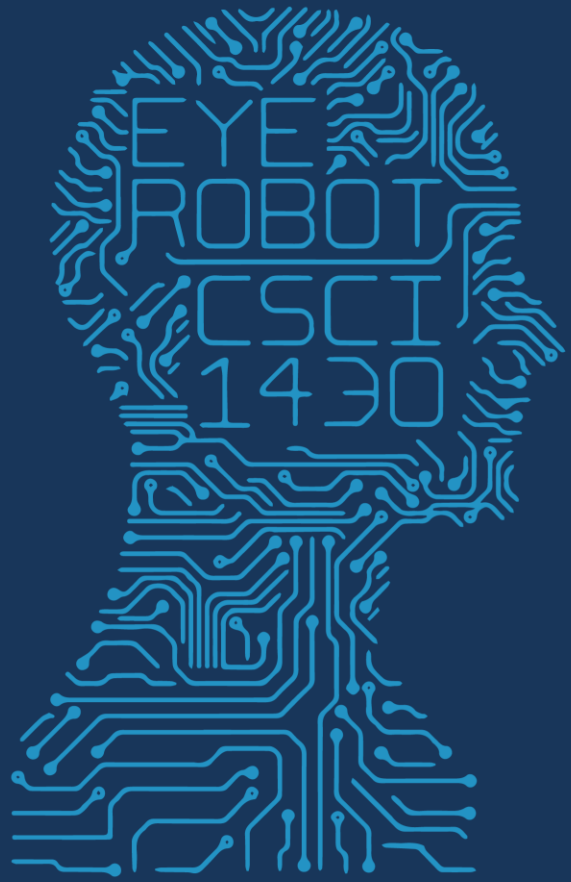
1950

Future Vision

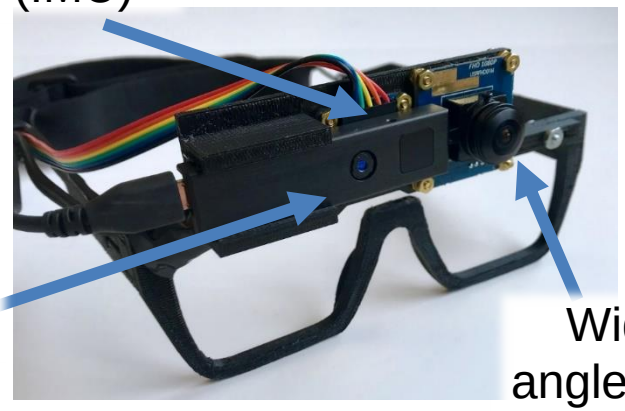


4 February 2019

Computer Vision

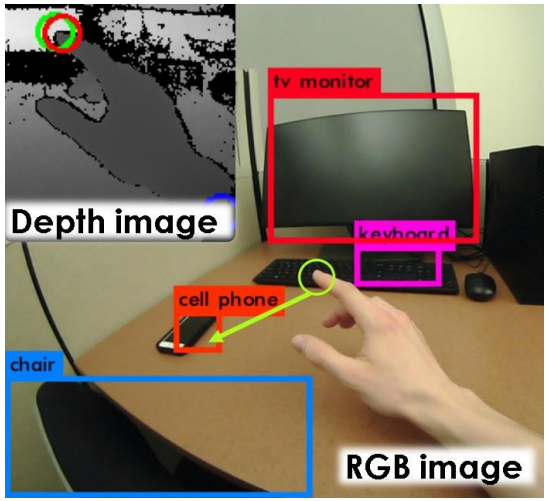
Intelligent Visual Prosthesis

Orientation sensor (IMU)



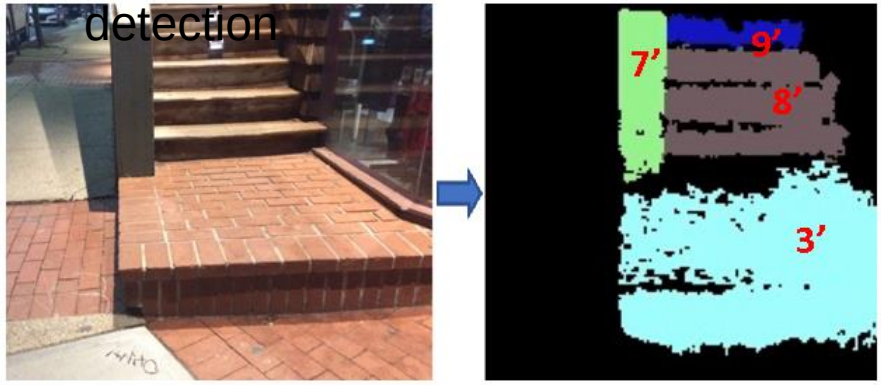
Depth camera

Wide-angle RGB camera



Simultaneous object recognition, localization, and hand tracking

Depth image-based obstacle detection



New projects:

- Barcode reading
- Face recognition
- Text reading
- Currency recognition

Michael_Paradiso@brown.edu
Morgan_Talbot@alumni.brown.edu

Will Povell ‘20

University memorial service

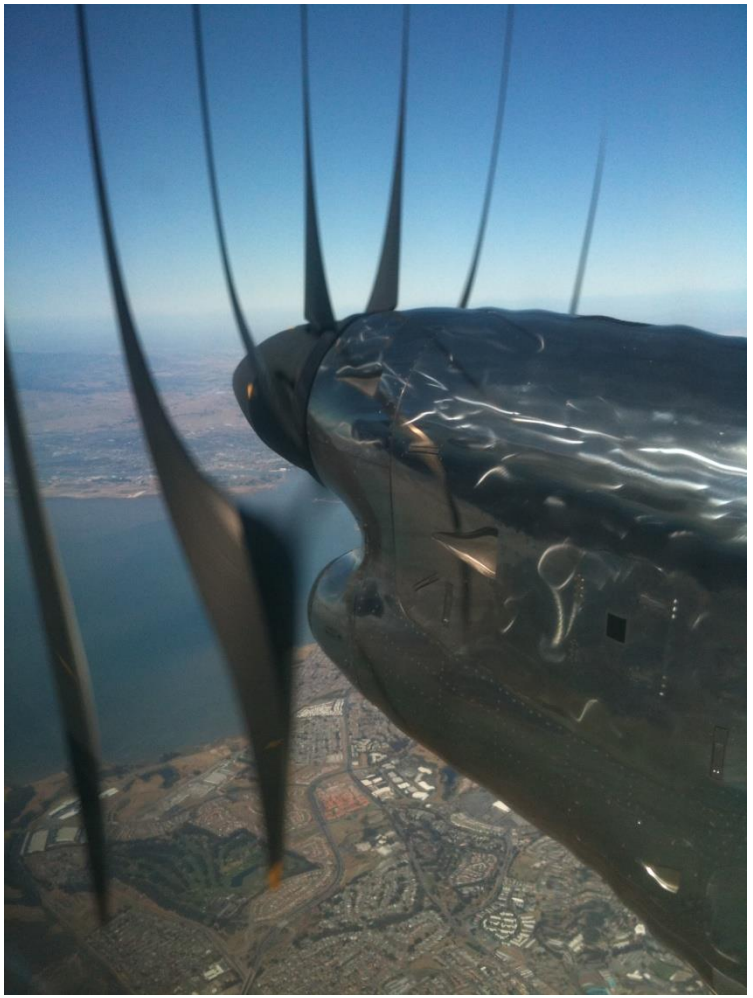
4pm today

Leung Family Gallery in the
Stephen Robert ’62 Campus Center

Class will end early.


DO NOT STEP

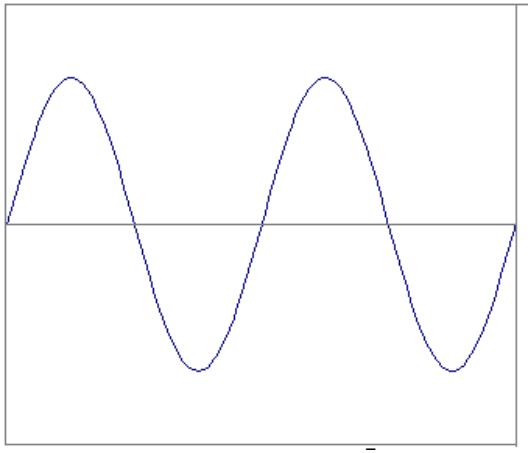
Capture Frequency - Rolling `Shutter’

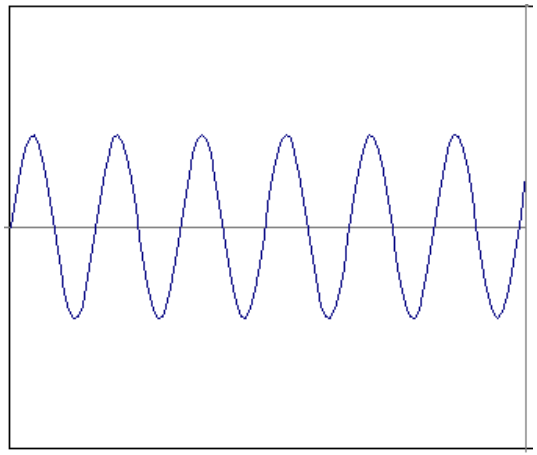


James Hays

High pass vs low pass filters

I understand frequency as in waves…





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

…to image frequency?

Another way of thinking about frequency

FOURIER SERIES & FOURIER TRANSFORMS

Fourier series

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.*

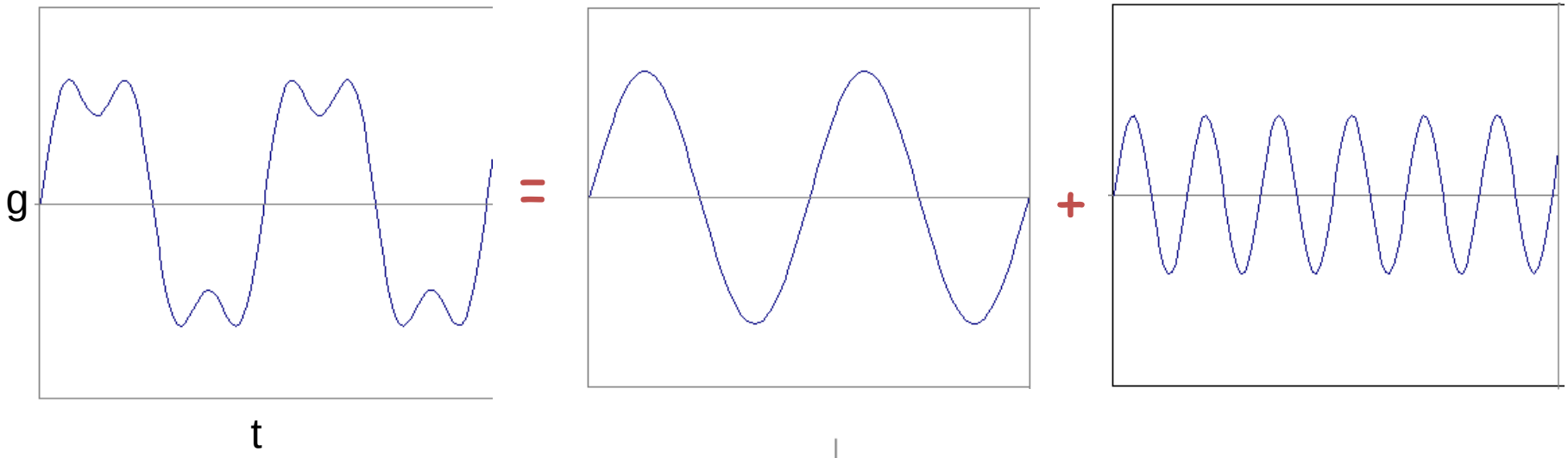
Our building block:

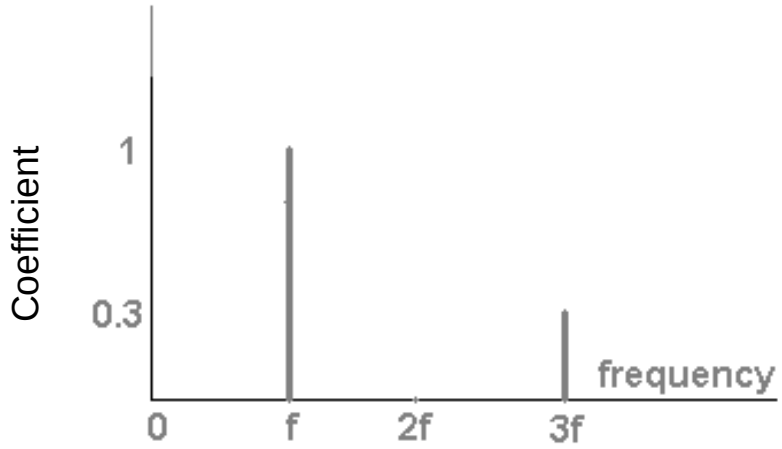
$$\sum_{n=1}^{\infty} (a_n \cos(nt) + b_n \sin(nt))$$

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

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

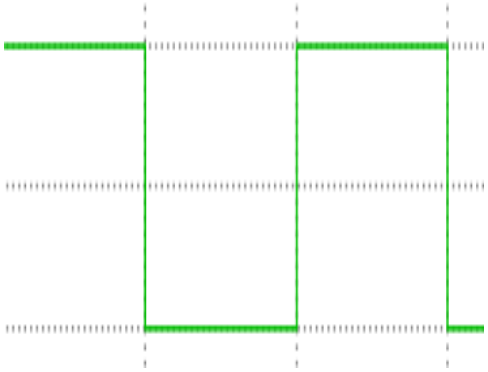
$$g(t) = (1)\sin(2\pi f\, t) + (1/3)\sin(2\pi(3f)\, t)$$



