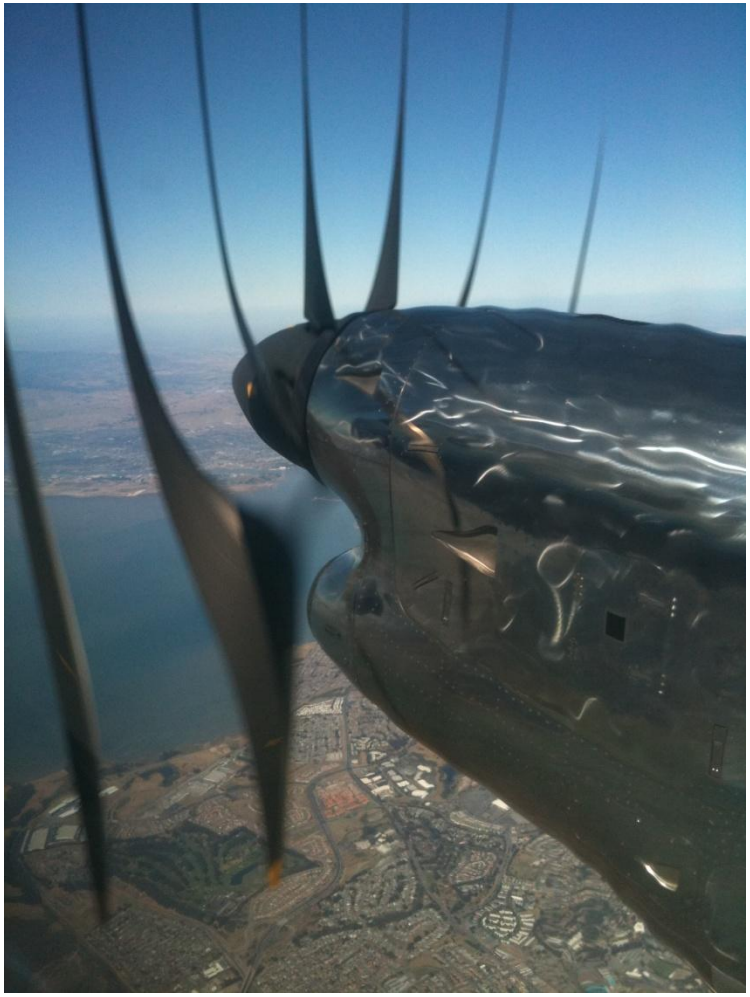




1st Derivative of Gaussian



Capture Frequency - Rolling `Shutter’

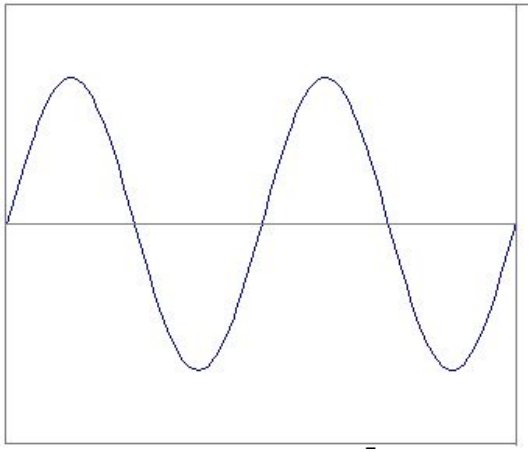


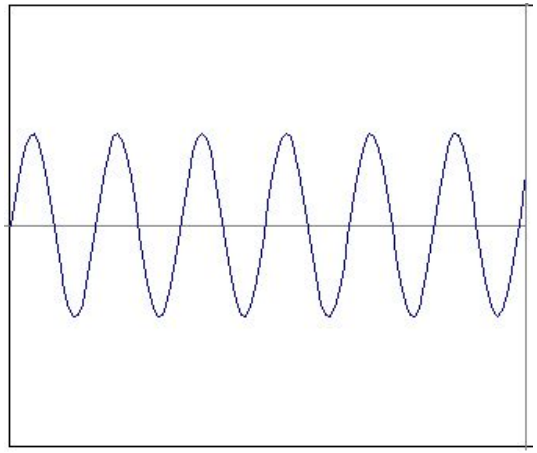
James Hays

Thinking in frequency

FOURIER SERIES & FOURIER TRANSFORMS

I understand frequency as in waves…





…but how does this relate to the complex signals we see in natural images?

…to image frequency?

Fourier series

A bold idea (1807):

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

Our building block:

$$\sum_{n=1}^{\infty} (a_n \cos(nt) + b_n \sin(nt))$$

Add enough of them to get any signal *g(t)* you want!

Jean Baptiste Joseph Fourier (1768-1830)

Fourier series

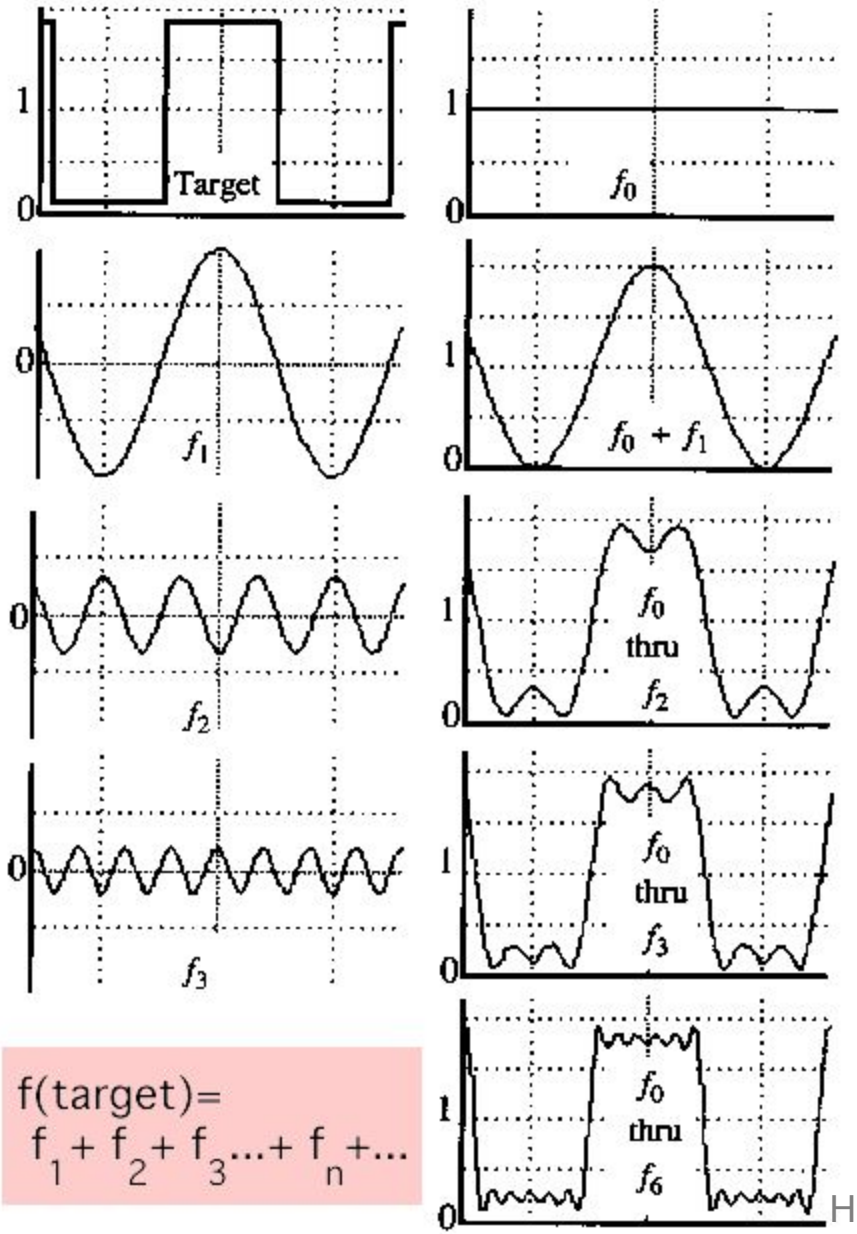
A bold idea (1807):

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

Our building block:

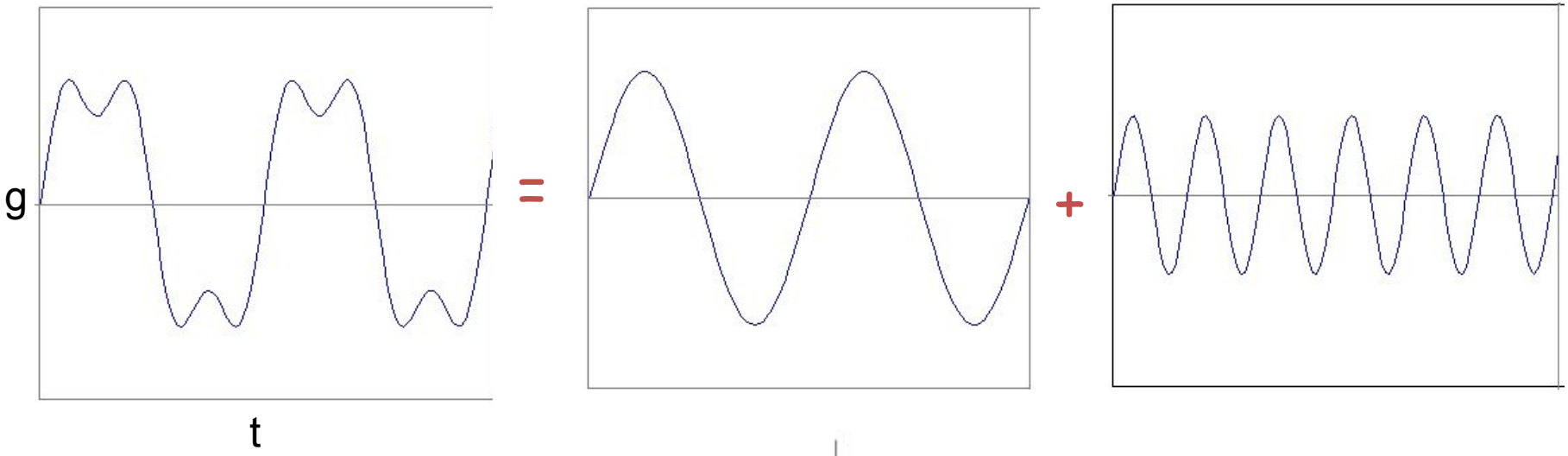
$$A\sin(\omega t) + B\cos(\omega t)$$

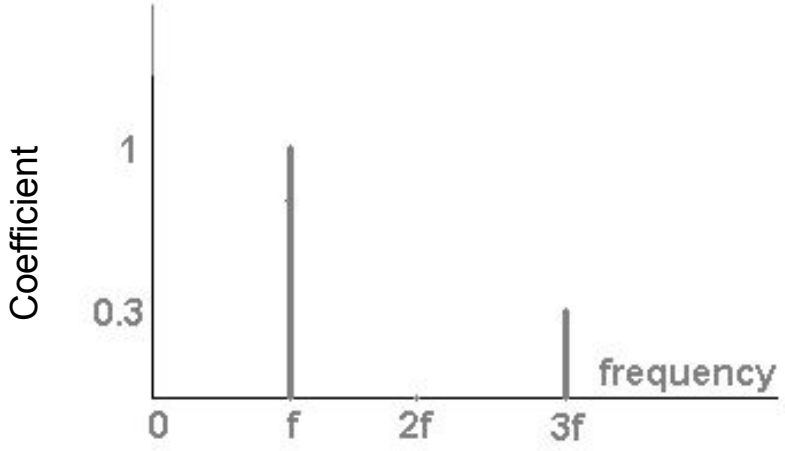
Add enough of them to get any signal *g(t)* you want!



$t = [0,2], f = 1$

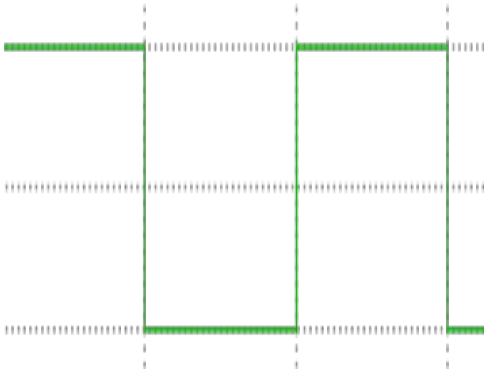
$$g(t) = (1)\sin(2\pi f\, t) + (1/3)\sin(2\pi(3f)\, t)$$



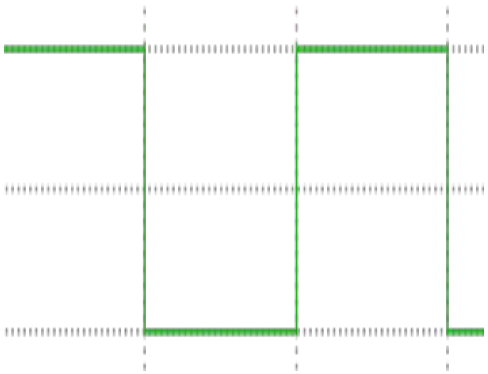


Slides: Efros

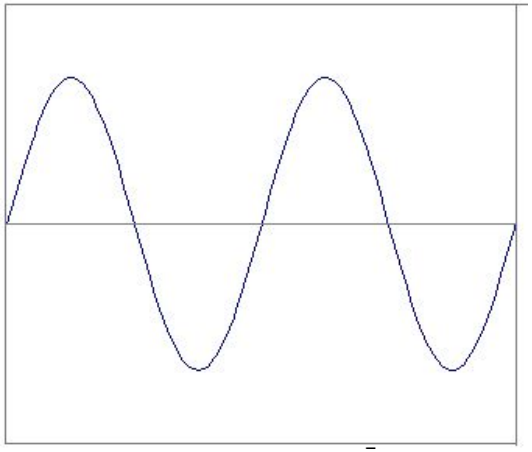
Square wave spectra



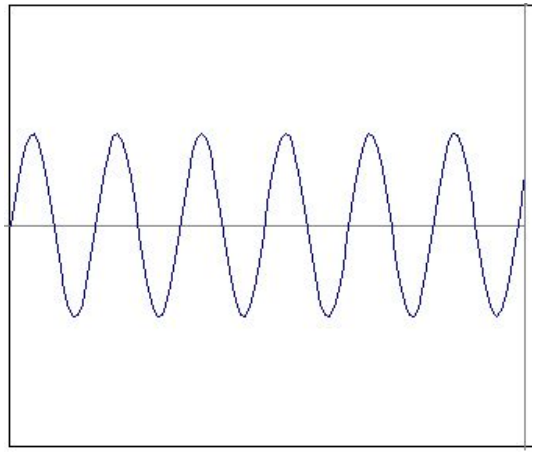
Square wave spectra



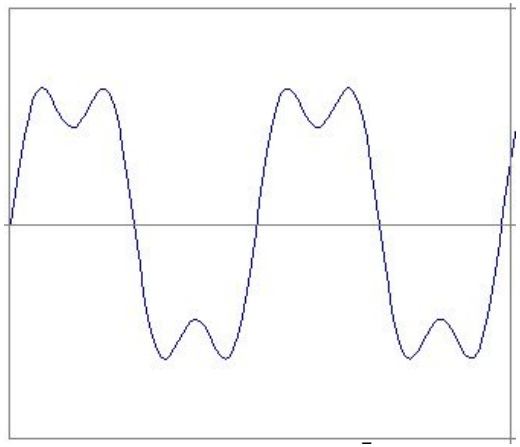
=



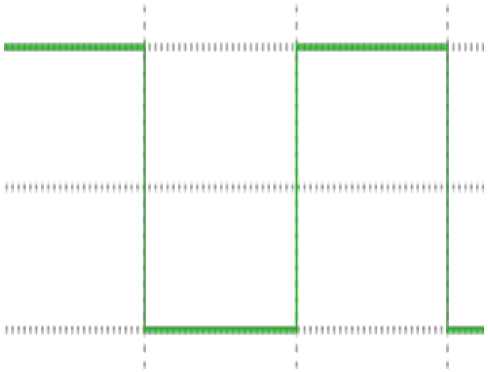
+



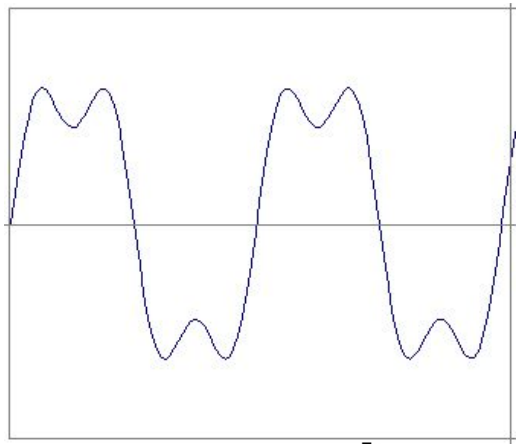
=



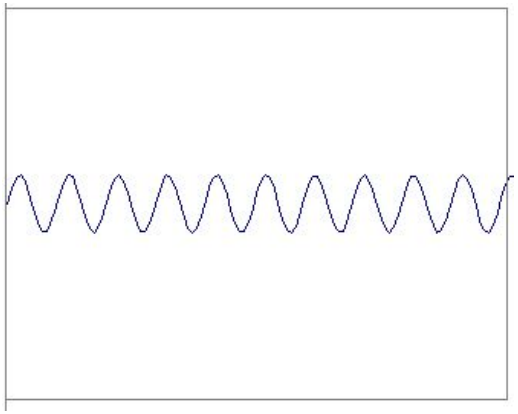
Square wave spectra



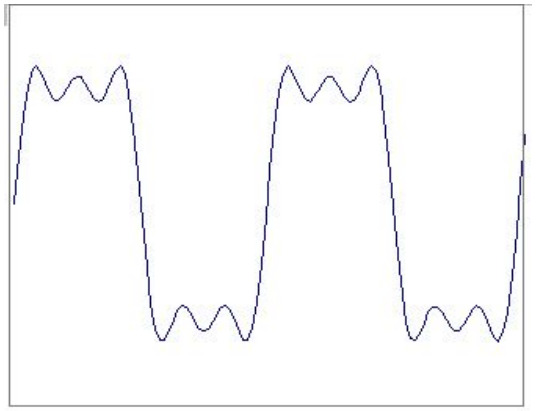
=



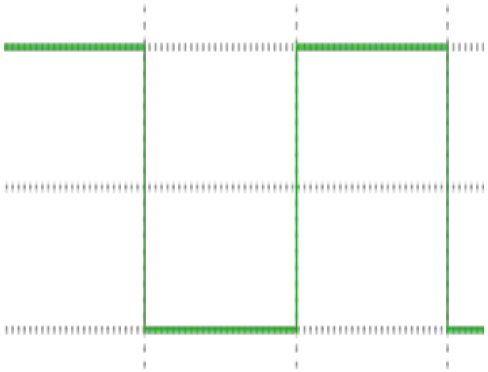
+

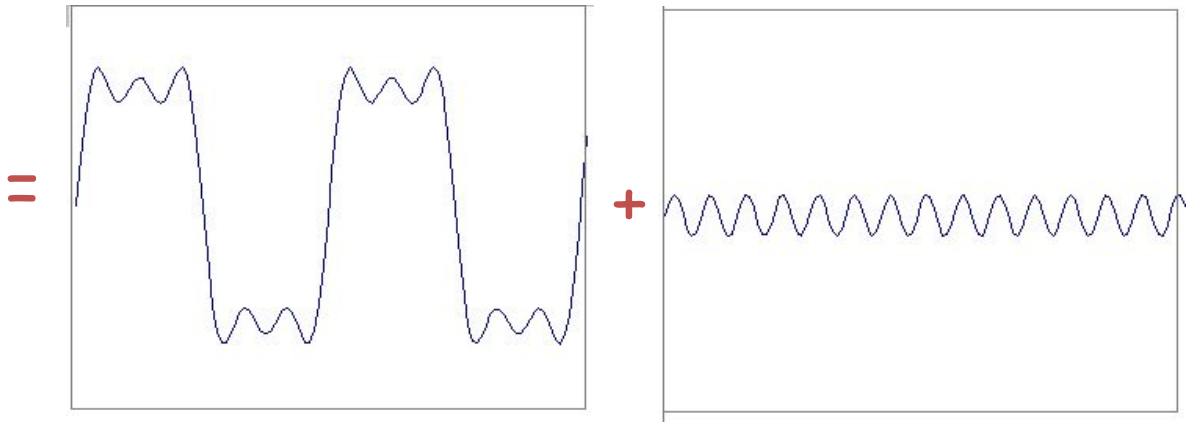


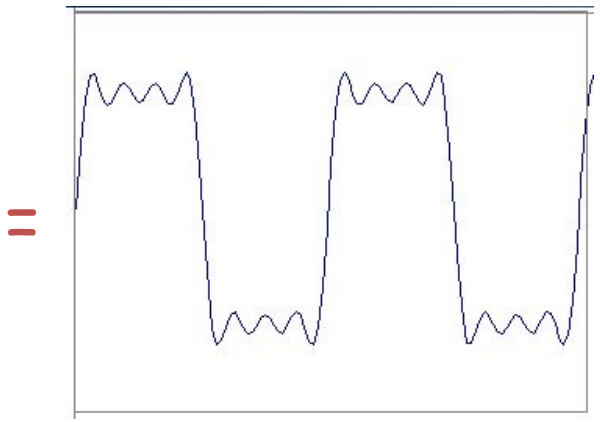
=



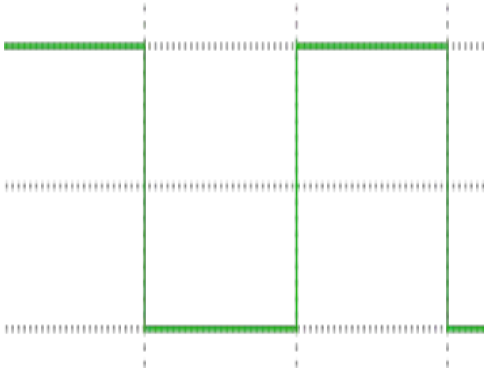
Square wave spectra

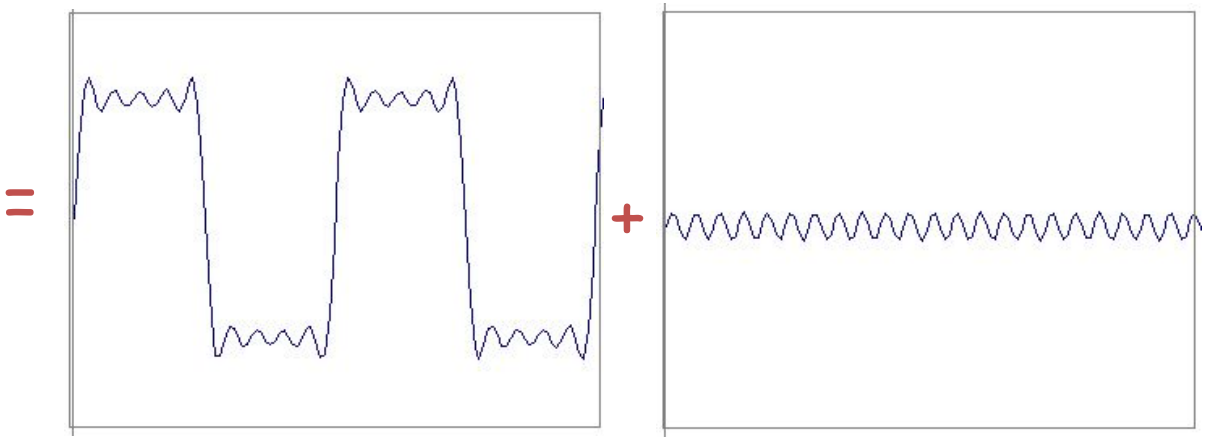


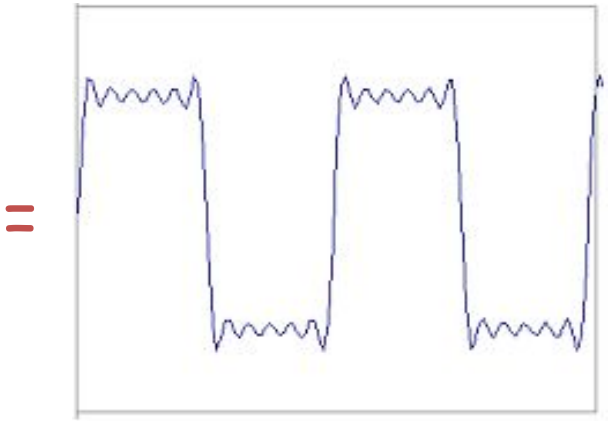




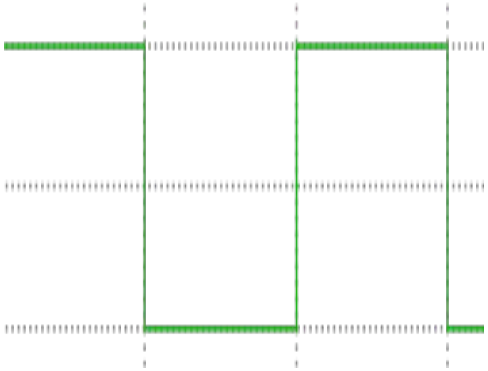
Square wave spectra

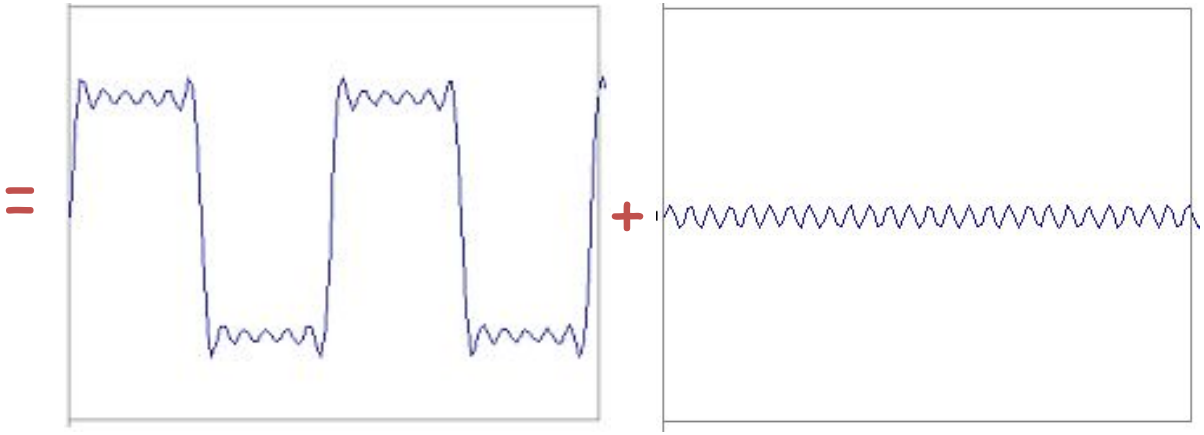


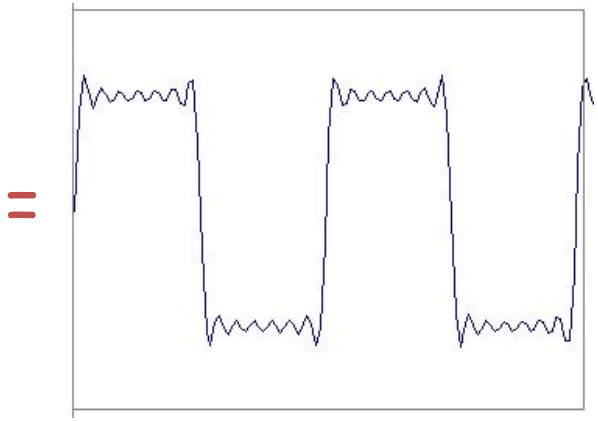




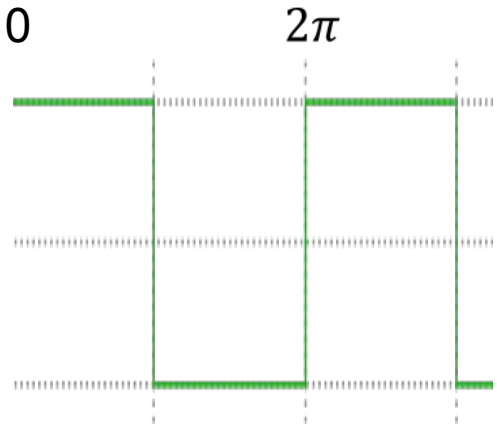
Square wave spectra







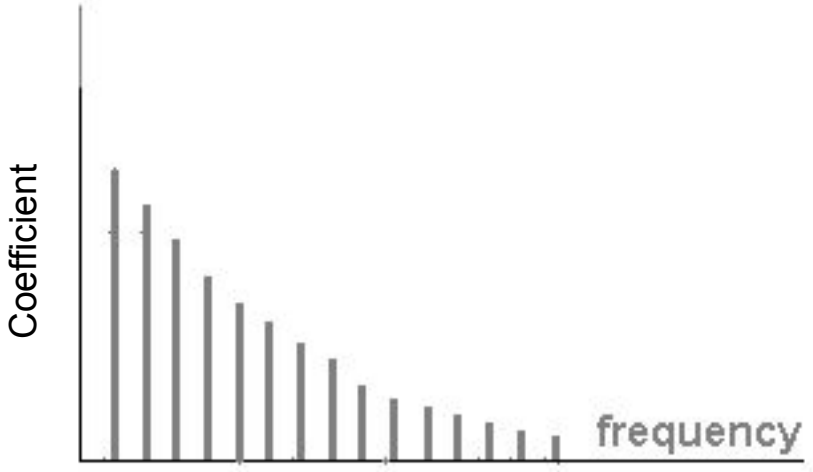
Square wave spectra



=

$$\sum_{n=1}^{\infty} \frac{1}{n} \sin(nt)$$

For periodic signals defined over $[0, 2\pi]$



Jean Baptiste Joseph Fourier (1768-1830)

A bold idea (1807):

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

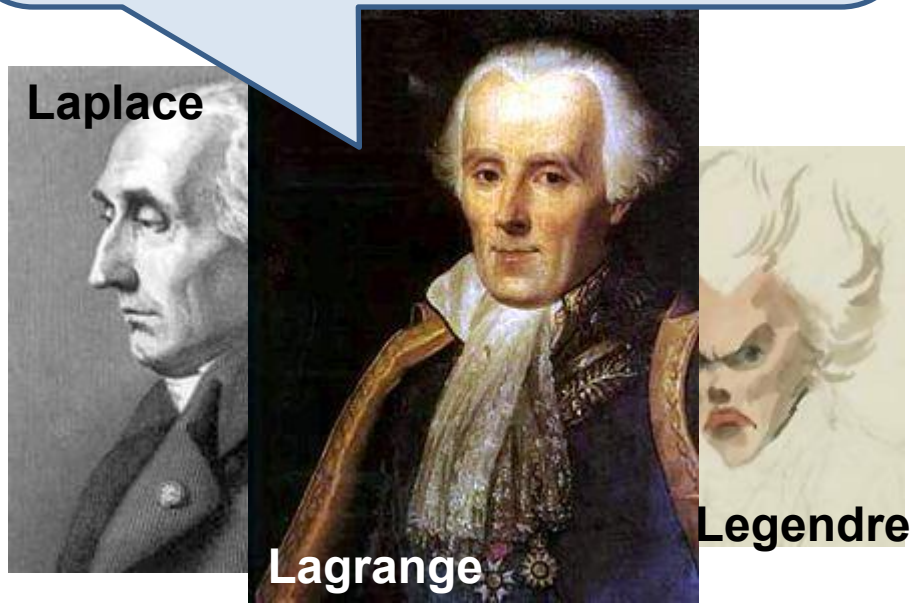
Don't believe it?

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

But it's (mostly) true!

- Called Fourier Series
- *Applies to periodic signals*

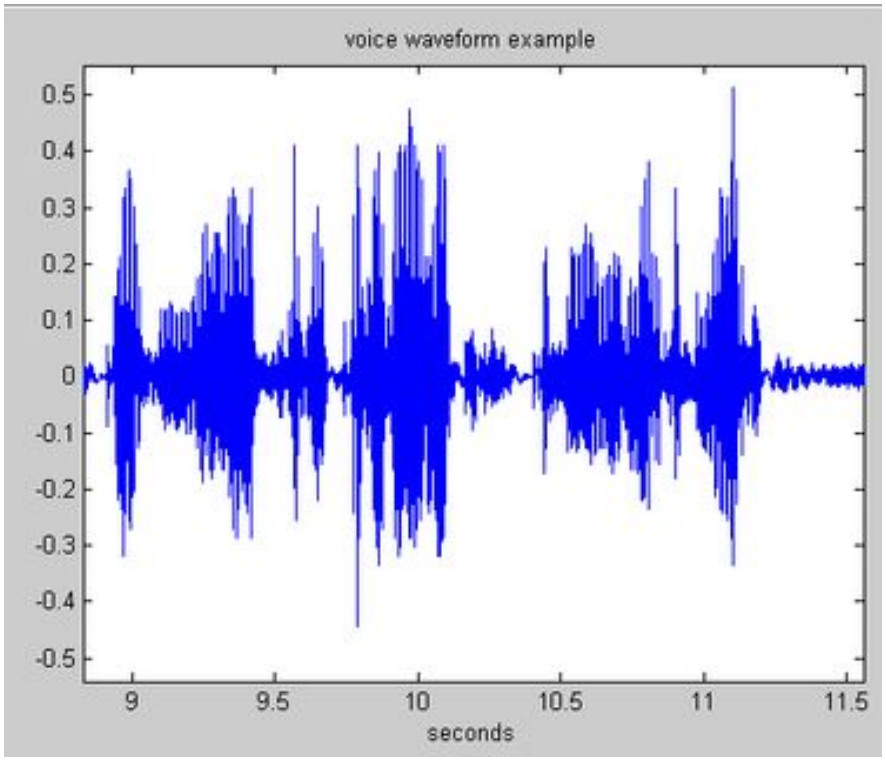
...the manner in which the author arrives at these equations is not exempt of difficulties and...his analysis to integrate them still leaves something to be desired on the score of generality and even rigour.