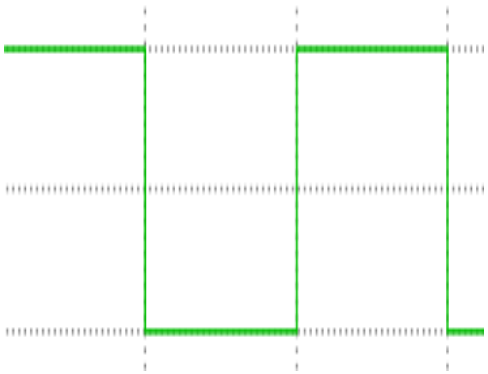


Slides: Efros

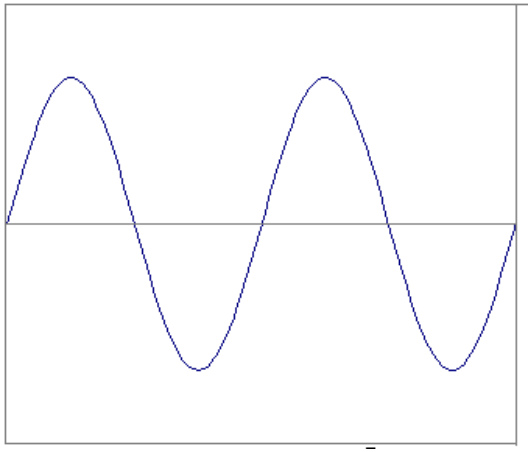
Square wave spectra



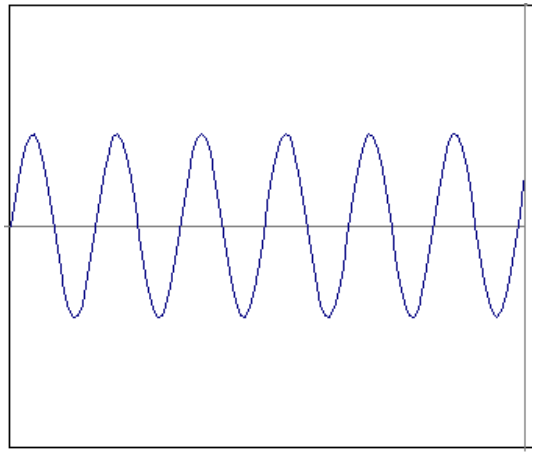
Square wave spectra



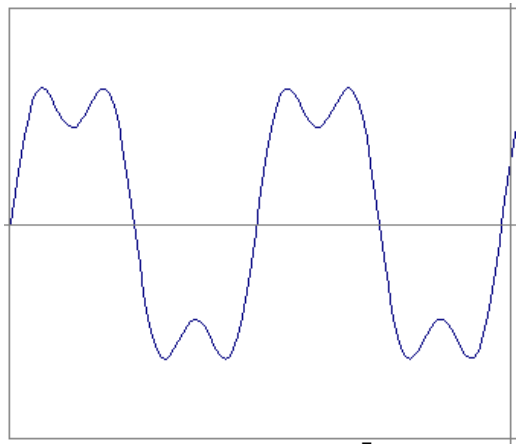
=



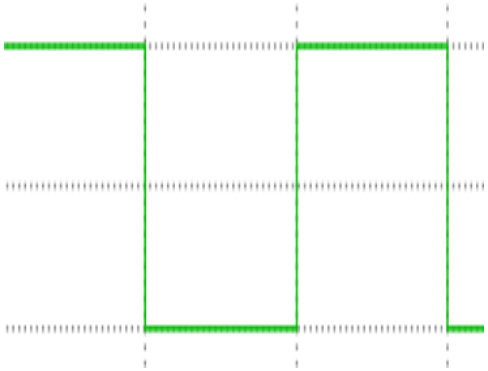
+



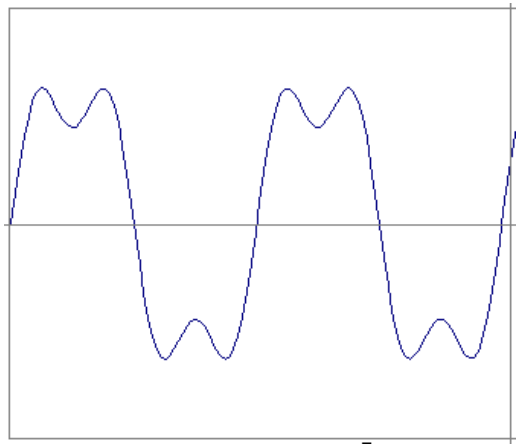
=



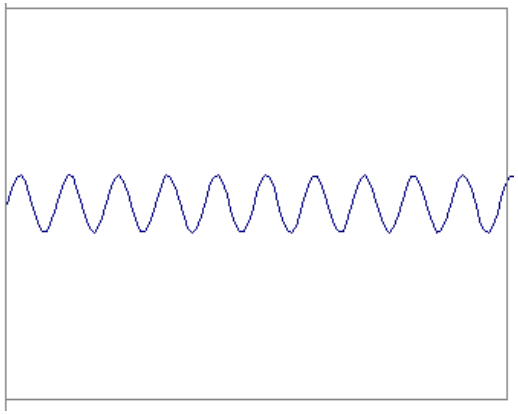
Square wave spectra



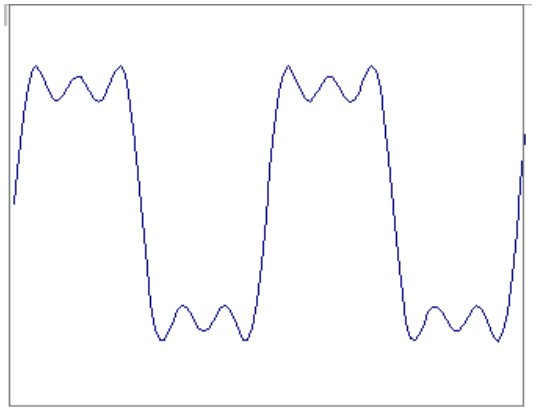
=



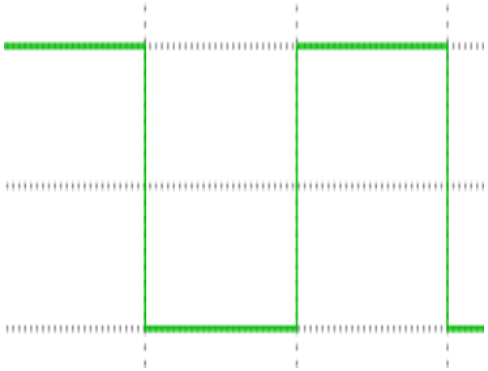
+

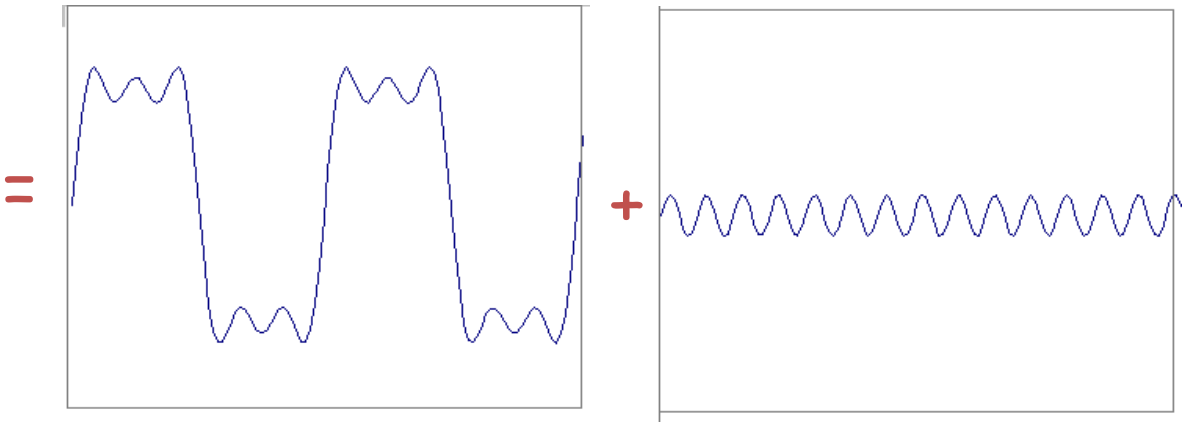


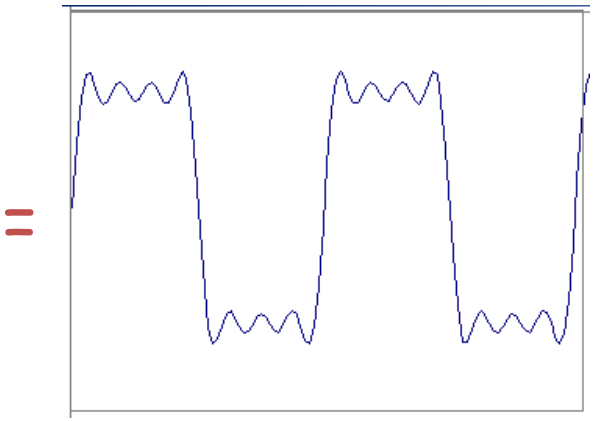
=



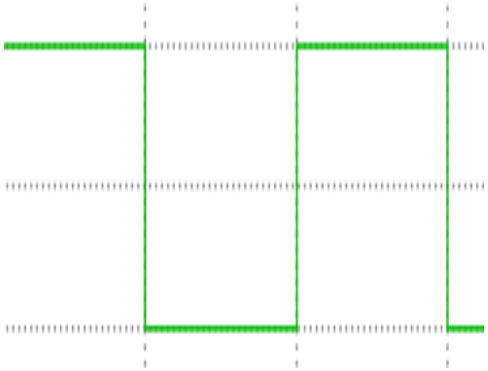
Square wave spectra

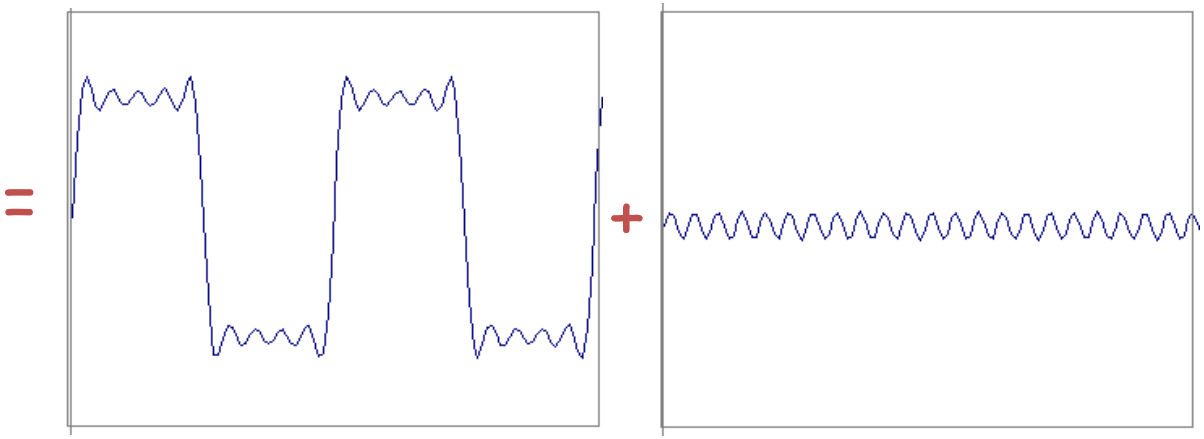


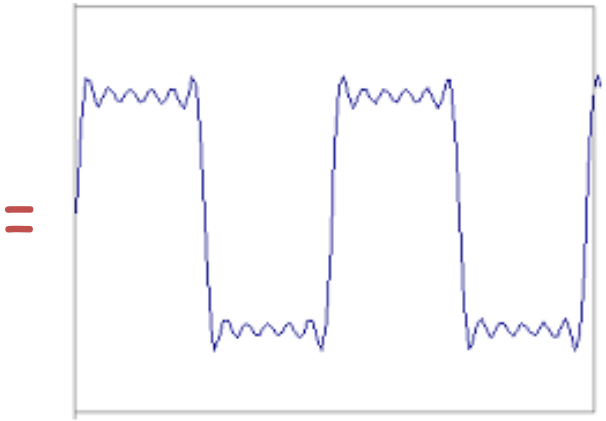




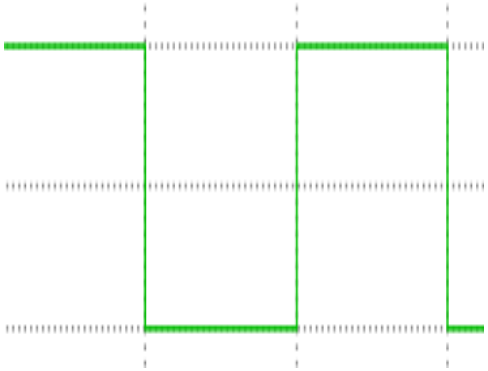
Square wave spectra

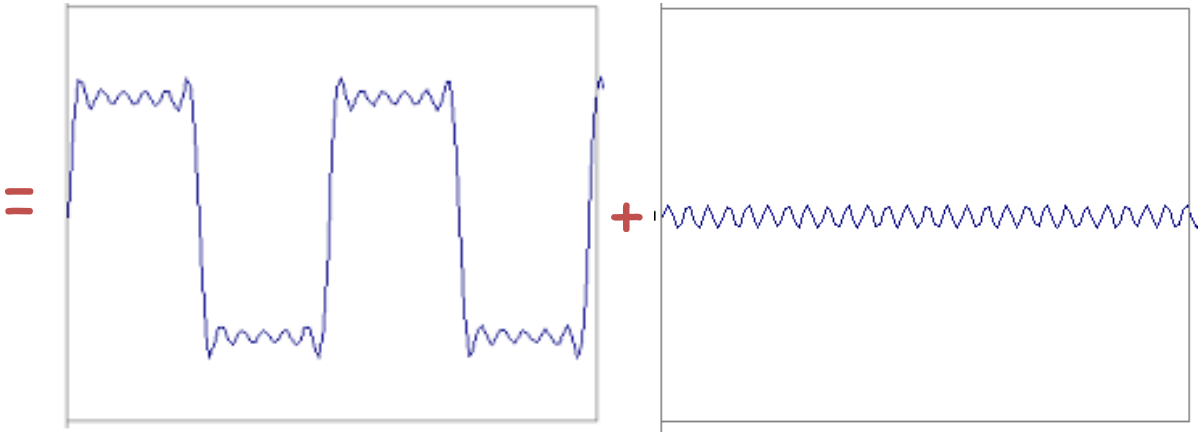


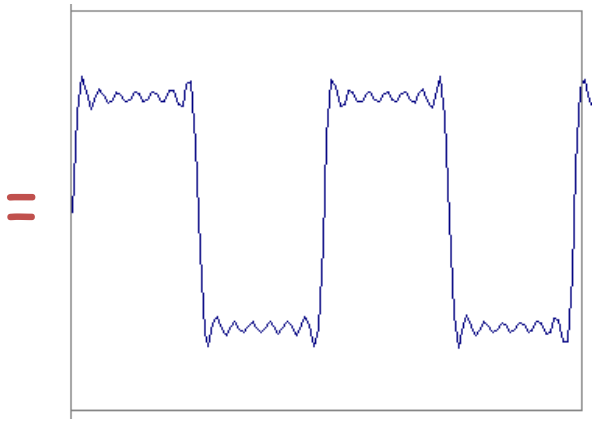




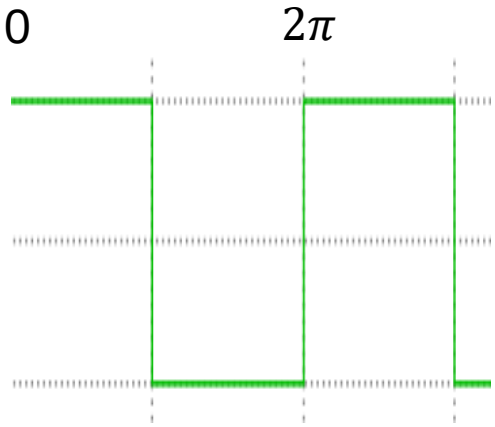
Square wave spectra







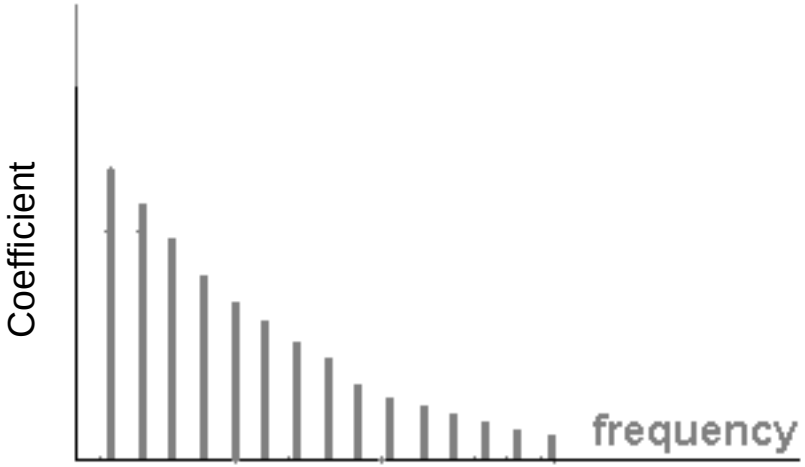
Square wave spectra



=

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

For periodic signals defined over [0, 2π]



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.*

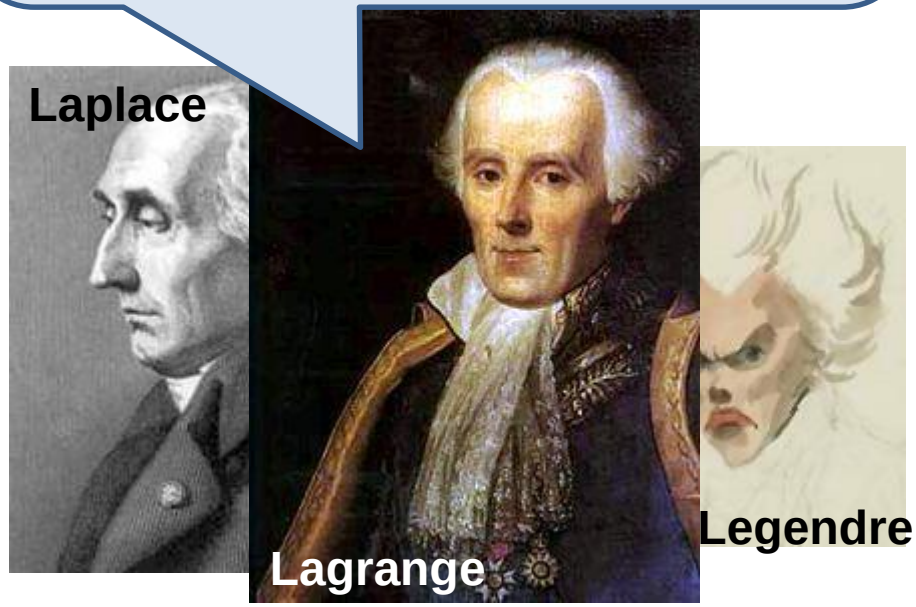
Don't believe it?

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

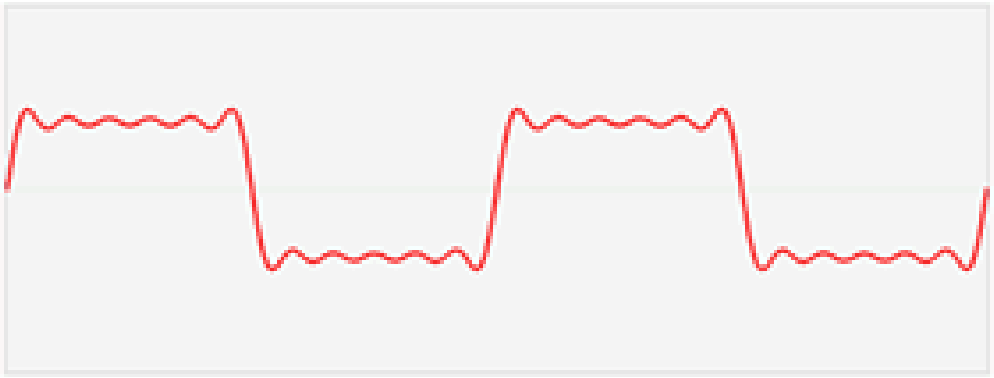
But it's (mostly) true!

- Called Fourier Series
- *Applies to periodic signals*

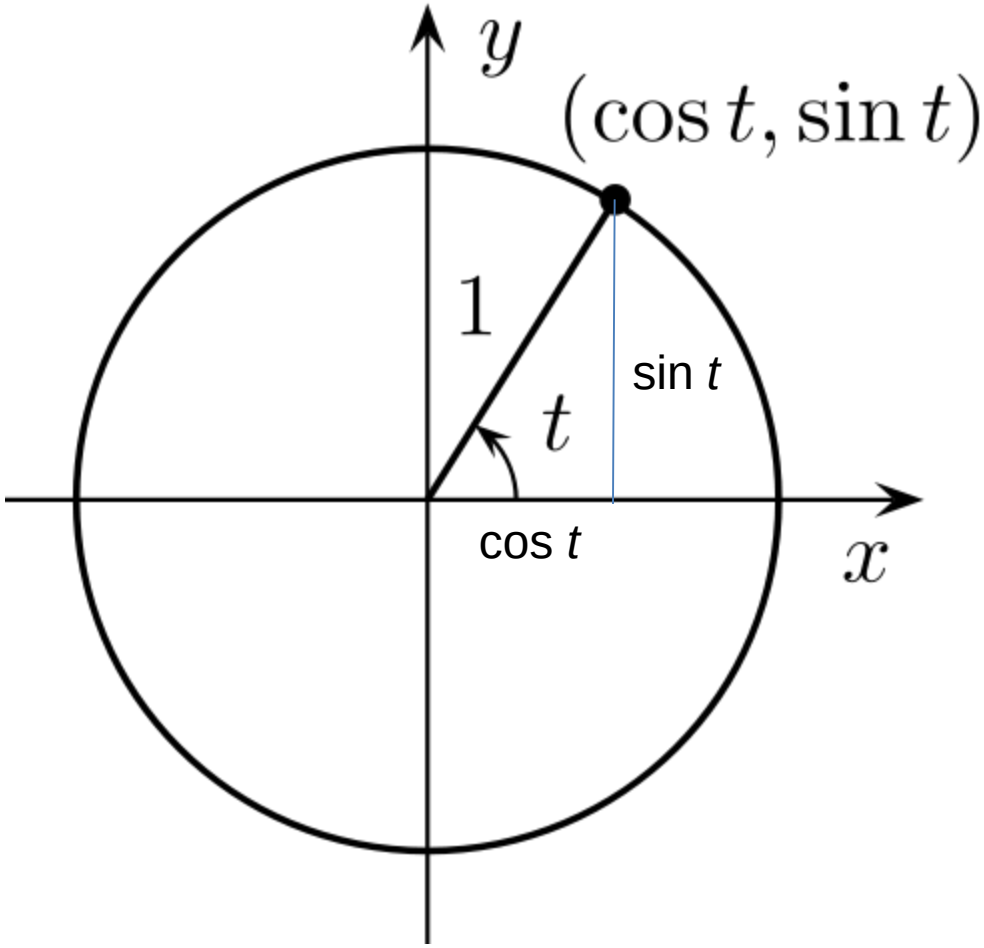
...the manner in which the author arrives at these equations is not exempt of difficulties and...his analysis to integrate them still leaves something to be desired on the score of generality and even rigour.