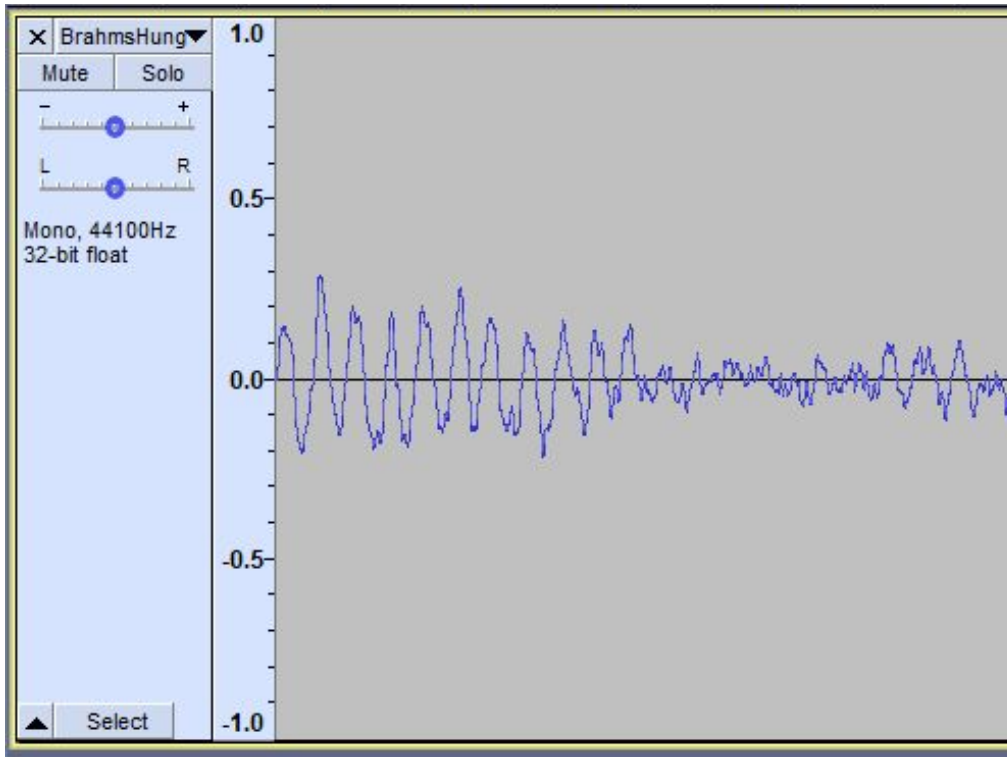
Reviewer #2

Example: Music

We think of music in terms of pitches (frequencies) at different loudnesses (amplitudes).



Zoom in:

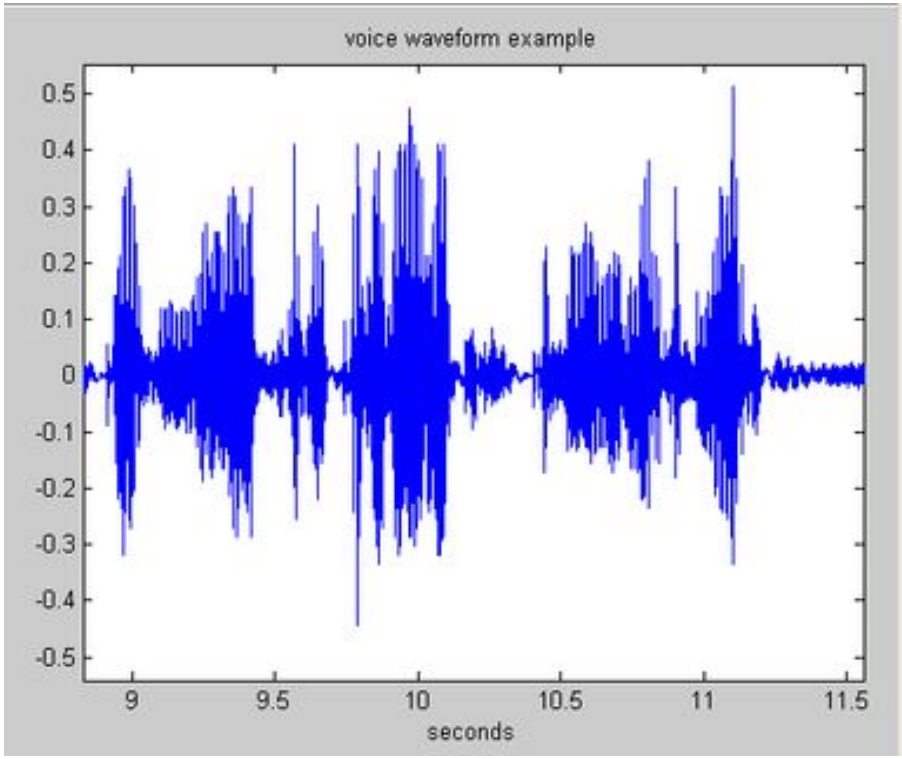


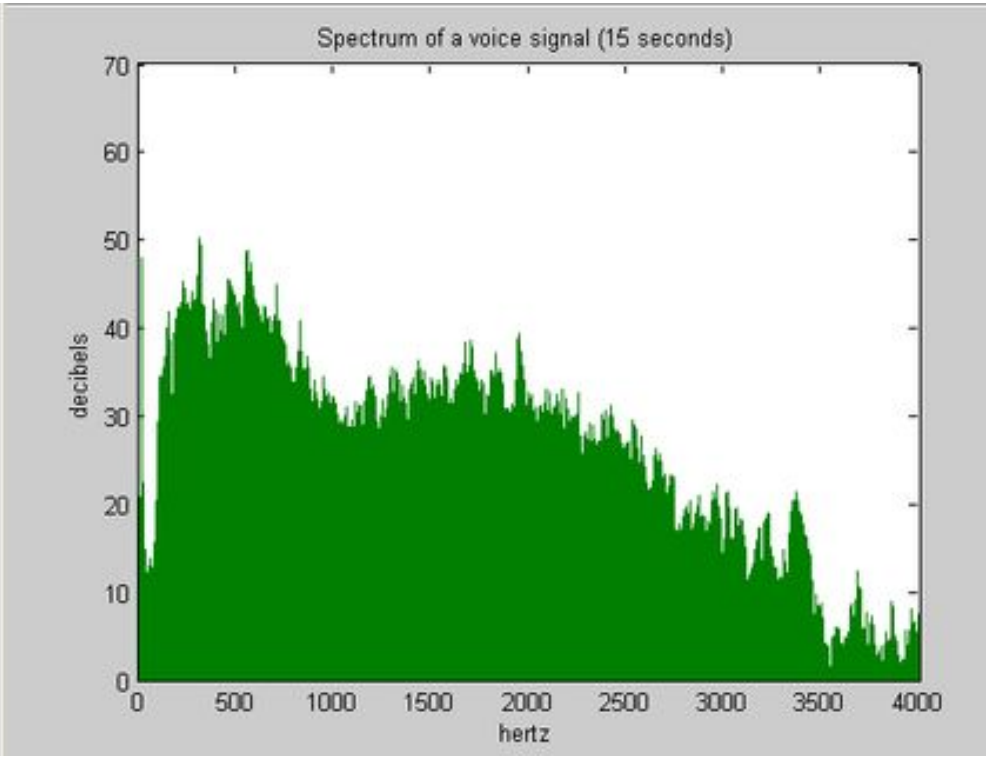
Samples

Slide: Hoiem

Example: Music

We think of music in terms of pitches (frequencies) at different loudnesses (amplitudes).

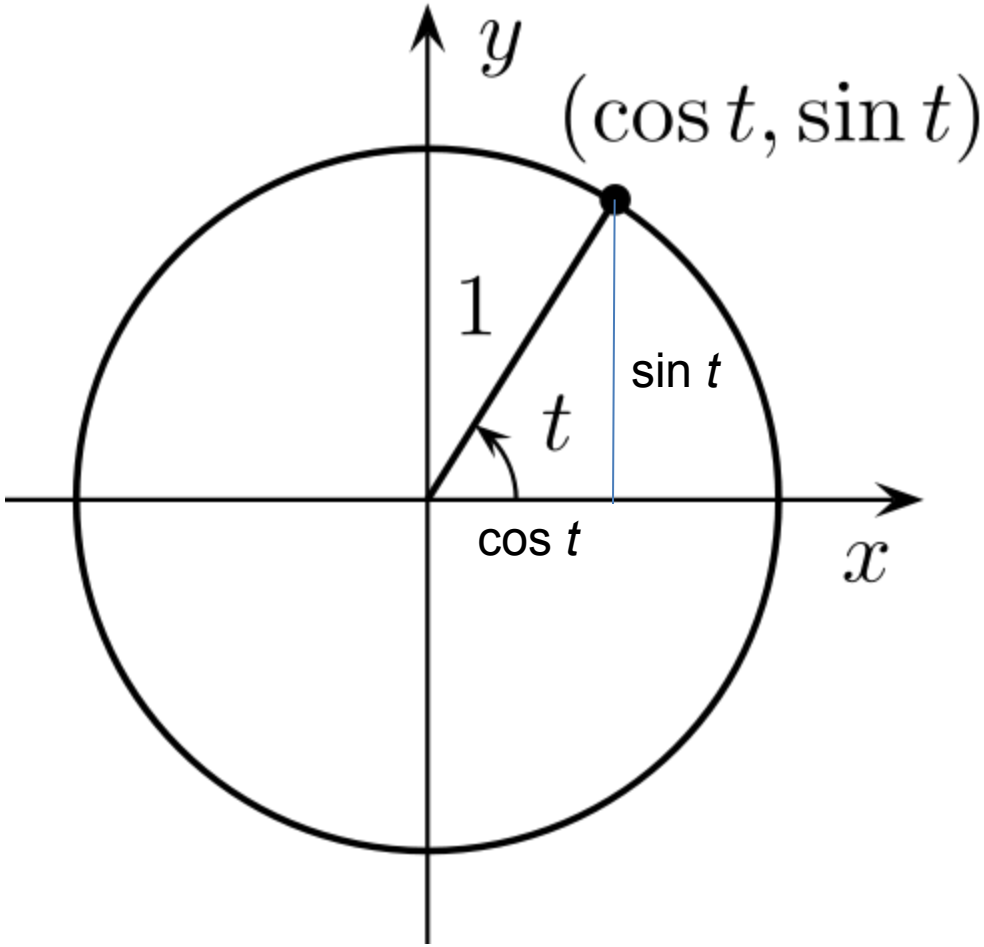




Samples

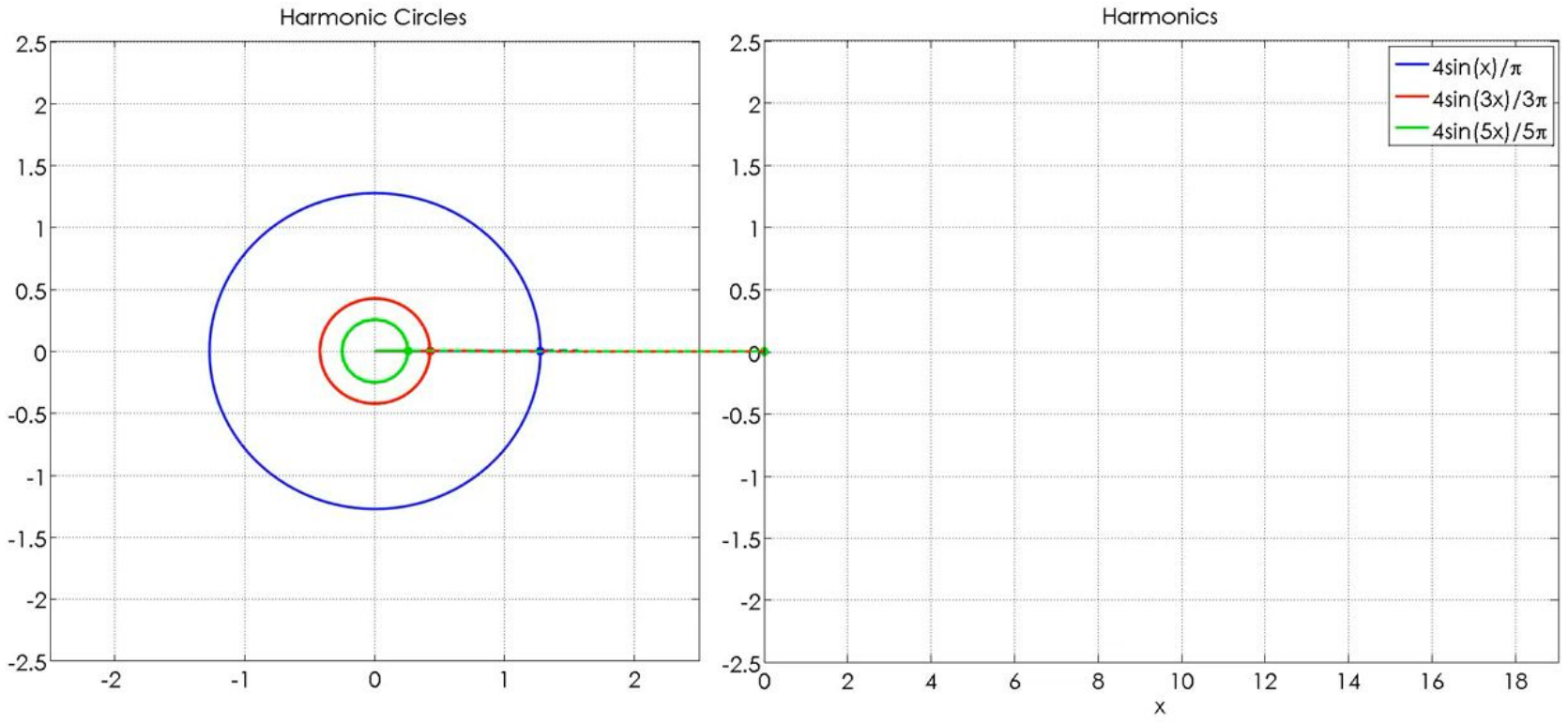
Slide: Hoiem

Sine/cosine and circle

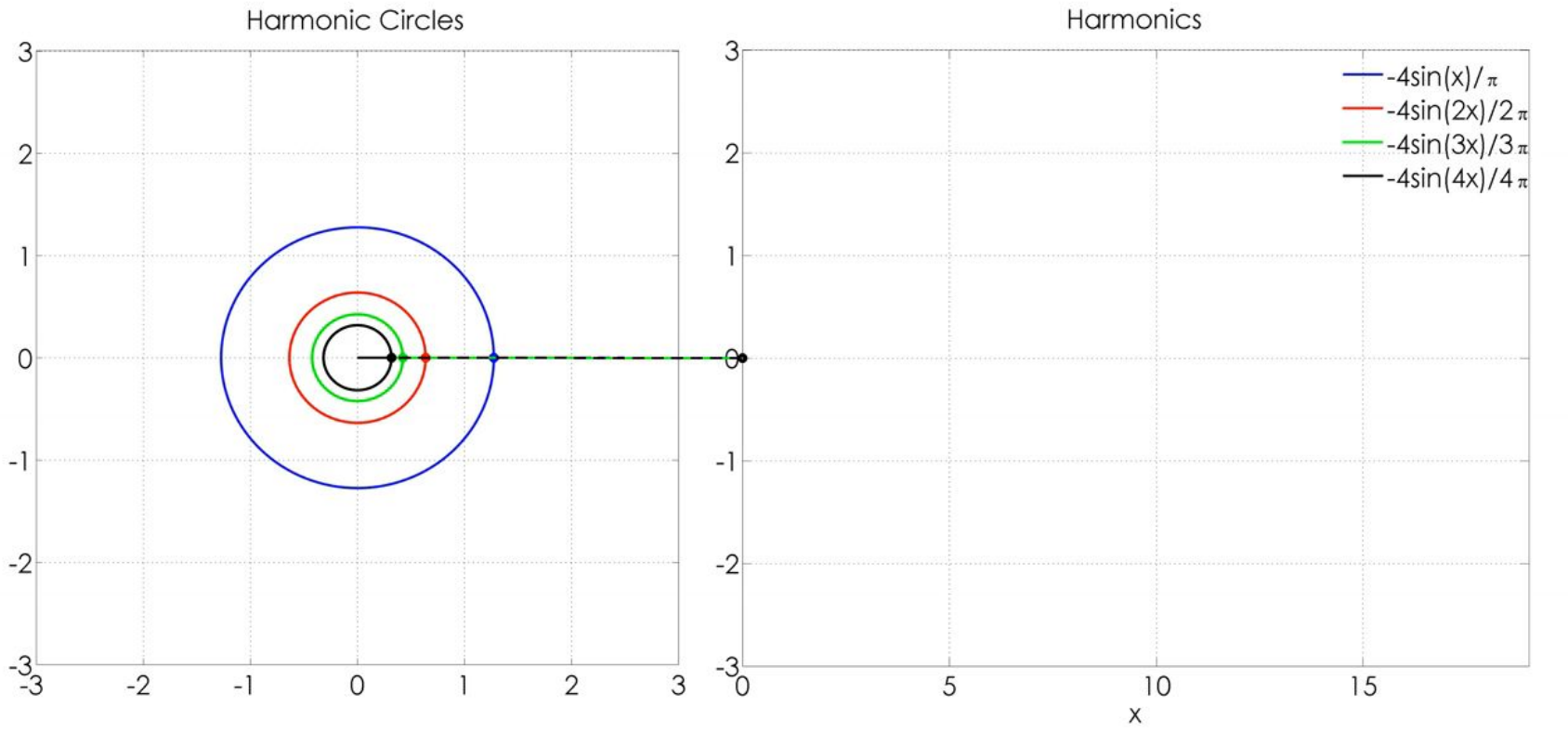


Wikipedia – Unit Circle

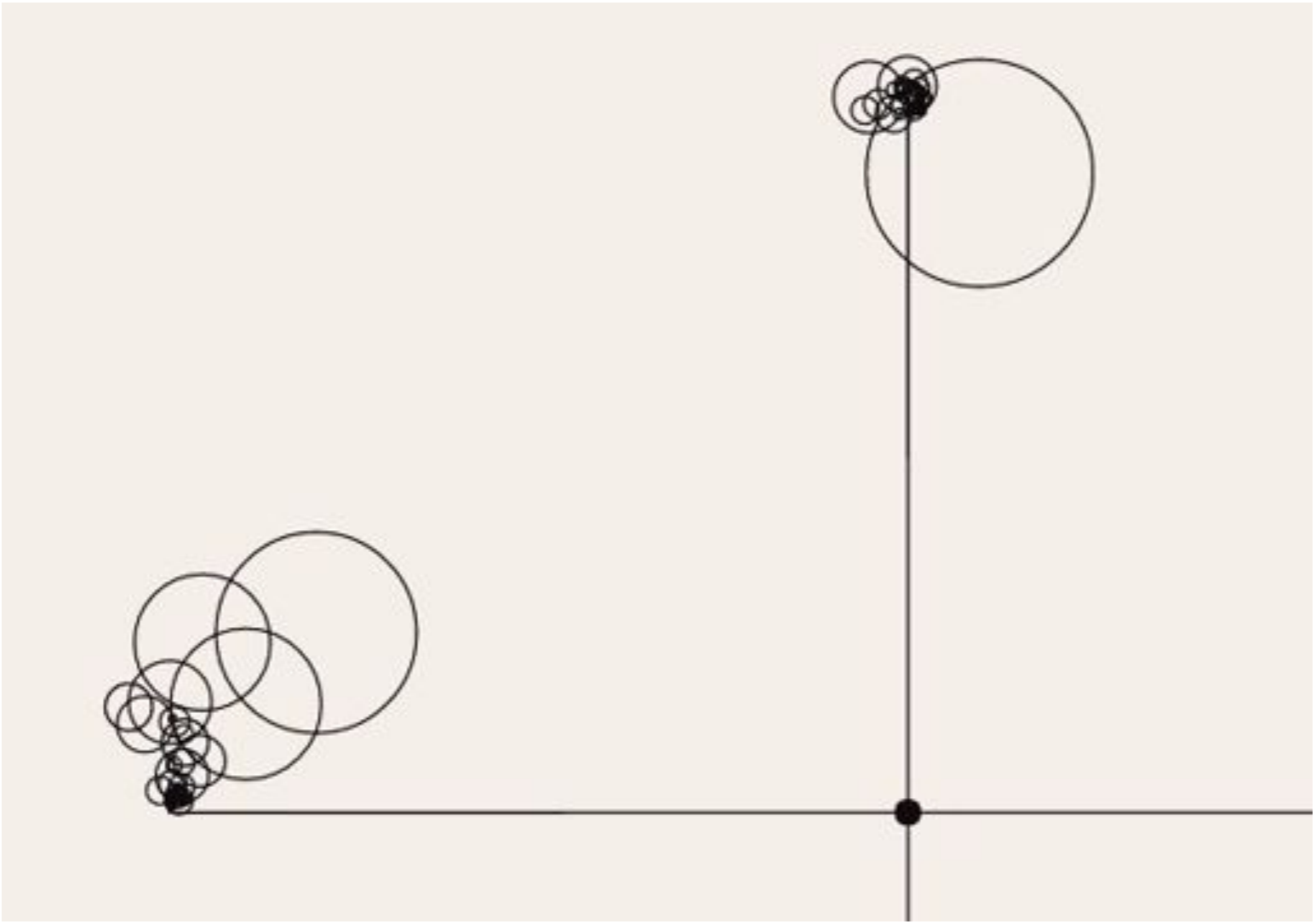
Square wave (approx.)



Sawtooth wave (approx.)

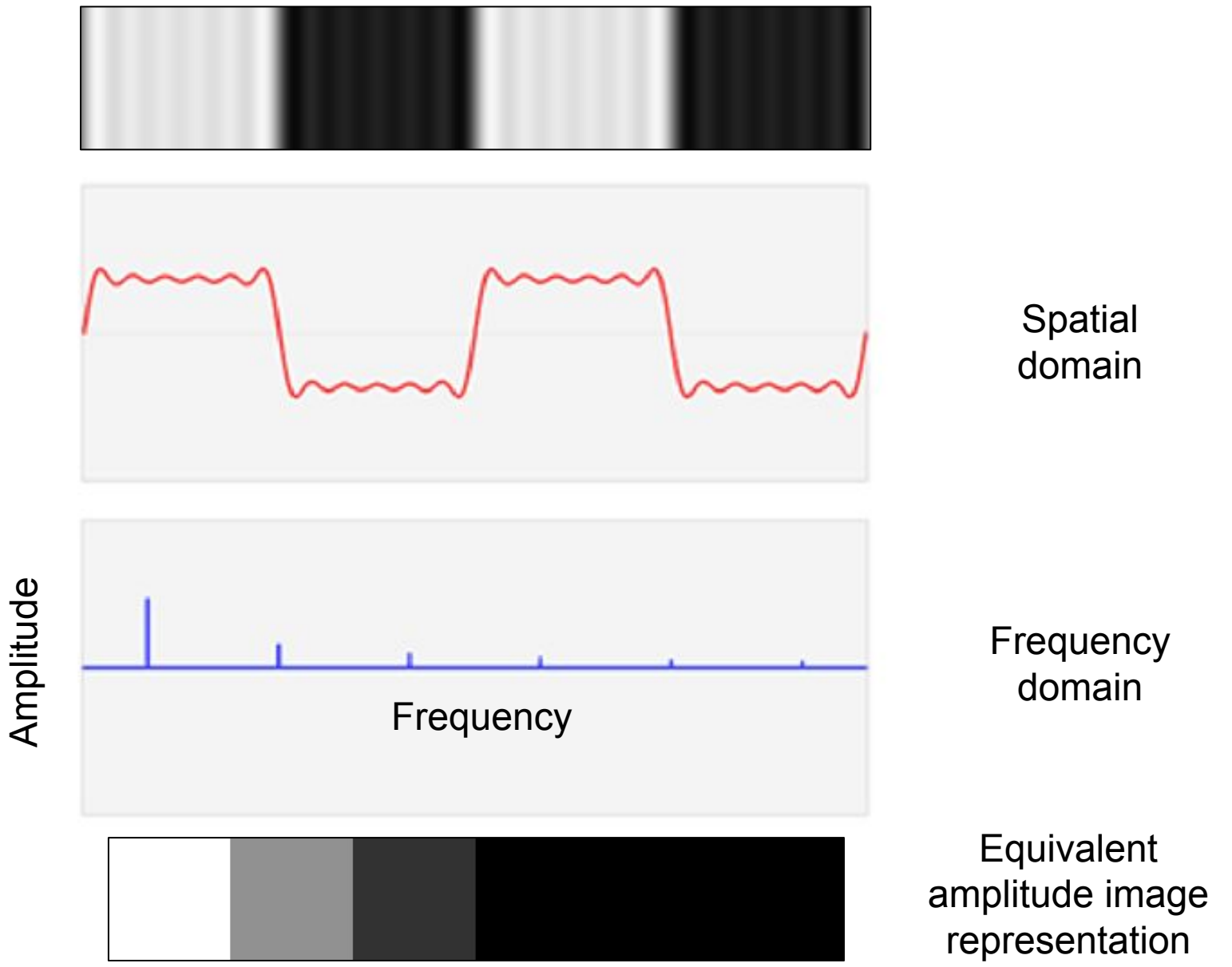


One series in each of x and y



Generative Art, じゃがりきん, Video, 2018

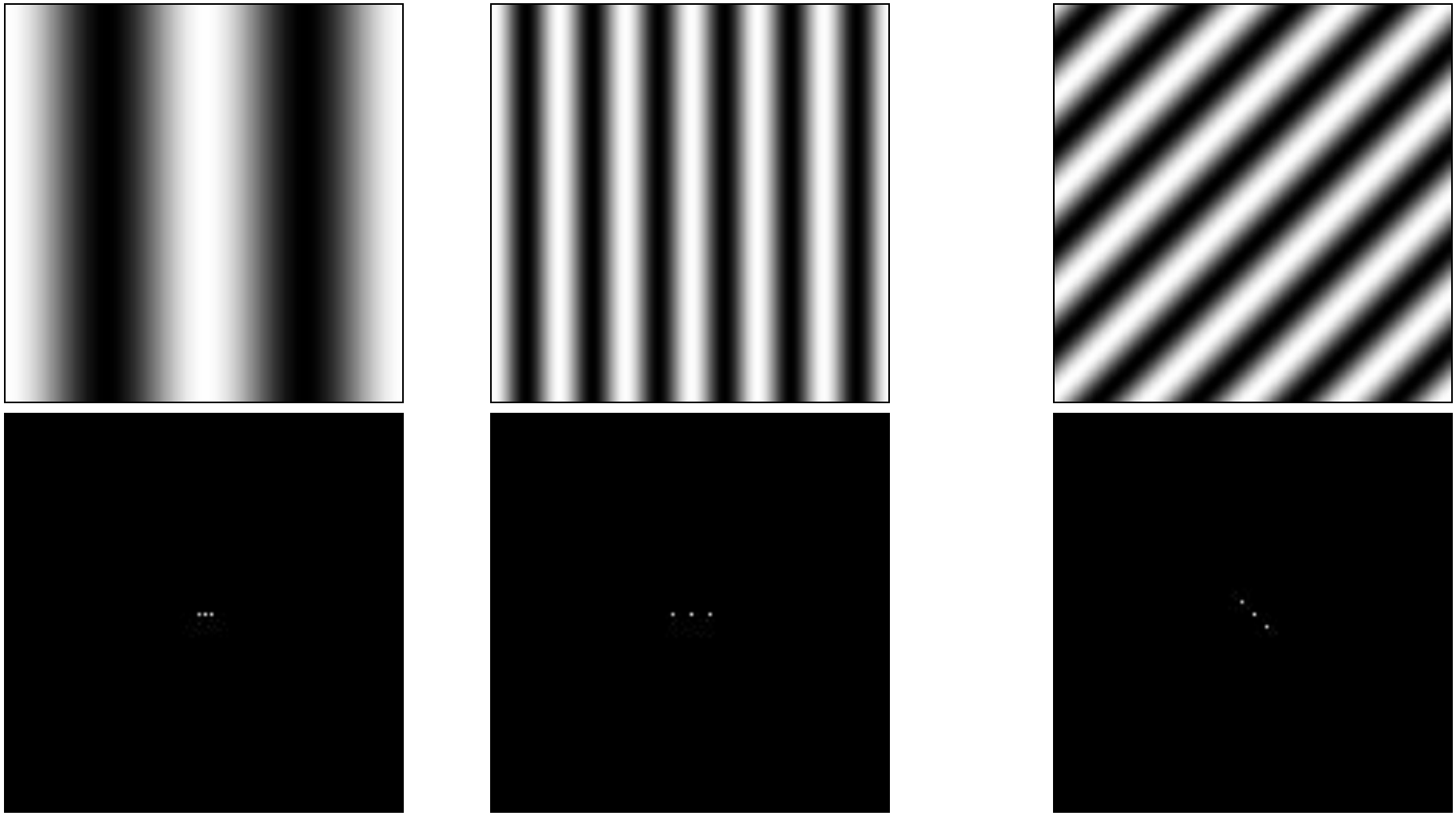
Wikipedia – Fourier transform



Wikipedia – Fourier transform

Fourier analysis in images

Spatial domain images



Fourier decomposition amplitude images

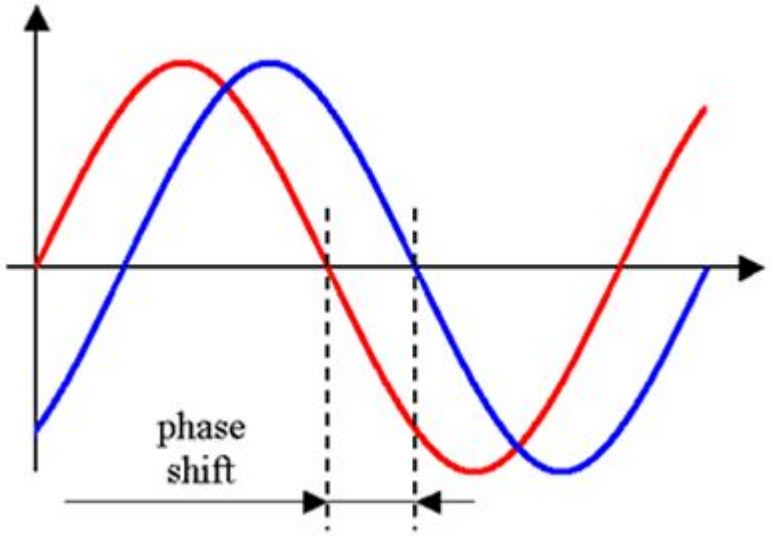
http://sharp.bu.edu/~slehar/fourier/fourier.html#filtering More: http://www.cs.unm.edu/~brayer/vision/fourier.html

Amplitude-phase form

Add phase term to shift
cos values into sin values

$$\sum_{n=1}^{N} (a_n \sin(nx + \emptyset_n))$$

Phase

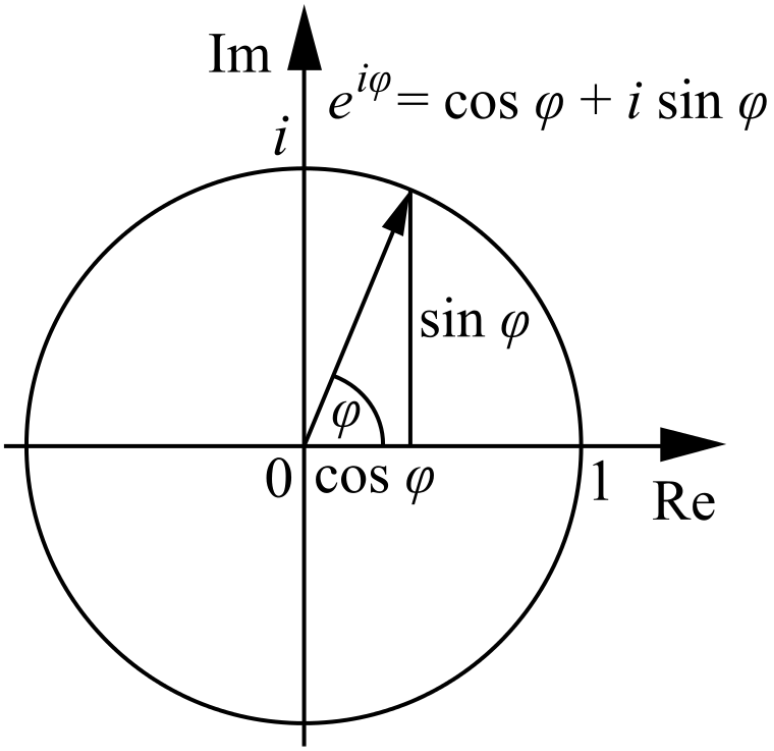


Morse

[Peppergrower; Wikipedia]

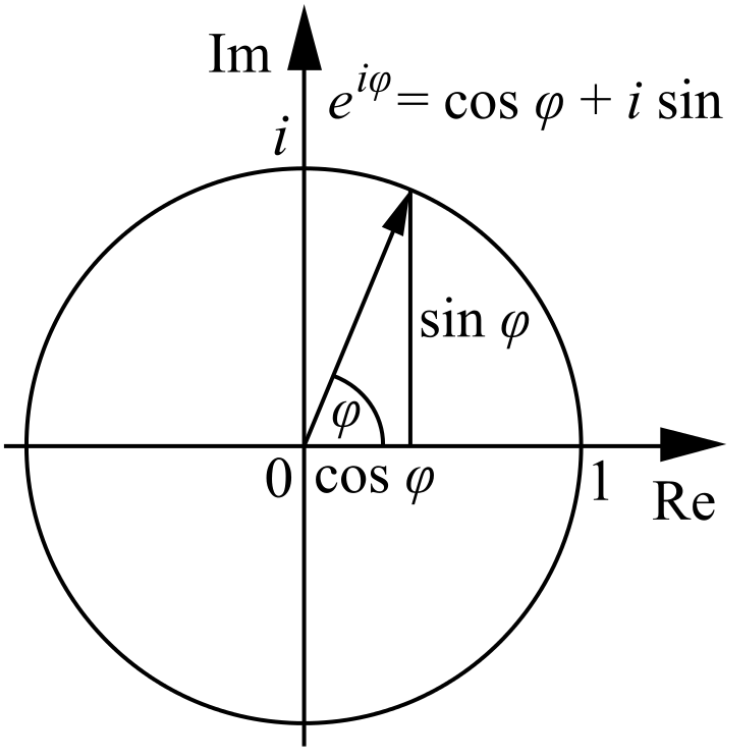
Fourier Transform

- Stores the amplitude and phase at each frequency:
  - For mathematical convenience, this is often notated in terms of real and complex numbers
  - Related by Euler's formula



Fourier Transform

- Stores the amplitude and phase at each frequency:
  - For mathematical convenience, this is often notated in terms of real and complex numbers
  - Related by Euler's formula



Amplitude encodes how much signal there is at a particular frequency:

$$A = \pm\sqrt{\mathrm{Re}(\varphi)^2 + \mathrm{Im}(\varphi)^2}$$

Phase encodes spatial information (indirectly):

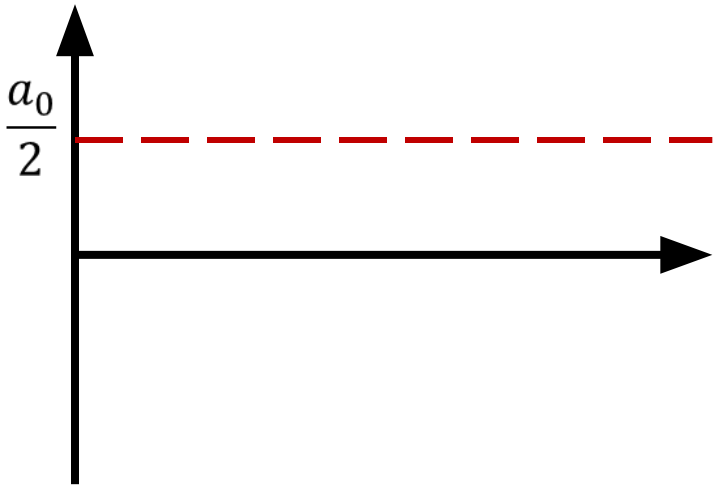
$$\phi = \tan^{-1}\frac{\mathrm{Im}(\varphi)}{\mathrm{Re}(\varphi)}$$

Amplitude-phase form

Add component of “zero” frequency
= mean of signal over period
= value around which signal fluctuates

$$\frac{a_0}{2} + \sum_{n=1}^{N} (a_n \sin(nx + \emptyset_n))$$

Average of signal over period



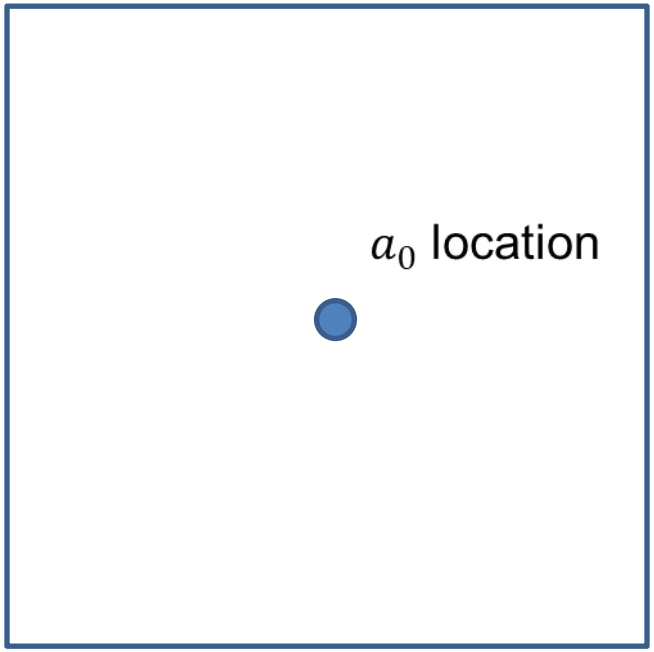
Electronics signal processing calls this the ‘DC offset’

How to read 2D Fourier transform images

We display the space such that zero-frequency coefficient is in the center.

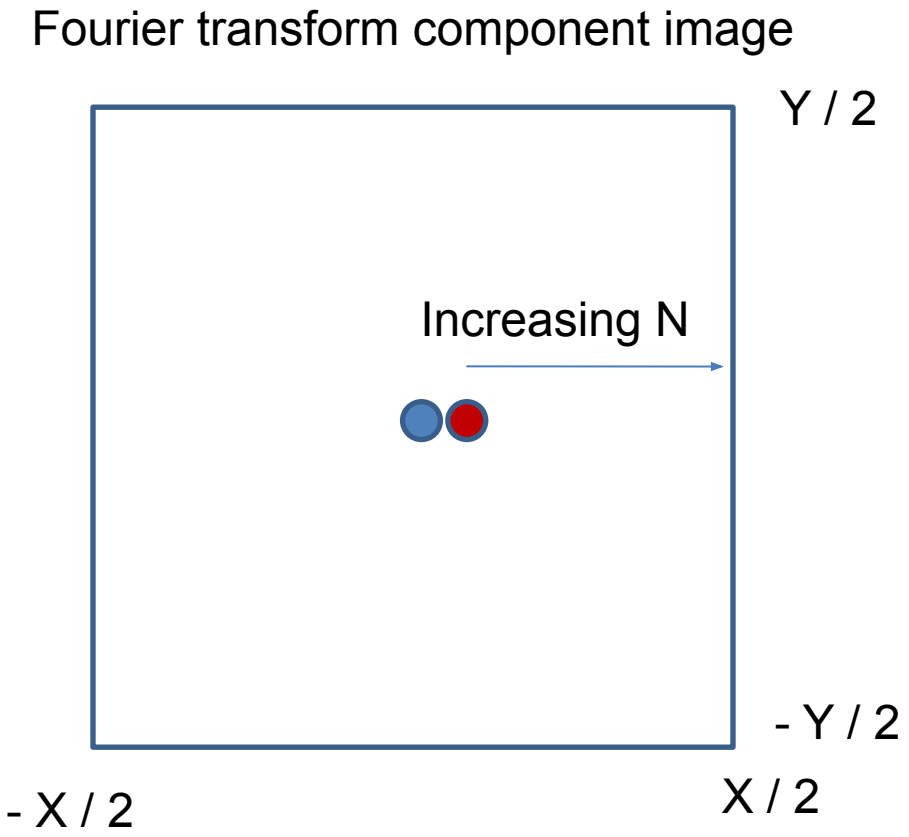
$$\frac{a_0}{2} + \sum_{n=1}^{N} (a_n \sin(nx + \emptyset_n))$$

Fourier transform component image



How to read 2D Fourier transform images

$$\frac{a_0}{2} + \sum_{n=1}^{N} (a_n \sin(nx + \emptyset_n))$$



How to read 2D Fourier transform images

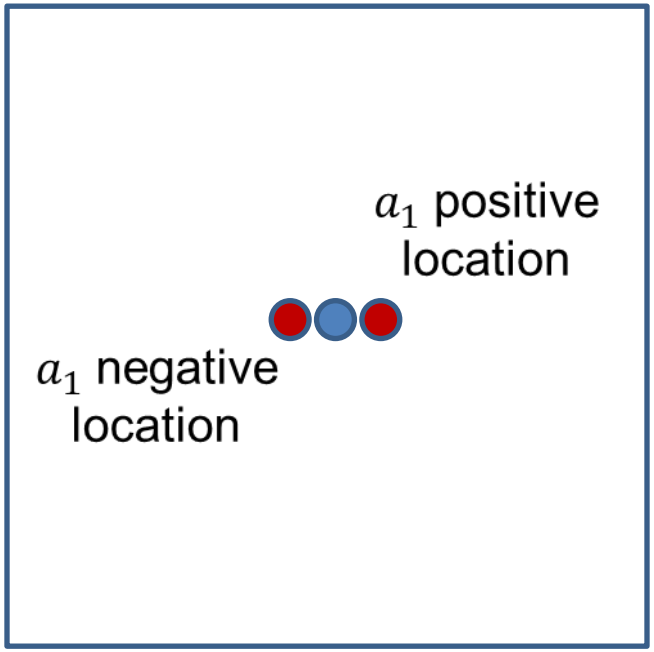
Image is rotationally symmetric about center because of negative frequencies

$$\frac{a_0}{2} + \sum_{n=1}^{N} (a_n \sin(nx + \emptyset_n))$$

e.g., wheel rotating one way or the other

For real-valued signals, positive and negative frequencies are complex conjugates (see additional slides).

Fourier transform component image



A few questions

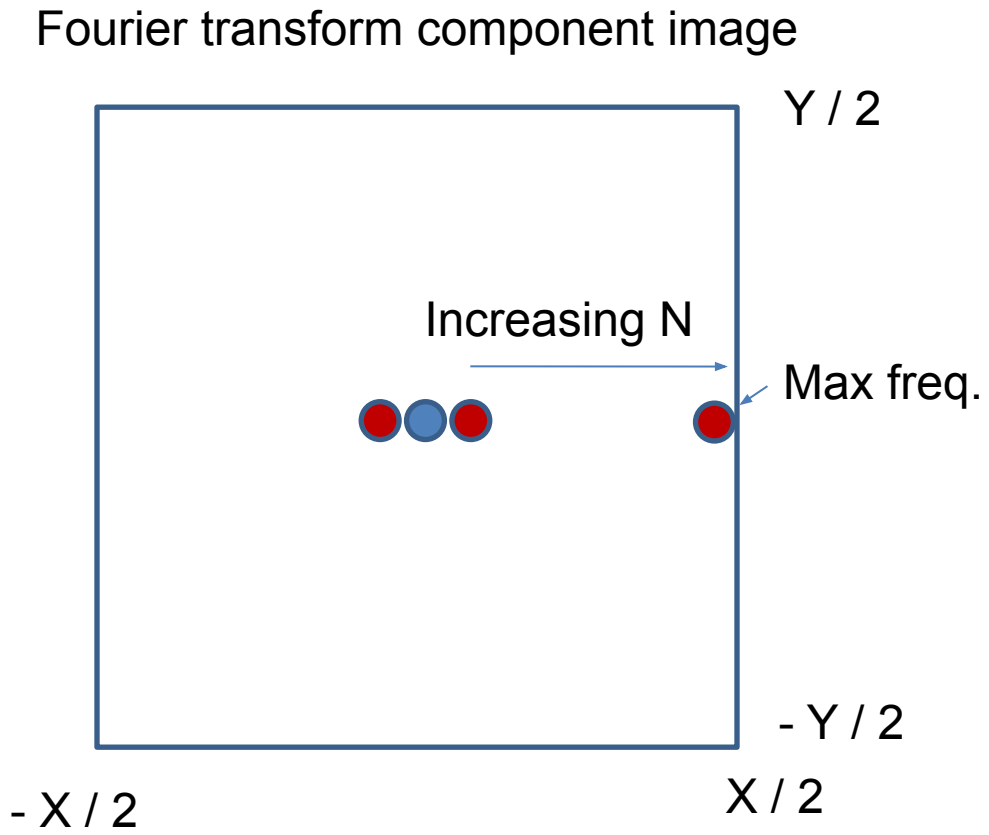
- Why is frequency decomposition centered in middle, and duplicated and rotated?
  - From Euler:
    - $\cos(x) + i\sin(x) = e^{ix}$
    - $\cos(\omega t) = \frac{1}{2}\left(e^{-i\omega t} + e^{+i\omega t}\right)$
  - Coefficients for negative frequencies (i.e., 'backwards traveling' waves)
  - FFT of a real signal is conjugate symmetric
    - i.e., $f(-x) = f^*(x)$

How to read 2D Fourier transform images

Image is as large as maximum frequency by resolution of input under Nyquist frequency

$$\frac{a_0}{2} + \sum_{n=1}^{N} (a_n \sin(nx + \emptyset_n))$$

Nyquist frequency is half the sampling rate of the signal. Sampling rate is size of image (X and Y), so Fourier transform images are (±X/2, ±Y/2).



A few questions…

If we have infinite frequencies, why does the image end?

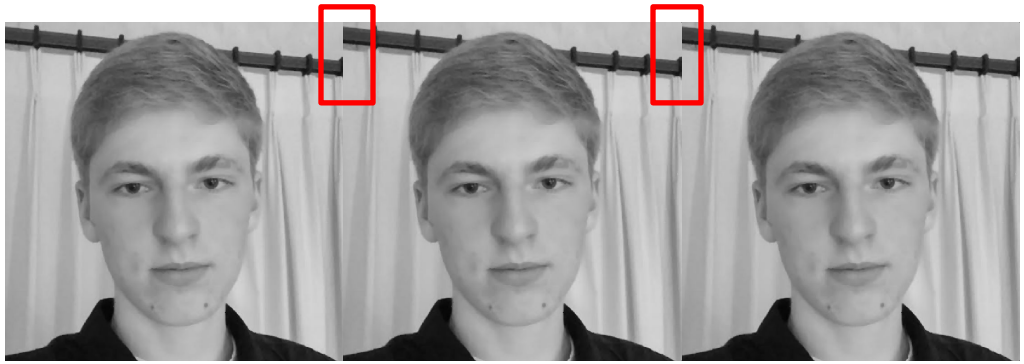
- Sampling theory. Frequencies higher than Nyquist frequency end up falling on an existing sample.
  - i.e., they are ‘aliases’ for existing samples!
- Nyquist frequency is half the sampling frequency.

- (Nyquist frequency is not Nyquist-Shannon rate, which is sampling required to reconstruct alias-free signal. Both are derived from same theory.)

Periodicity

- Fourier decomposition assumes periodic signals with infinite extent.
  - E.G., our image is assumed to repeated forever

- ‘Fake’ signal is image wrapped around



Convention is to pad with zeros to reduce effect.

A few questions

How is the Fourier decomposition computed?

Intuitively, by correlating the signal with a set of waves of increasing frequency!

Computing the Fourier Transform

$$H(\omega) = \mathcal{F}\{h(x)\} = Ae^{j\phi}$$

Continuous

$$H(\omega) = \int_{-\infty}^{\infty} h(x)e^{-j\omega x}dx$$

Discrete

$$H(k) = \frac{1}{N}\sum_{x=0}^{N-1} h(x)e^{-j\frac{2\pi kx}{N}}$$

k = -N/2..N/2

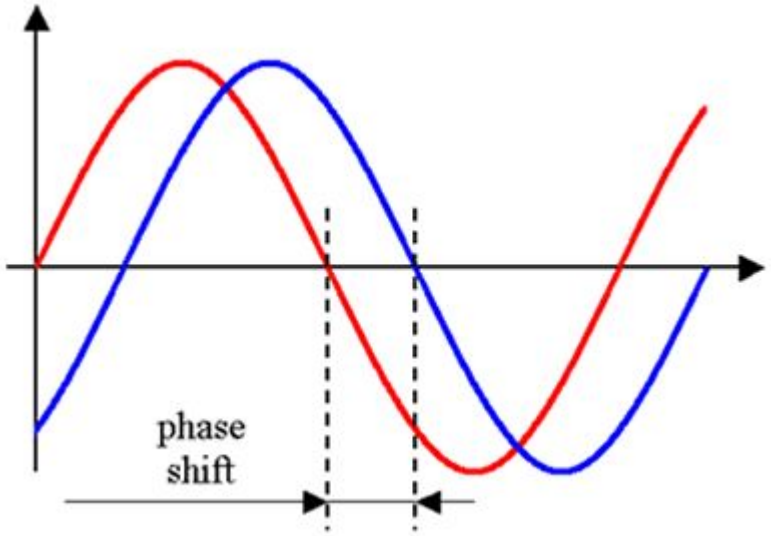
Fast Fourier Transform (FFT): NlogN

Amplitude-phase form

Add phase term to shift
cos values into sin values

$$\sum_{n=1}^{N} (a_n \sin(nx + \emptyset_n))$$

Phase

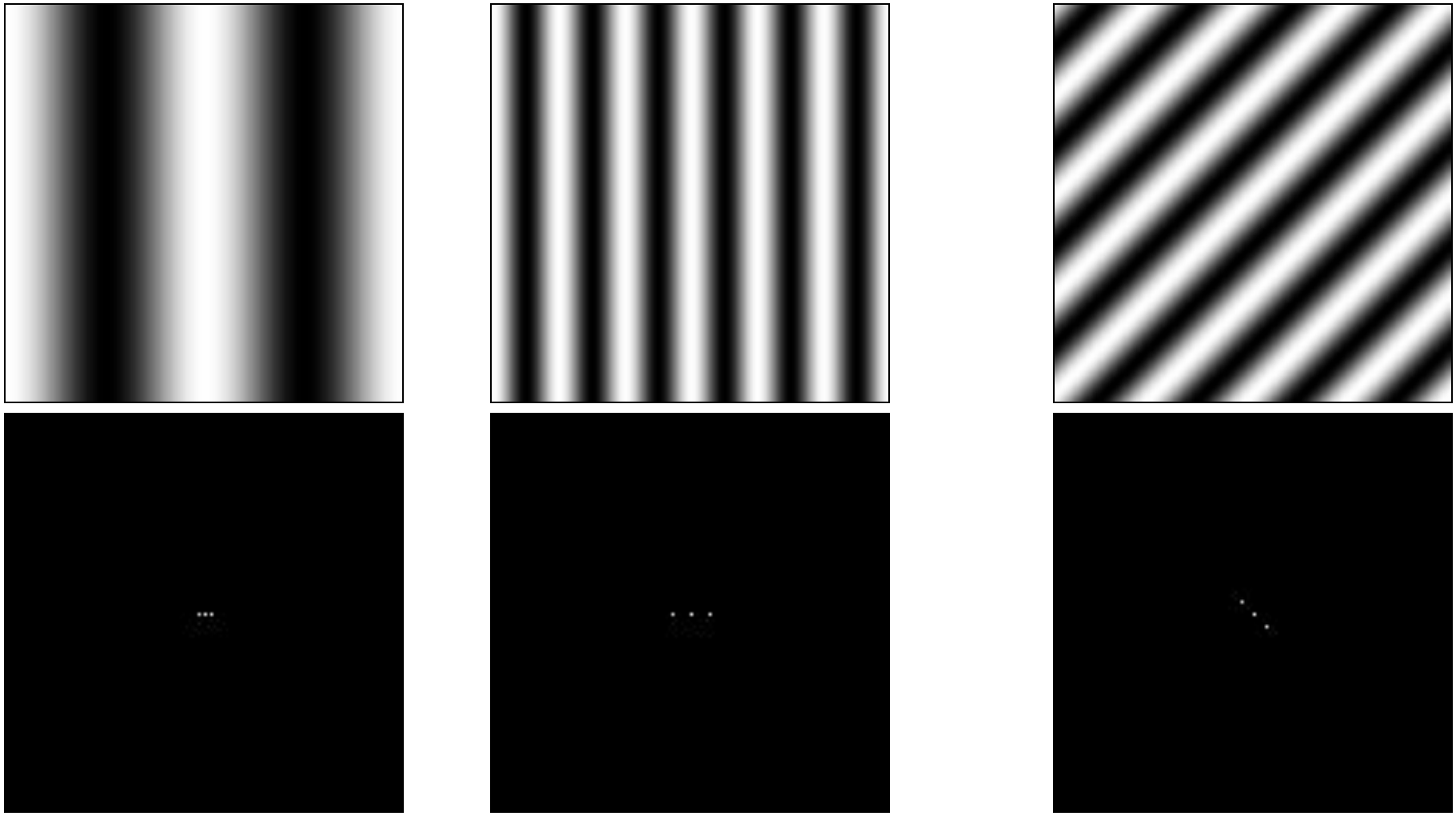


Morse

[Peppergrower; Wikipedia]

Fourier analysis in images

Spatial domain images

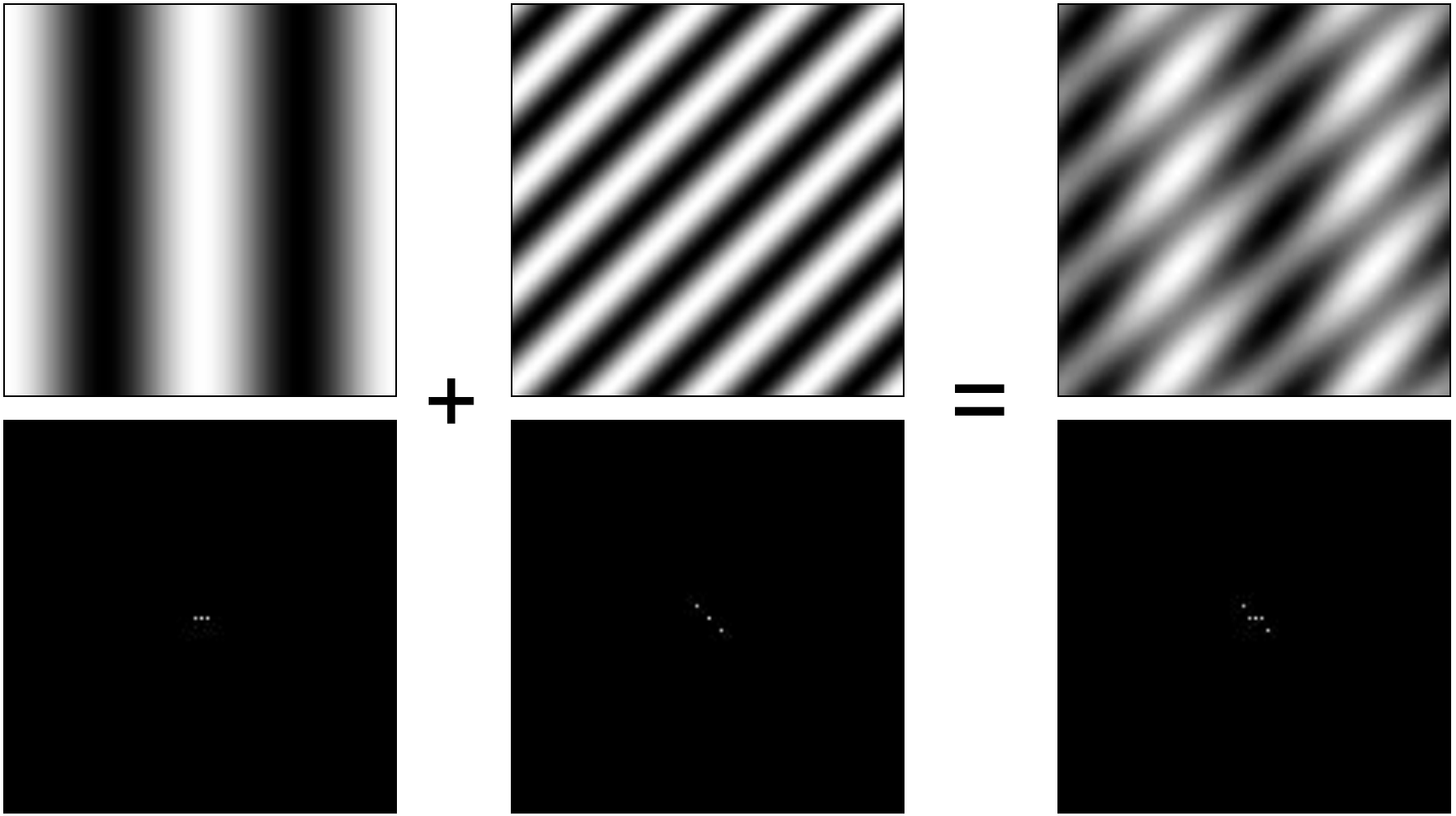


Fourier decomposition amplitude images

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

Signals can be composed

Spatial domain images



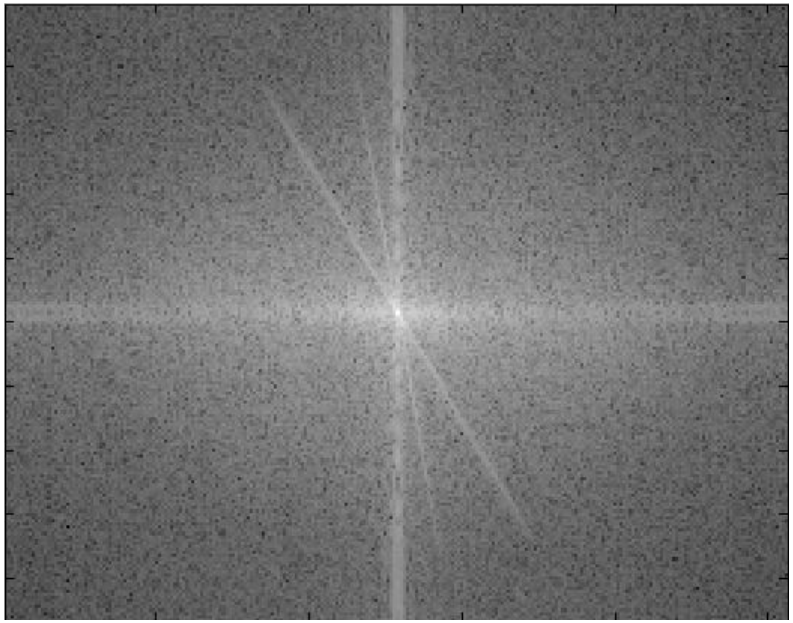
Fourier decomposition amplitude images

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

Natural image

Natural image

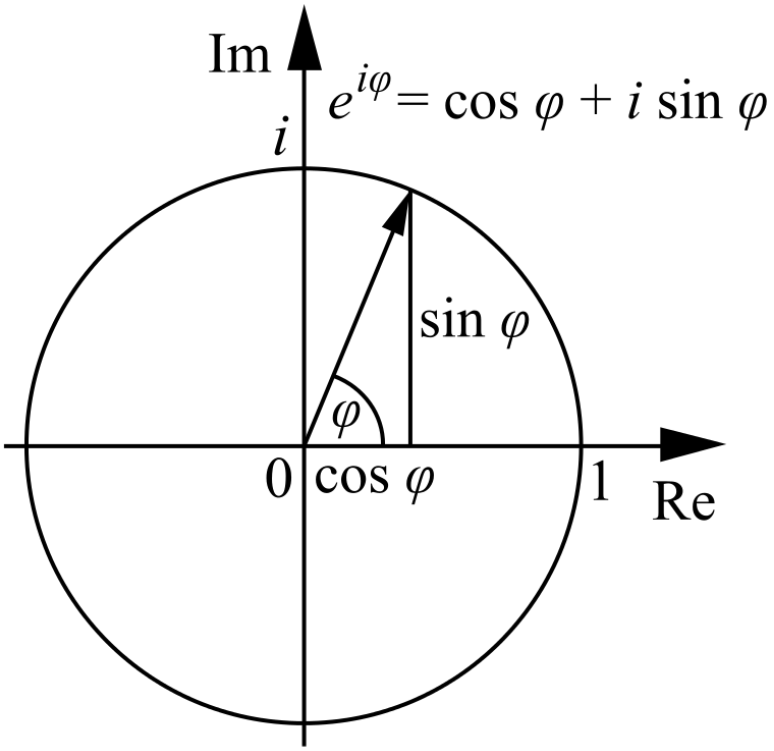
Fourier decomposition
Amplitude image



What does it mean to be more or less bright in the Fourier decomposition image?
What about edges?

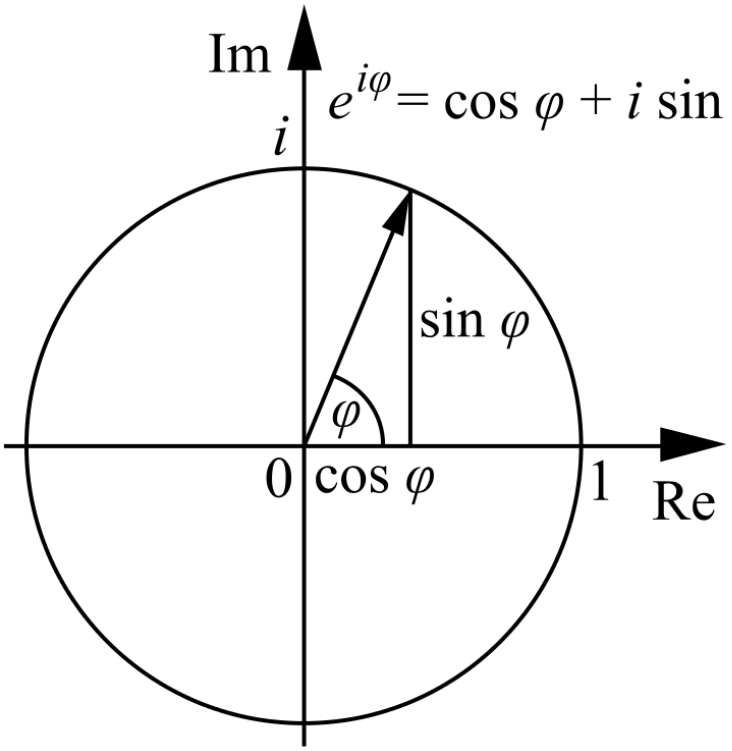
Euler's Formula

- Stores the amplitude and phase at each frequency:
  - For mathematical convenience, this is often notated in terms of real and complex numbers
  - Related by Euler's formula



Euler's Formula – Amplitude and Phase

- Stores the amplitude and phase at each frequency:
  - For mathematical convenience, this is often notated in terms of real and complex numbers
  - Related by Euler's formula



Amplitude encodes how much signal there is at a particular frequency:

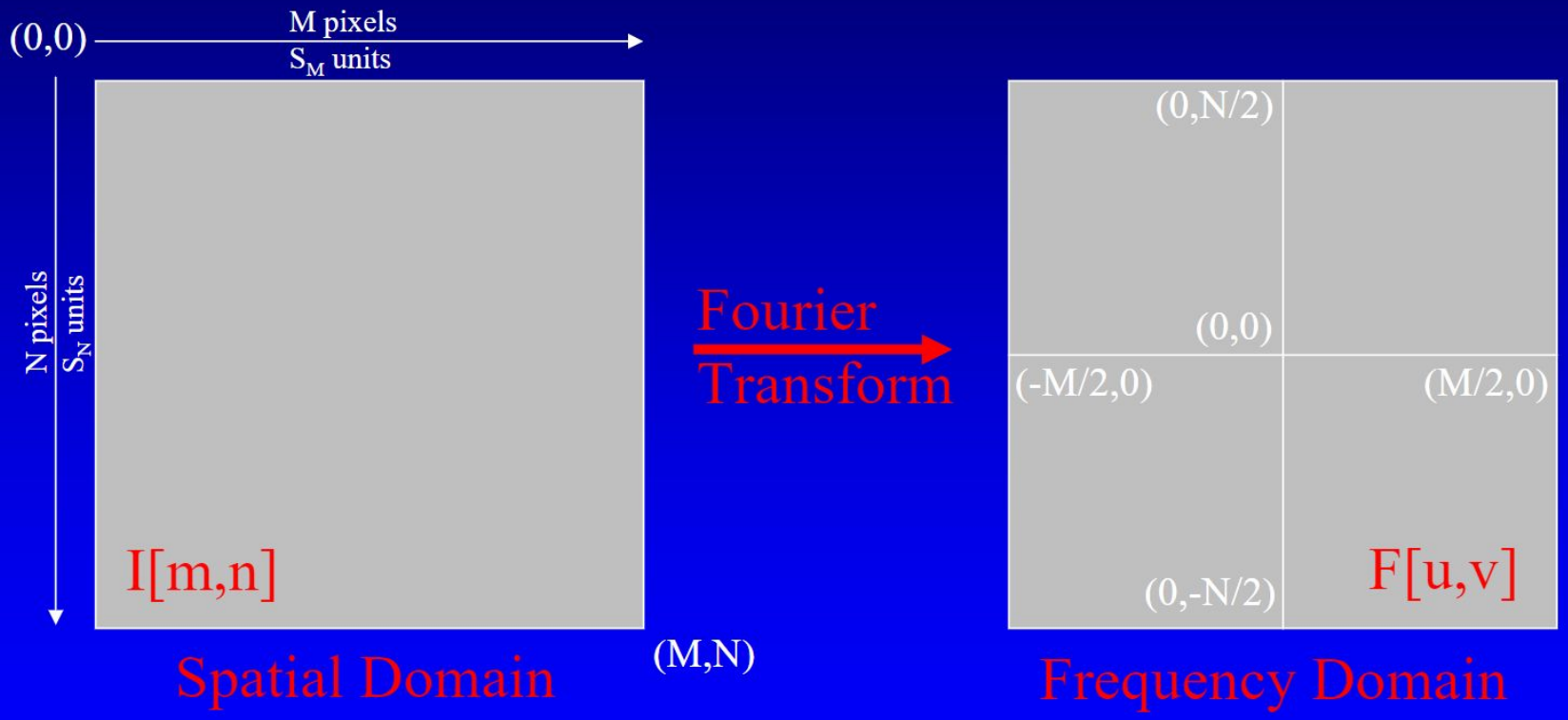
$$A = \pm\sqrt{\mathrm{Re}(\varphi)^2 + \mathrm{Im}(\varphi)^2}$$

Phase encodes spatial information (indirectly):

$$\phi = \tan^{-1}\frac{\mathrm{Im}(\varphi)}{\mathrm{Re}(\varphi)}$$

2D Discrete Fourier Transform

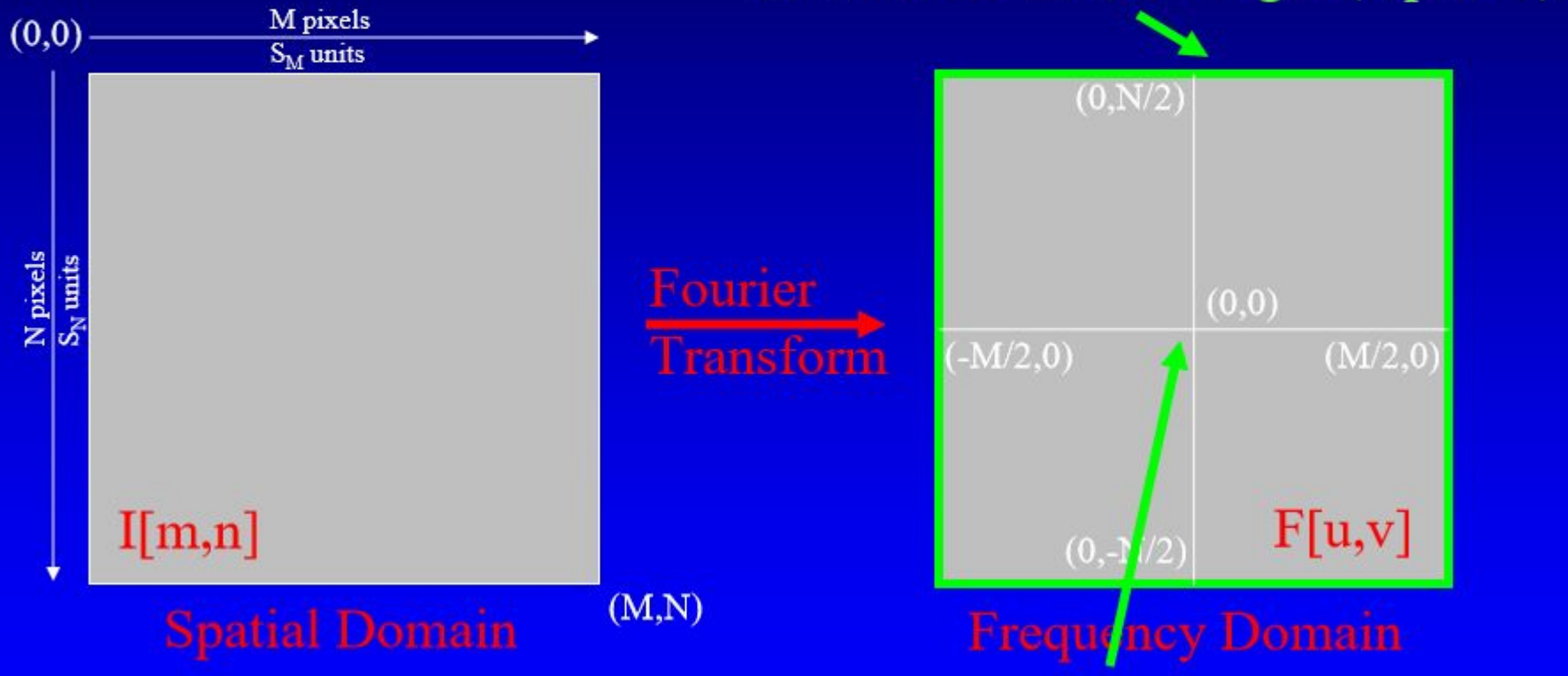
$$F[u,v] = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I[m,n] \cdot e^{-i2\pi \left(\frac{um}{M} + \frac{vn}{N} \right)}$$



Source: Seul et al, *Practical Algorithms for Image Analysis*, 2000, p. 249, 262.