Reviewer #2

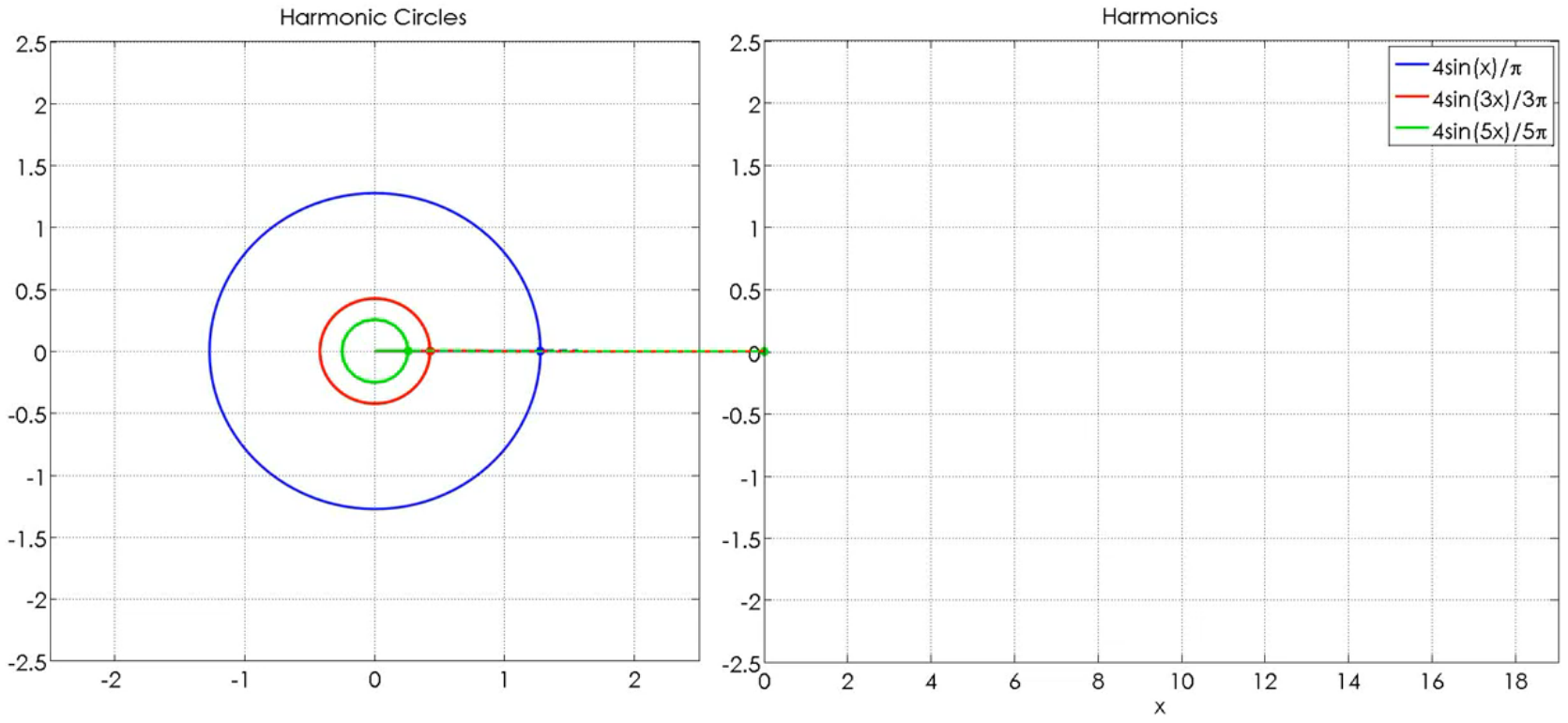


Sine/cosine and circle

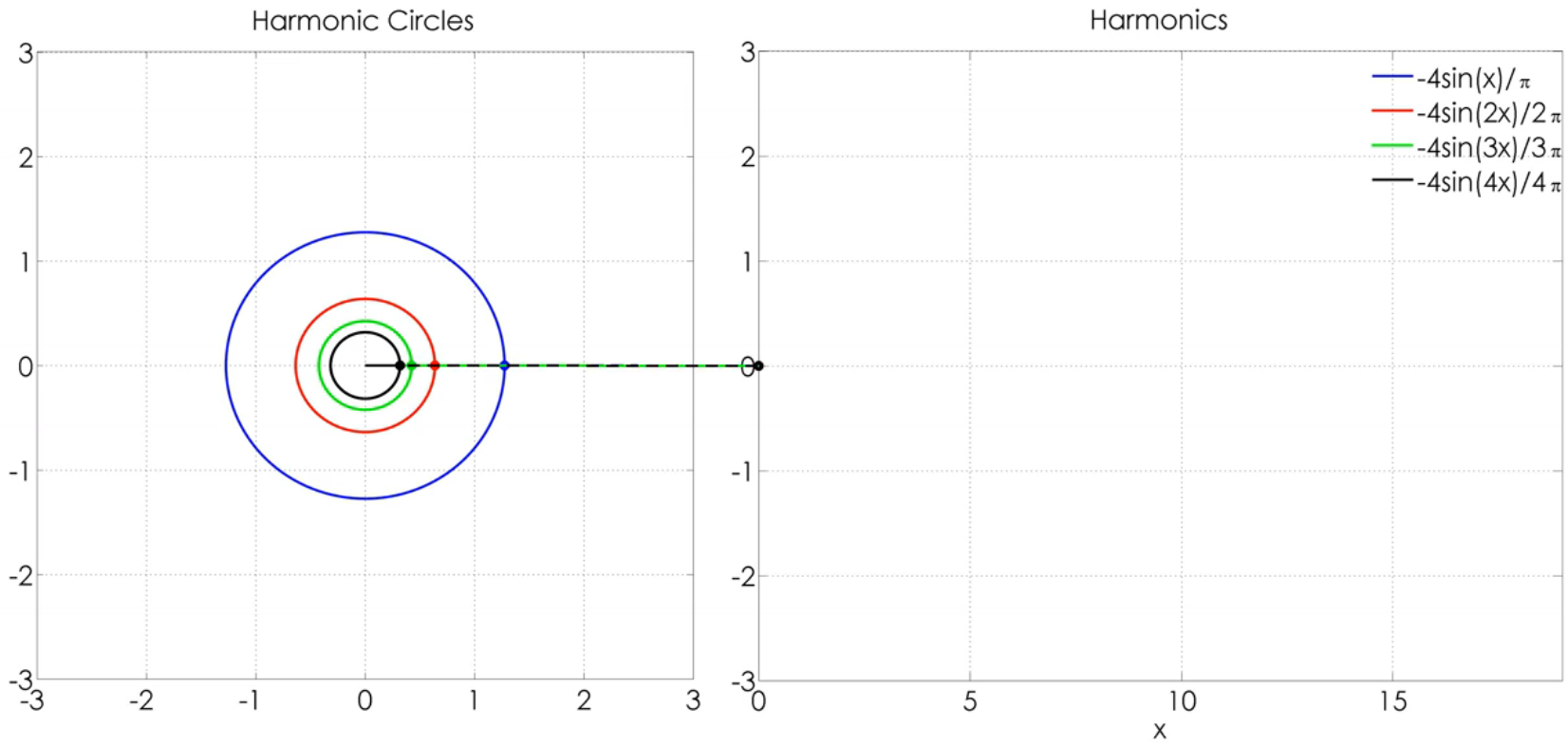


Wikipedia – Unit Circle

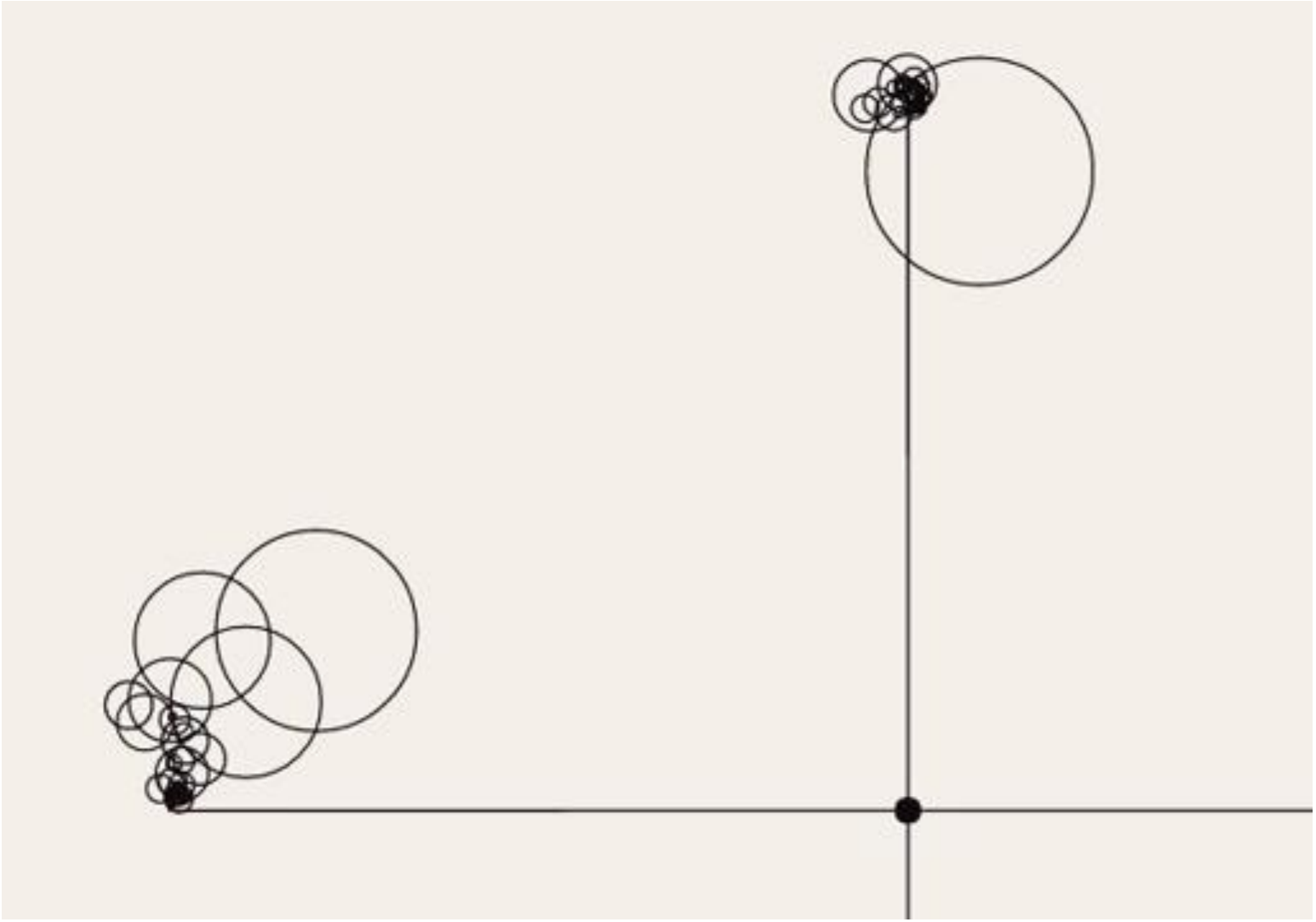
Square wave (approx.)



Sawtooth wave (approx.)



One series in each of x and y



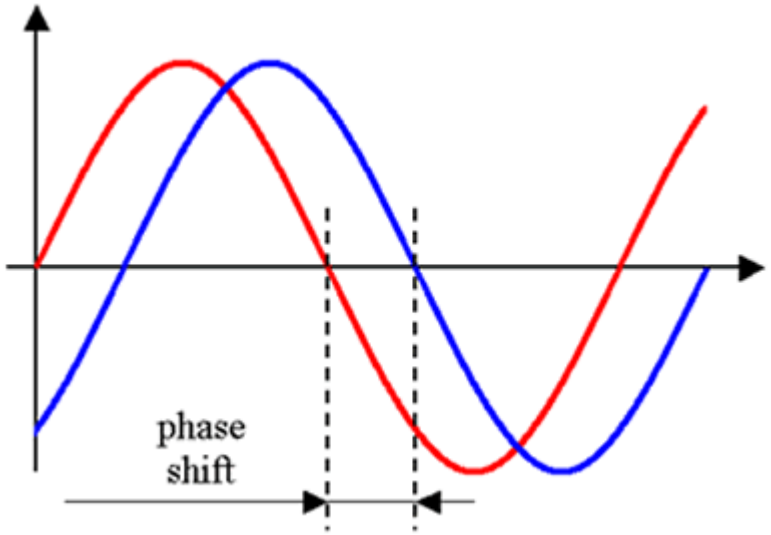
Generative Art, じゃがりきん, Video, 2018

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]

Amplitude-phase form

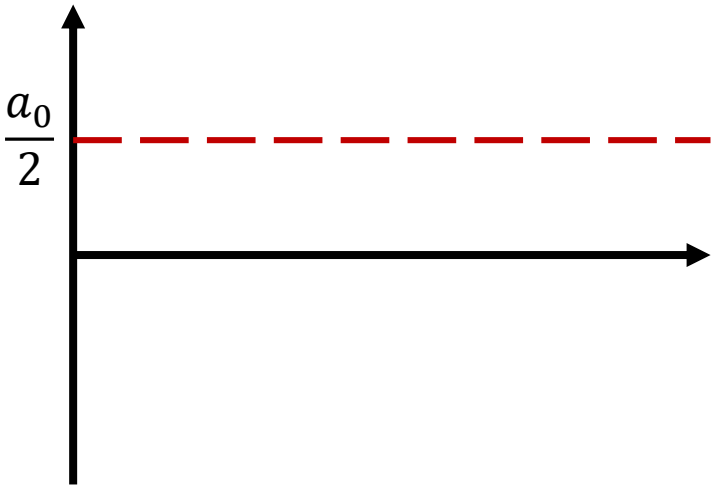
Add component of infinite frequency

= mean of signal over period

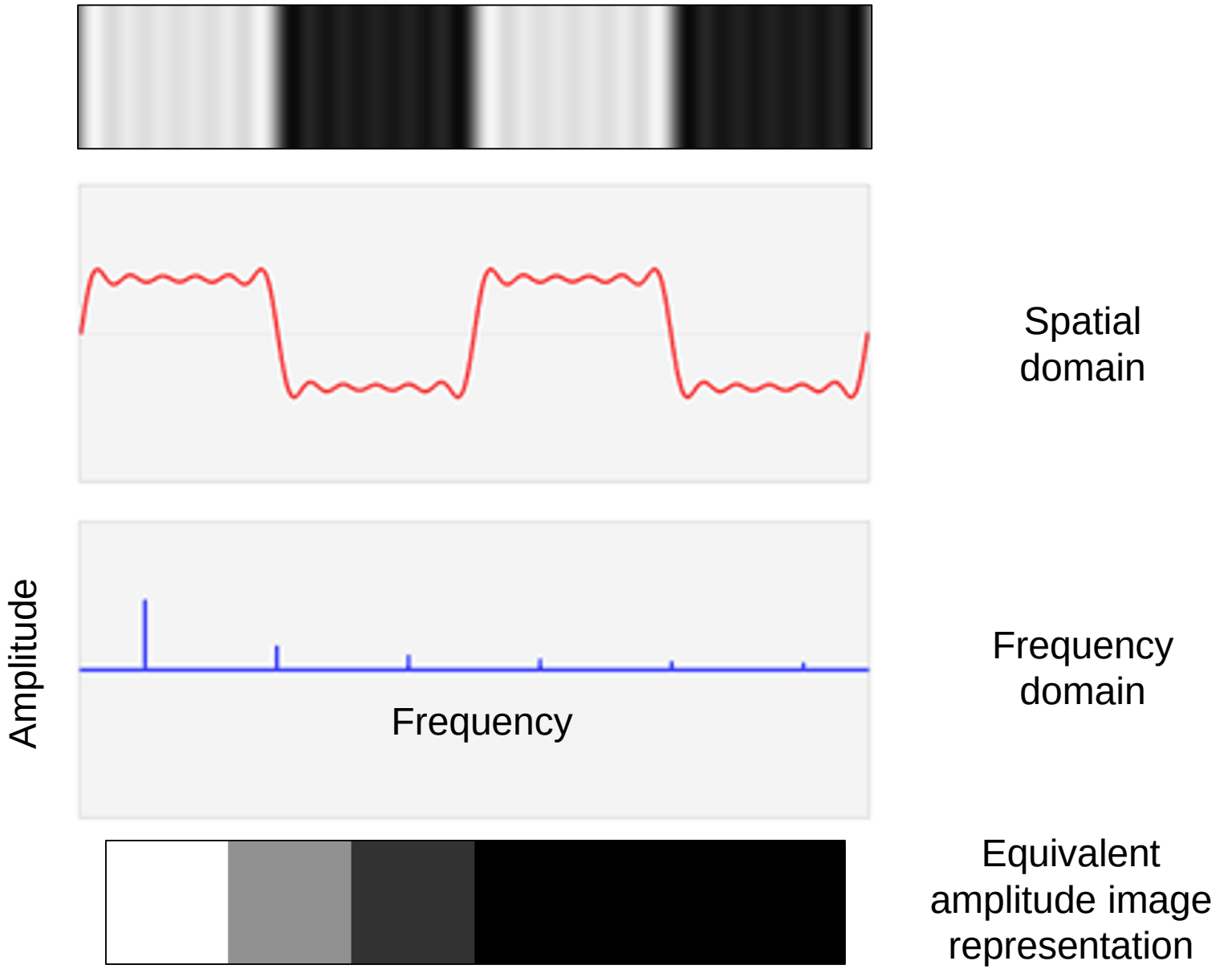
= value around which signal fluctuates

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

Average of signal over period



Electronics signal processing calls this the ‘DC offset’

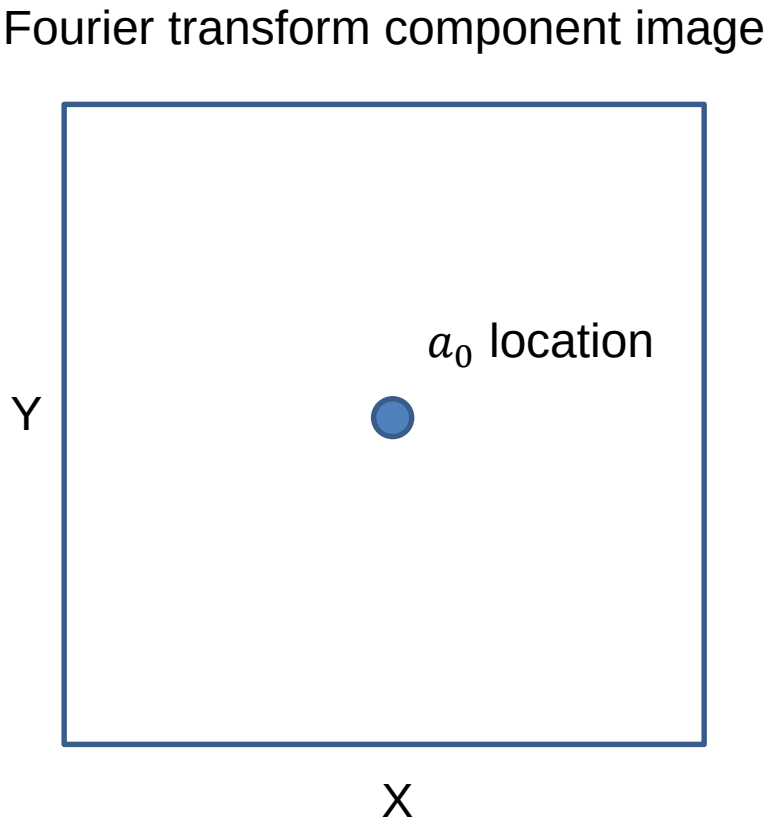


Wikipedia – Fourier transform

How to read Fourier transform images

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

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



How to read Fourier transform images

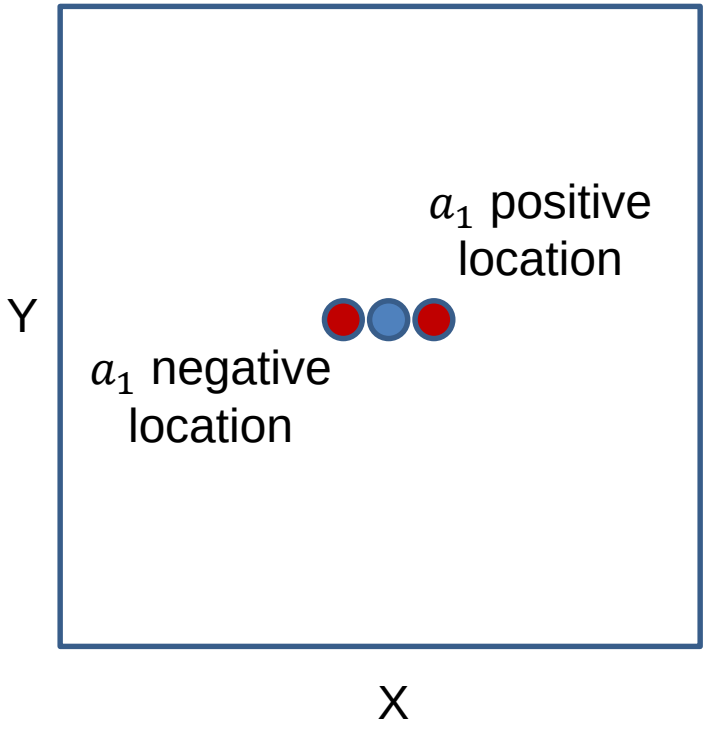
Image is rotationally symmetric about center because of negative frequencies