2D FFT can be computed as two discrete Fourier transforms in 1 dimension

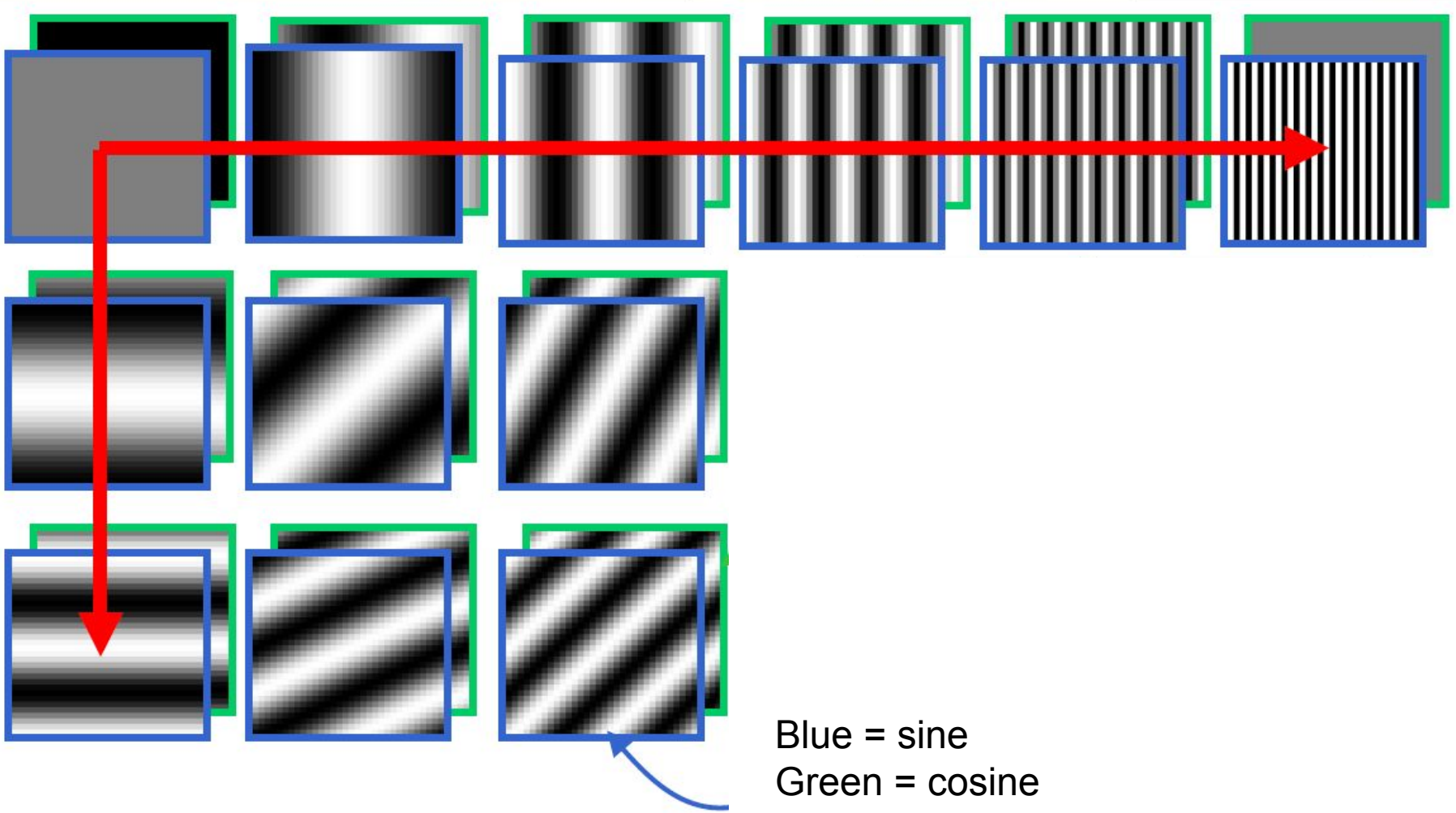
2D Discrete Fourier Transform



Earl F. Glynn

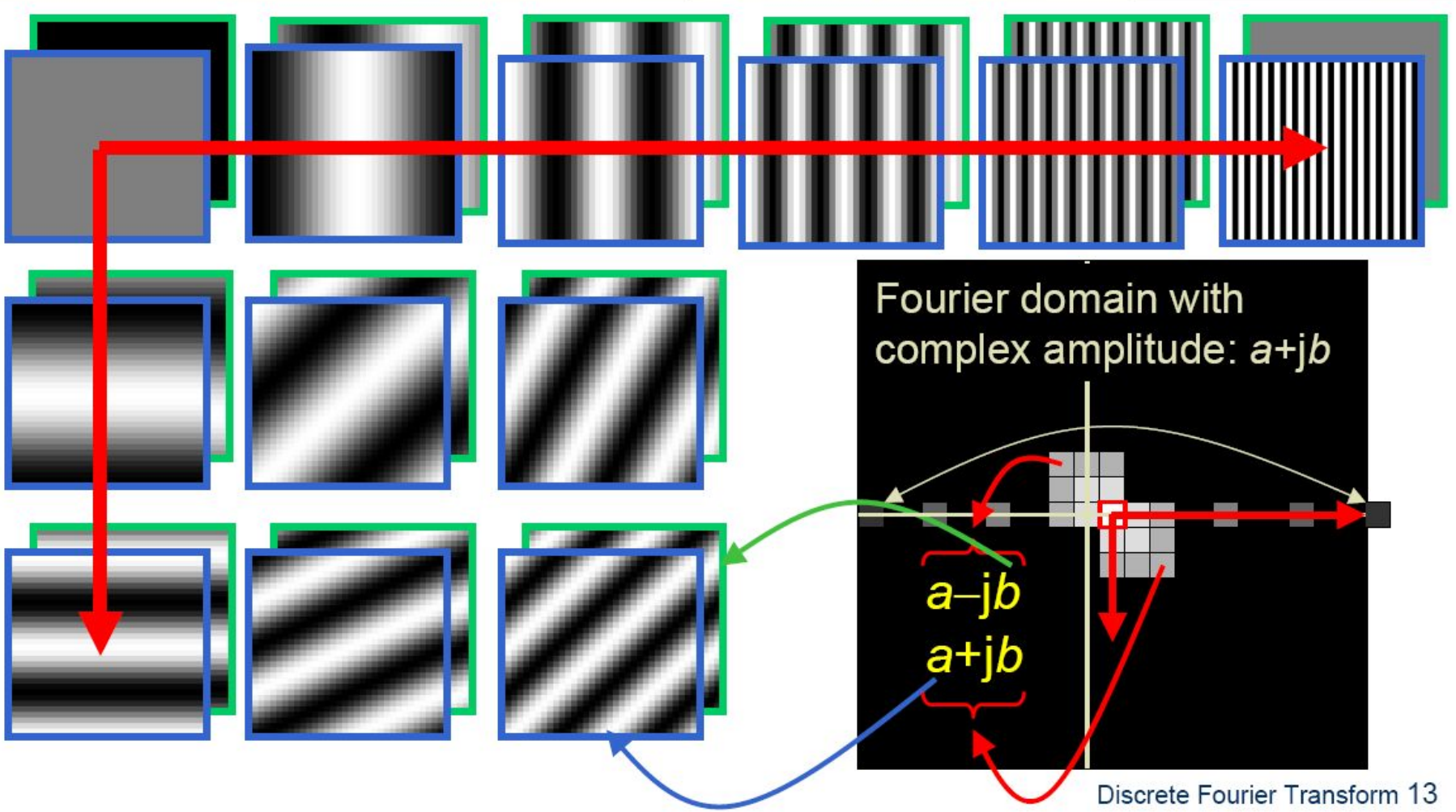
Fourier Bases

Teases away ‘fast vs. slow’ changes in the image.



This change of basis is the Fourier Transform

Fourier Bases

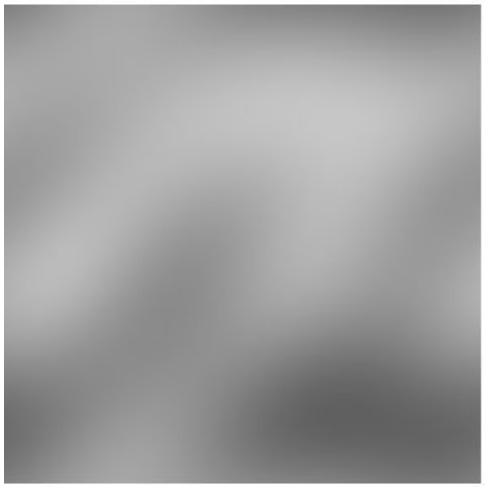


Basis reconstruction

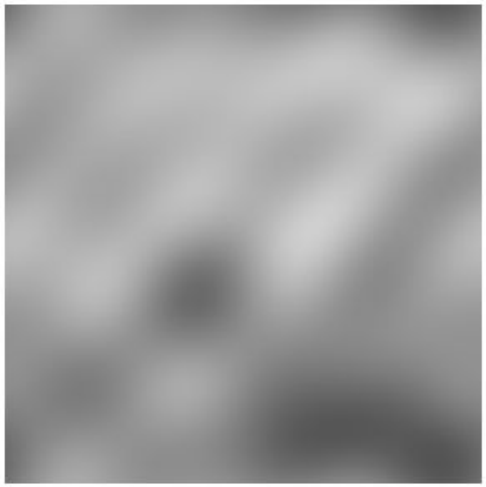
Full image

First 1 basis fn

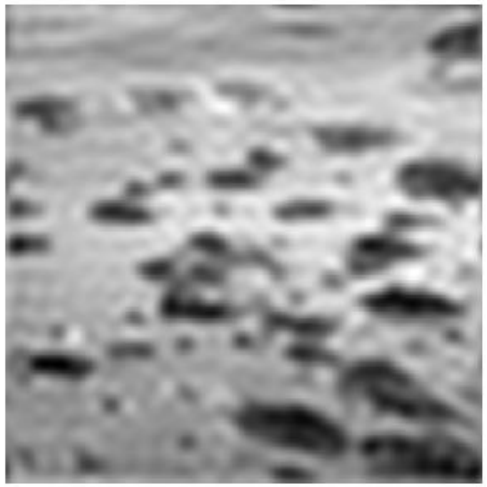
First 4 basis fns



First 9 basis fns



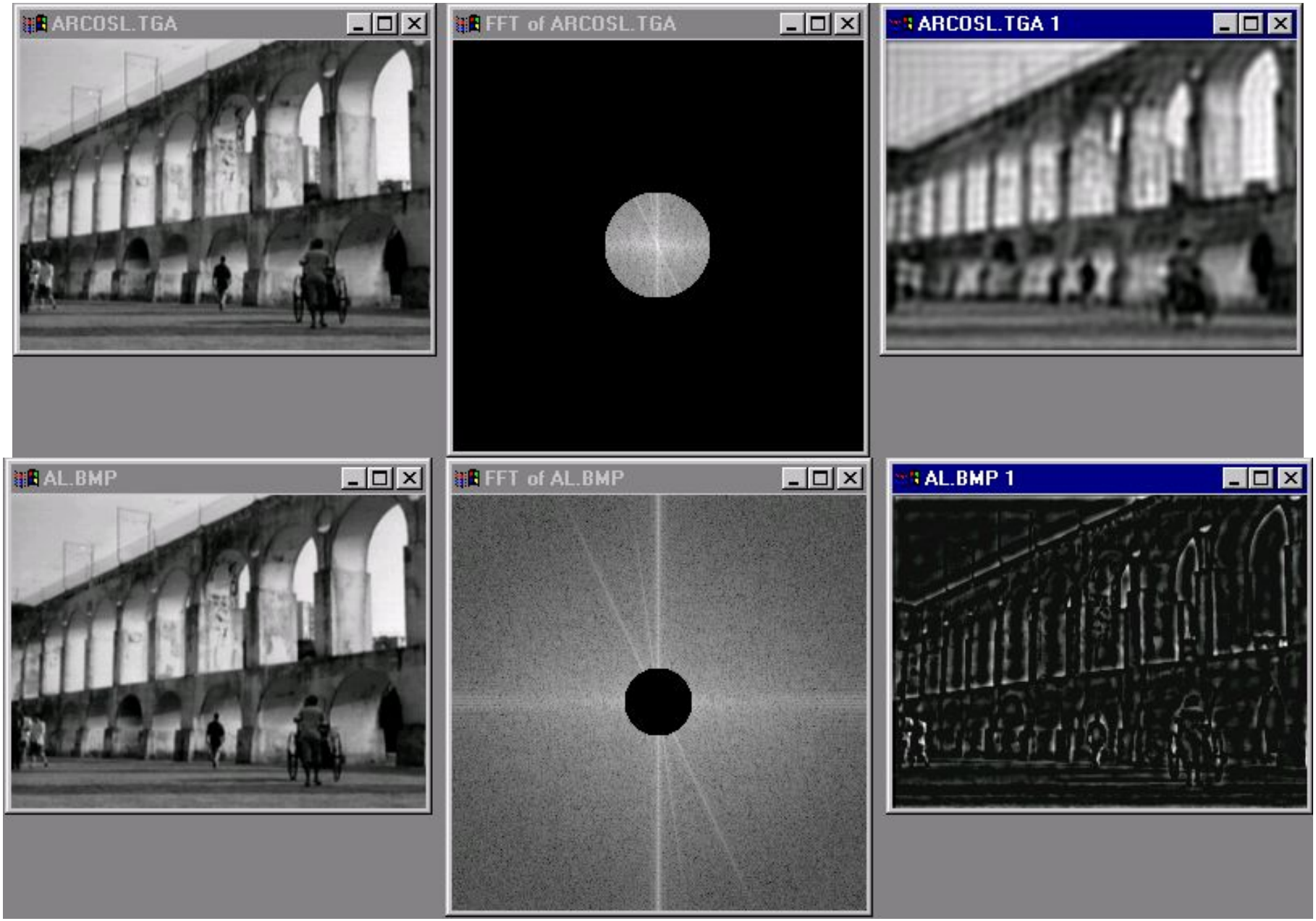
First 16 basis fns



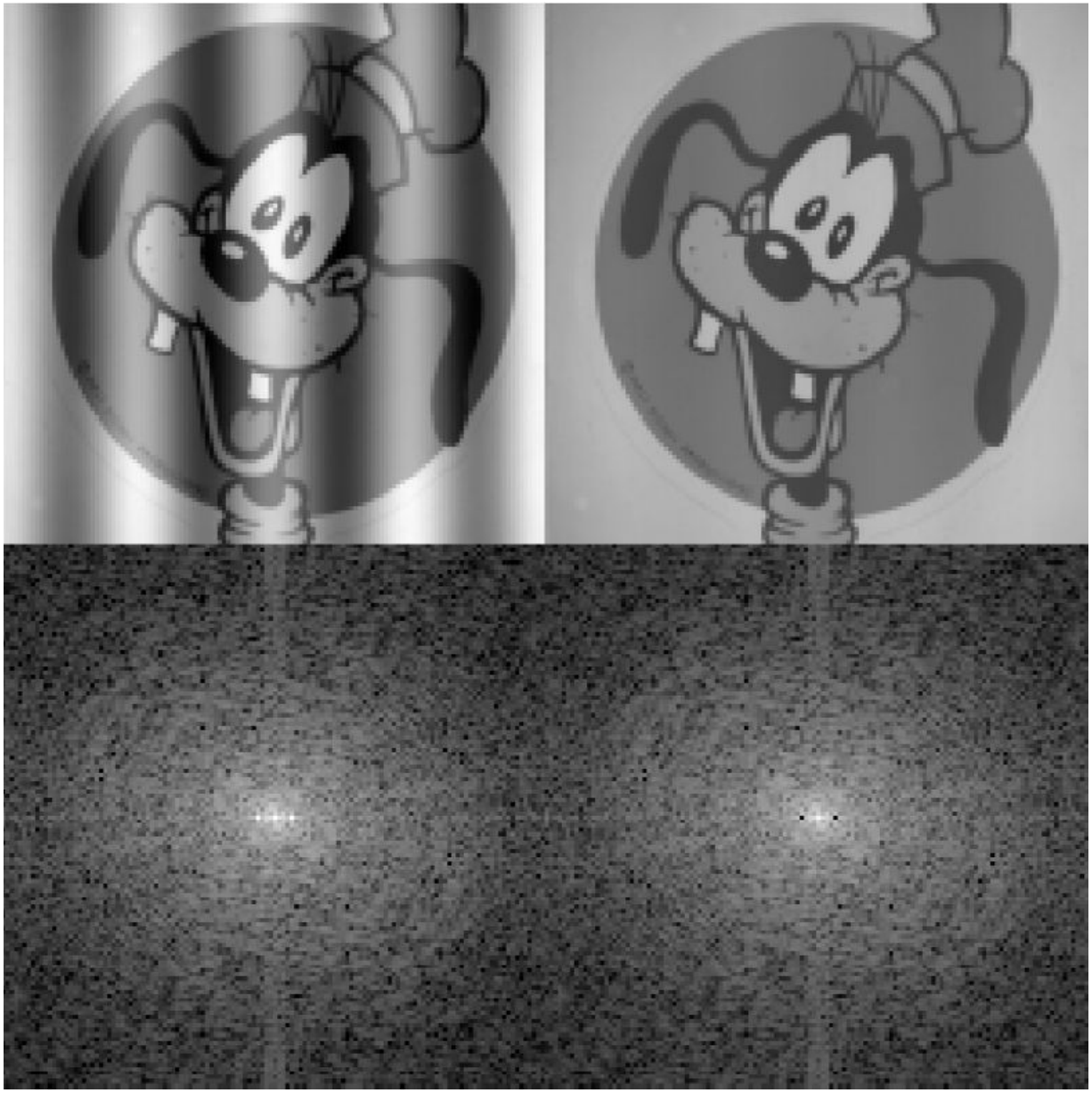
First 400 basis fns

Now we can edit frequencies!

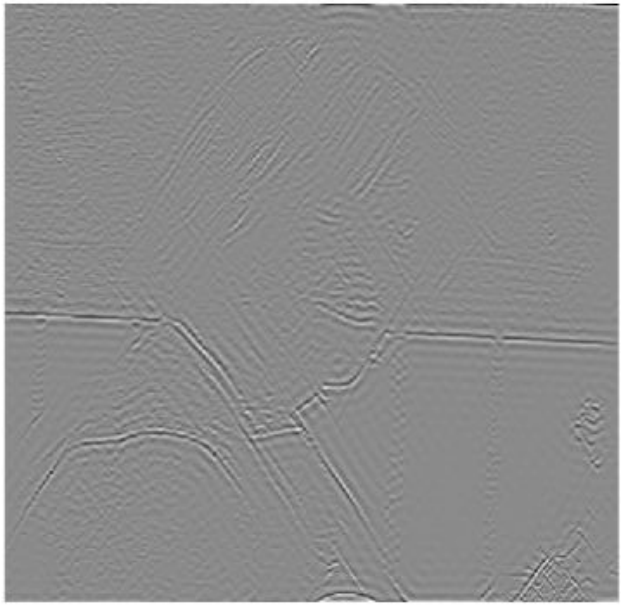
Low and High Pass filtering

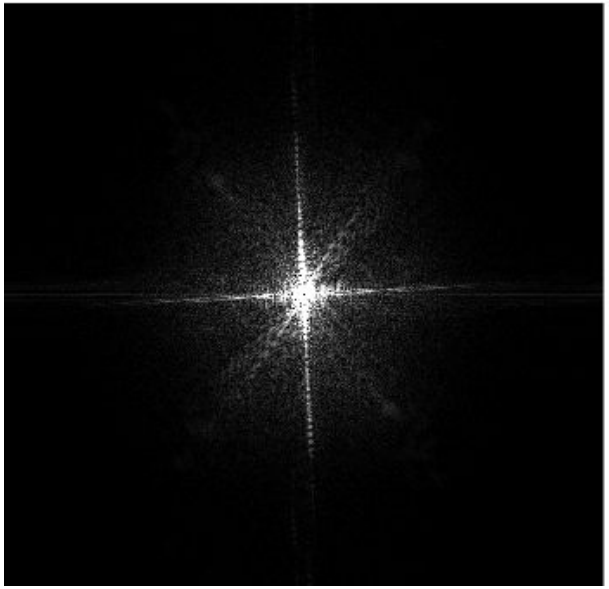


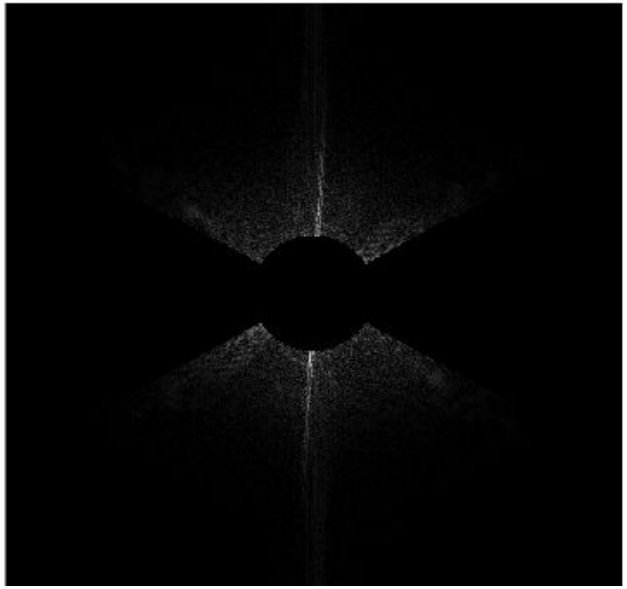
Removing frequency bands



High pass filtering + orientation



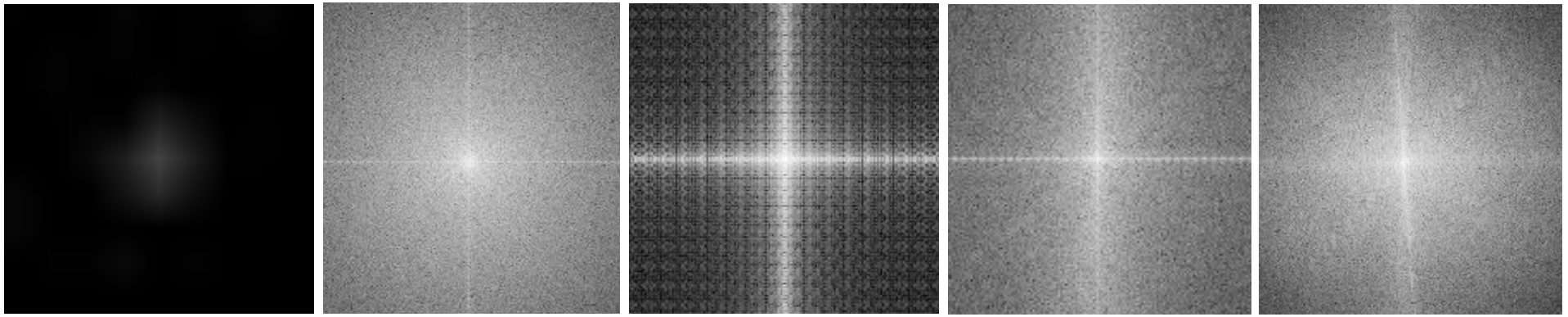




Think-Pair-Share

Match the spatial domain image to the Fourier amplitude image

1 2 3 4 5



A B C D E

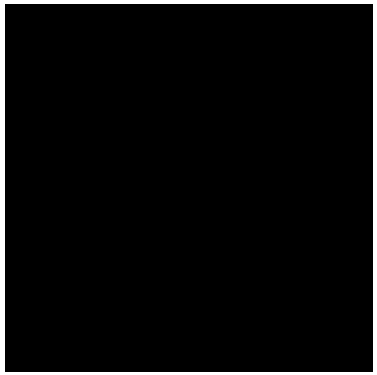
Brian Pauw demo

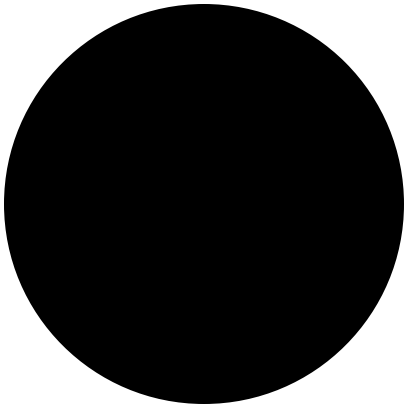
- Let’s look at some images and edit images.

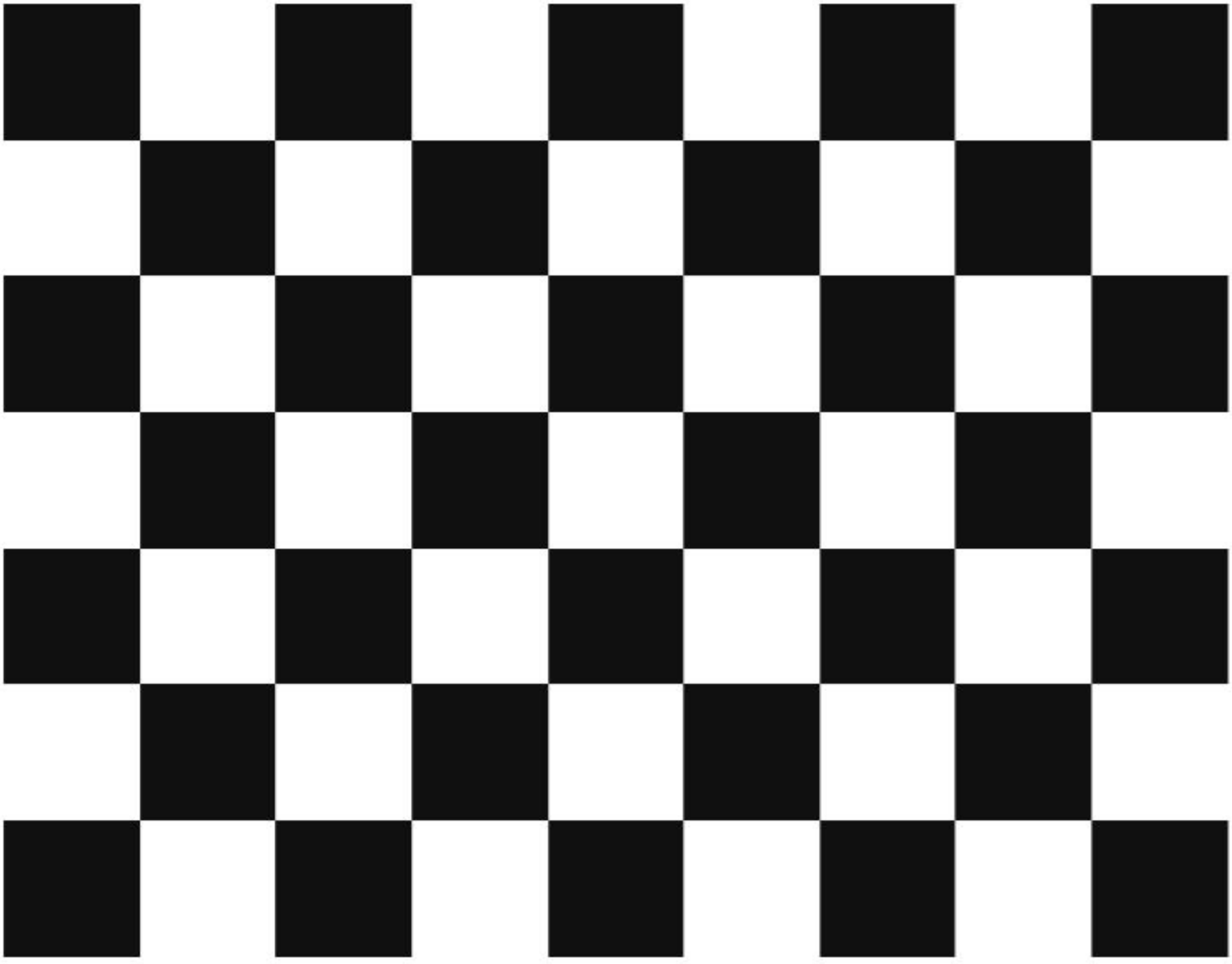
- Live Fourier decomposition images
  - Using FFT2 function

- http://www.lookingatnothing.com/index.php/archives/991

DEMO



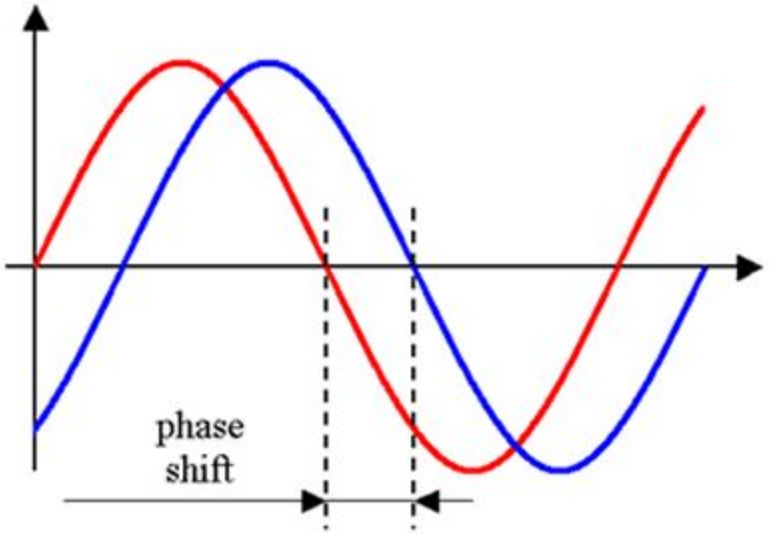




Fourier Transform

- Stores the amplitude and phase at each frequency:
  - For mathematical convenience, this is often notated in terms of real and complex numbers
  - Related by Euler's formula

Amplitude / Phase



- Amplitude tells you ¨how much”
- Phase tells you “where”

- Translate the image?
  - Amplitude unchanged
  - Adds a constant to the phase.