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

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

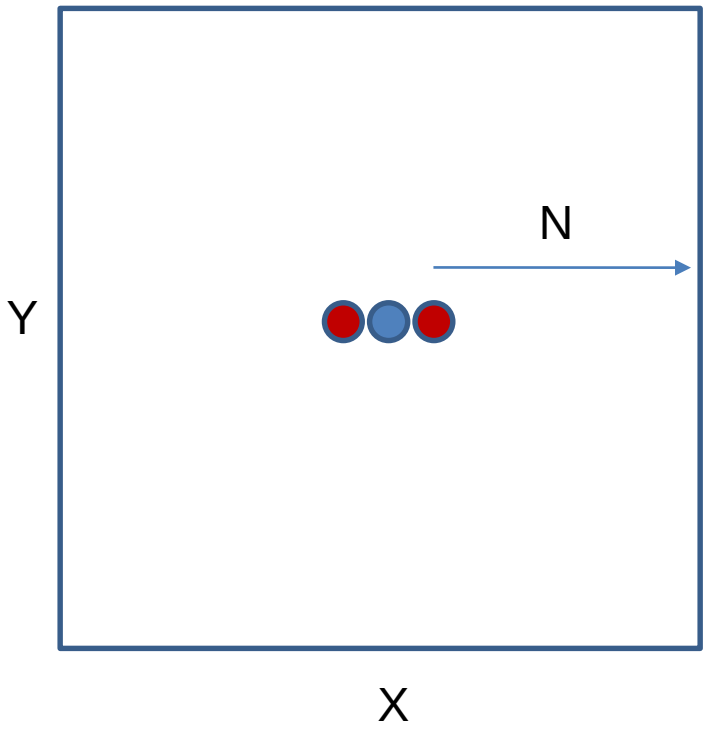


How to read Fourier transform images

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

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

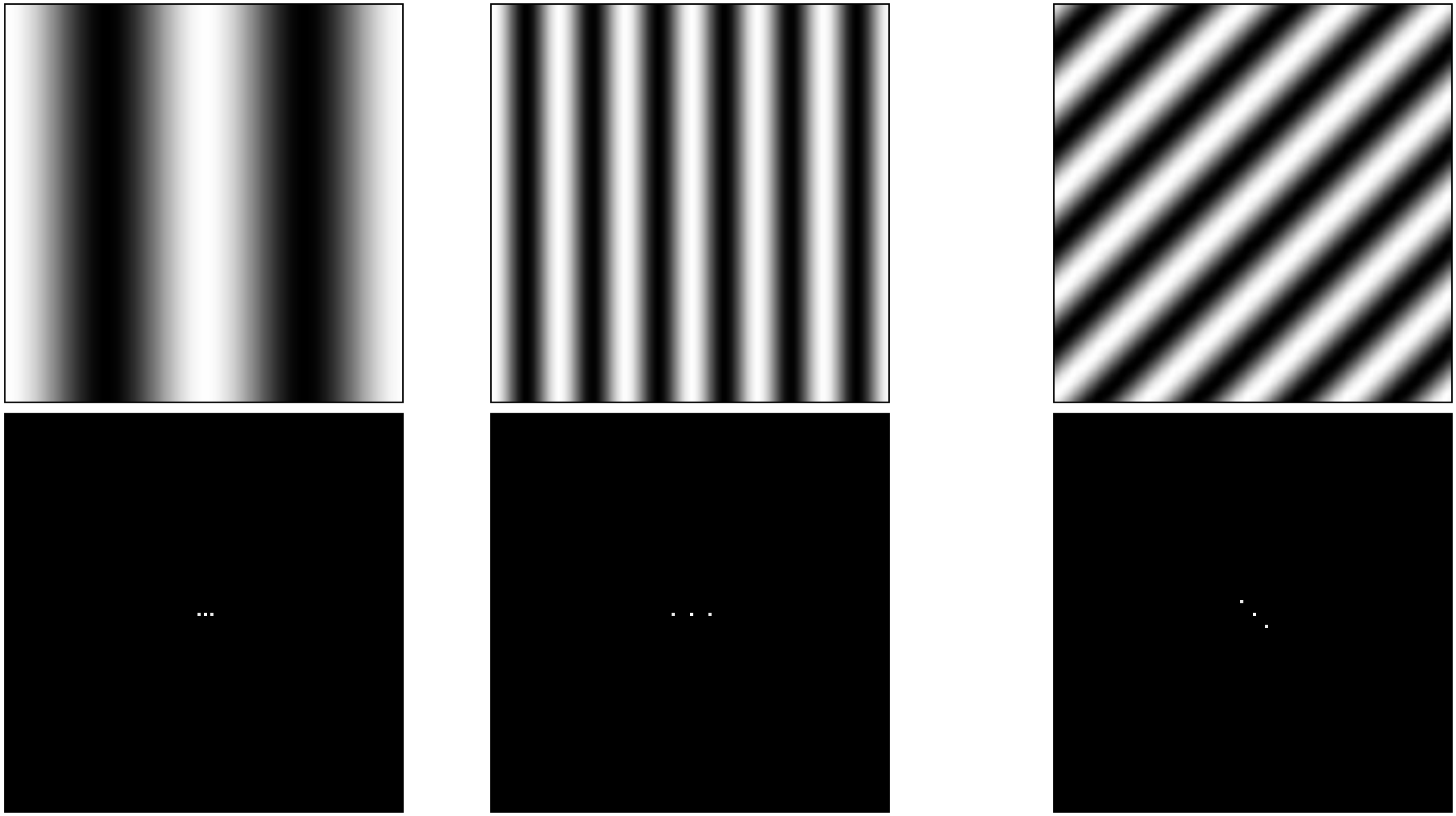
Fourier transform component image



Nyquist frequency is half the sampling rate of the signal. Sampling rate is size of X (or Y), so Fourier transform images are (2X+1,2Y+1).

Fourier analysis in images

Spatial domain images

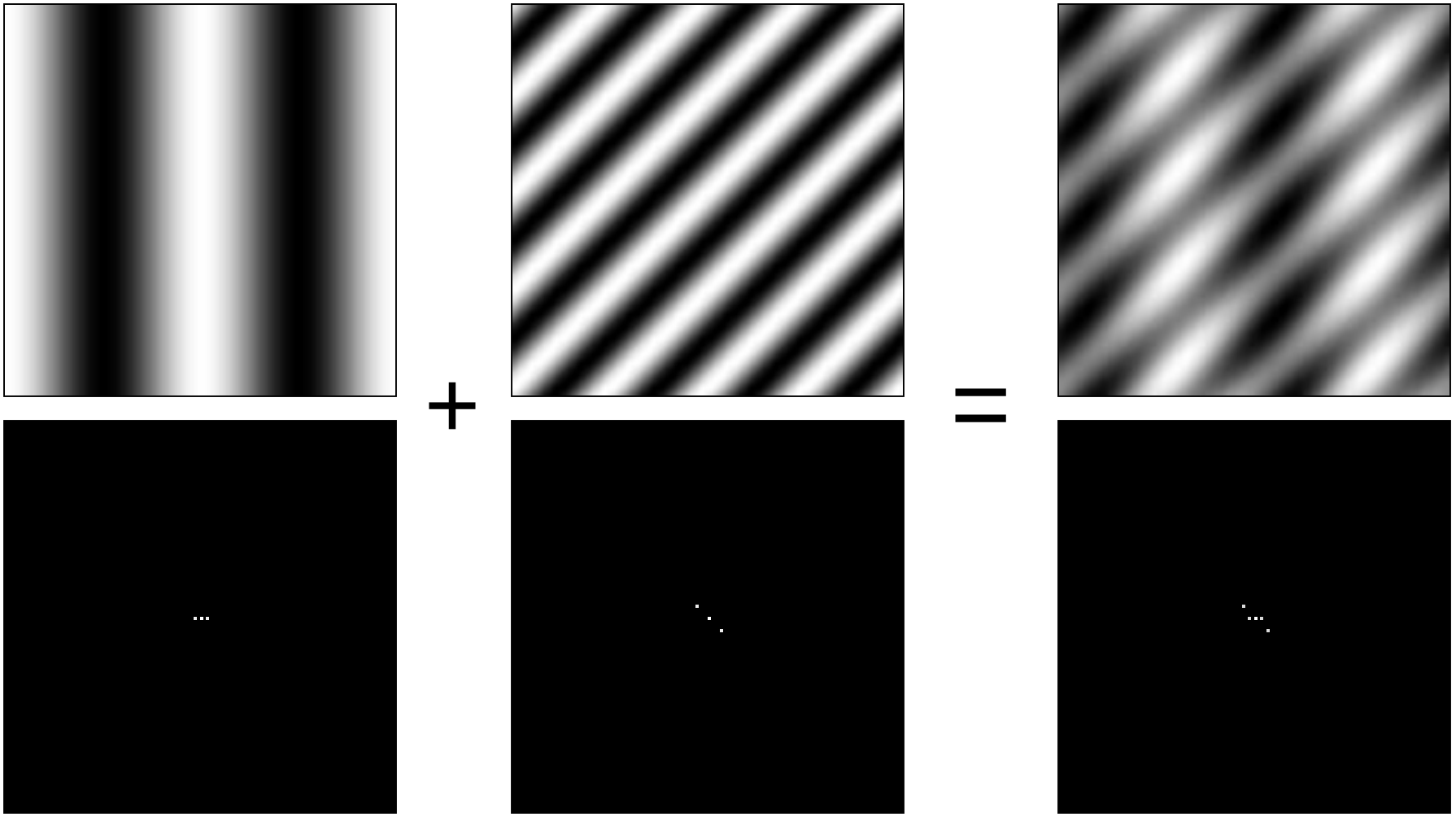


Fourier decomposition amplitude images

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



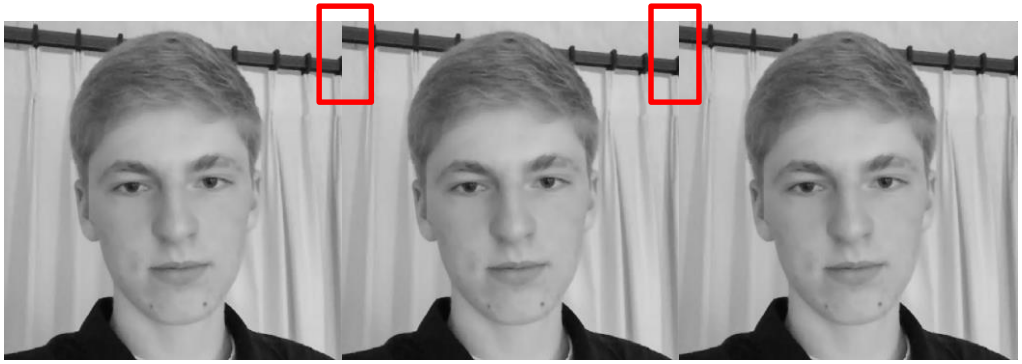
Fourier decomposition amplitude images

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

Periodicity

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

- ‘Fake’ signal is image wrapped around

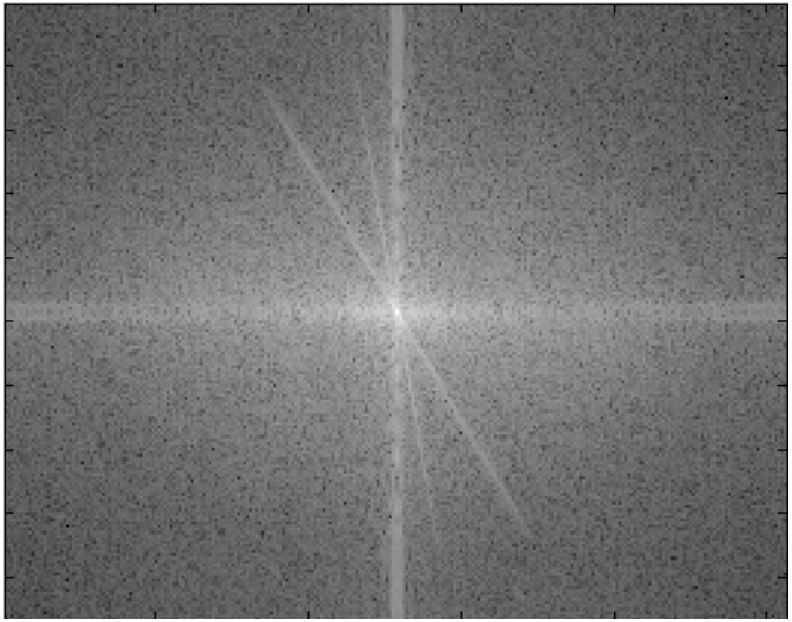


Convention is to pad with zeros to reduce effect.

Natural image

Natural image

Fourier decomposition
Amplitude image

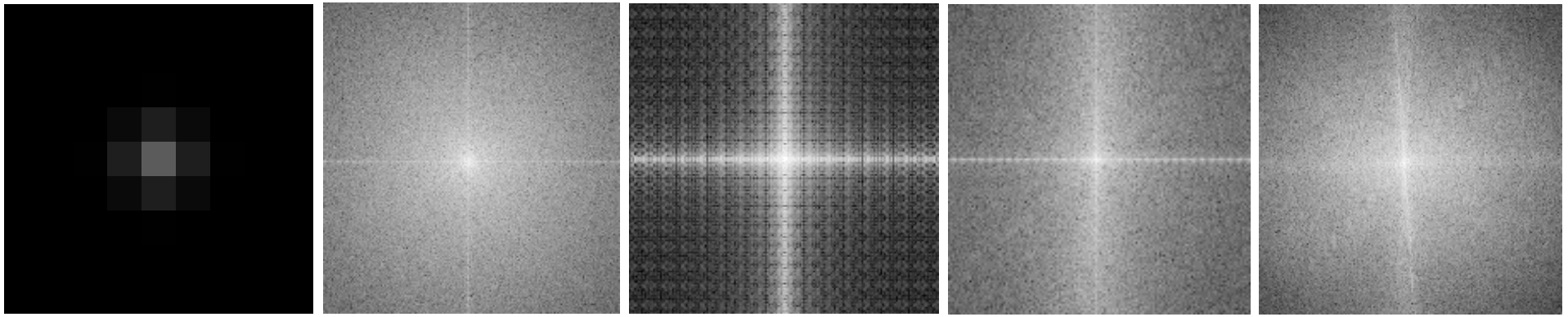


What does it mean to be at pixel x,y?
What does it mean to be more or less bright in the Fourier decomposition image?

Think-Pair-Share

Match the spatial domain image to the Fourier amplitude image

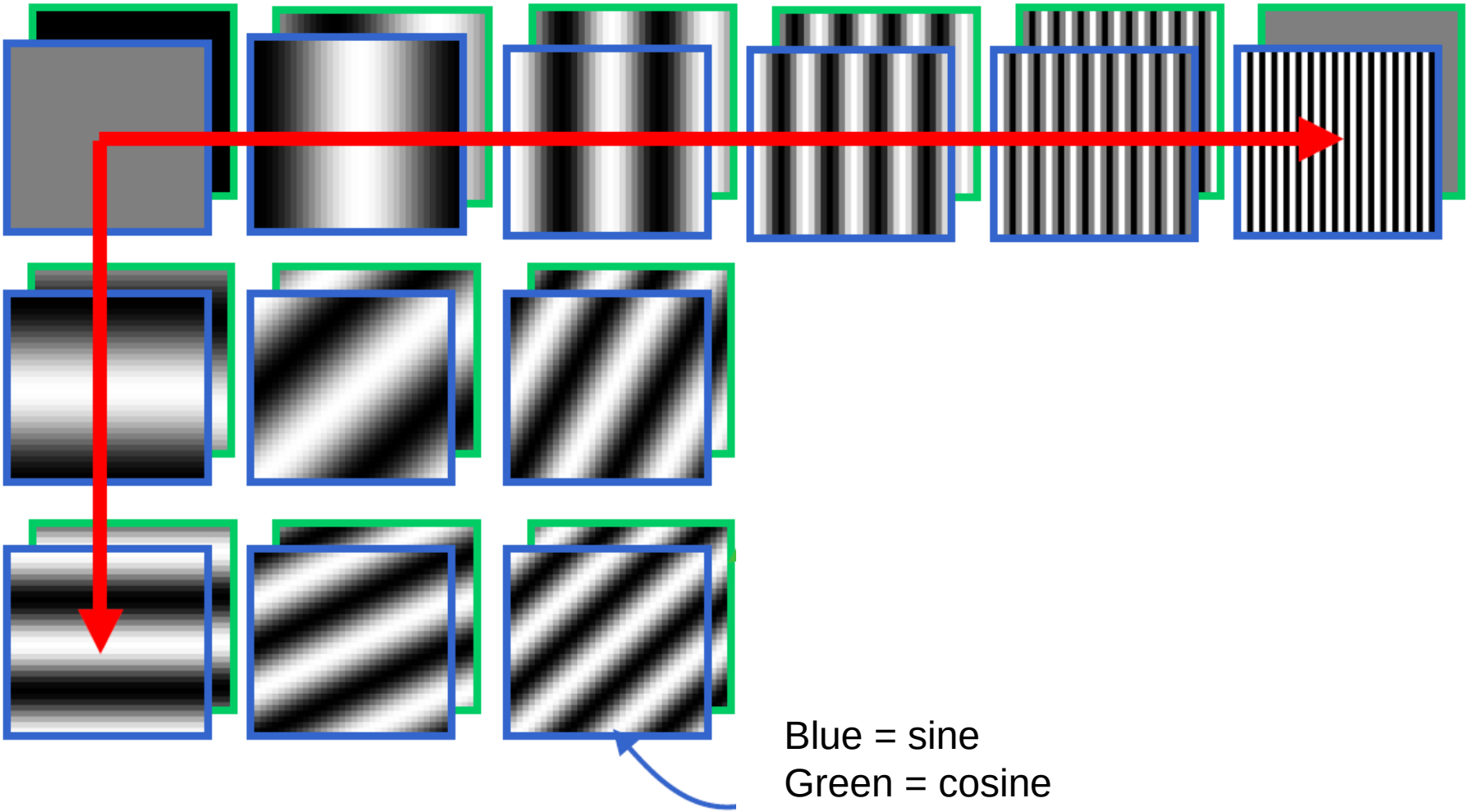
1 2 3 4 5



A B C D E

Fourier Bases

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

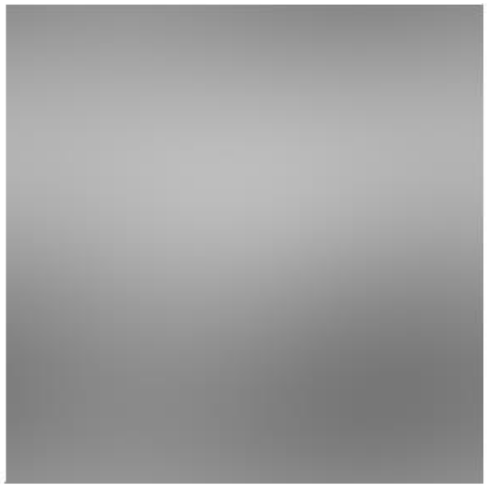


This change of basis is the Fourier Transform

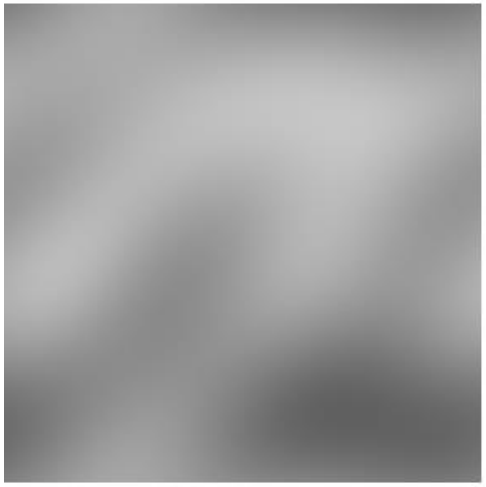
Basis reconstruction

Full image

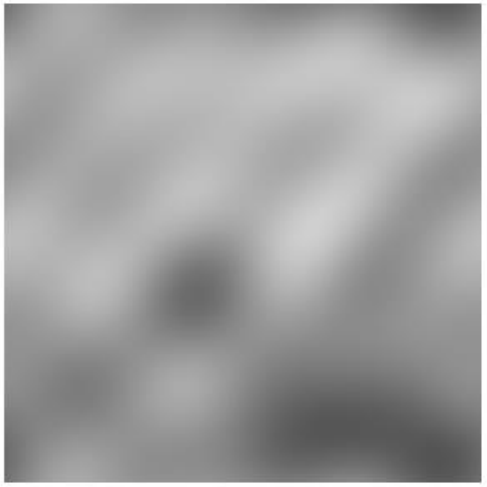
First 1 basis fn



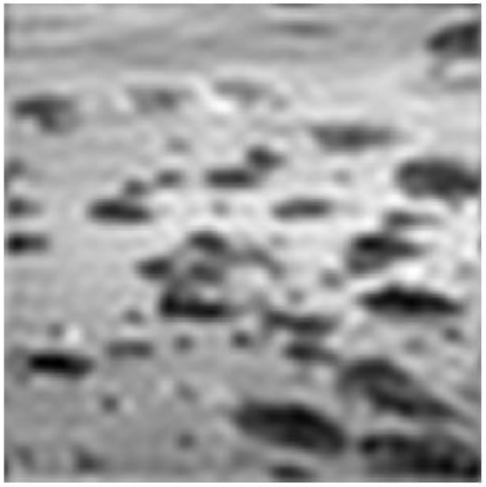
First 4 basis fns



First 9 basis fns



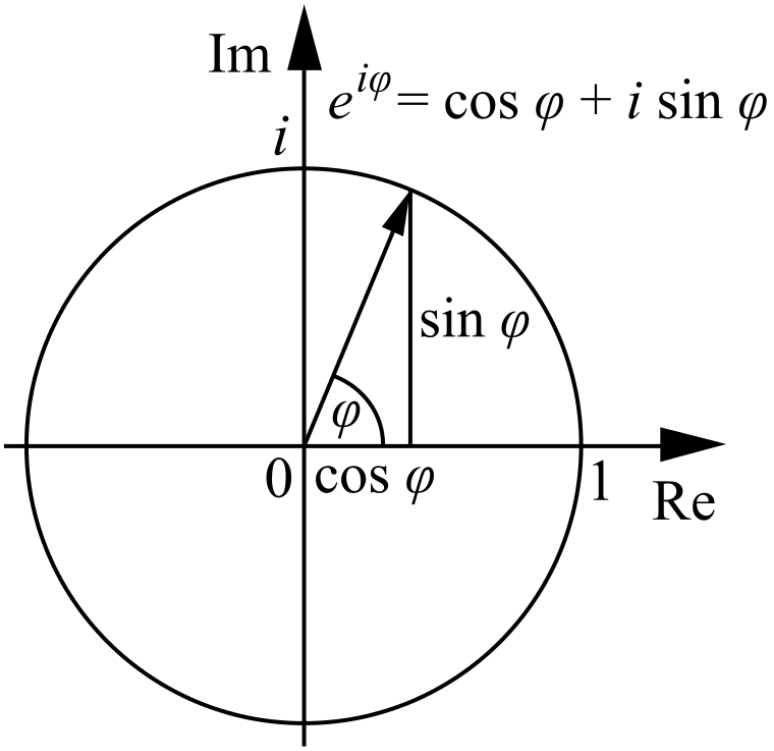
First 16 basis fns



First 400 basis fns

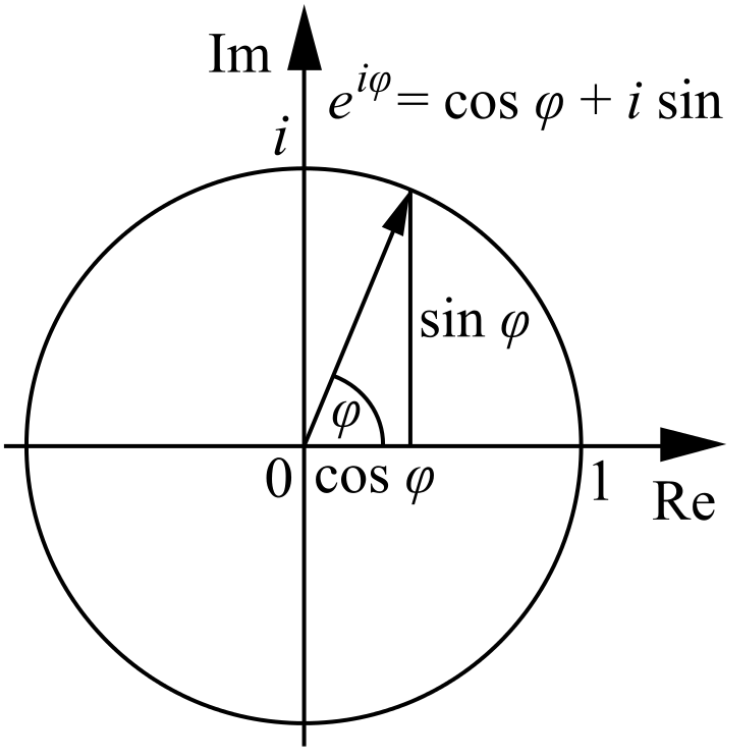
Fourier Transform

- 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:

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

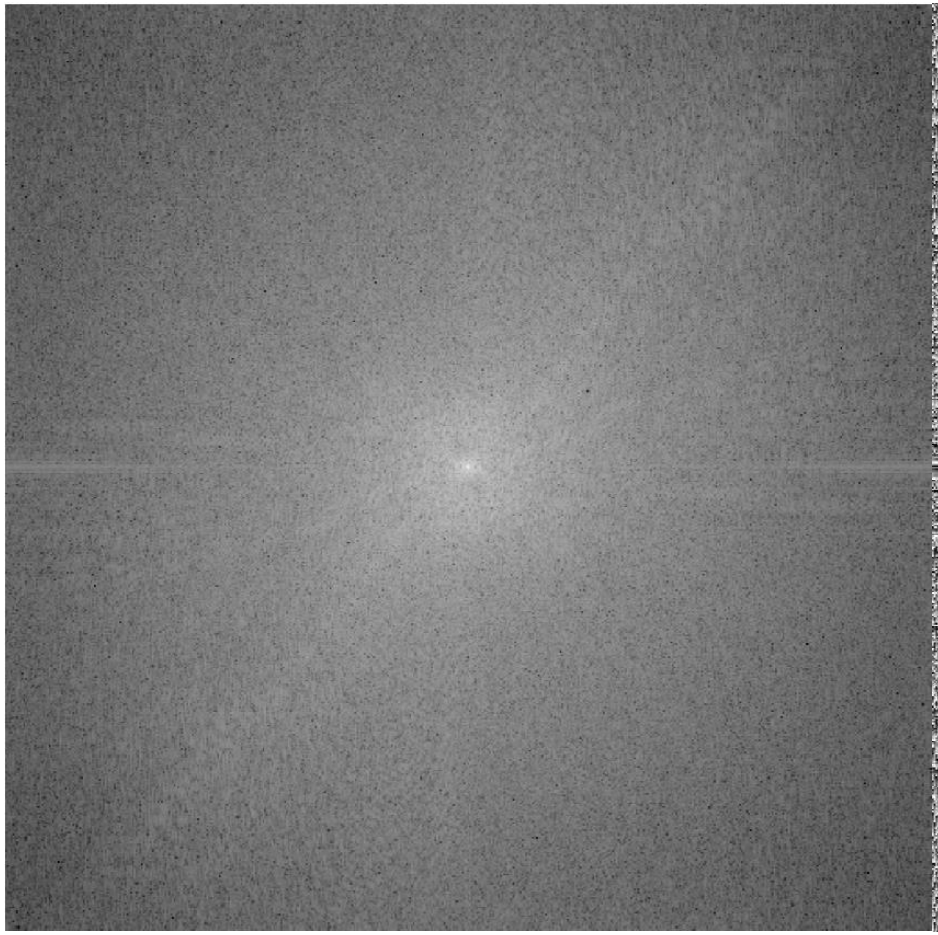
Phase encodes spatial information (indirectly):

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

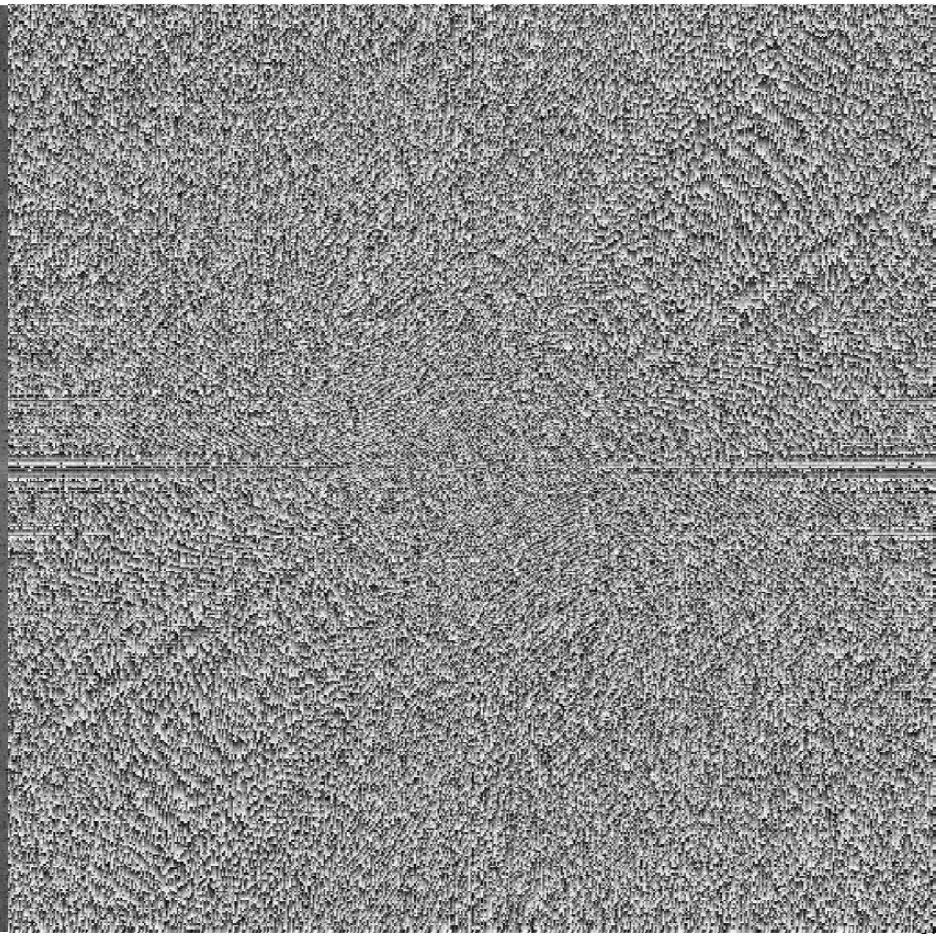
What about phase?

What about phase?

Amplitude



Phase

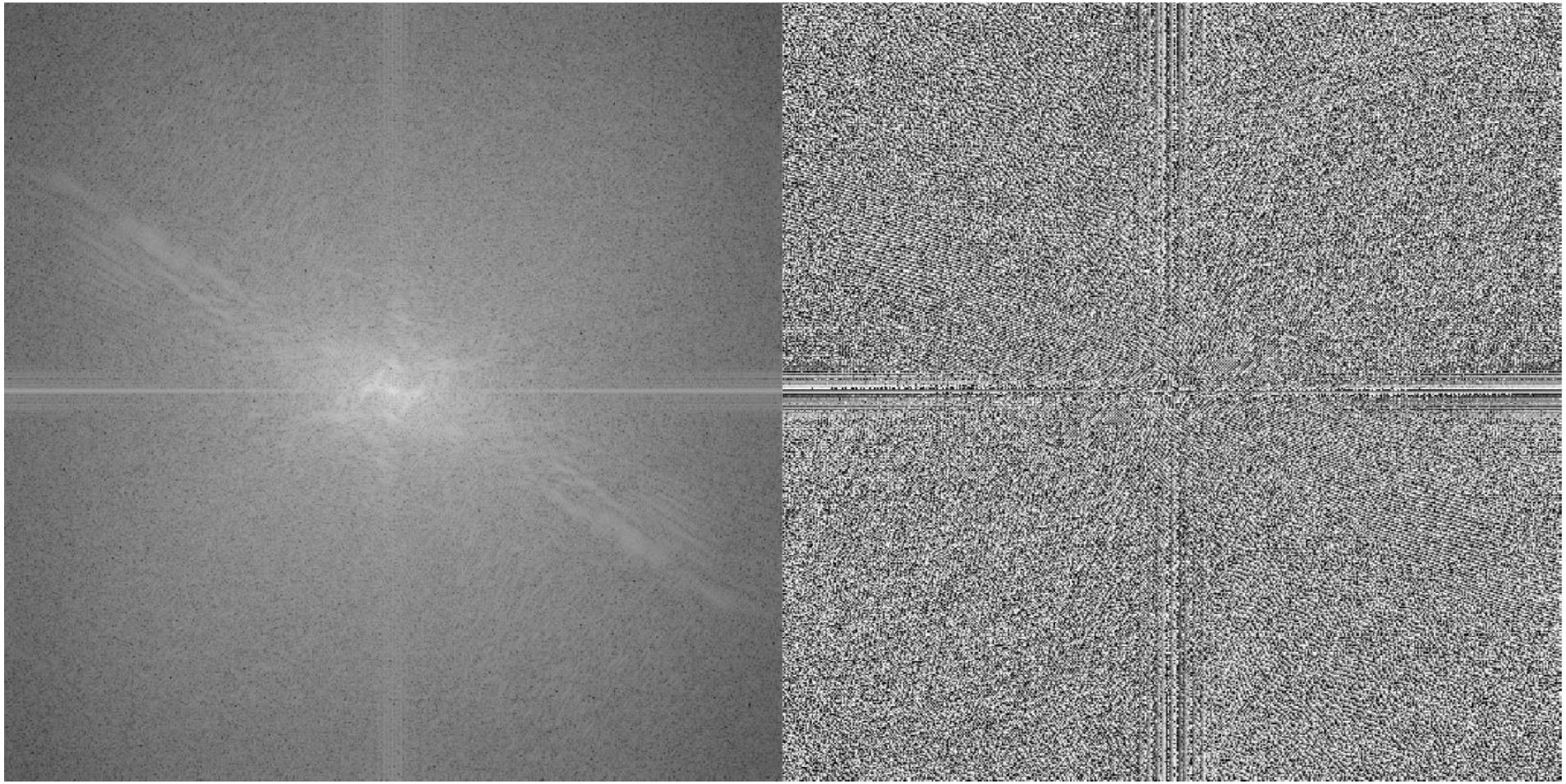


What about phase?

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.”

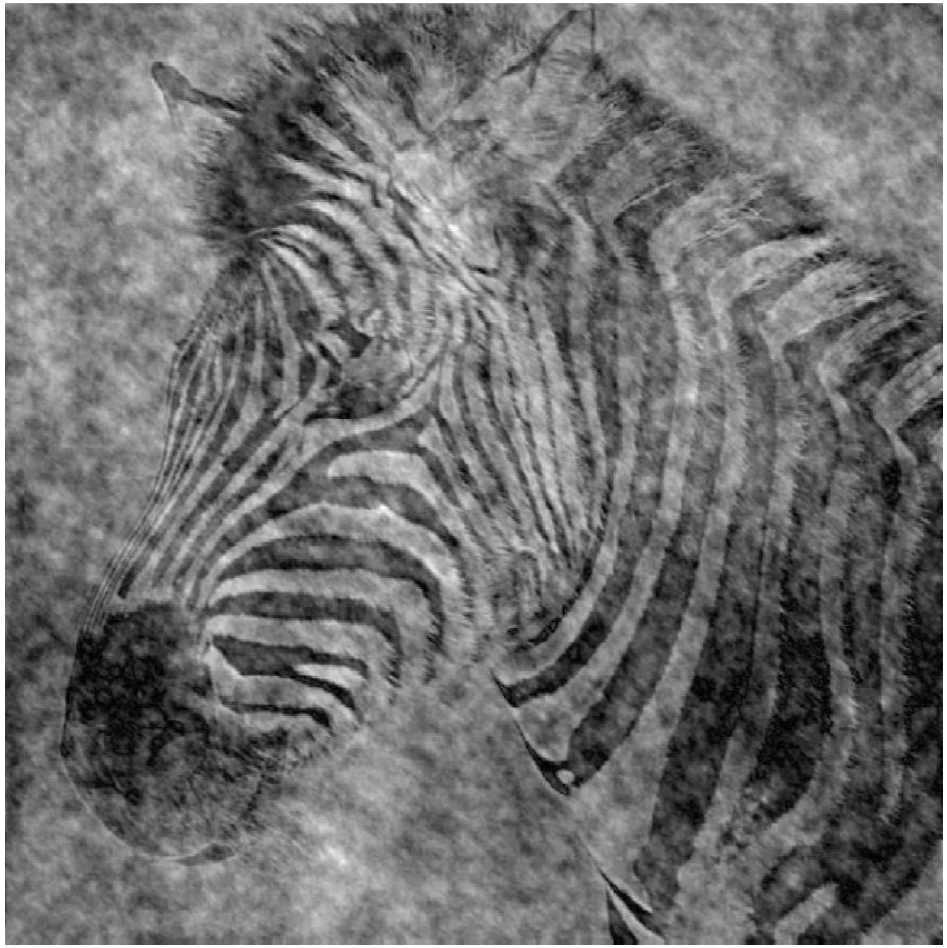
- https://www.cs.unm.edu/~brayer/vision/fourier.html

Think-Pair-Share

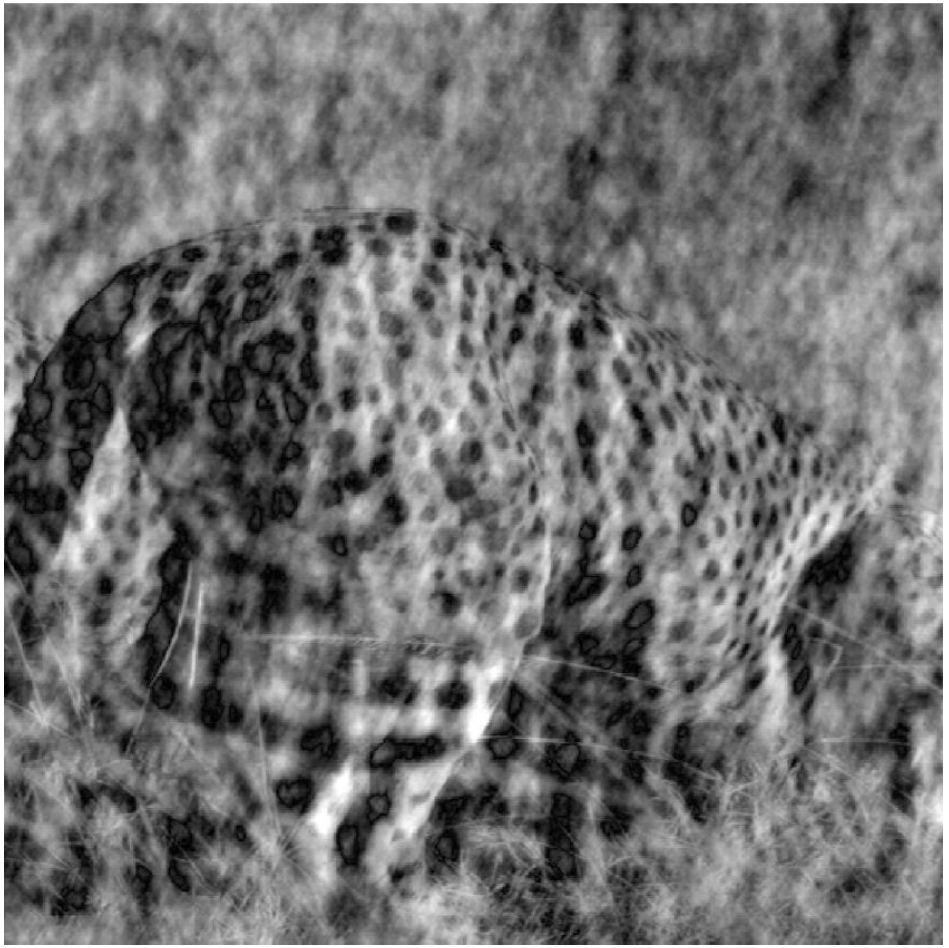
- In Fourier space, where is more of the information that we see in the visual world?
  - Amplitude
  - Phase

Cheebra

Zebra phase, cheetah amplitude



Cheetah phase, zebra amplitude



- 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
  - Not quite clear why

What is the relationship to phase in audio?