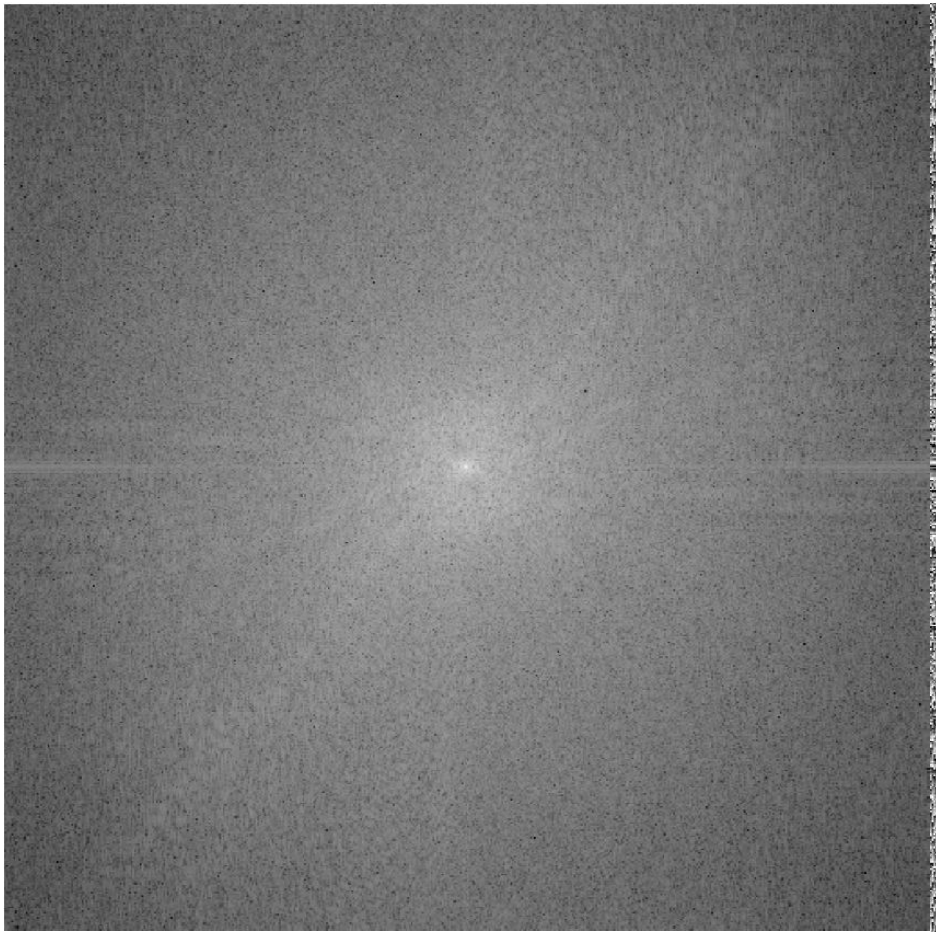
Morse

[Peppergrower; Wikipedia]

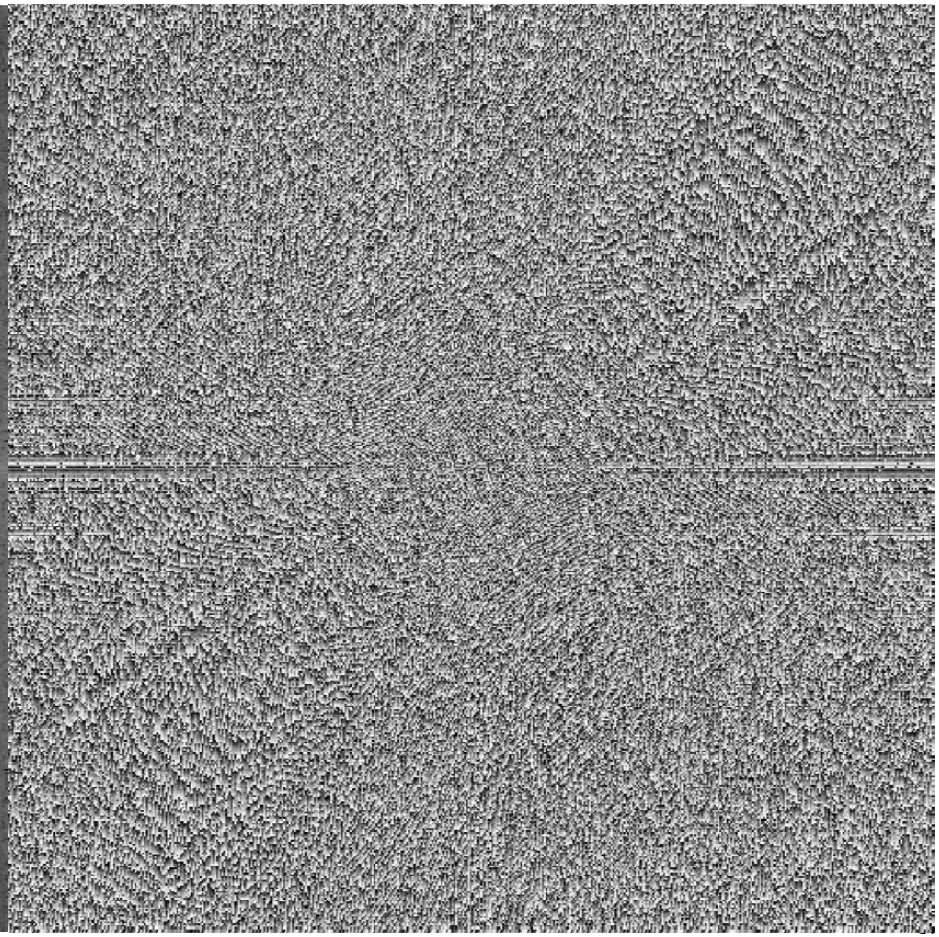
What about phase?

What about phase?

Amplitude



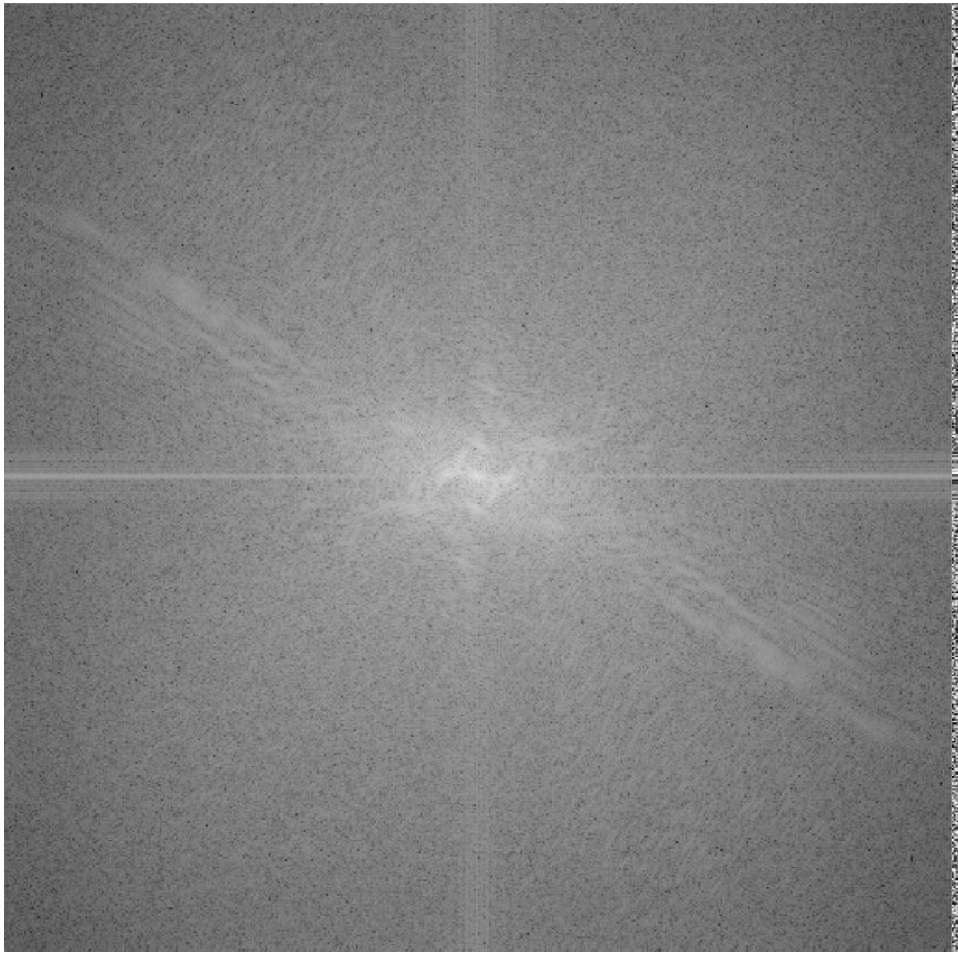
Phase



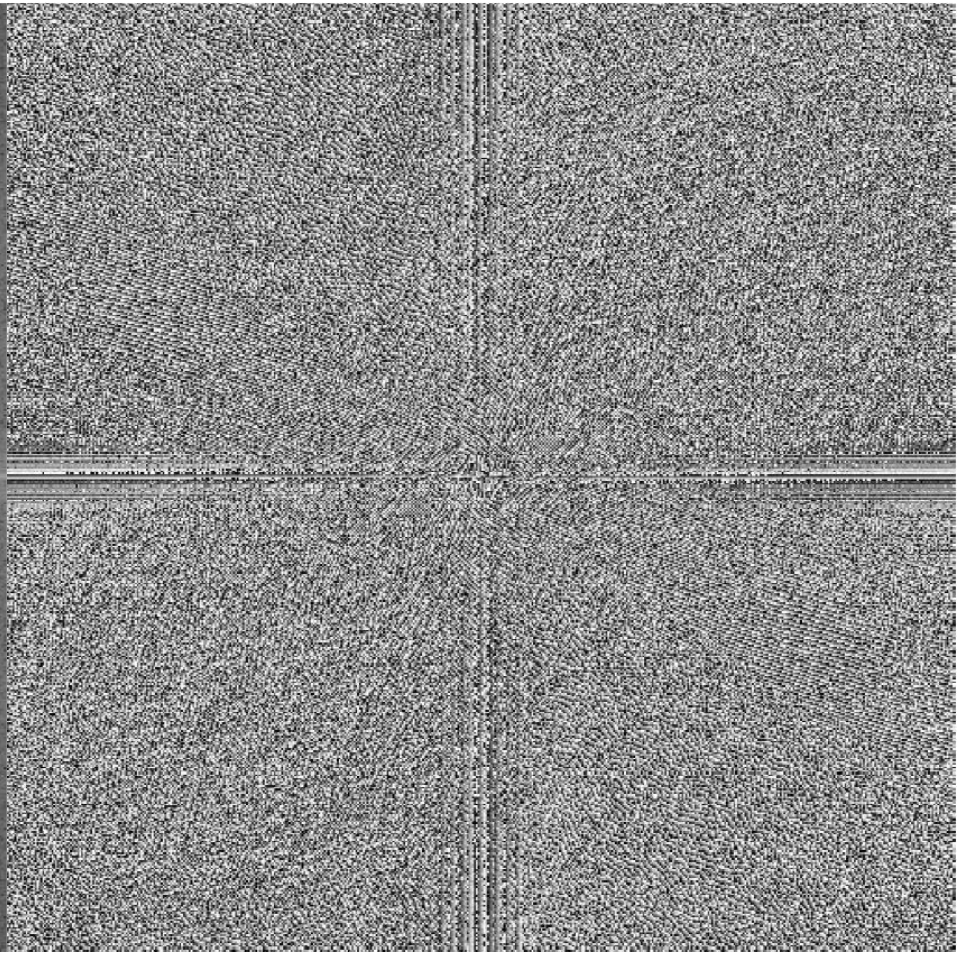
What about phase?

What about phase?

Amplitude



Phase



John Brayer, Uni. New Mexico

- “We generally do not display PHASE images because most people who see them shortly thereafter succumb to hallucinogenics or end up in a Tibetan monastery.”

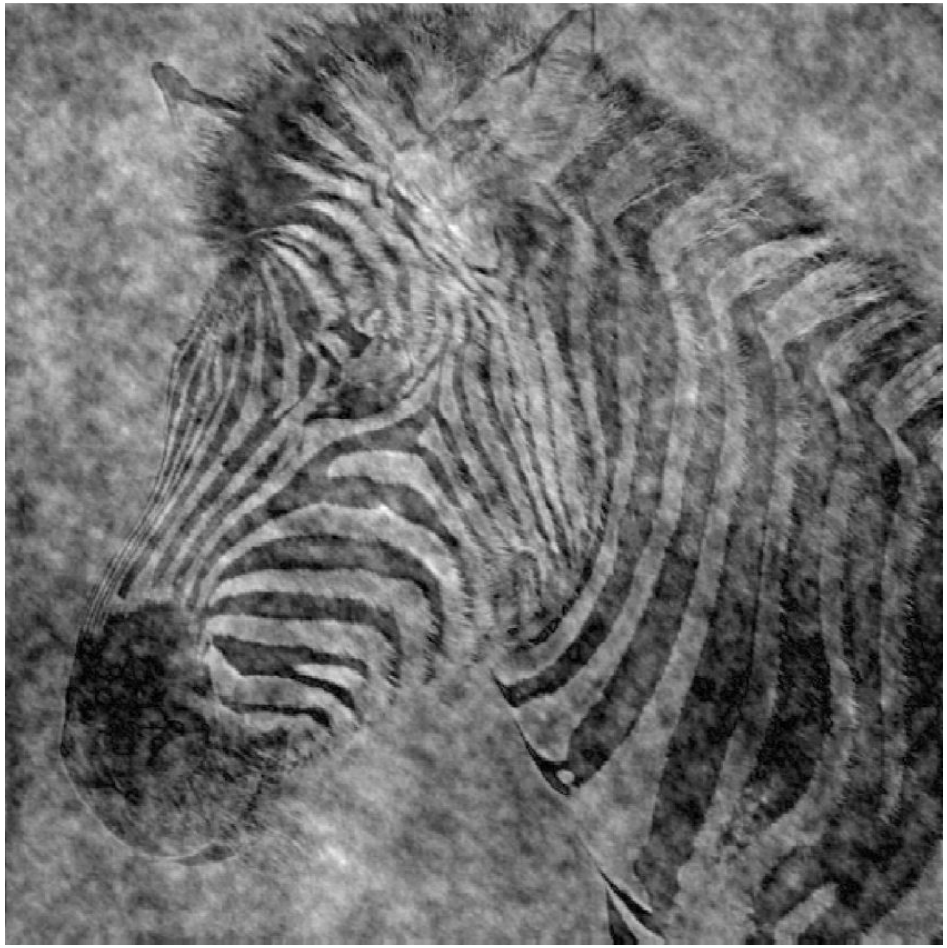
- https://www.cs.unm.edu/~brayer/vision/fourier.html

Think-Pair-Share

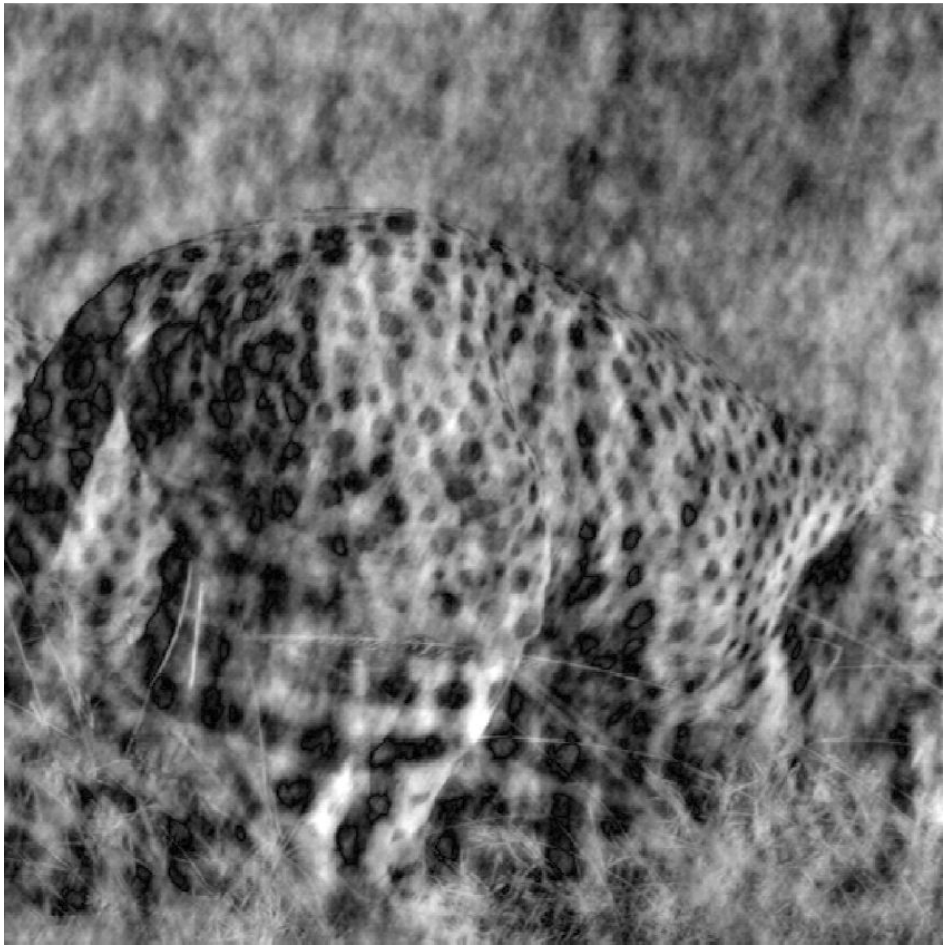
- In Fourier space, where is more of the information that we see in the visual world?
  - Amplitude
  - Phase

Cheebra

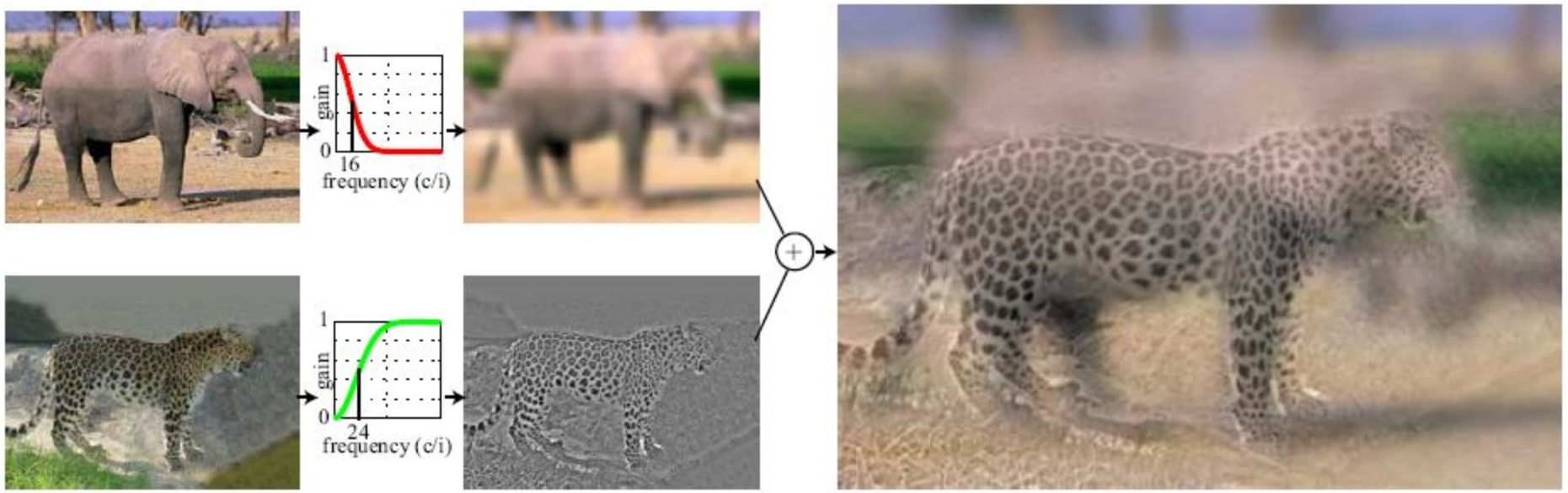
Zebra phase, cheetah amplitude



Cheetah phase, zebra amplitude

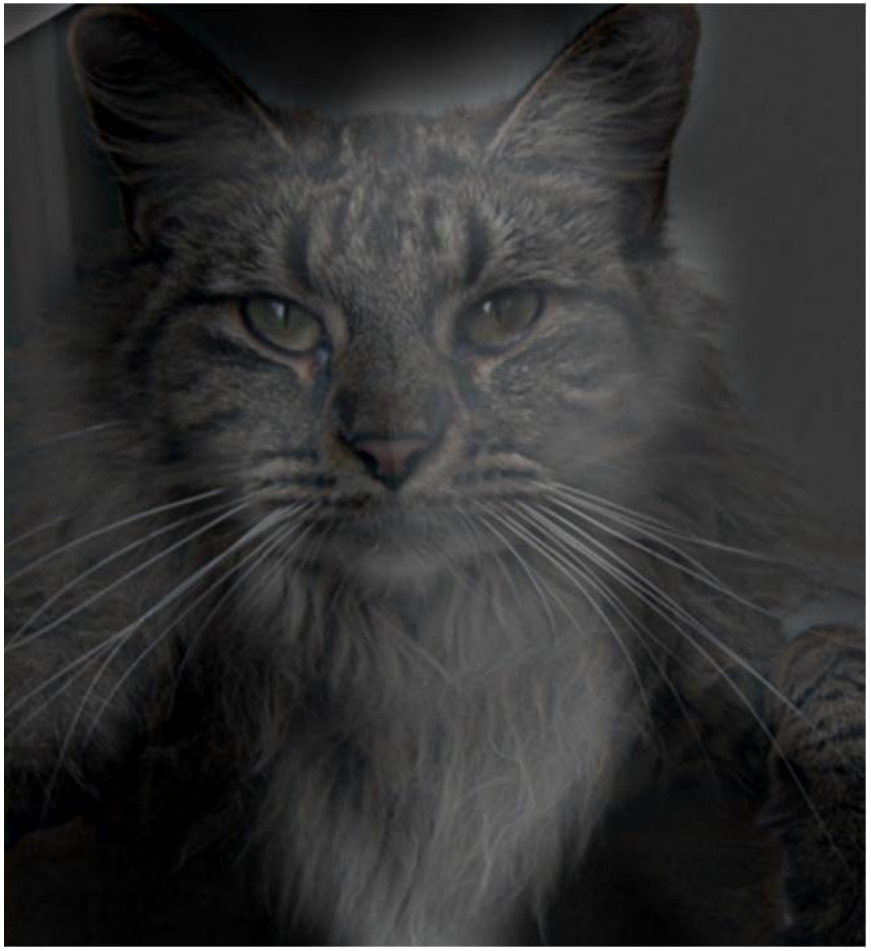


Hybrid Images



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

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

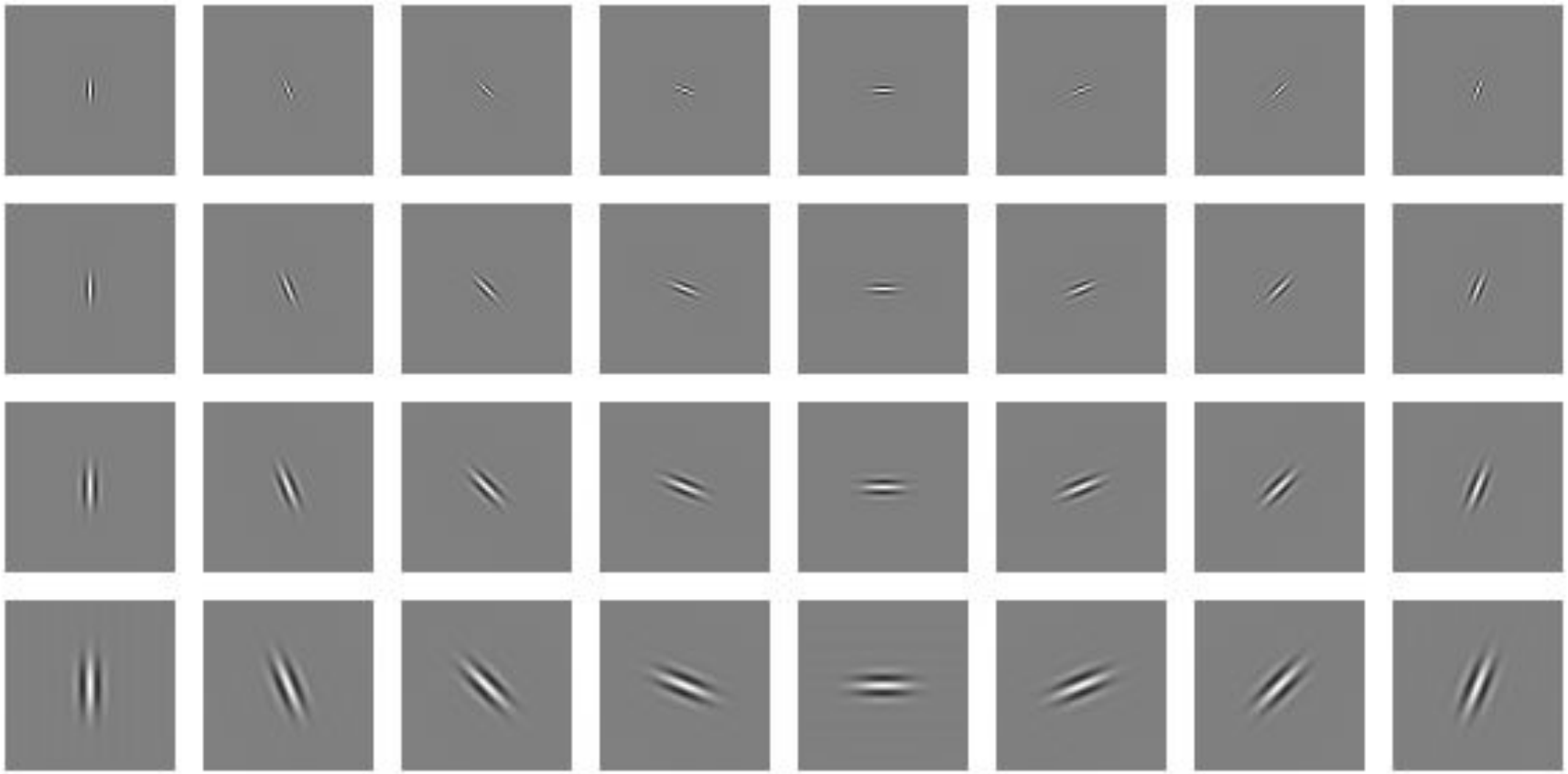


?

Slide: Hoiem

Clues from Human Perception

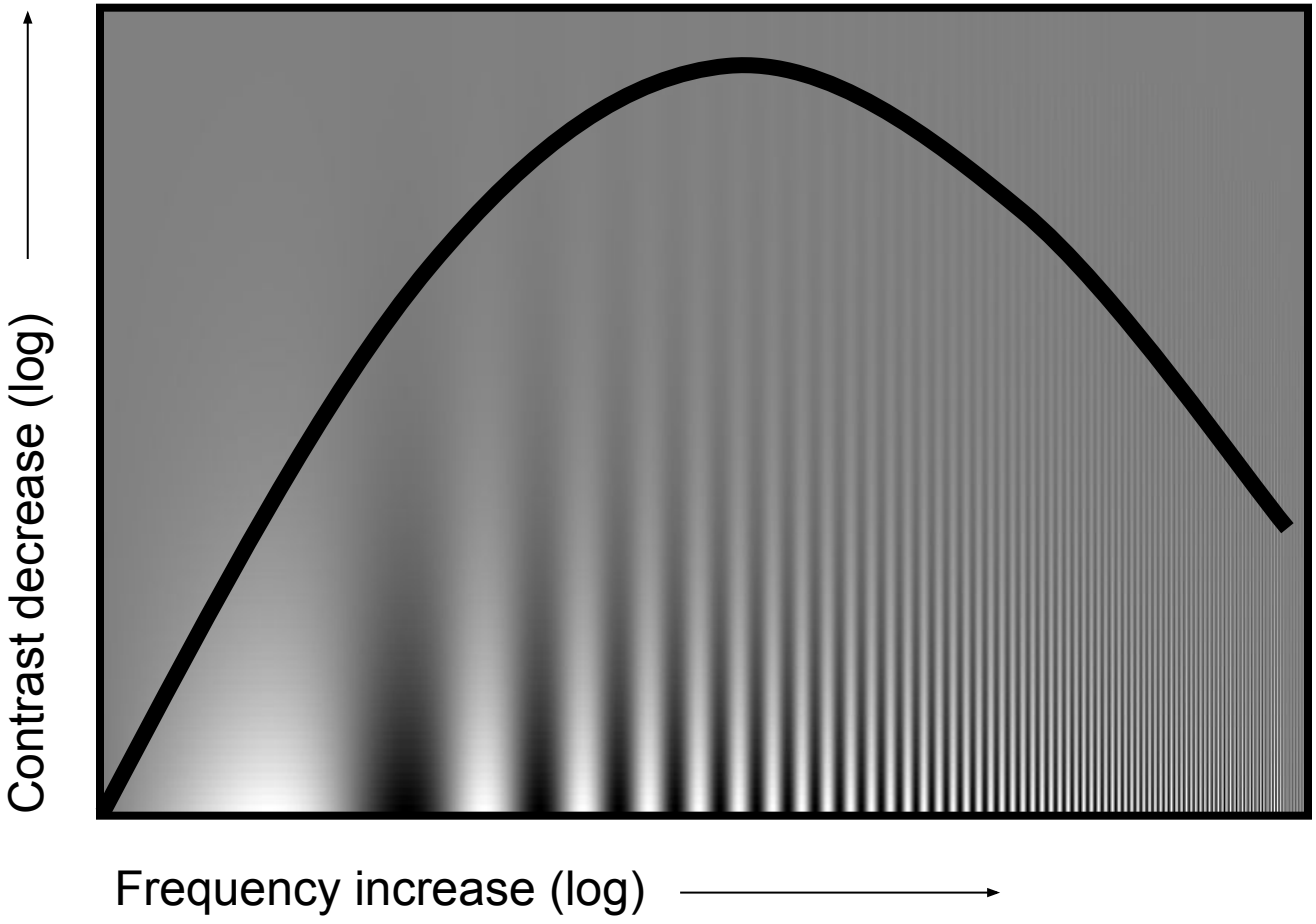
Early visual processing in human perception
filters for orientations and scales of frequency.



Related to *Gabor filters:* sinusoids convolved with Gaussians

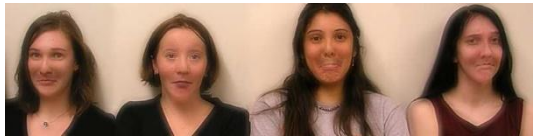
Campbell-Robson contrast sensitivity curve

Perceptual cues in the mid-high frequencies dominate perception.



Application: Hybrid Images

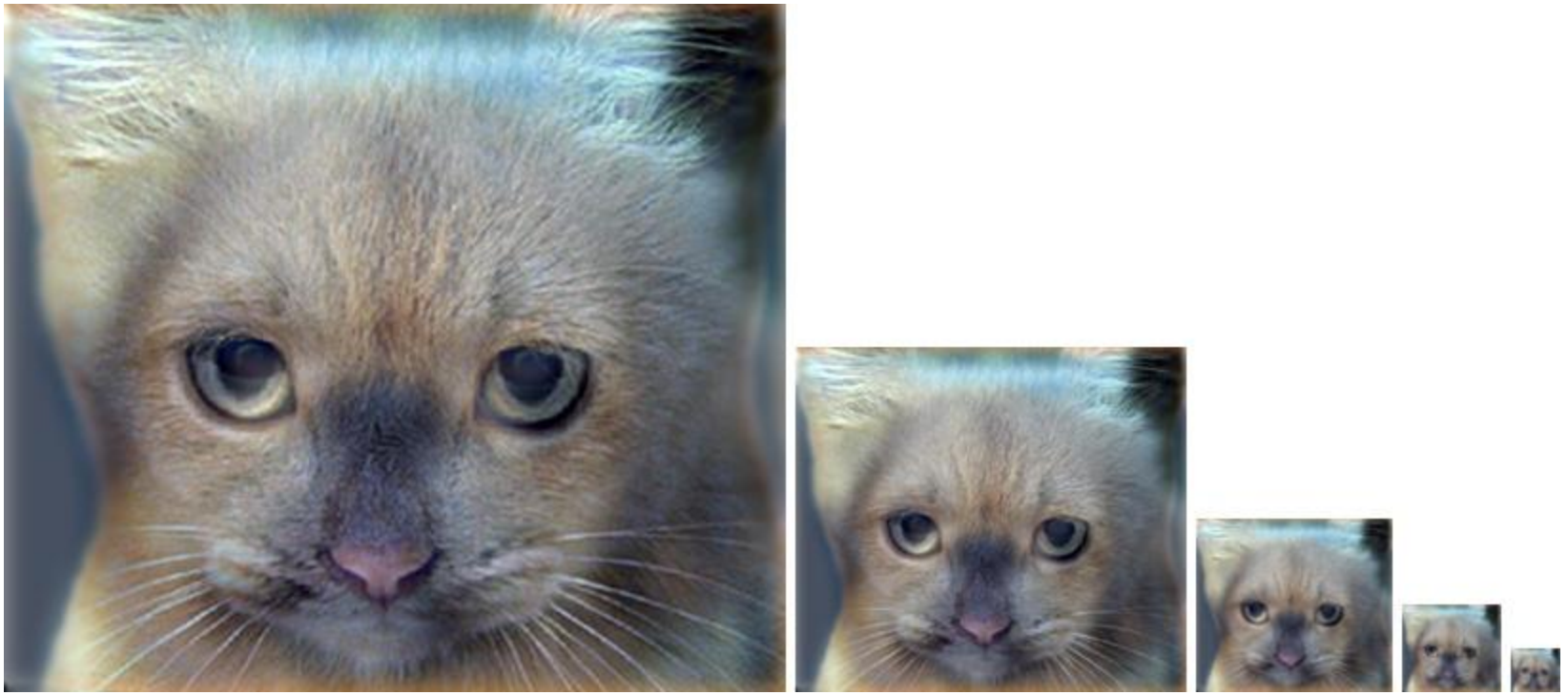
When we see an image from far away, we are effectively subsampling it!