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

- ??? : (

Brian Pauw demo

- Live Fourier decomposition images
  - Using FFT2 function

- I hacked it a bit for MATLAB
- http://www.lookingatnothing.com/index.php/archives/991

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

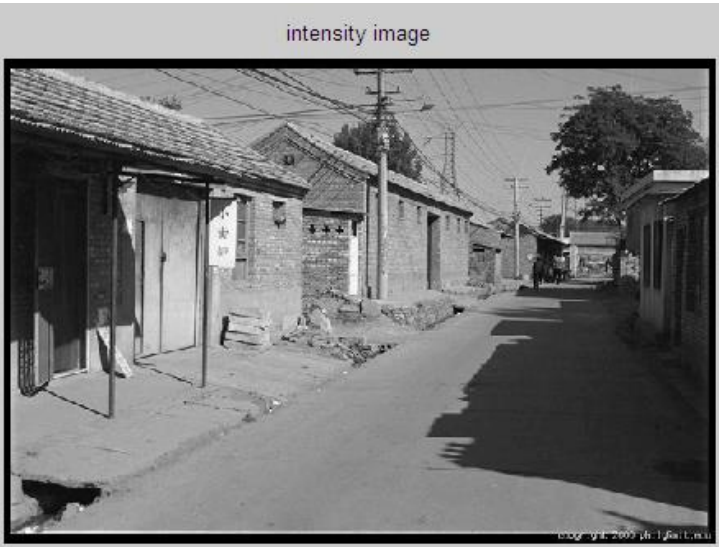
$$\mathrm{F}[g * h] = \mathrm{F}[g]\,\mathrm{F}[h]$$

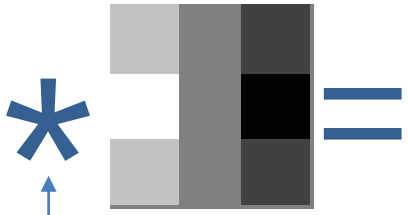
- **Convolution** in spatial domain is equivalent to **multiplication** in frequency domain!

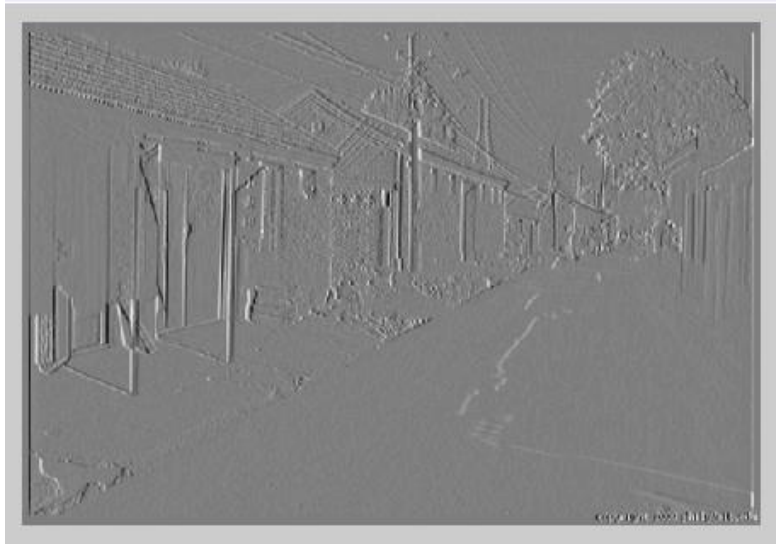
$$g * h = \mathrm{F}^{-1}[\mathrm{F}[g]\,\mathrm{F}[h]]$$

Filtering in spatial domain

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

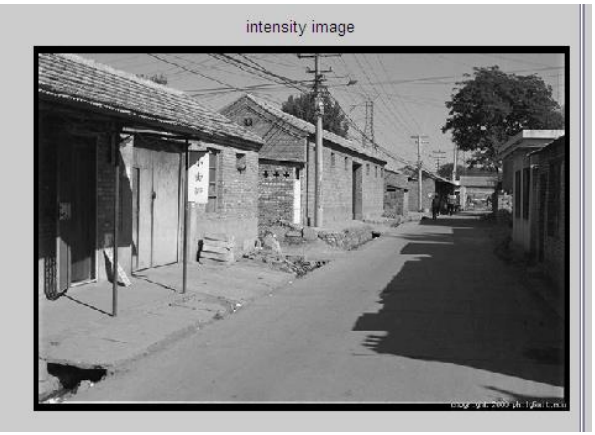






Convolution

Filtering in frequency domain

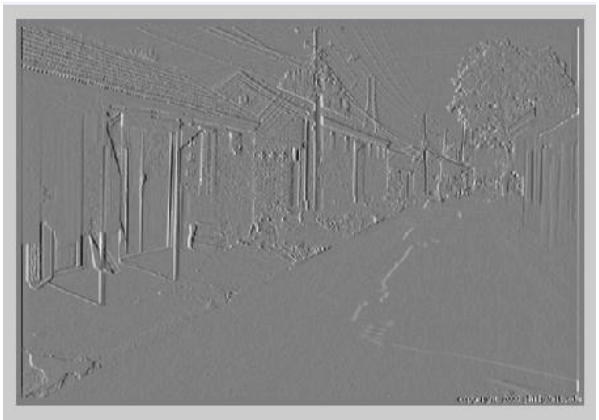


Fourier transform

Element-wise product

Fourier transform

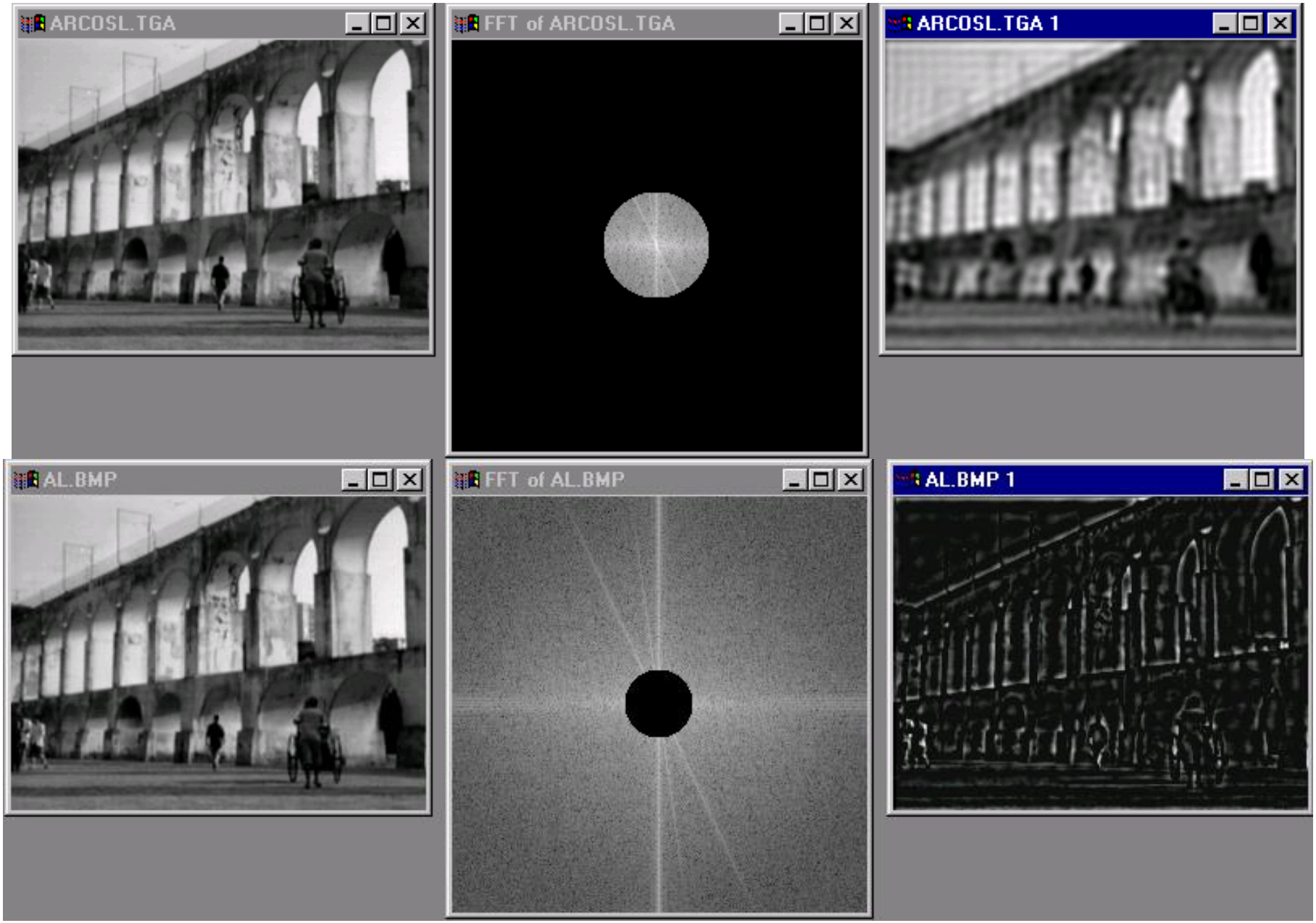
Inverse Fourier transform



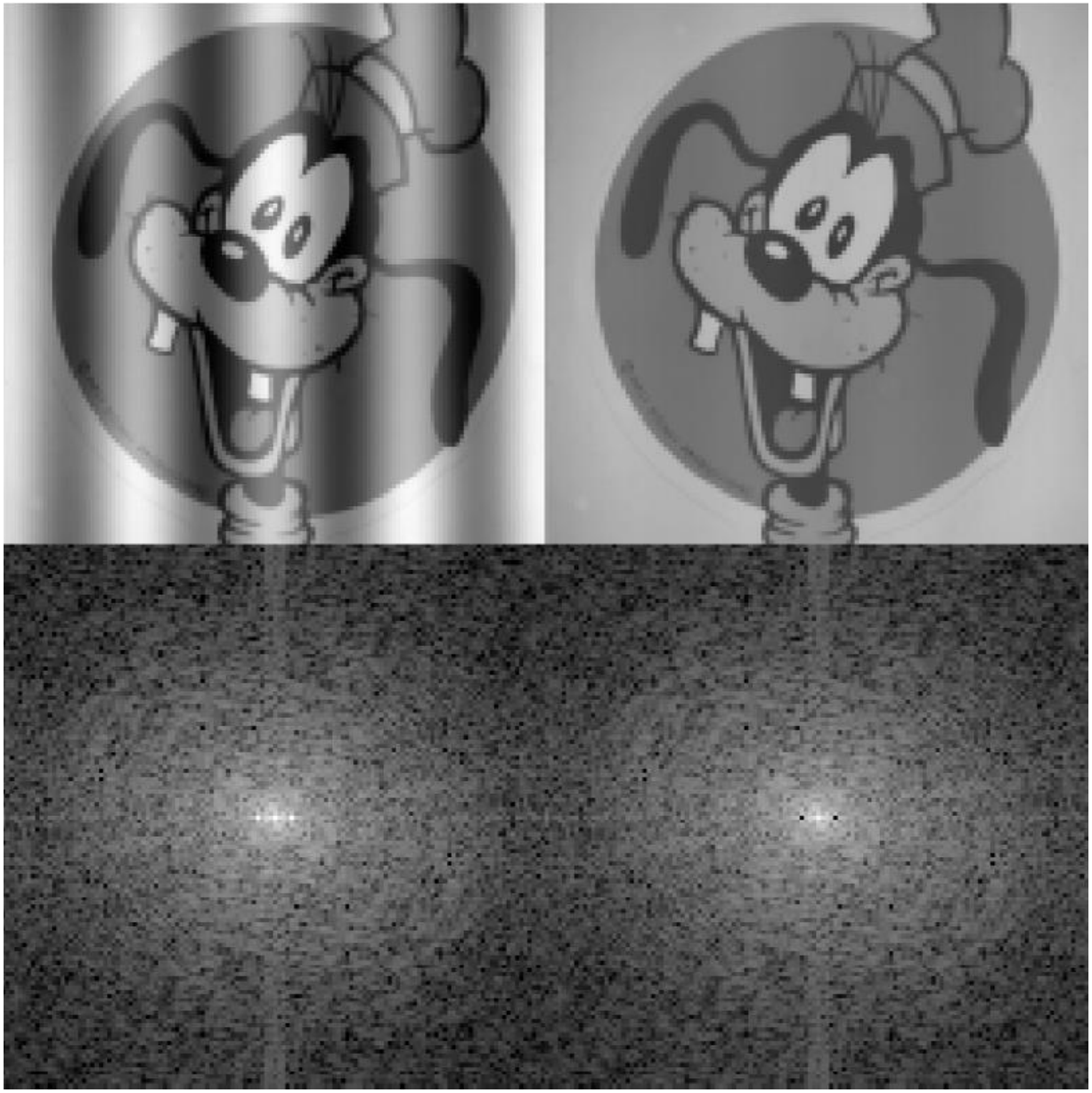
Slide: Hoiem

Now we can edit frequencies!