A. Oliva, A. Torralba, P.G. Schyns, SIGGRAPH 2006

HW 1: Image Filtering and Hybrid Images

- Implement image filtering to separate high and low frequencies.
- Combine high frequencies and low frequencies from different images to create a scale-dependent image.

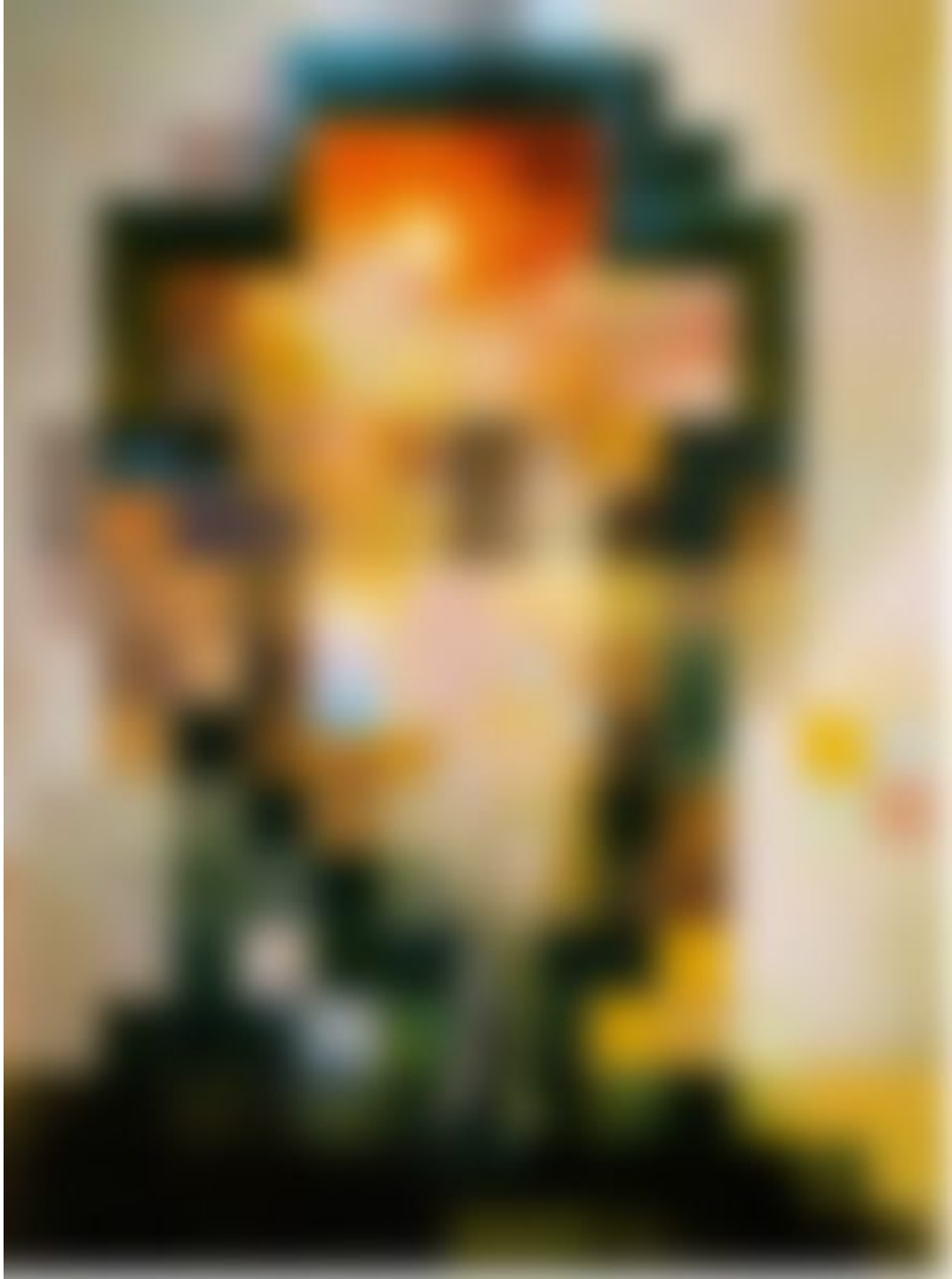


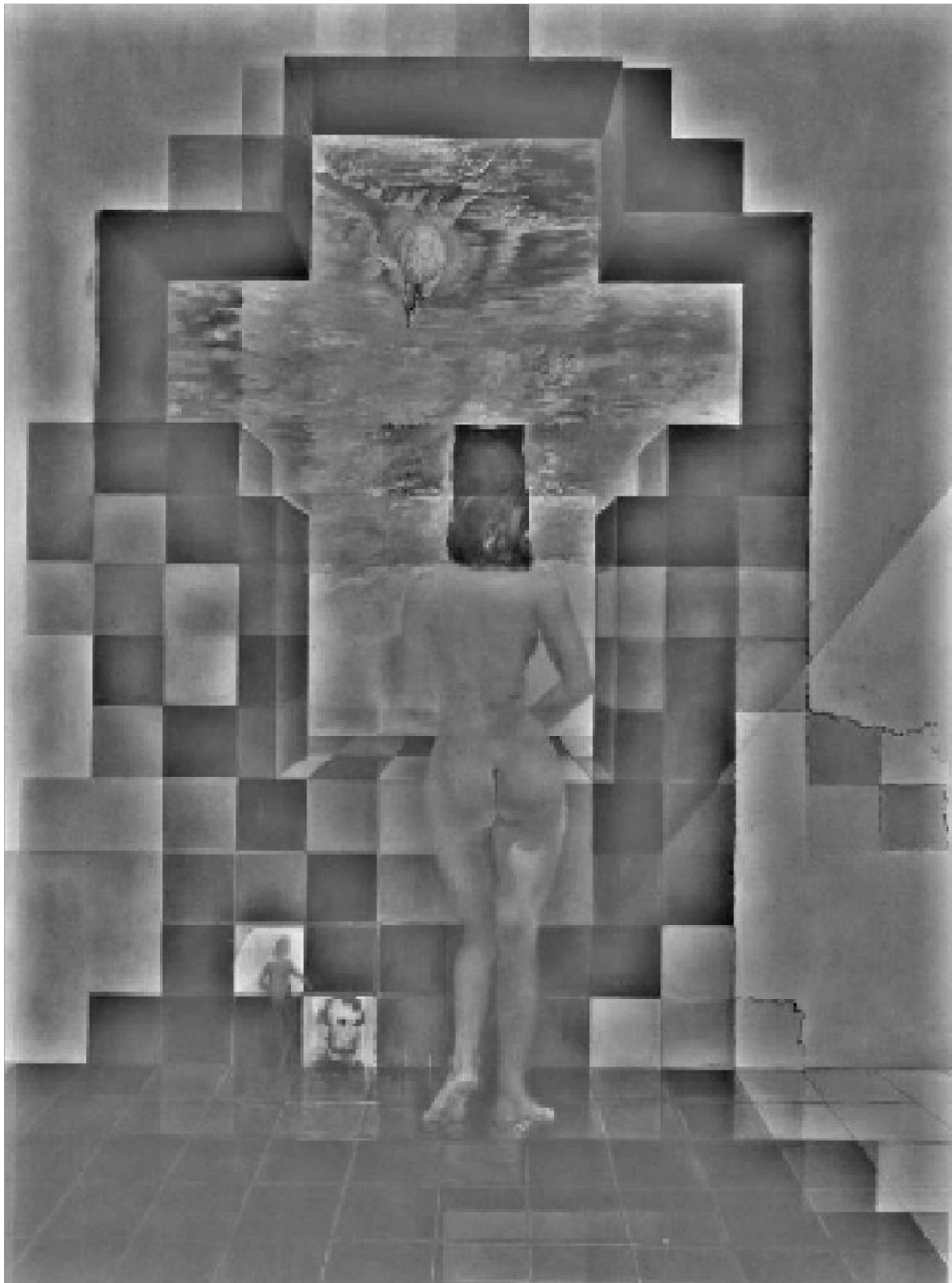
JH

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

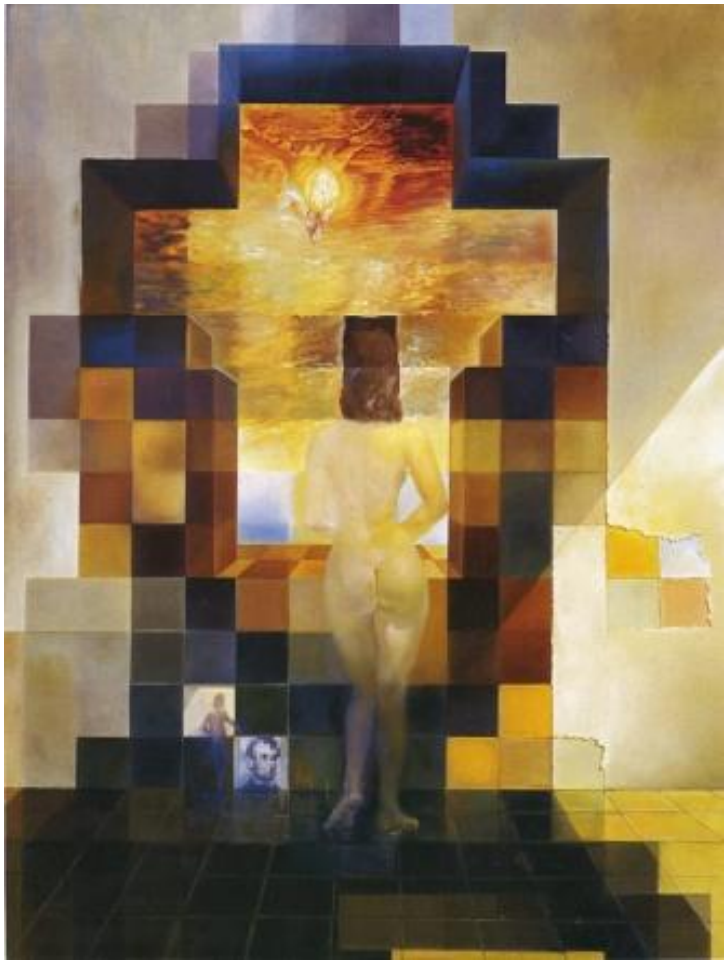
Salvador Dali invented Hybrid Images?

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





Next class: Continue to Think in Frequency



Next Class: Edge Detection

The frequency amplitude of natural images are quite similar.

- Heavy in low frequencies, falling off in high frequencies
- Will *any* image be like that, or is it a property of the world we live in?

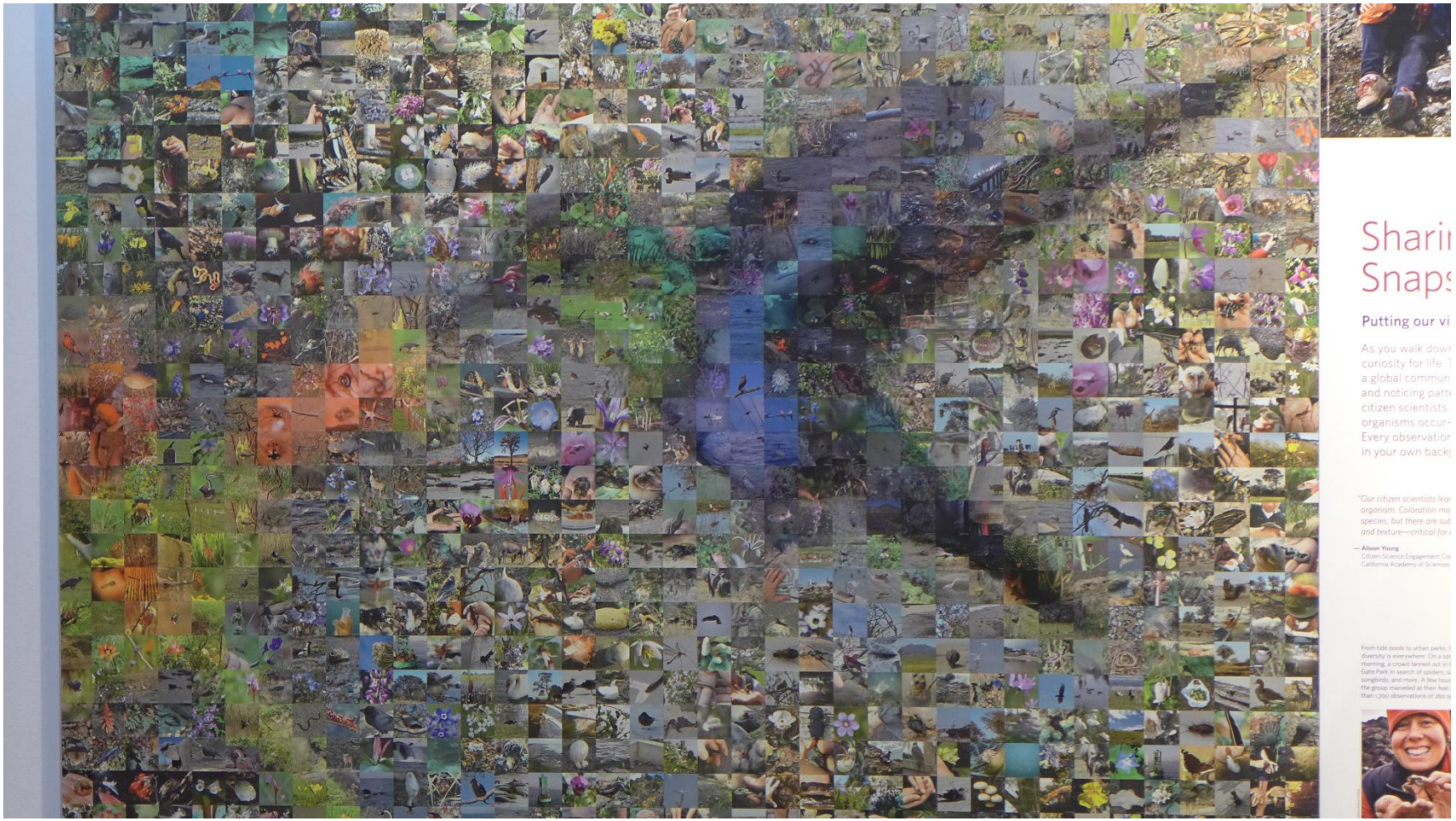
Most information in the image is carried in the phase, not the amplitude.

What is the relationship to phase in audio?

- In audio perception, frequency is important but phase is not.
- In visual perception, both are important.

- ??? : (


Ken-ichi Ueda
sodutchbaby
flapack
tueda
tsokrau
Mark Wilkinson
Ken-ichi Ueda
antonyw
flapack
Mark Wilkinson
annetanne
drew_avery
flapack
Mark Wilkinson



Sharin
Snaps

Putting our vi

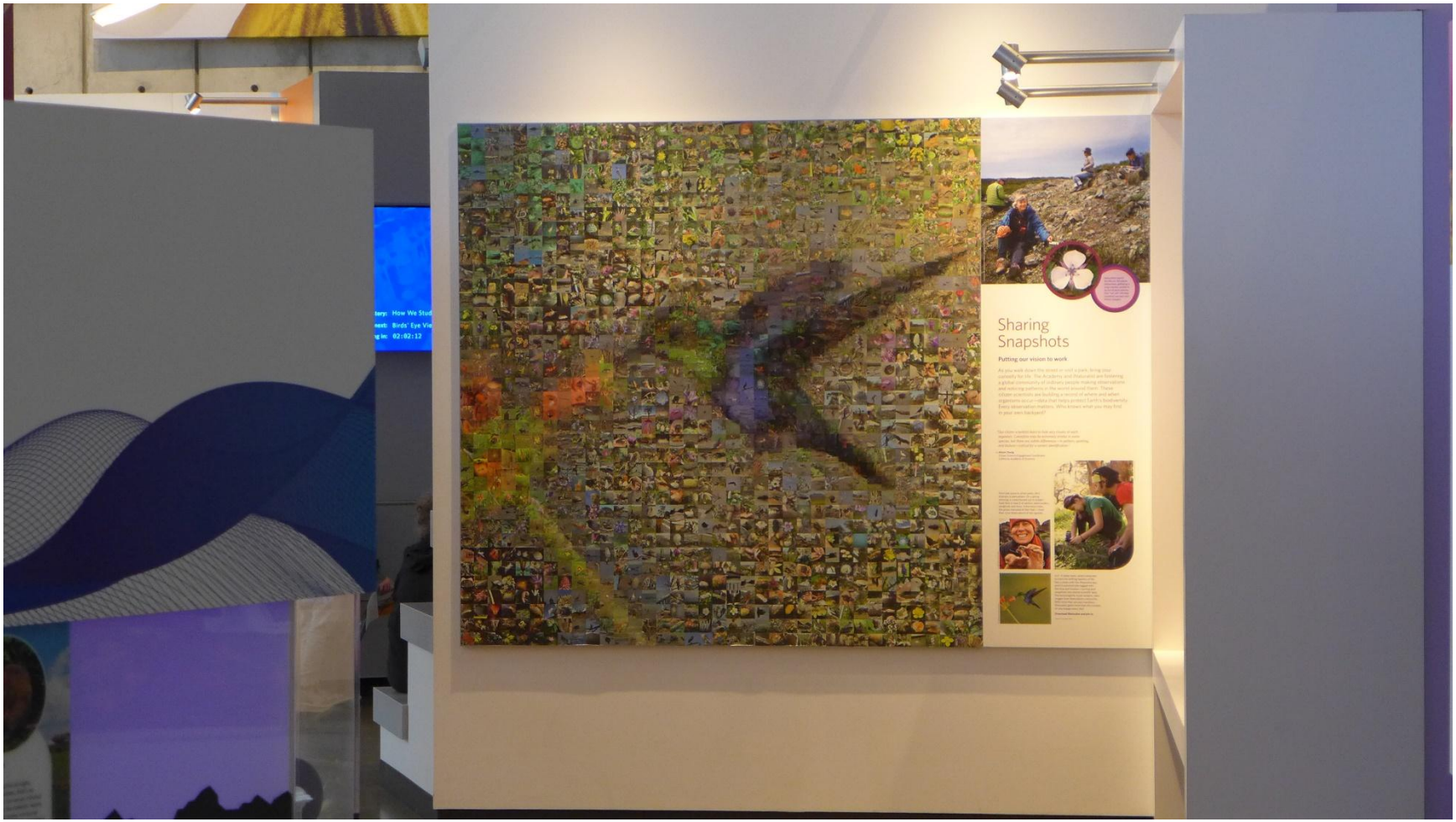
As you walk down
curiosity for life. T
a global commun
and noticing patte
citizen scientists
organisms occur-
Every observation
in your own back

"Our citizen scientists lea
organism. Coloration ma
species, but there are sub
and texture—critical for

— Alison Young
Citizen Science Engagement Coo
California Academy of Sciences

From tide pools to urban parks, b
diversity is everywhere. On a spr
morning, a crowd fanned out in
Gate Park in search of spiders, s
songbirds, and more. A few hour
the group marveled at their findi
than 1,700 observations of 280 s


Sharing
Snapshots
Putting our vision to work
As you walk down the street or visit a park, bring your curiosity for life. The Academy and iNaturalist are fostering a global community of ordinary people making observations and noticing patterns in the world around them. These citizen scientists are building a record of where and when organisms occur—data that helps protect Earth's biodiversity. Every observation matters. Who knows what you may find in your own backyard?
"Our citizen scientists learn to look very closely at each organism. Coloration may be extremely similar in some species, but there are subtle differences—in pattern, spotting, and texture—critical for a correct identification."
— Alison Young
Citizen Science Engagement Coordinator
California Academy of Sciences


Sharing
Snapshots
Putting our vision to work


Seeing in
Color
Want to see the world differently? Try on another animal's eyes. Birds and bees see colors of light invisible to the human eye. Other animals detect only a few colors—dogs see shades of yellows, blues, and grays. Explore how every animal eye, from simple to complex, is built to pick up a certain slice of light.

Severe Moiré pattern

Properties of Fourier Transforms

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

The Convolution Theorem

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

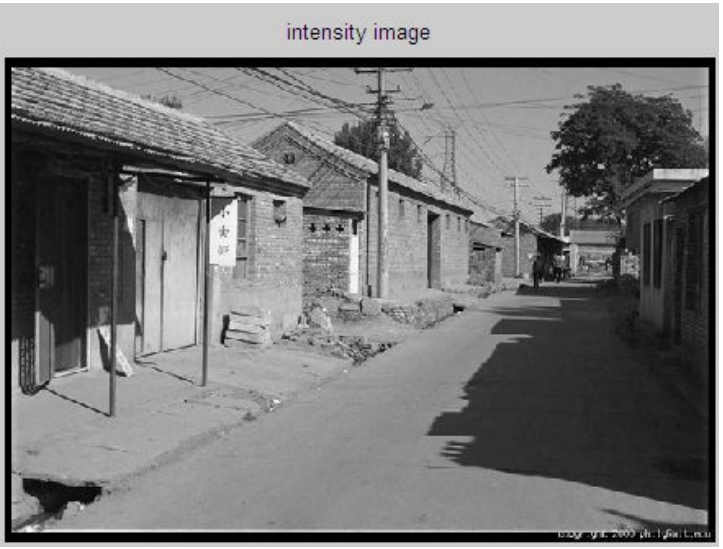
$$\mathrm{F}[\, g * h] = \mathrm{F}[\, g]\, \mathrm{F}[\, h]$$

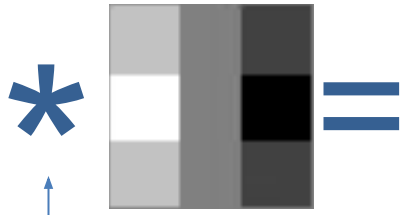
- **Convolution** in spatial domain is equivalent to **multiplication** in frequency domain!

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

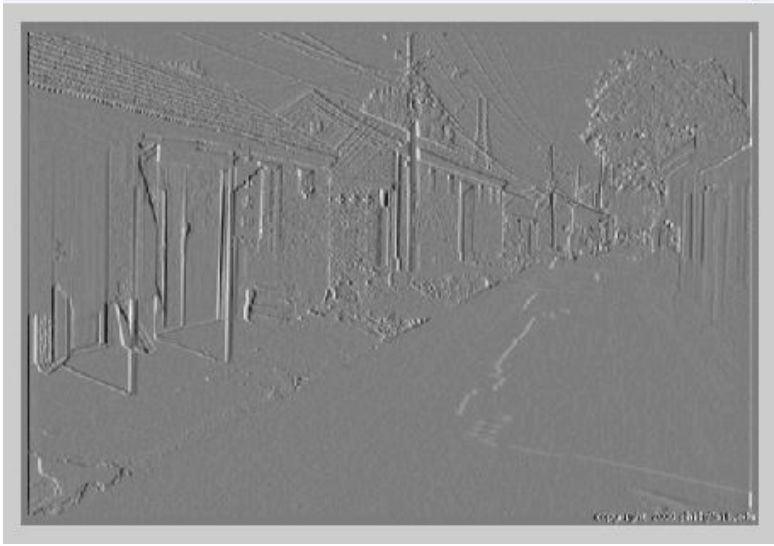
Filtering in spatial domain

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

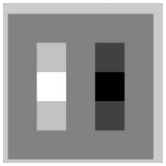




Convolution

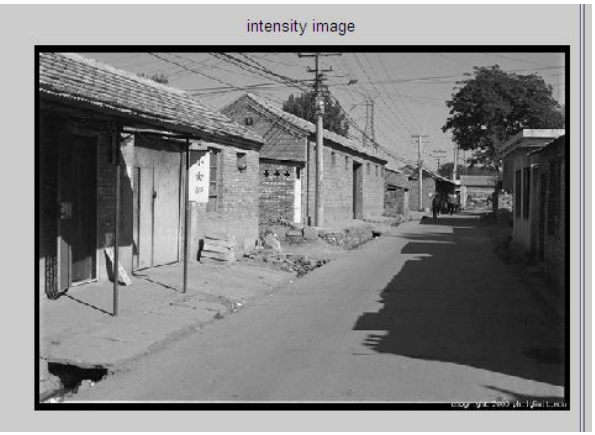


Filtering in frequency domain

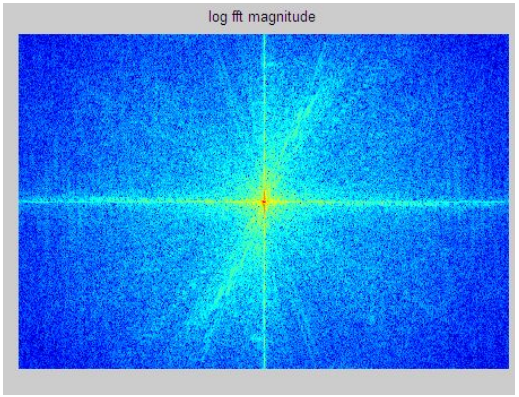


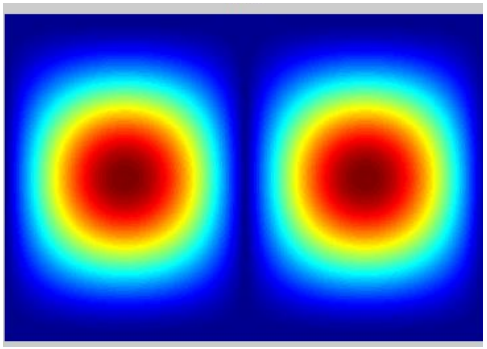
Element-wise product

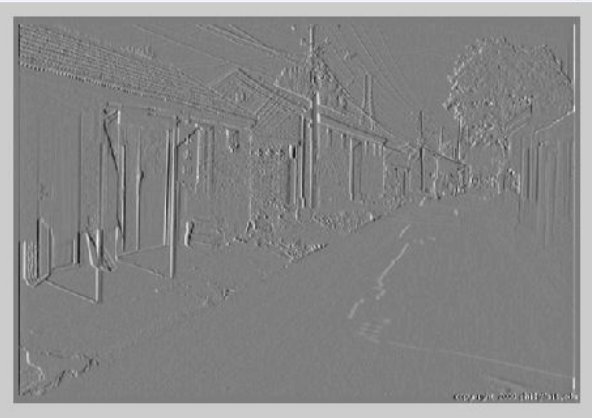
FFT



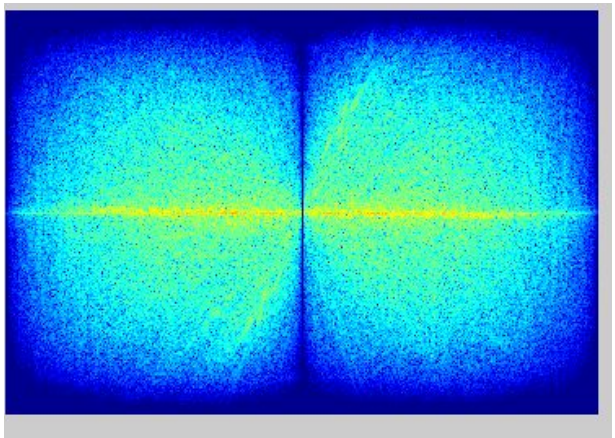
FFT





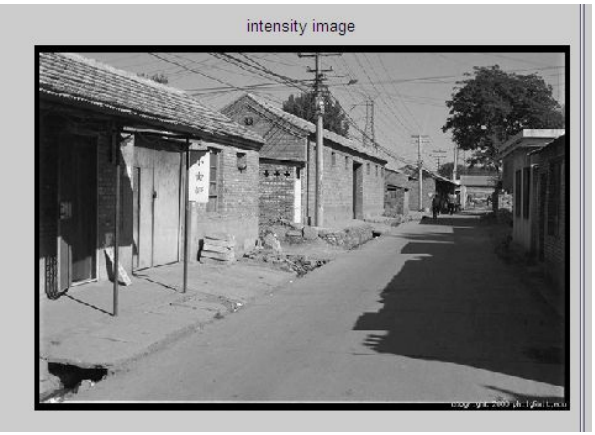


Inverse FFT



Slide: Hoiem

Filtering in frequency domain

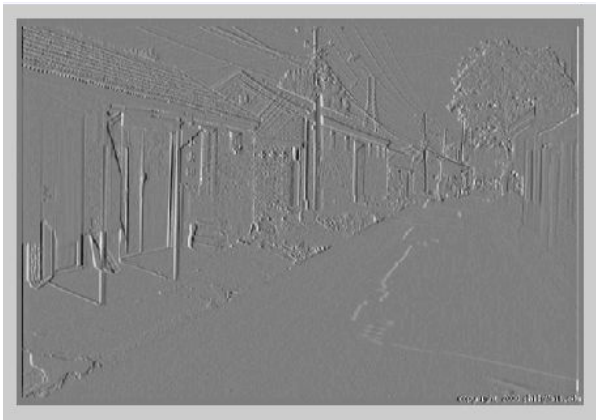


Fourier transform

Element-wise product

Fourier transform

Inverse Fourier transform



Slide: Hoiem

What about complexity?

Fourier transform = correlation of signal with basis frequency signals

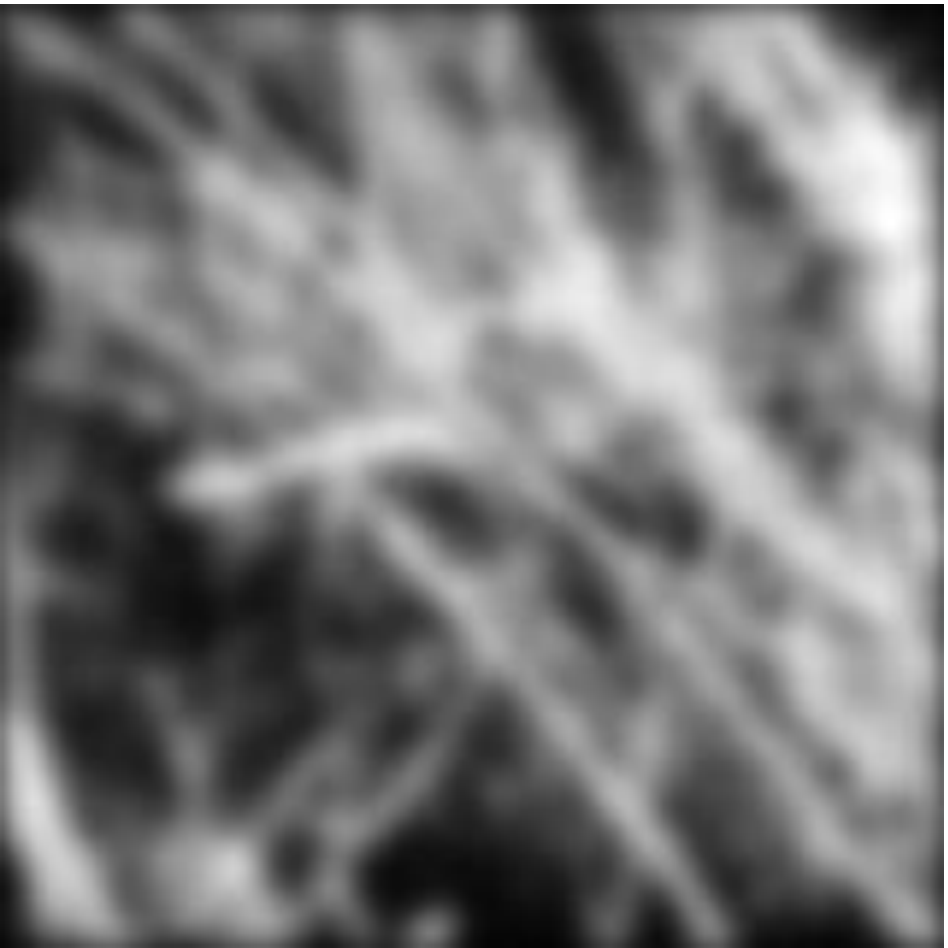
- Intuitively, in 2D, an $O(N^2)$ operation

Fast Fourier Transform (FFT)

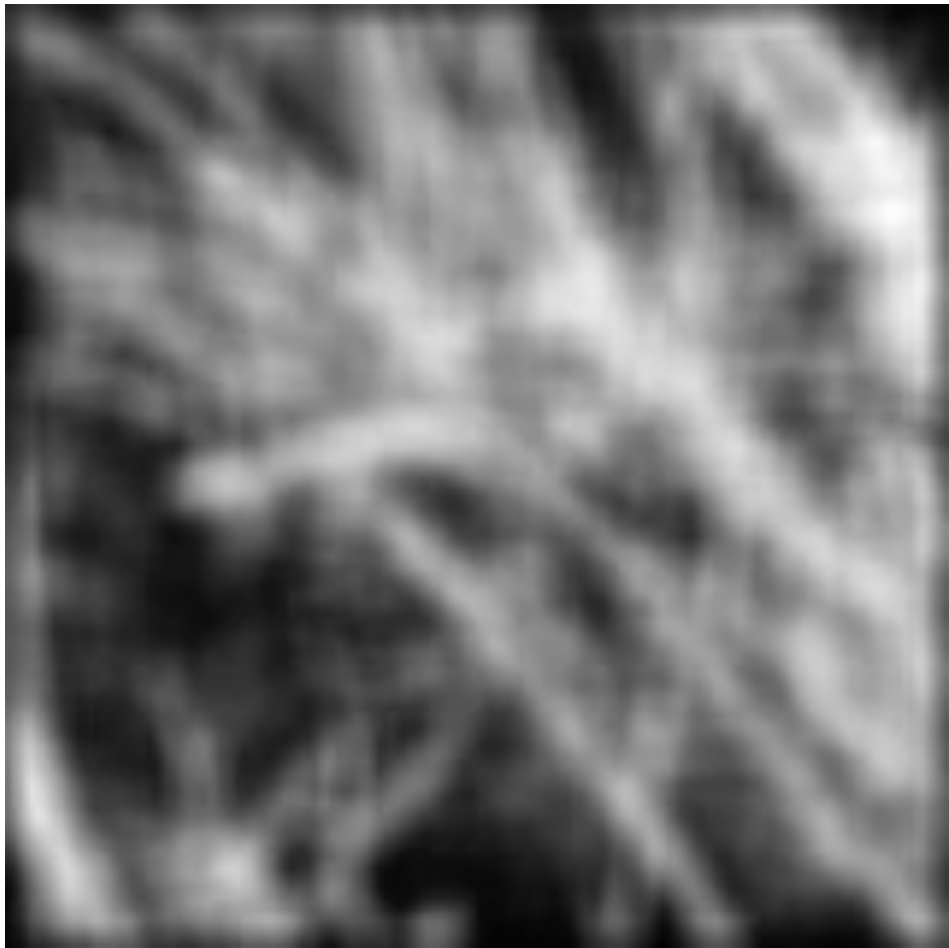
- O(NlogN)
- For large kernel sizes, it can be faster to exploit convolution theorem and operate via FFT

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

Gaussian 

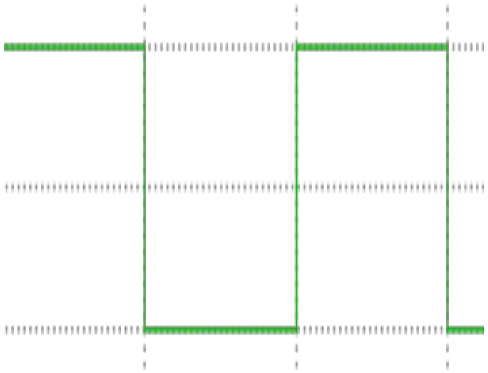


Box filter 



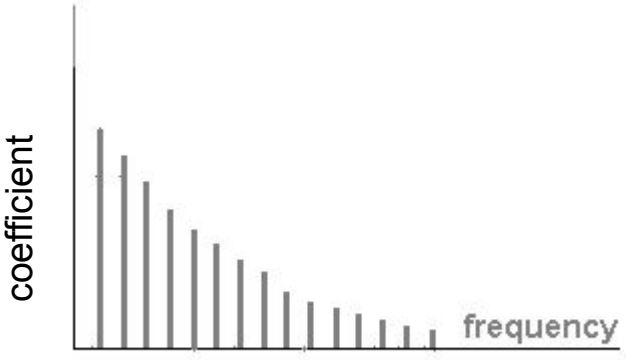
Why do we have those lines in the image?

Sharp edges in an image or kernel need _all_ frequencies to represent them.



=

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

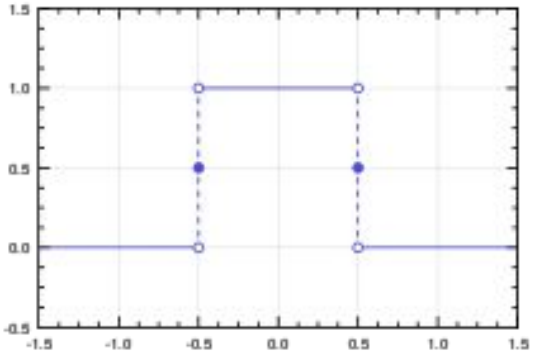


Box filter / sinc filter duality

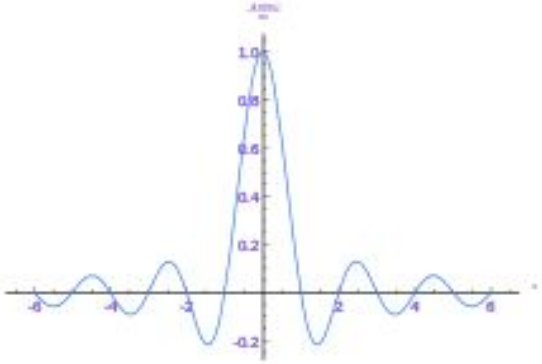
What is the box filter in the frequency domain?

What is the sinc filter in the frequency domain?

Box filter



Sinc filter



sinc(x) = sin(x) / x

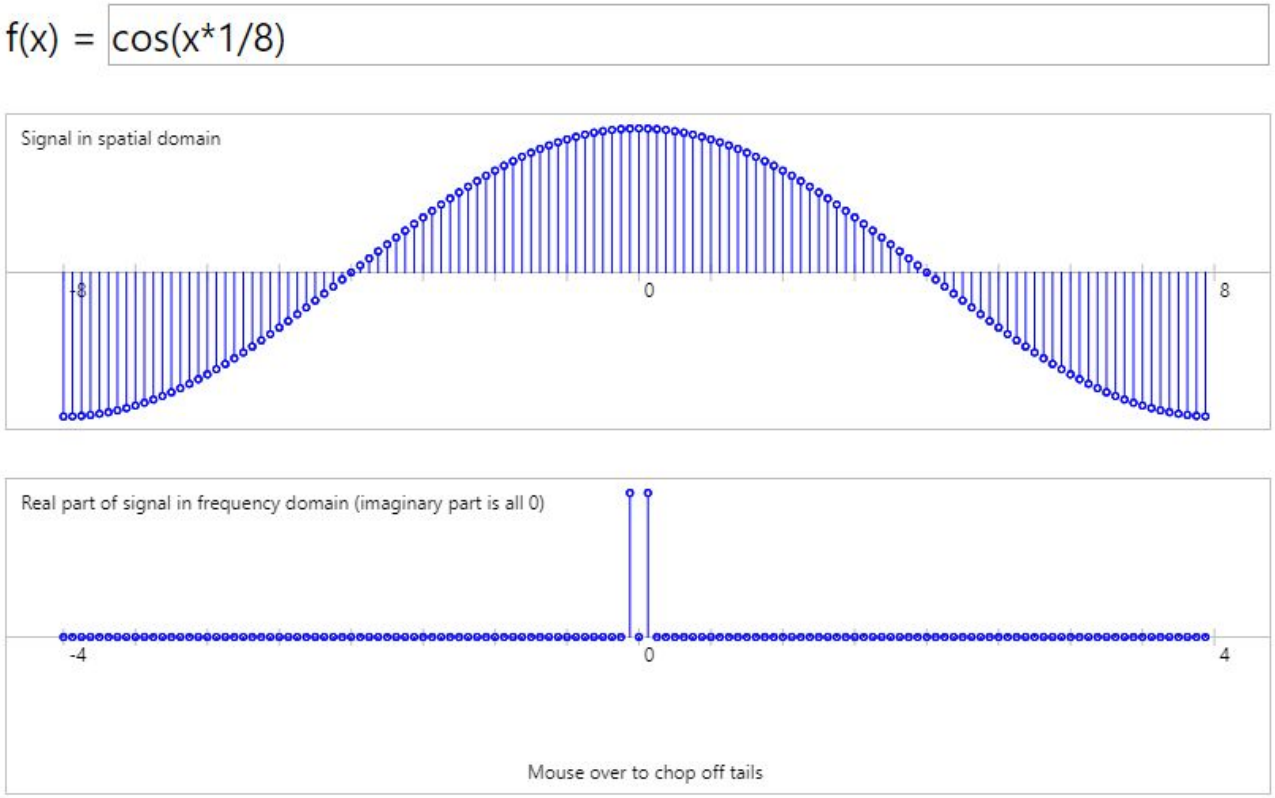
Spatial Domain ⟺ Frequency Domain

Frequency Domain ⟺ Spatial Domain

Box filter / sinc filter duality

What is the box filter in the frequency domain?

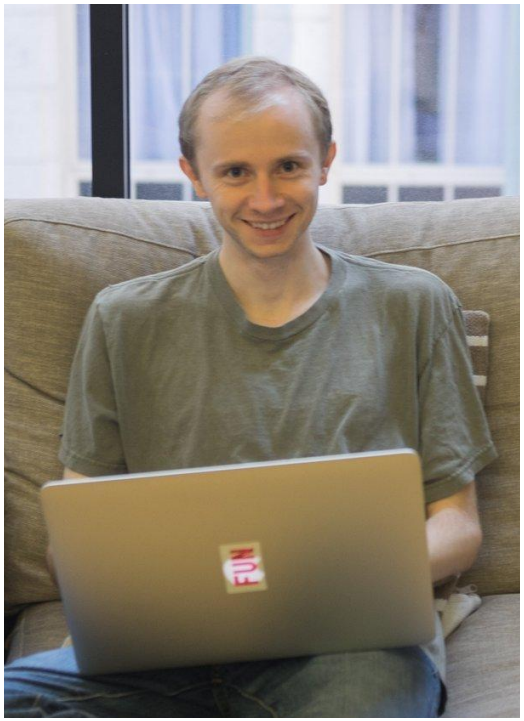
http://madebyevan.com/dft/



Evan Wallace demo

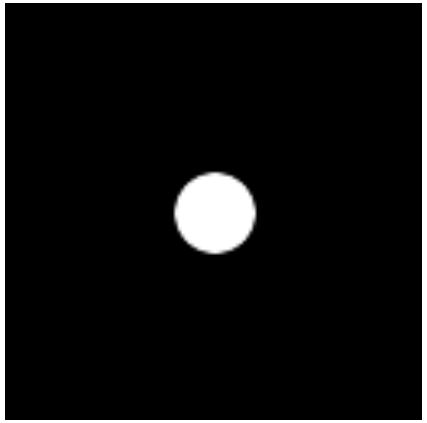
- Made for CSCI1230
- 1D example
- Forbes 30 under 30
  - Figma (collaborative design tools)

- http://madebyevan.com/dft/

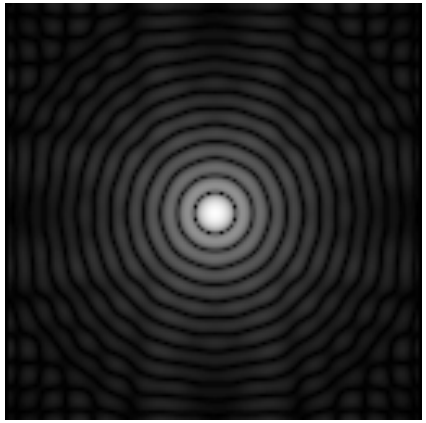


with Dylan Field

Box filter (spatial)



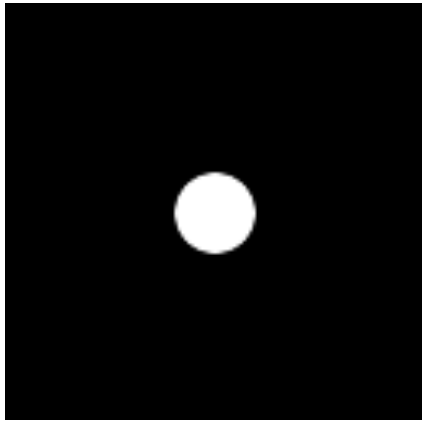
Frequency domain magnitude



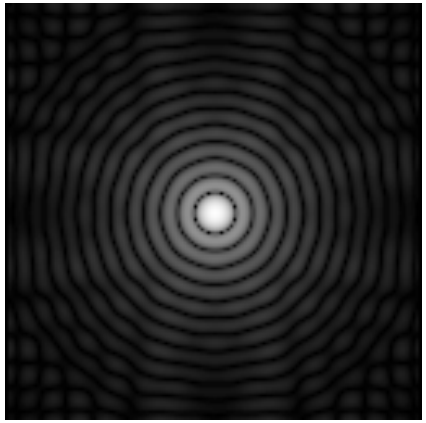
Gaussian filter duality

- Fourier transform of one Gaussian…
  …is *another Gaussian (with inverse variance).*
- Why is this useful?
  - Smooth degradation in frequency components
  - No sharp cut-off
  - No negative values
  - Never zero (infinite extent)

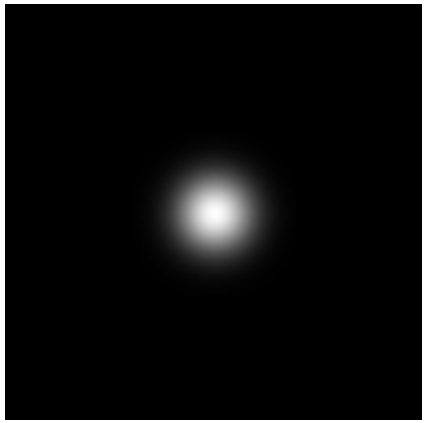
Box filter (spatial)



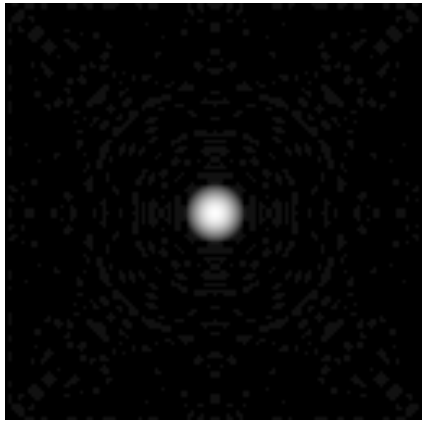
Frequency domain magnitude



Gaussian filter (spatial)



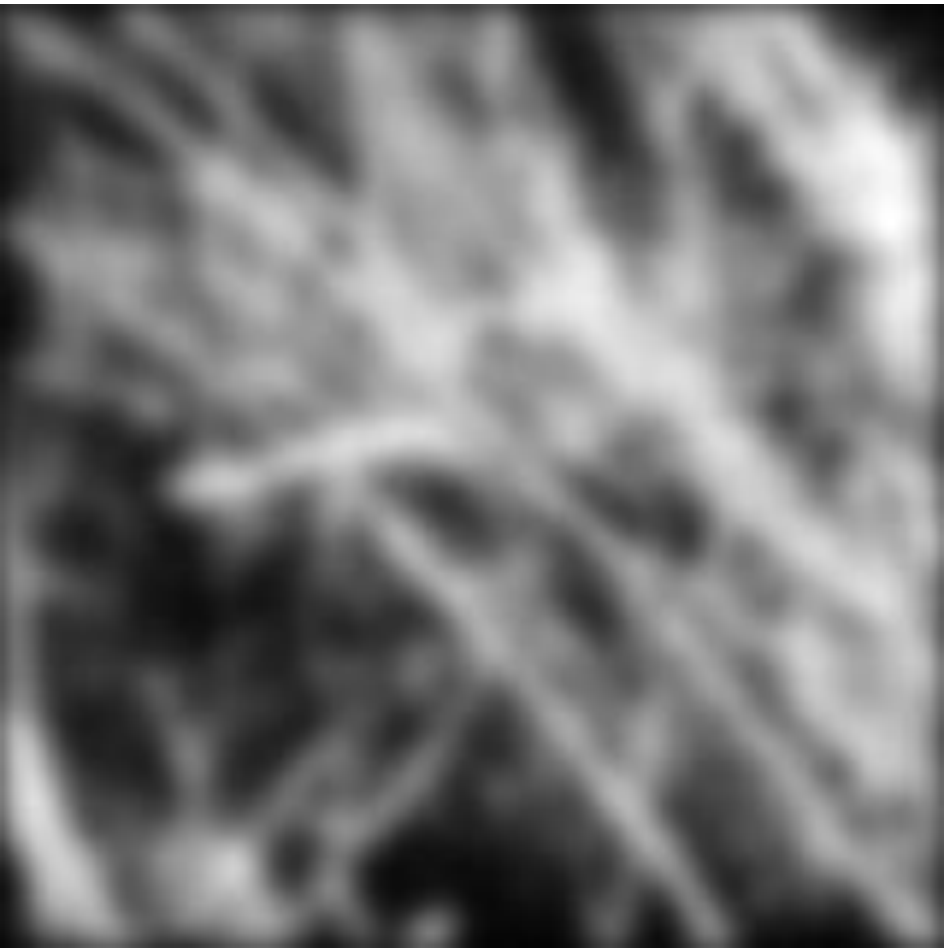
Frequency domain amplitude



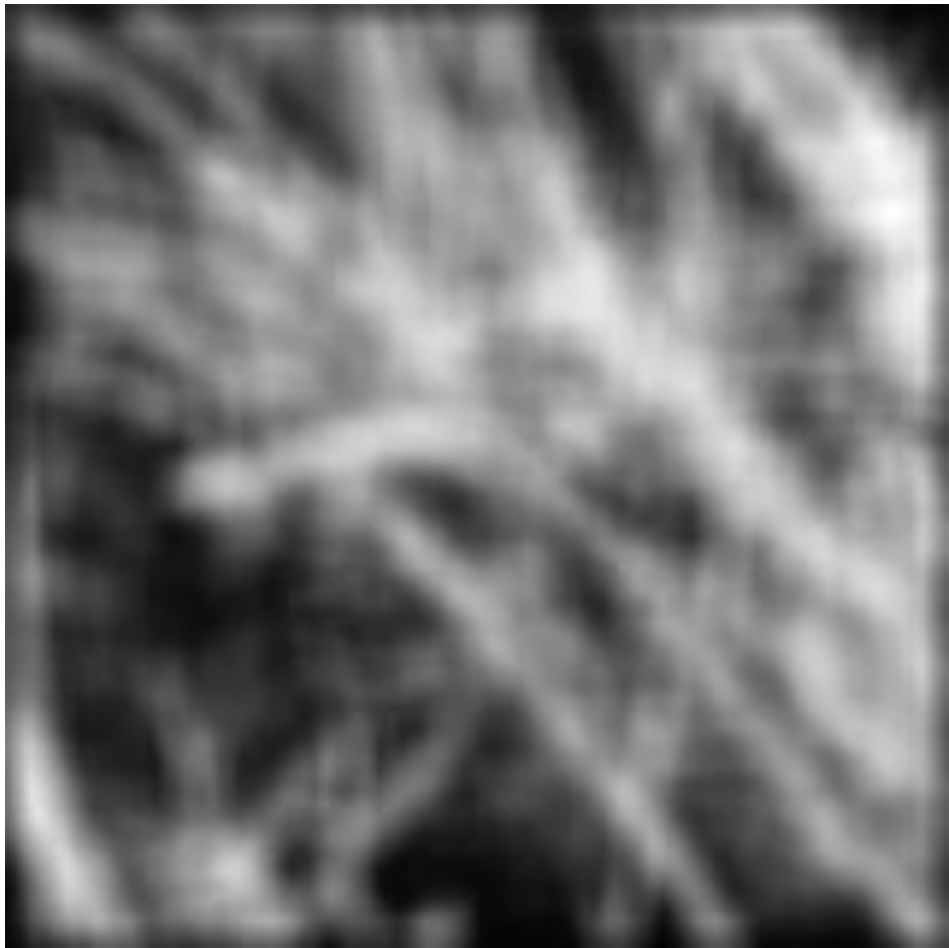
Ringing artifacts -> ‘Gibbs effect’

Where *infinite* series can never be reached

Gaussian 



Box filter 



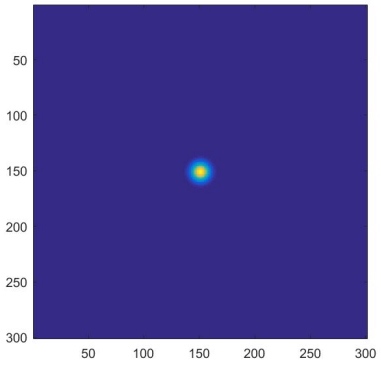
Is convolution invertible?

- If convolution is just multiplication in the Fourier domain, isn't deconvolution just division?

- Sometimes, it clearly is invertible (e.g. a convolution with an identity filter)
- In one case, it clearly isn't invertible (e.g. convolution with an all zero filter)
- What about for common filters like a Gaussian?

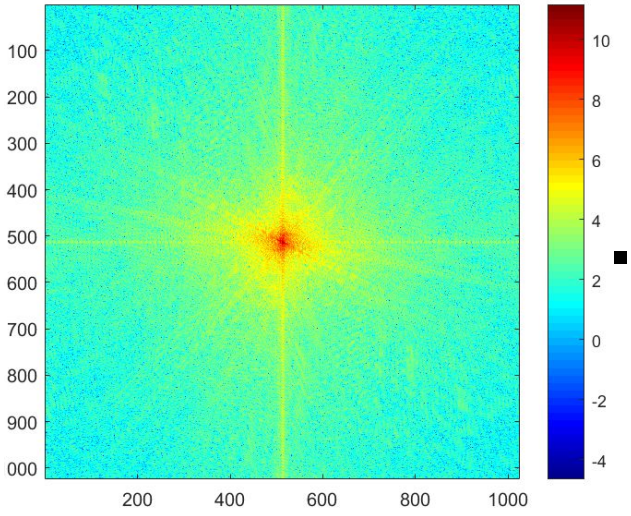
Convolution

*

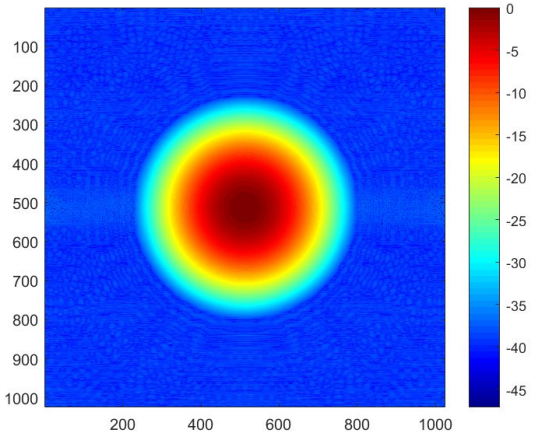


=

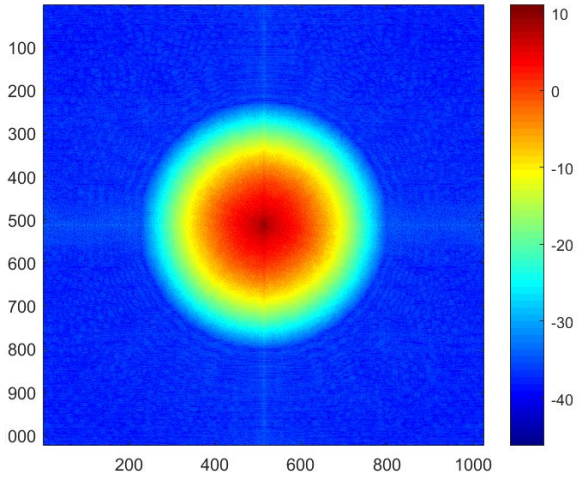
FFT ⬇ FFT ⬇ iFFT ⬆



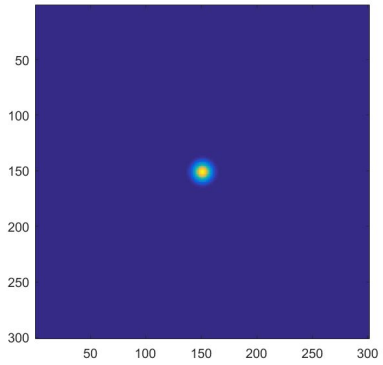
.X



=



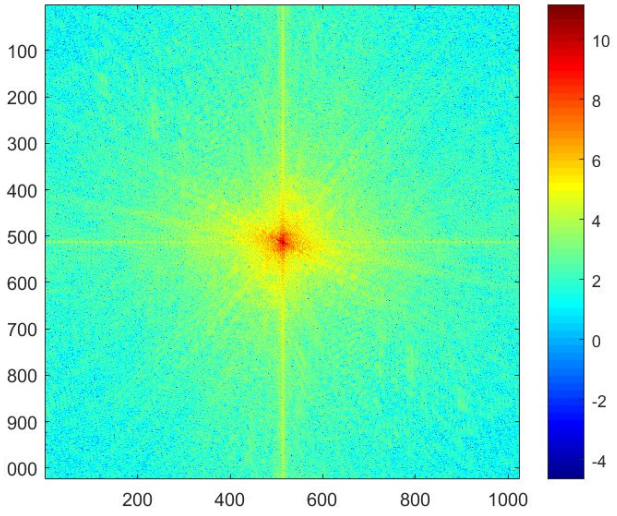
Deconvolution?