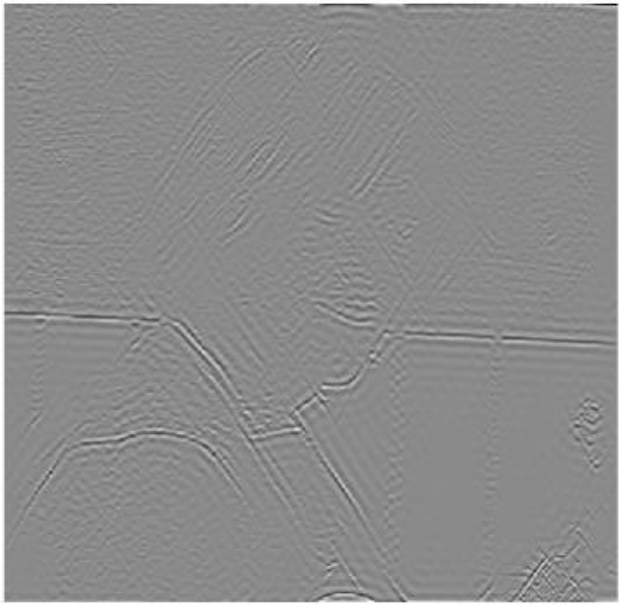
Low and High Pass filtering

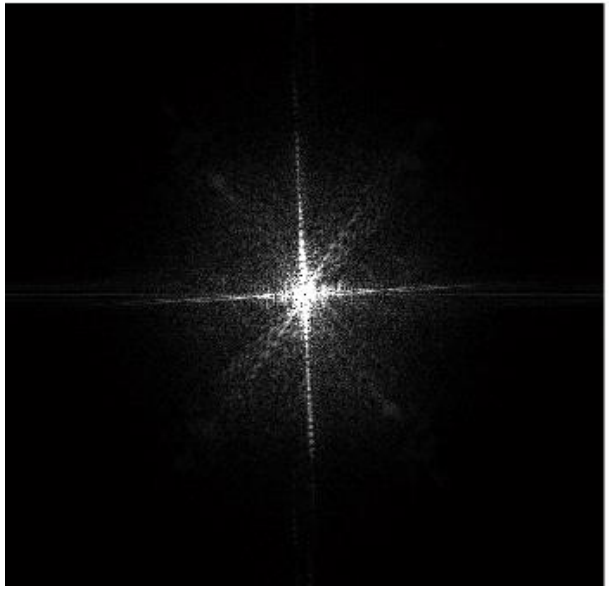


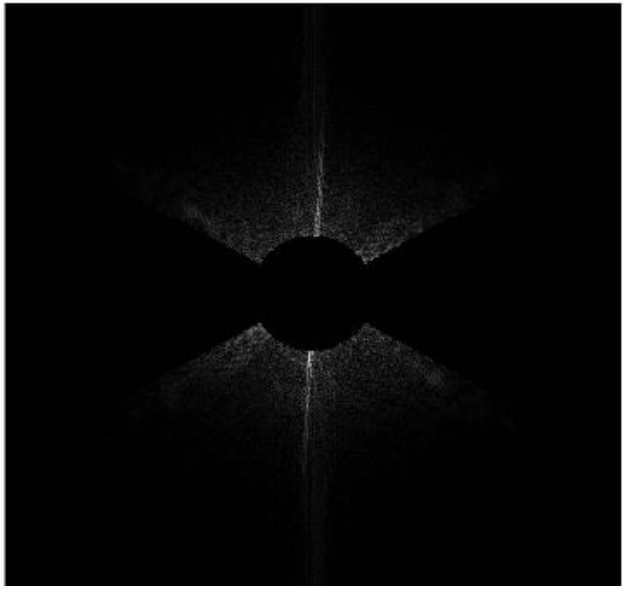
Removing frequency bands



High pass filtering + orientation

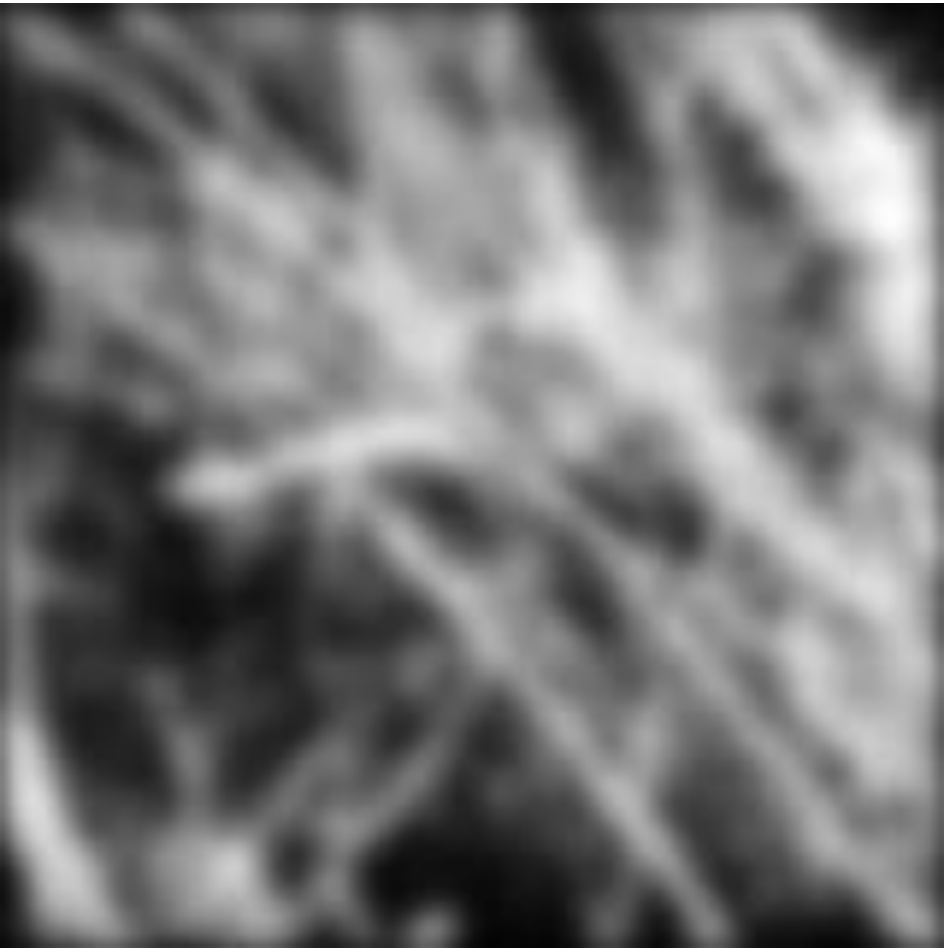




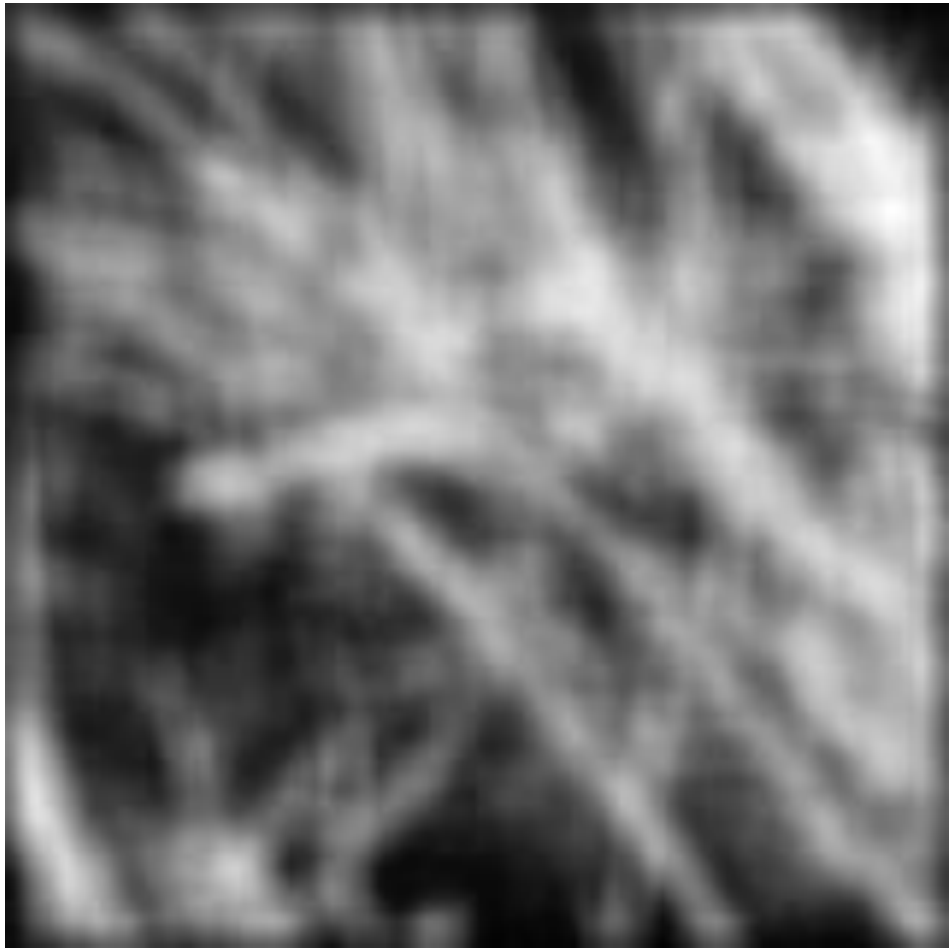


Why does the Gaussian filter give a nice smooth image, but the square filter give edgy artifacts?

Gaussian 

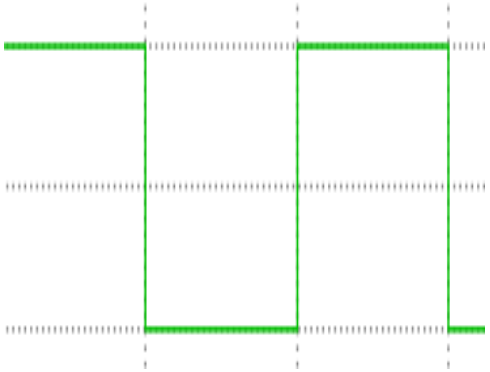


Box filter 



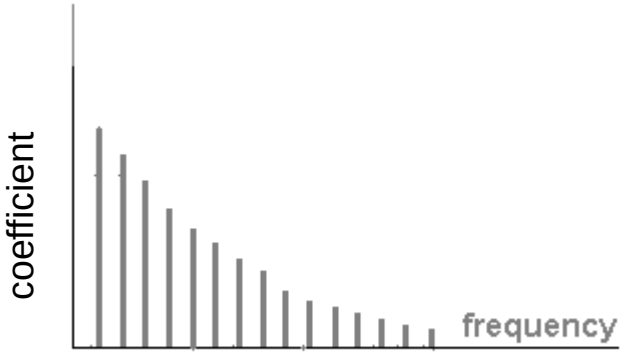
Why do we have those lines in the image?

- Sharp edges in the image need _all_ frequencies to represent them.



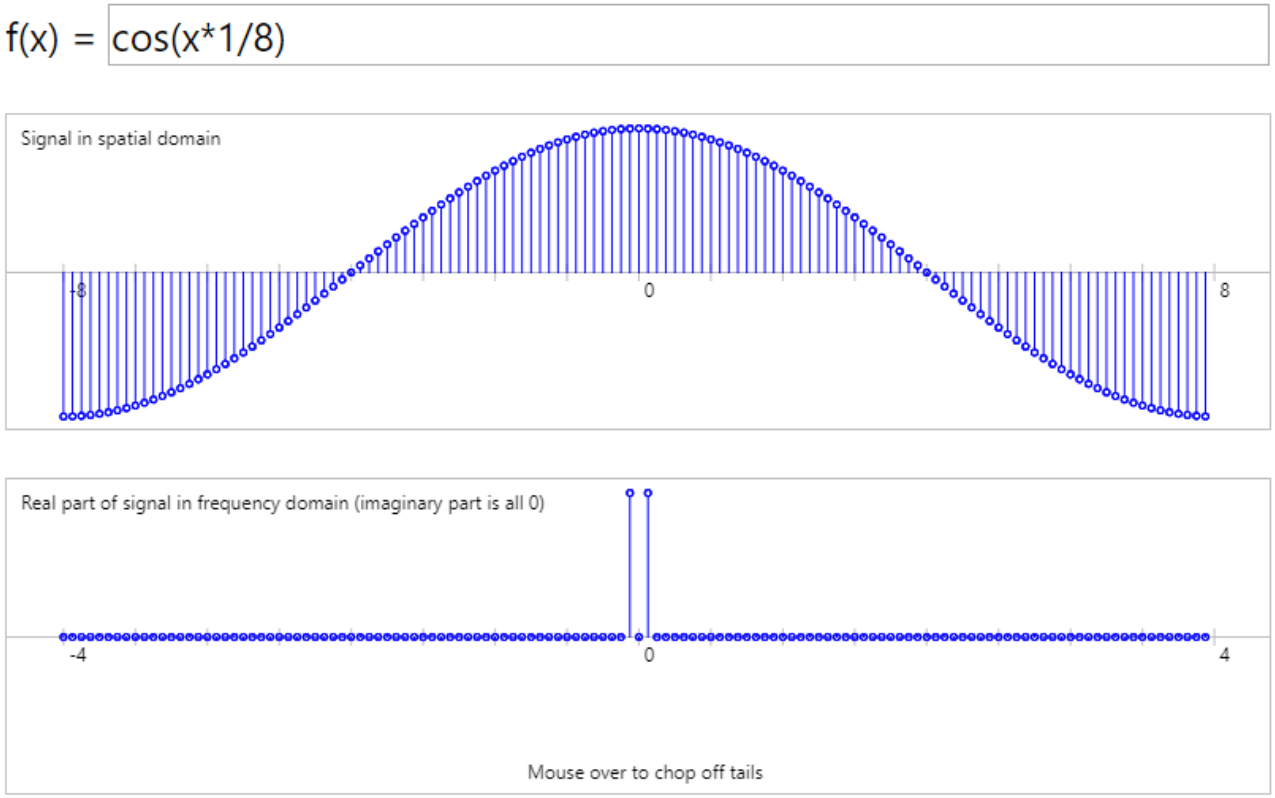
=

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



Box filter / sinc filter duality

- What is the spatial representation of the hard cutoff (box) in the frequency domain?
- http://madebyevan.com/dft/



Evan Wallace demo

- Made for CS123
- 1D example
- Forbes 30 under 30
  - Figma (collaborative design tools)

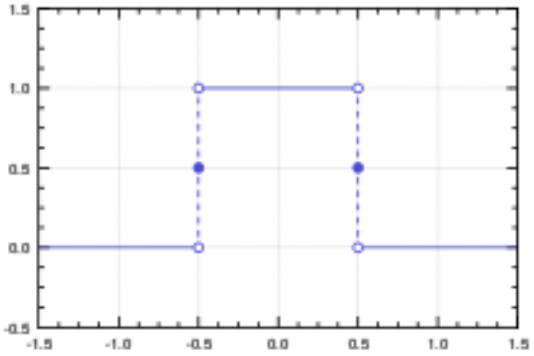
- http://madebyevan.com/dft/

with Dylan Field

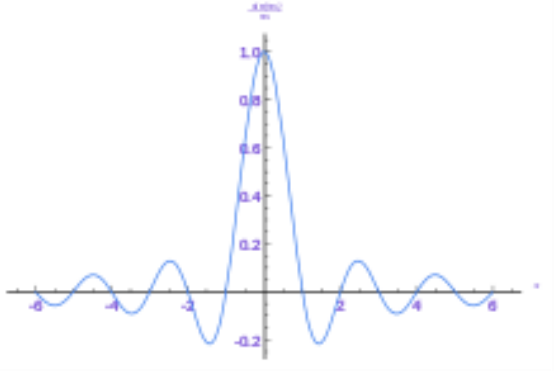
Box filter / sinc filter duality

- What is the spatial representation of the hard cutoff (box) in the frequency domain?
- http://madebyevan.com/dft/

Box filter



Sinc filter

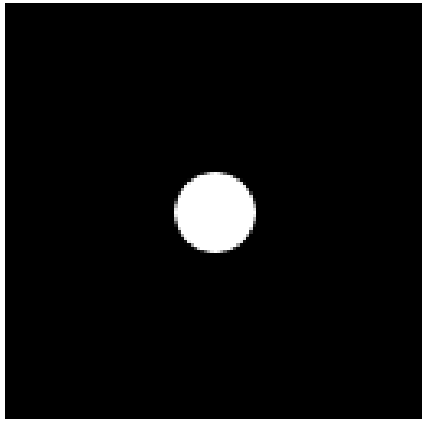


sinc(x) = sin(x) / x

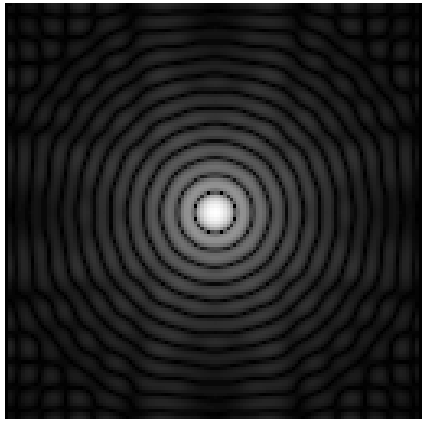
Spatial Domain ⟺ Frequency Domain

Frequency Domain ⟺ Spatial Domain

Box filter (spatial)



Frequency domain magnitude



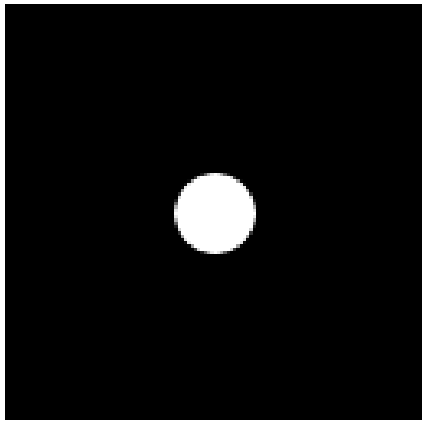
Gaussian filter duality

- Fourier transform of one Gaussian…

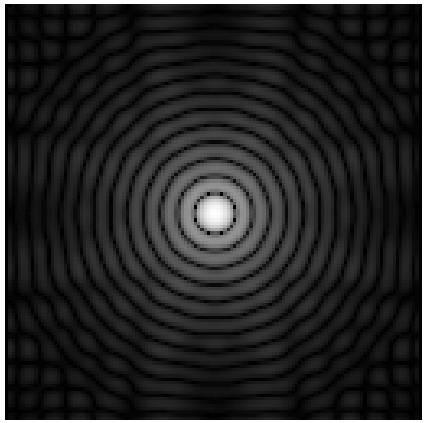
  …is *another Gaussian (with inverse variance).*

- Why is this useful?
  - Smooth degradation in frequency components
  - No sharp cut-off
  - No negative values
  - Never zero (infinite extent)

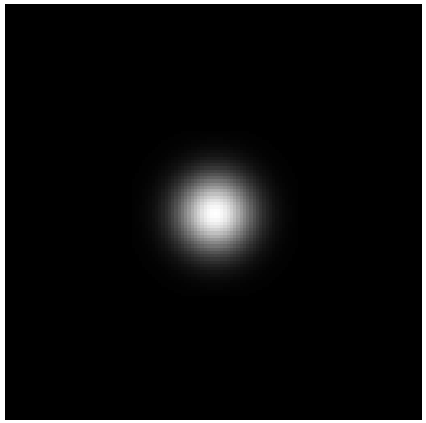
Box filter (spatial)



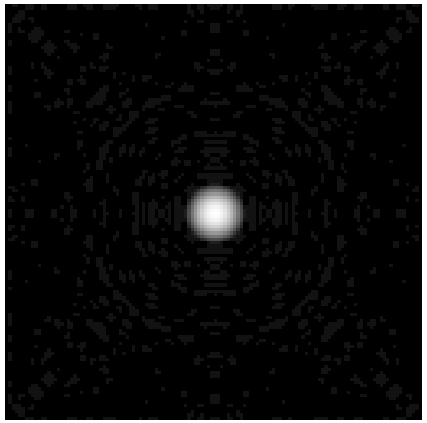
Frequency domain magnitude



Gaussian filter (spatial)



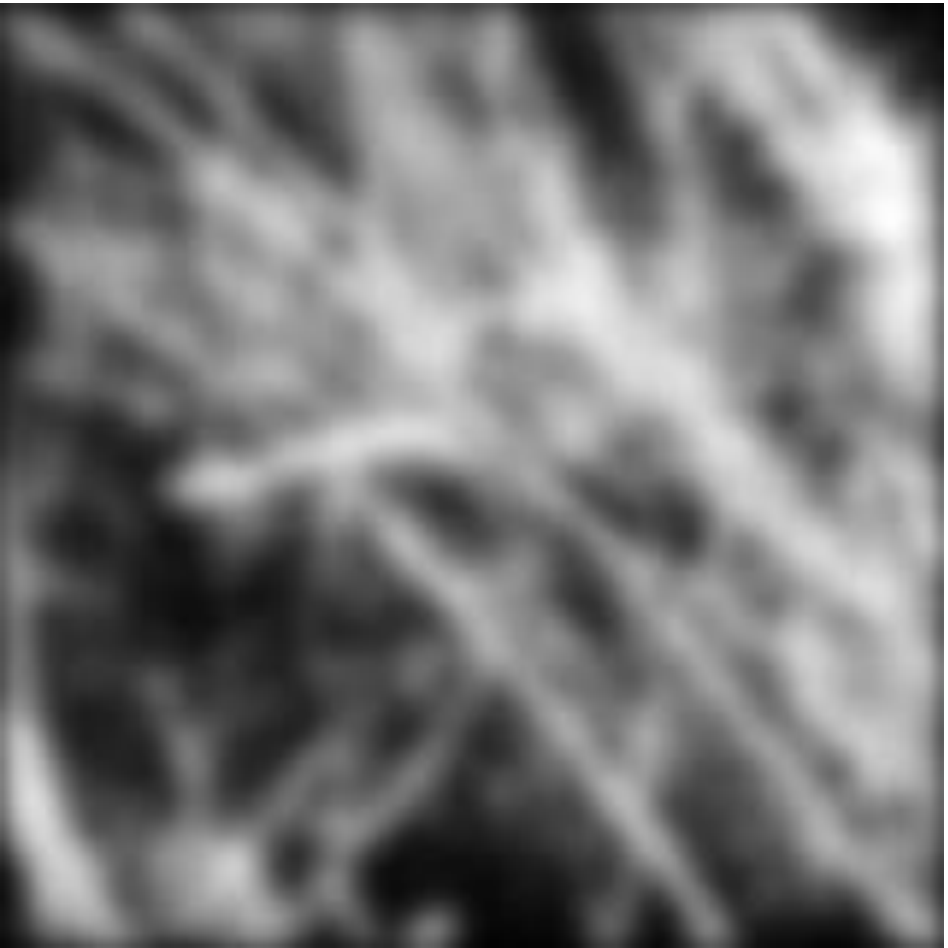
Frequency domain magnitude



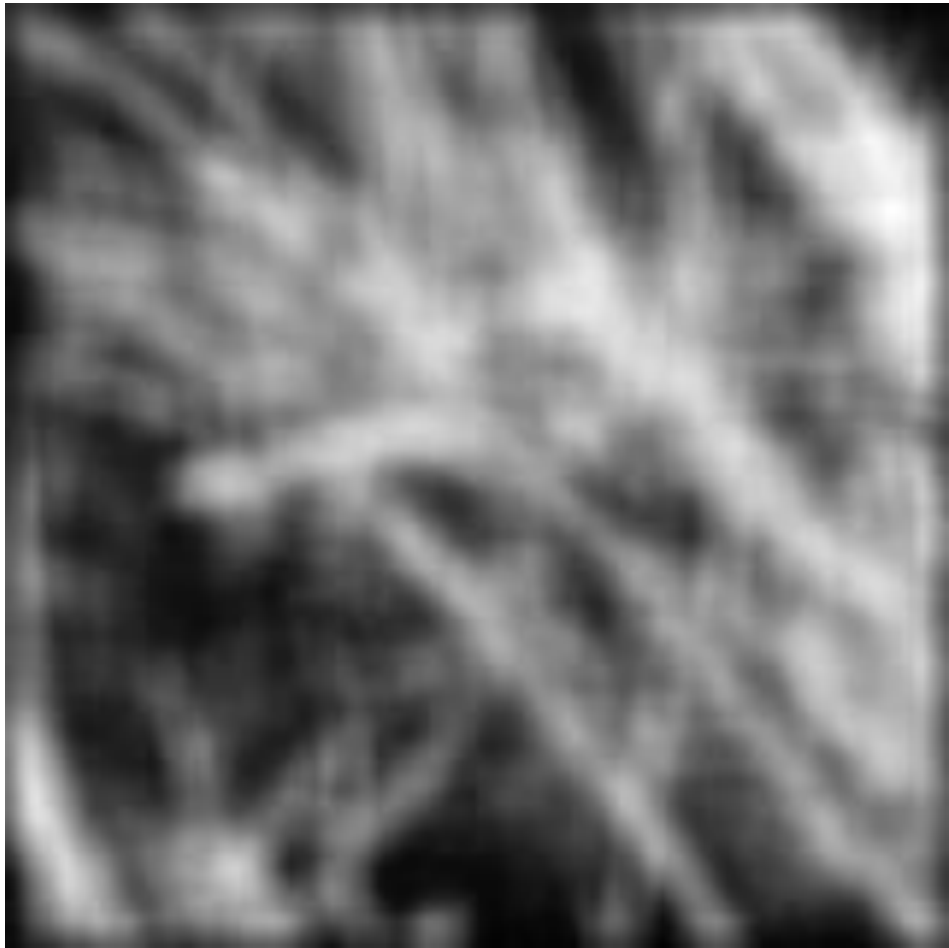
Ringing artifacts -> ‘Gibbs effect’

Where *infinite* series can never be reached

Gaussian 



Box filter 



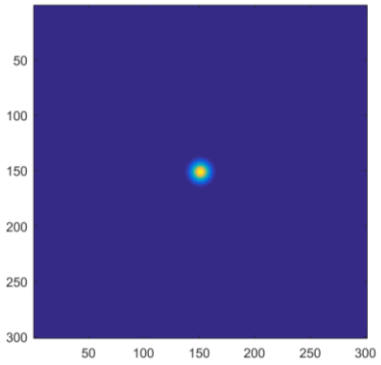
Is convolution invertible?