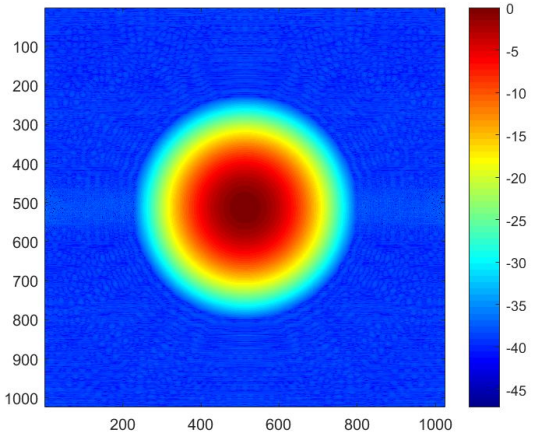
iFFT

FFT

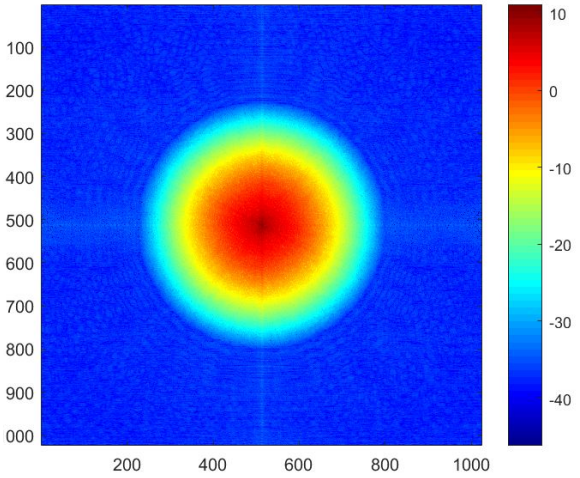
FFT



=

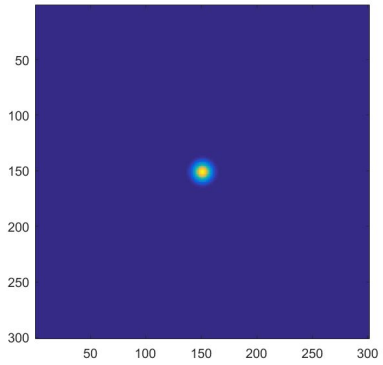


./



Hays

But under more realistic conditions

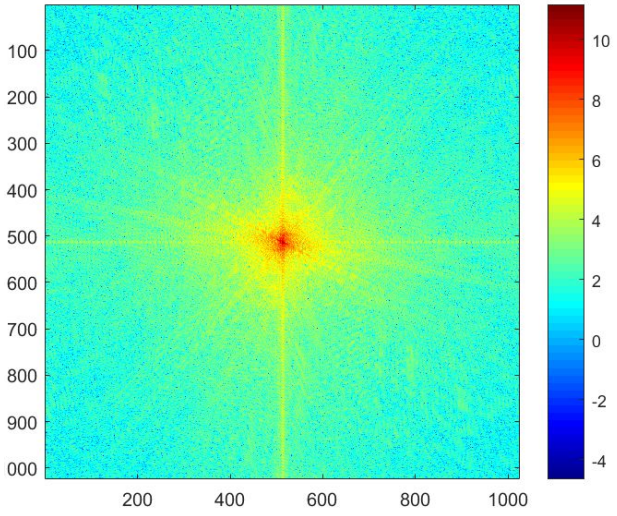


Random noise, .000001 magnitude

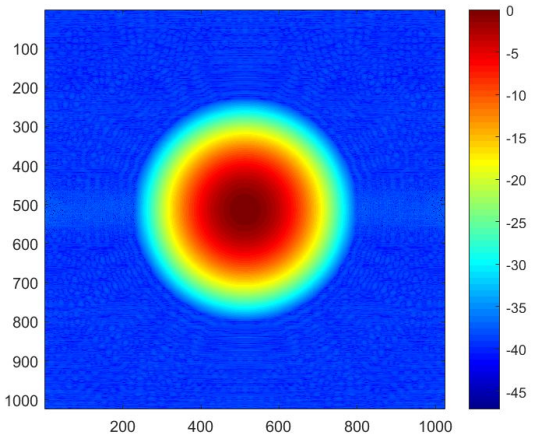
iFFT

FFT

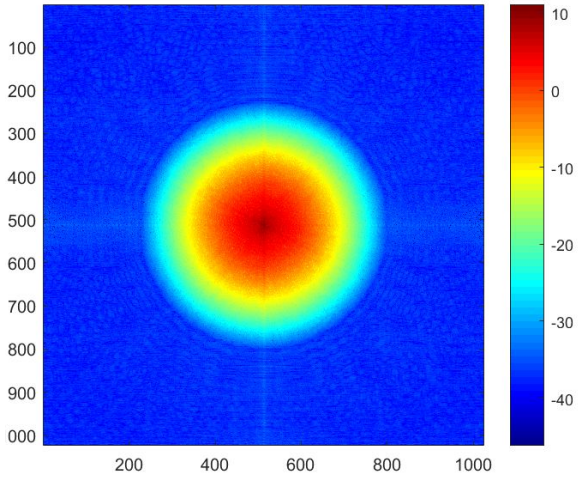
FFT



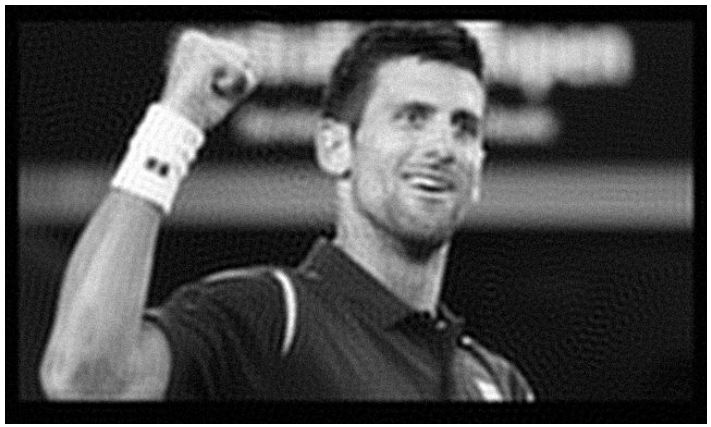
=

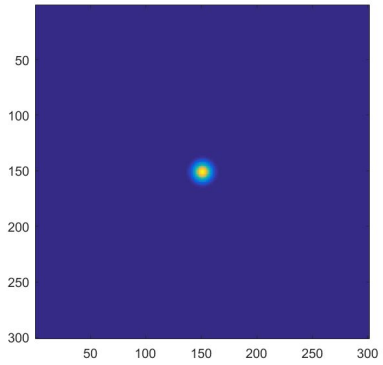


./



But under more realistic conditions



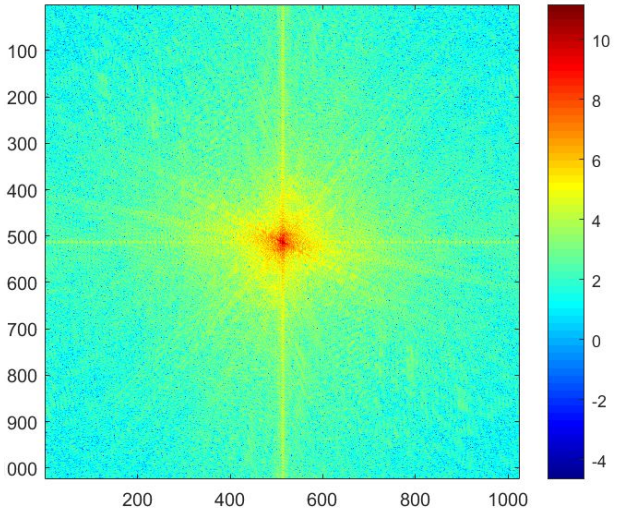


Random noise, .0001 magnitude

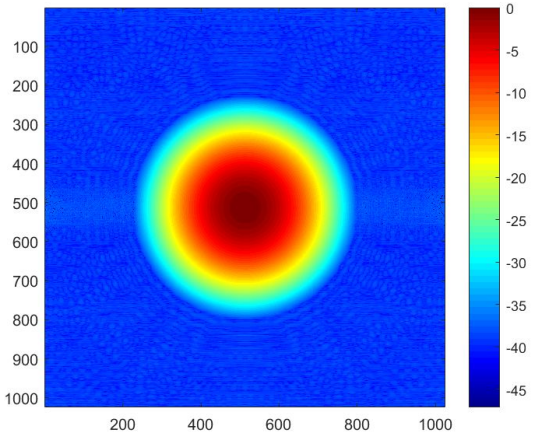
iFFT

FFT

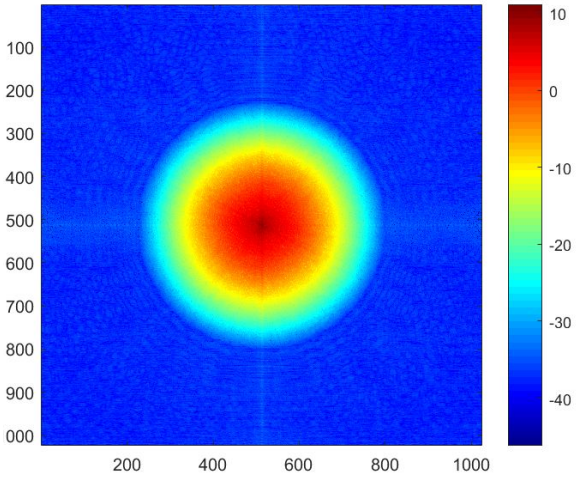
FFT



=

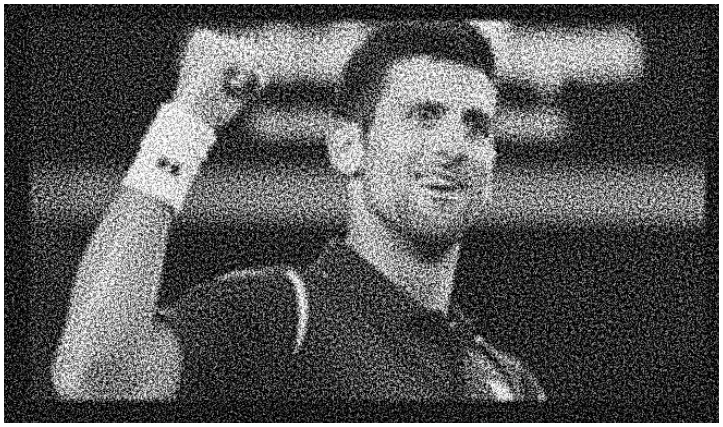


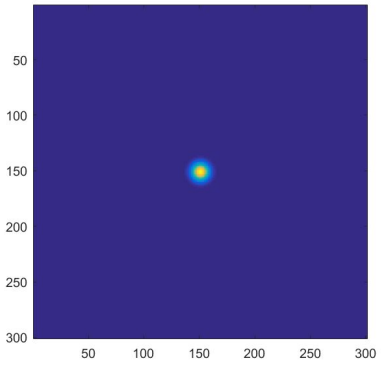
./



Hays

But under more realistic conditions



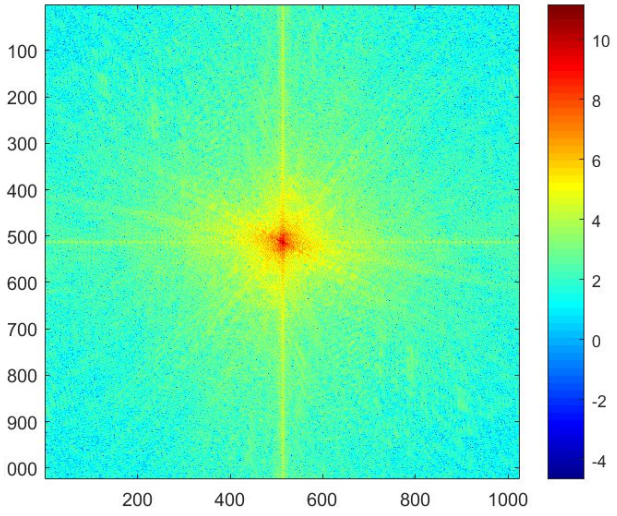


Random noise, .001 magnitude

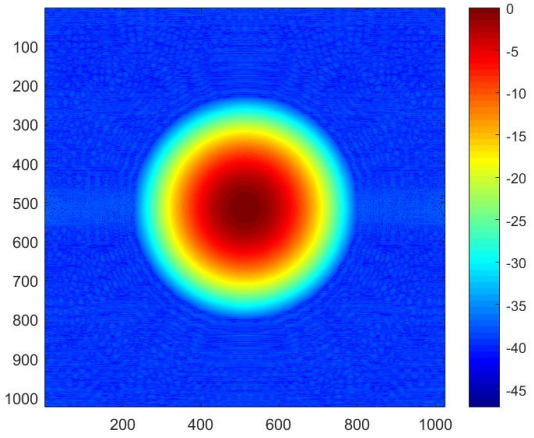
iFFT ↑

FFT ↓

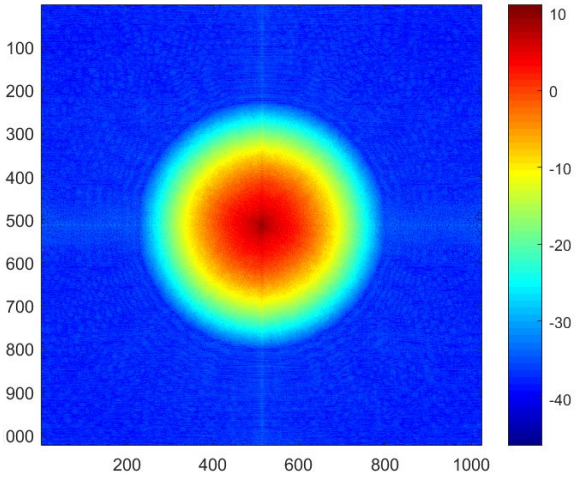
FFT ↓



=



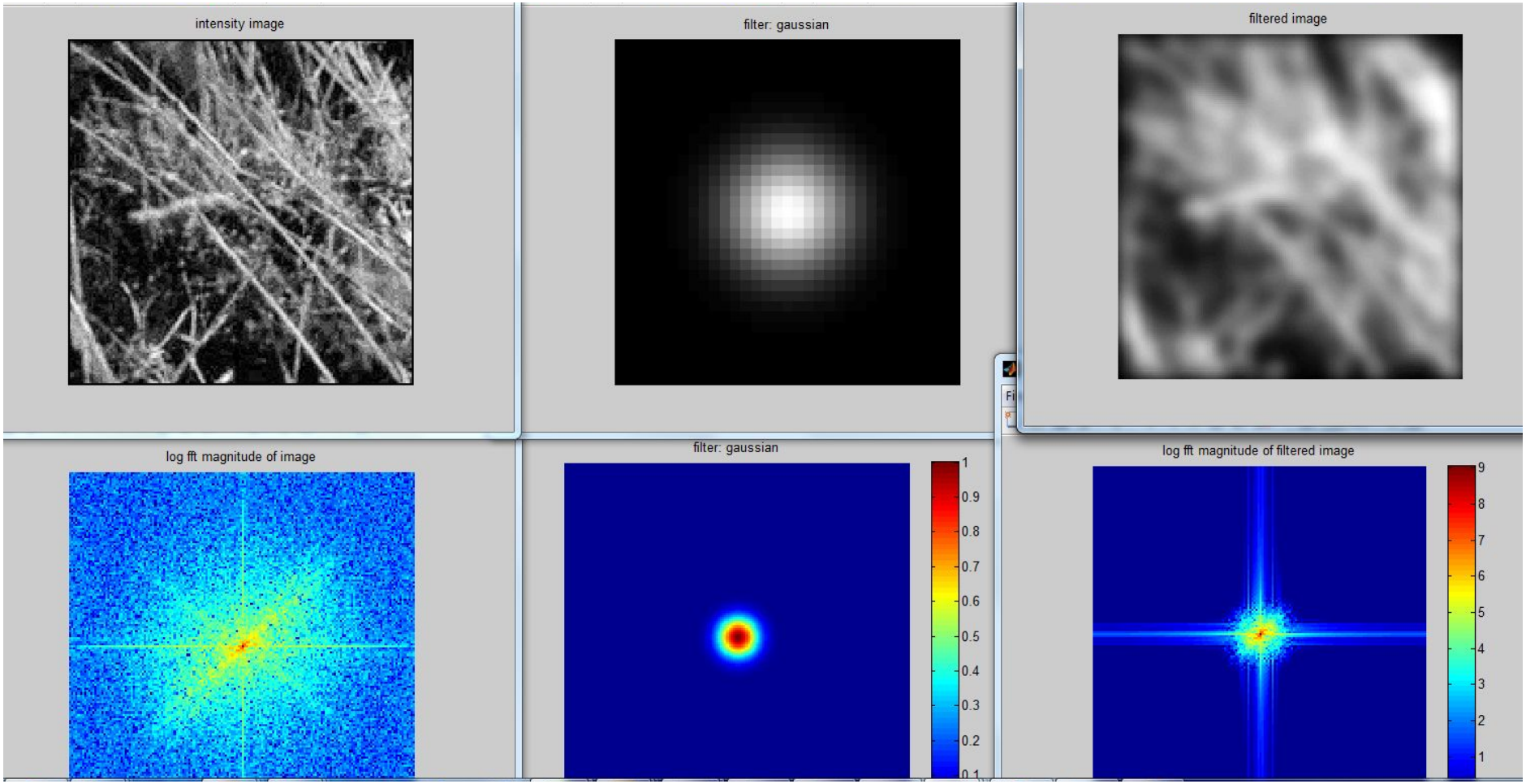
./



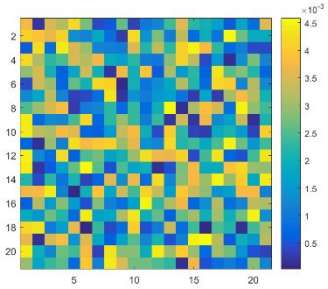
Hays

But you can't invert multiplication by 0!

- A Gaussian is only zero at infinity…



With a random filter…

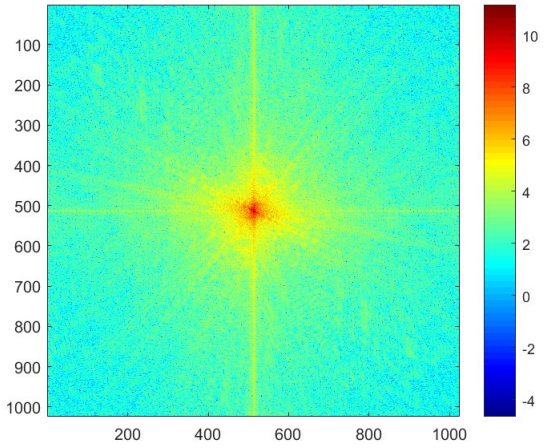


Random noise, .001 magnitude

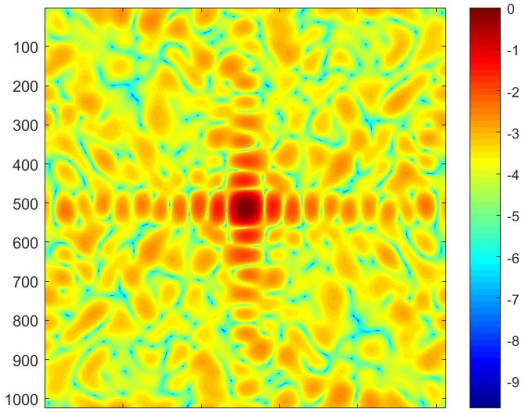
iFFT ↑

FFT ↓

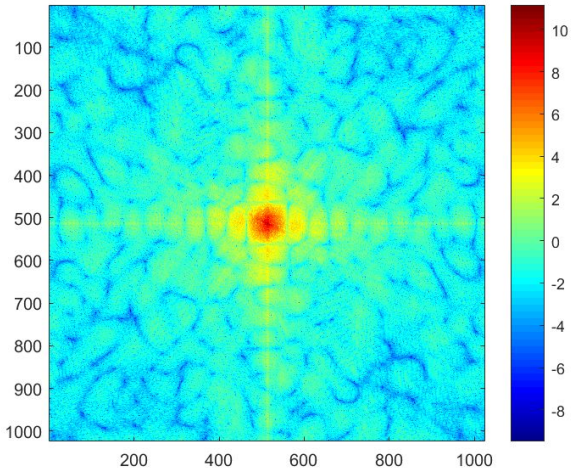
FFT ↓



=



./

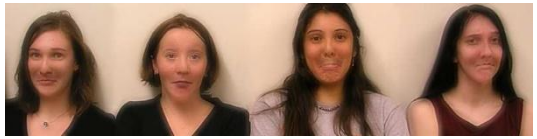


Deconvolution is hard.

- Active research area.
- Even if you know the filter (non-blind deconvolution), it is still hard and requires strong *regularization* to counteract noise.
- If you don’t know the filter (blind deconvolution), then it is harder still.

Application: Hybrid Images

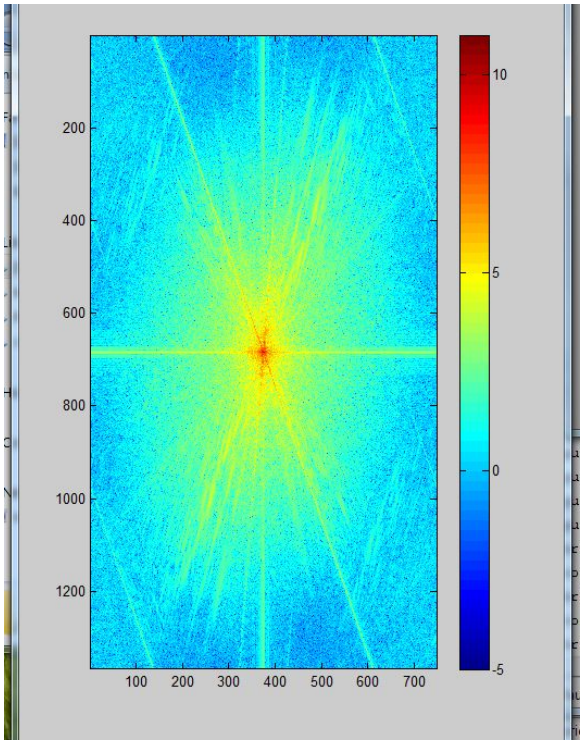
When we see an image from far away, we are effectively subsampling it!



A. Oliva, A. Torralba, P.G. Schyns, SIGGRAPH 2006

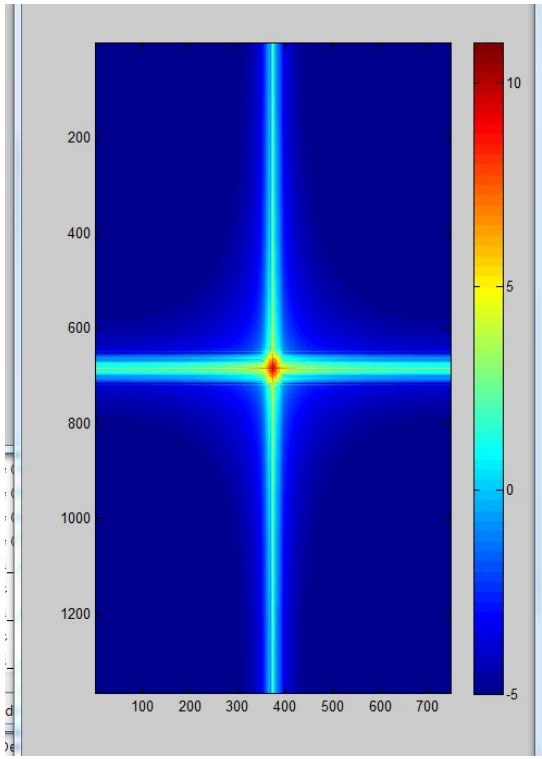
Hybrid Image in FFT

Hybrid Image



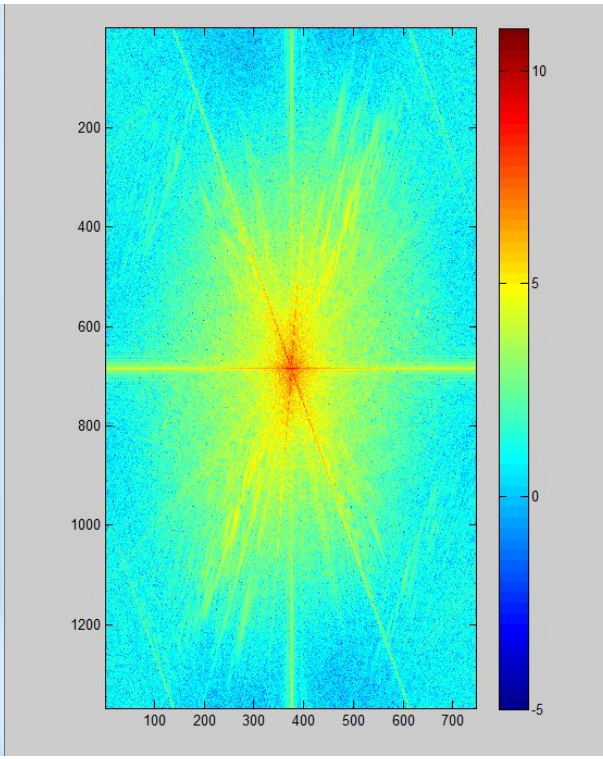
=

Low-passed Image



+

High-passed Image



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

Additional biographies: MacTutor (St. Andrews), Bonn

How would math have changed if the onesie had been invented?!?! : (

Fast Fourier Transform in Matlab

- Filtering with fft (fft2 -> 2D)

```
im = double(imread('…'))/255;
im = rgb2gray(im); % "im" should be a gray-scale floating point image
[imh, imw] = size(im);

hs = 50; % filter half-size
fil = fspecial('gaussian', hs*2+1, 10);

fftsize = 1024; % should be order of 2 (for speed) and include padding
im_fft  = fft2(im,  fftsize, fftsize);                    % 1) fft im with padding
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

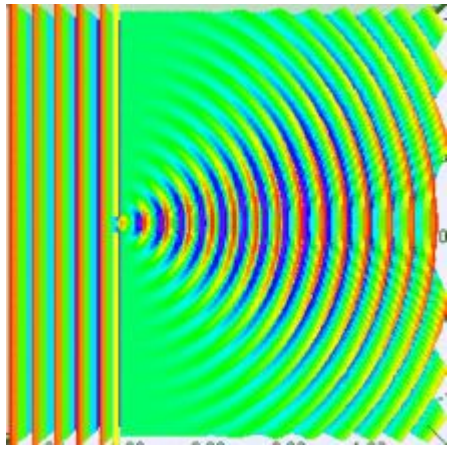
```
figure(1), imagesc(log(abs(fftshift(im_fft)))), axis image, colormap jet
```

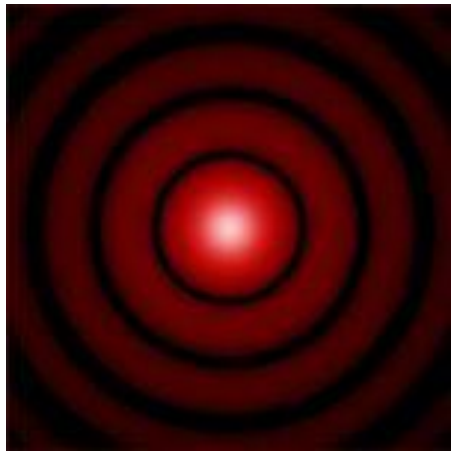
Slide: Hoiem

Applications of Fourier analysis

- Fast filtering with large kernels

- Fourier Optics
  - Fraunhofer diffraction is Fourier transform of slit in the ‘far field’.



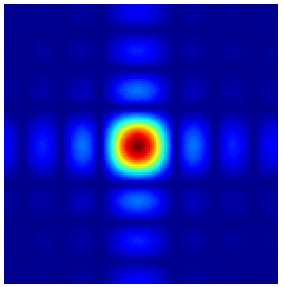


Circular aperture = Airy disc diffraction pattern

  - Light spectrometry for astronomy

Wikipedia for graphics

Things to Remember

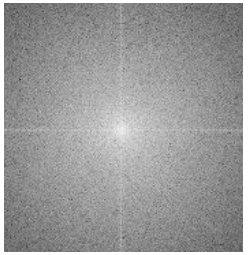


Sometimes it helps to think of images and filtering in the frequency domain

- Fourier analysis

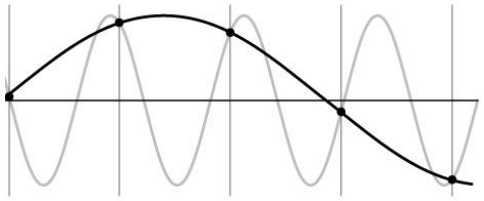
Can be faster to filter using FFT for large images

- N logN vs. N^2 for convolution/correlation



Images are mostly smooth

- Basis for compression



Remember to low-pass before sampling

- Otherwise you create aliasing

Review of Filtering

Filtering in frequency domain

- Can be faster than filtering in spatial domain (for large filters)
- Can help understand effect of filter
- Algorithm:
  1. Convert image and filter to Fourier domain (e.g., numpy.fft.fft2())
  2. Element-wise multiply their decompositions
  3. Convert result to spatial domain with inverse Fourier transform (e.g., numpy.fft.ifft2())

You will play with code in Proj2 questions

Before we leave

Fourier decomposition is tricky

I want to know what is confusing to you!

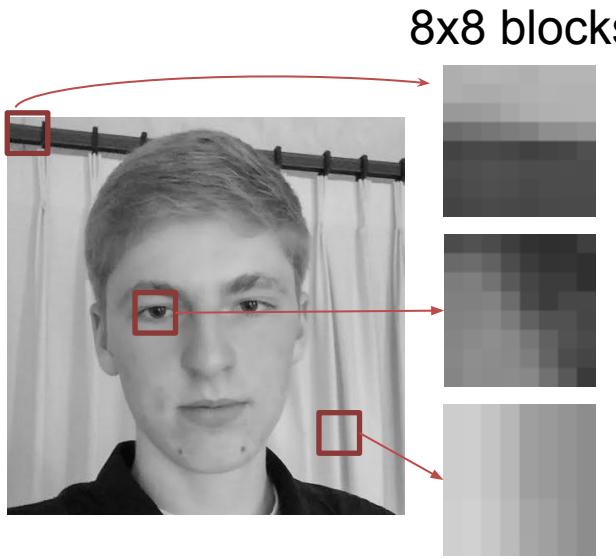
- Take 1 minute to think about the lecture

- Write down on Prismia.Chat something that you'd like clarifying.

Thinking in Frequency - Compression

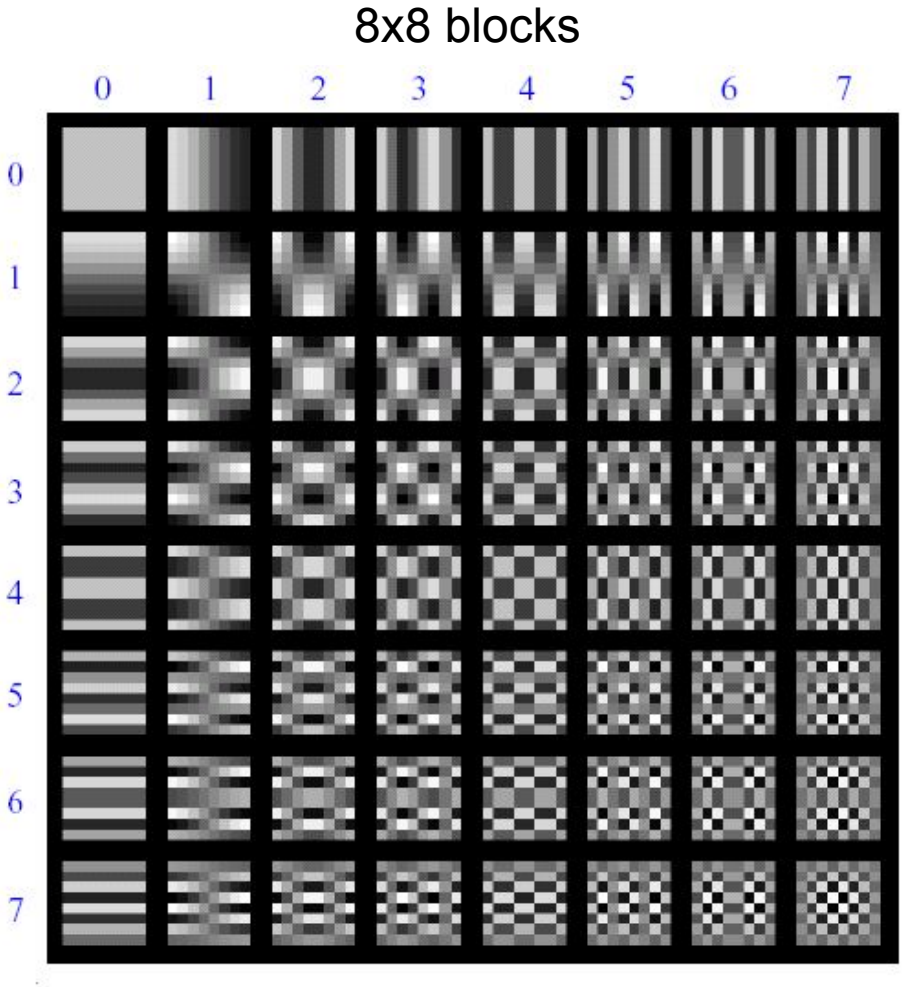
How is it that a 4MP image can be compressed to a few hundred KB without a noticeable change?

Lossy Image Compression (JPEG)



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

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

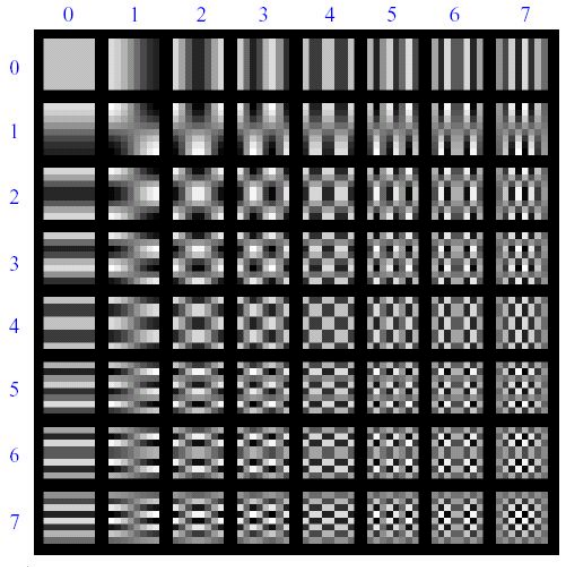


Block-based Discrete Cosine Transform (DCT)

Slides: Efros

Using DCT in JPEG

- The first coefficient $B(0,0)$ is the DC component, the average intensity
- The top-left coeffs represent low frequencies, the bottom right – high frequencies



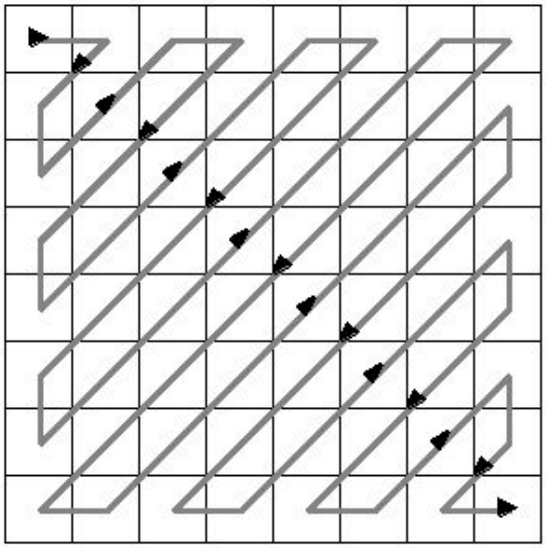


Image compression using DCT

- Compute DCT filter responses in each 8x8 block

Filter responses

$$G = \begin{bmatrix} -415.38 & -30.19 & -61.20 & 27.24 & 56.13 & -20.10 & -2.39 & 0.46 \\ 4.47 & -21.86 & -60.76 & 10.25 & 13.15 & -7.09 & -8.54 & 4.88 \\ -46.83 & 7.37 & 77.13 & -24.56 & -28.91 & 9.93 & 5.42 & -5.65 \\ -48.53 & 12.07 & 34.10 & -14.76 & -10.24 & 6.30 & 1.83 & 1.95 \\ 12.12 & -6.55 & -13.20 & -3.95 & -1.88 & 1.75 & -2.79 & 3.14 \\ -7.73 & 2.91 & 2.38 & -5.94 & -2.38 & 0.94 & 4.30 & 1.85 \\ -1.03 & 0.18 & 0.42 & -2.42 & -0.88 & -3.02 & 4.12 & -0.66 \\ -0.17 & 0.14 & -1.07 & -4.19 & -1.17 & -0.10 & 0.50 & 1.68 \end{bmatrix} \begin{matrix} \\ \\ \\ \\ \\ \\ \\ \downarrow v \end{matrix}$$

$u \longrightarrow$

- Quantize to integer (div. by magic number; round)
  - More coarsely for high frequencies (which also tend to have smaller values)
  - Many quantized high frequency values will be zero

Quantization divisers (element-wise)

$$Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

Quantized values

$$B = \begin{bmatrix} -26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\ 0 & -2 & -4 & 1 & 1 & 0 & 0 & 0 \\ -3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\ -3 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

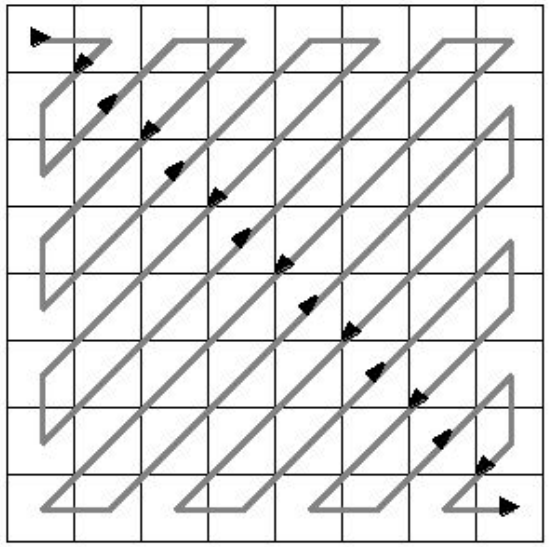
JPEG Encoding

- Entropy coding (Huffman-variant)

Quantized values

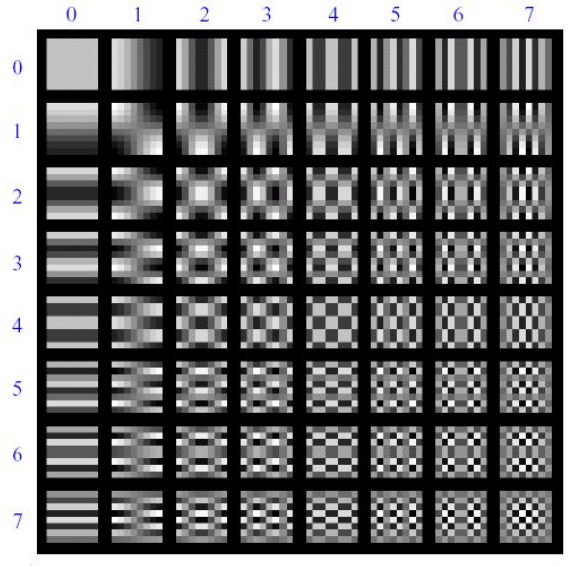
$$B = \begin{bmatrix} -26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\ 0 & -2 & -4 & 1 & 1 & 0 & 0 & 0 \\ -3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\ -3 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Linearize *B* like this.



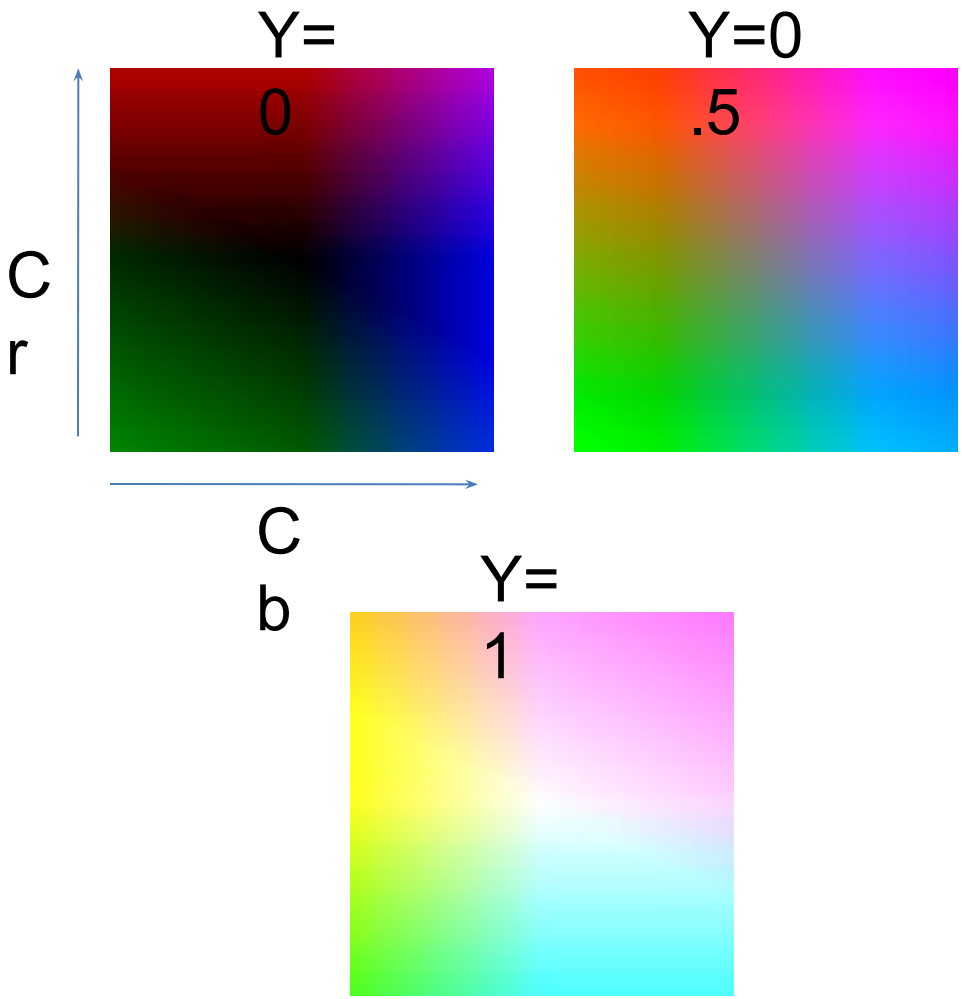
Helps compression:

- We throw away the high frequencies (‘0’).
- The zig zag pattern increases in frequency space, so long runs of zeros.



Color spaces: YCbCr

Fast to compute, good for compression, used by TV



Y
(Cb=0.5,Cr=0.5)

Cb
(Y=0.5,Cr=0.5)

Cr
(Y=0.5,Cb=05)

Most JPEG images & videos subsample chroma

PSP Comp 3
2x2 Chroma subsampling
285K

Original
1,261K lossless
968K PNG

JPEG Compression Summary

1. Convert image to YCrCb
2. Subsample color by factor of 2
   - People have bad resolution for color
3. Split into blocks (8x8, typically), subtract 128
4. For each block
   a. Compute DCT coefficients
   b. Coarsely quantize
      - Many high frequency components will become zero
   c. Encode (with run length encoding and then Huffman coding for leftovers)

http://en.wikipedia.org/wiki/YCbCr
http://en.wikipedia.org/wiki/JPEG

JPEG compression comparison

89 kb

12 kb