- 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

*

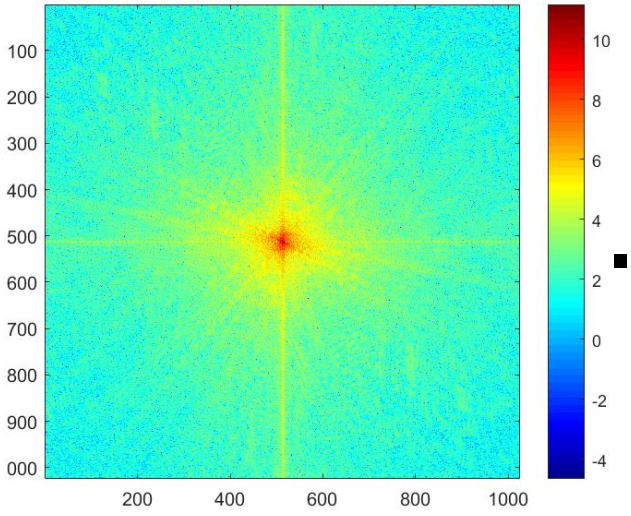


=

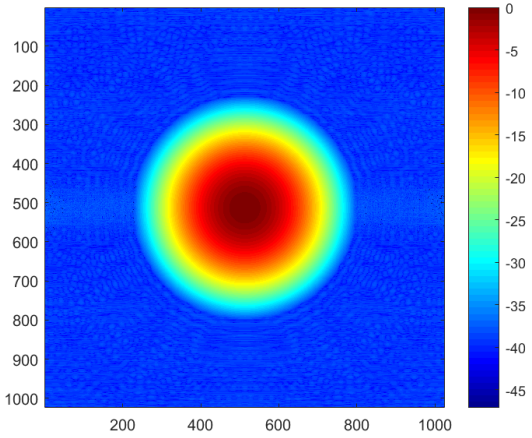
FFT

FFT

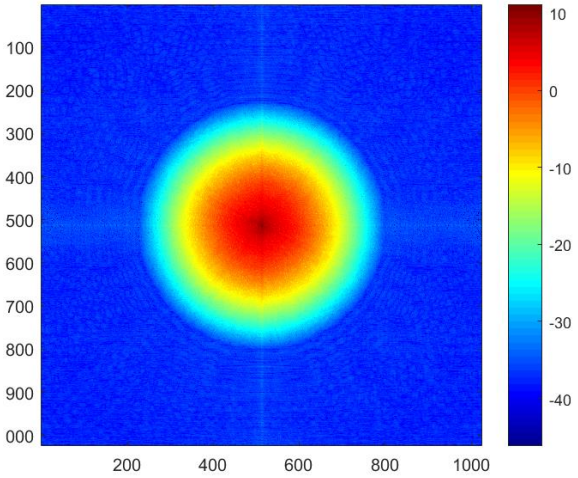
iFFT



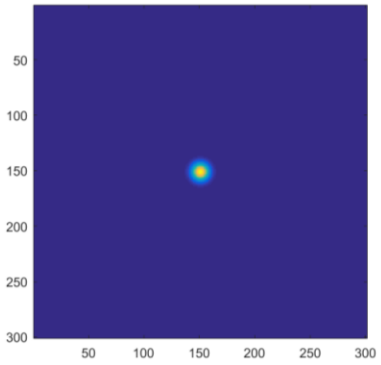
.X



=



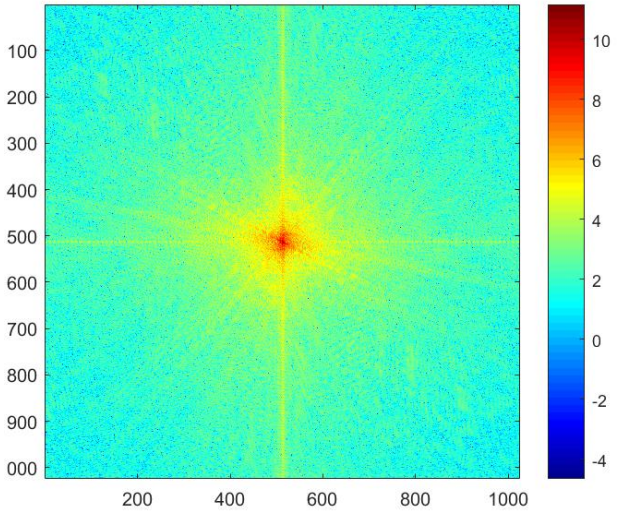
Deconvolution?



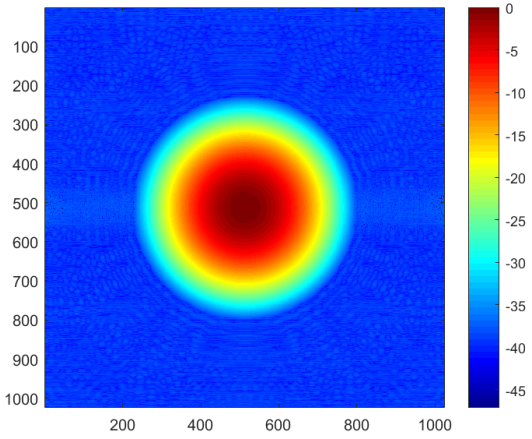
iFFT

FFT

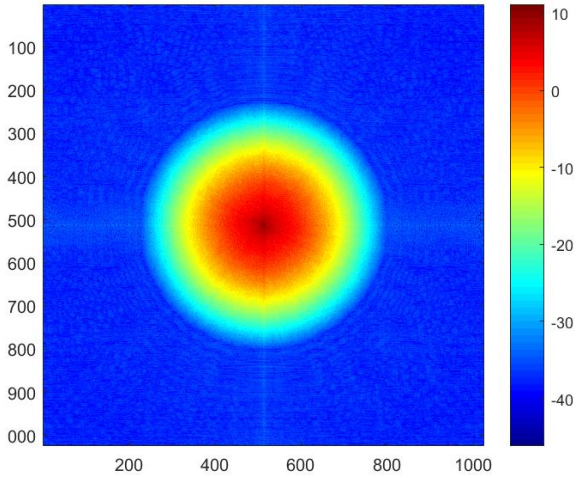
FFT



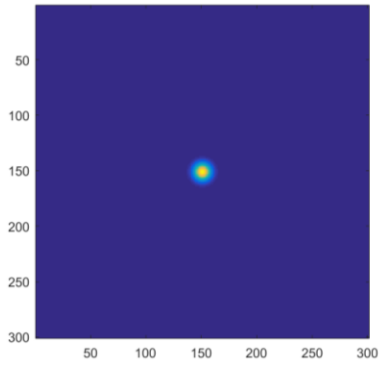
=



./



But under more realistic conditions

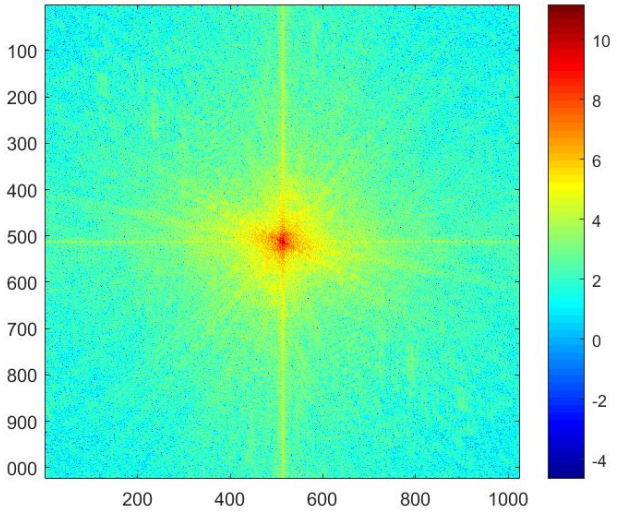


Random noise, .000001 magnitude

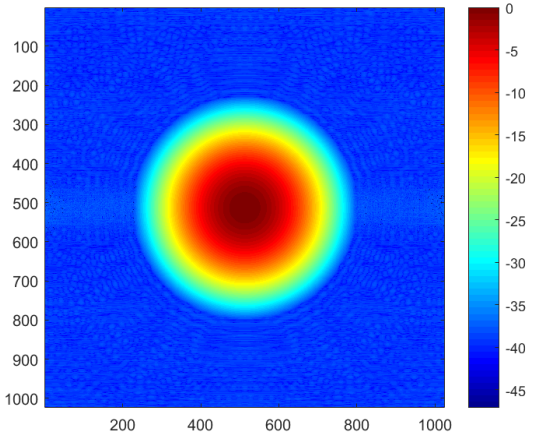
iFFT

FFT

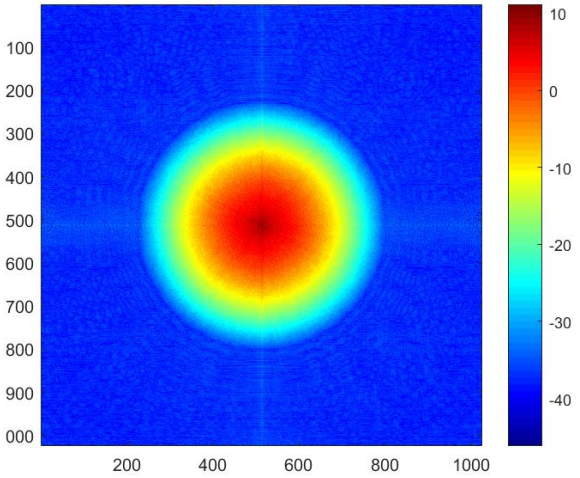
FFT



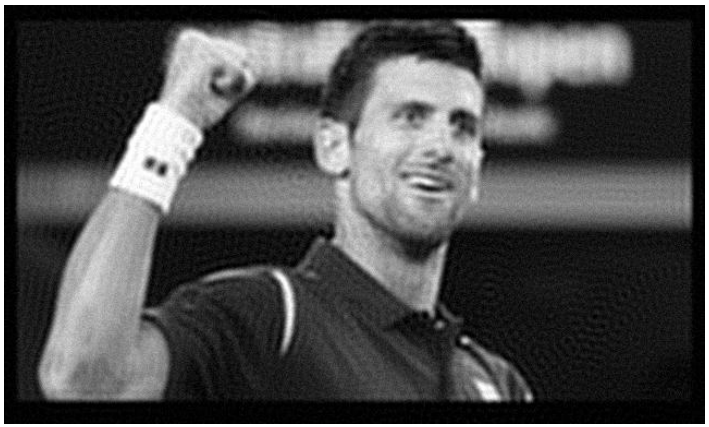
=

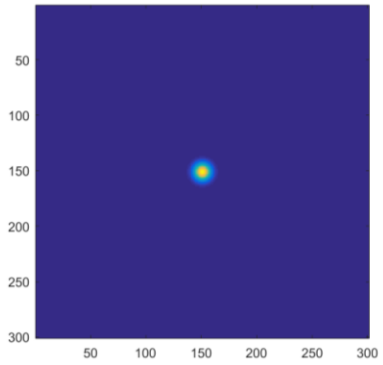


./



But under more realistic conditions



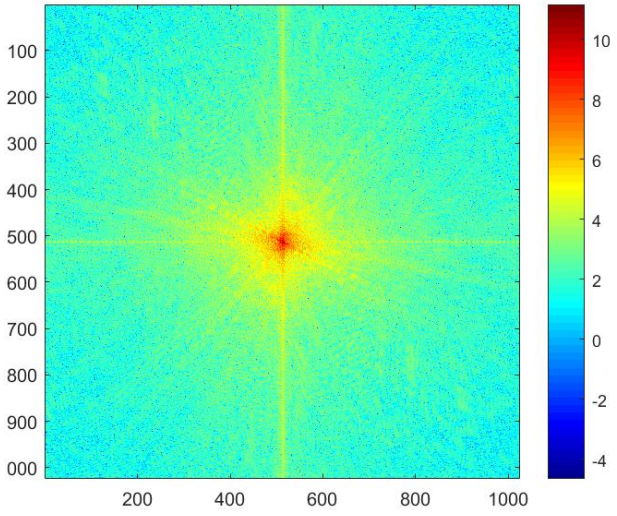


Random noise, .0001 magnitude

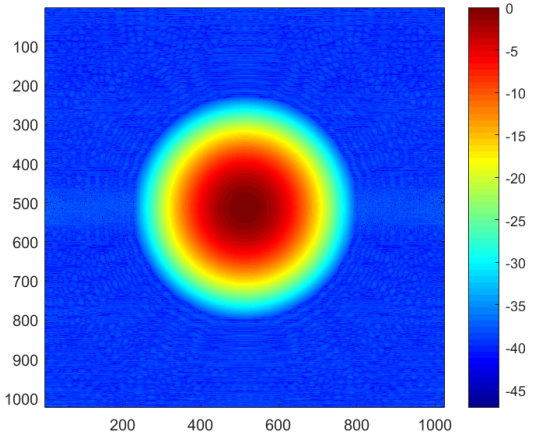
iFFT ↑

FFT ↓

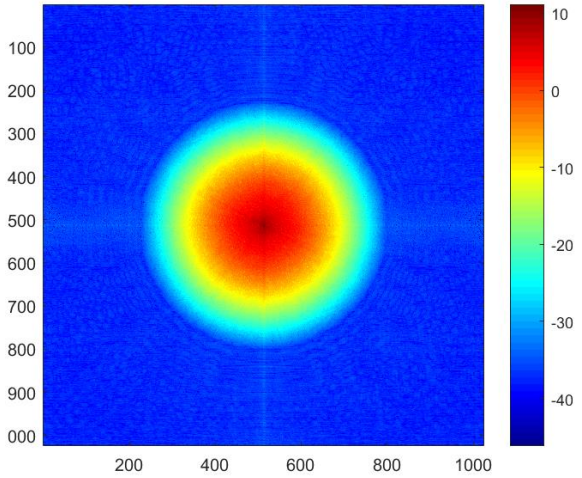
FFT ↓



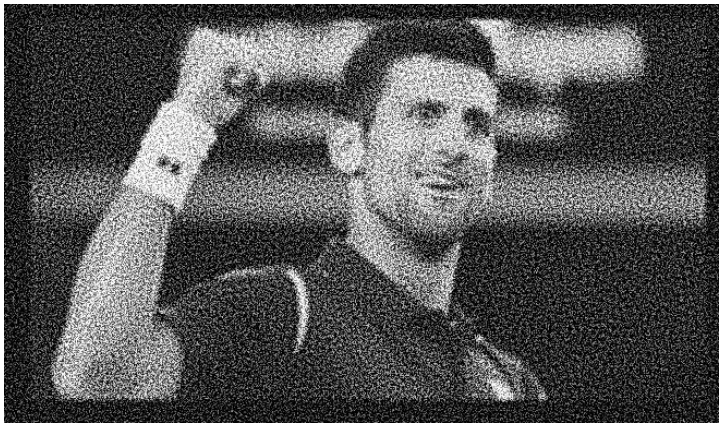
=

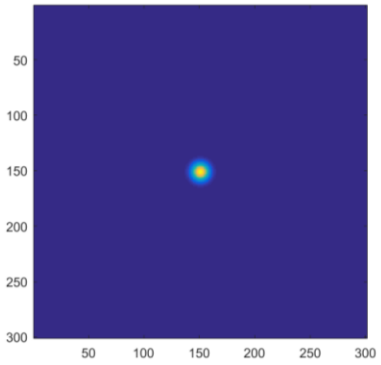


./



But under more realistic conditions



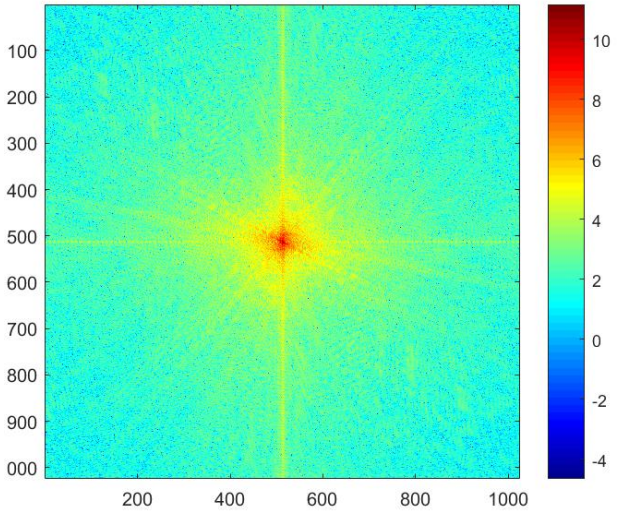


Random noise, .001 magnitude

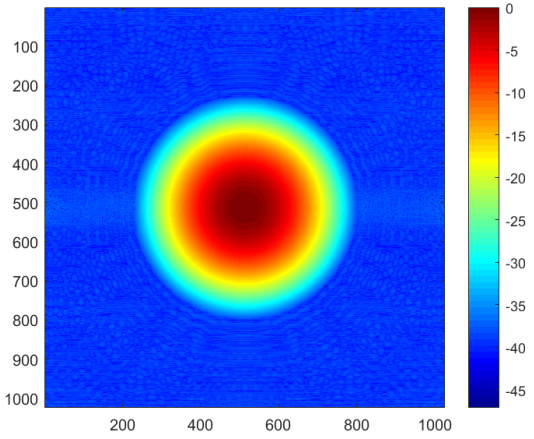
iFFT

FFT

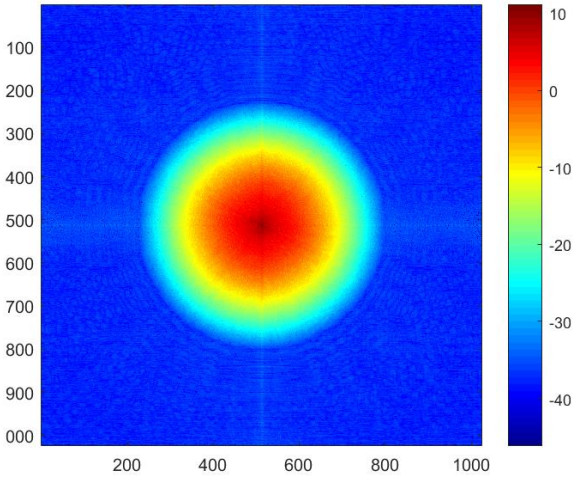
FFT



=



./



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.

A few questions

How is the Fourier decomposition computed?

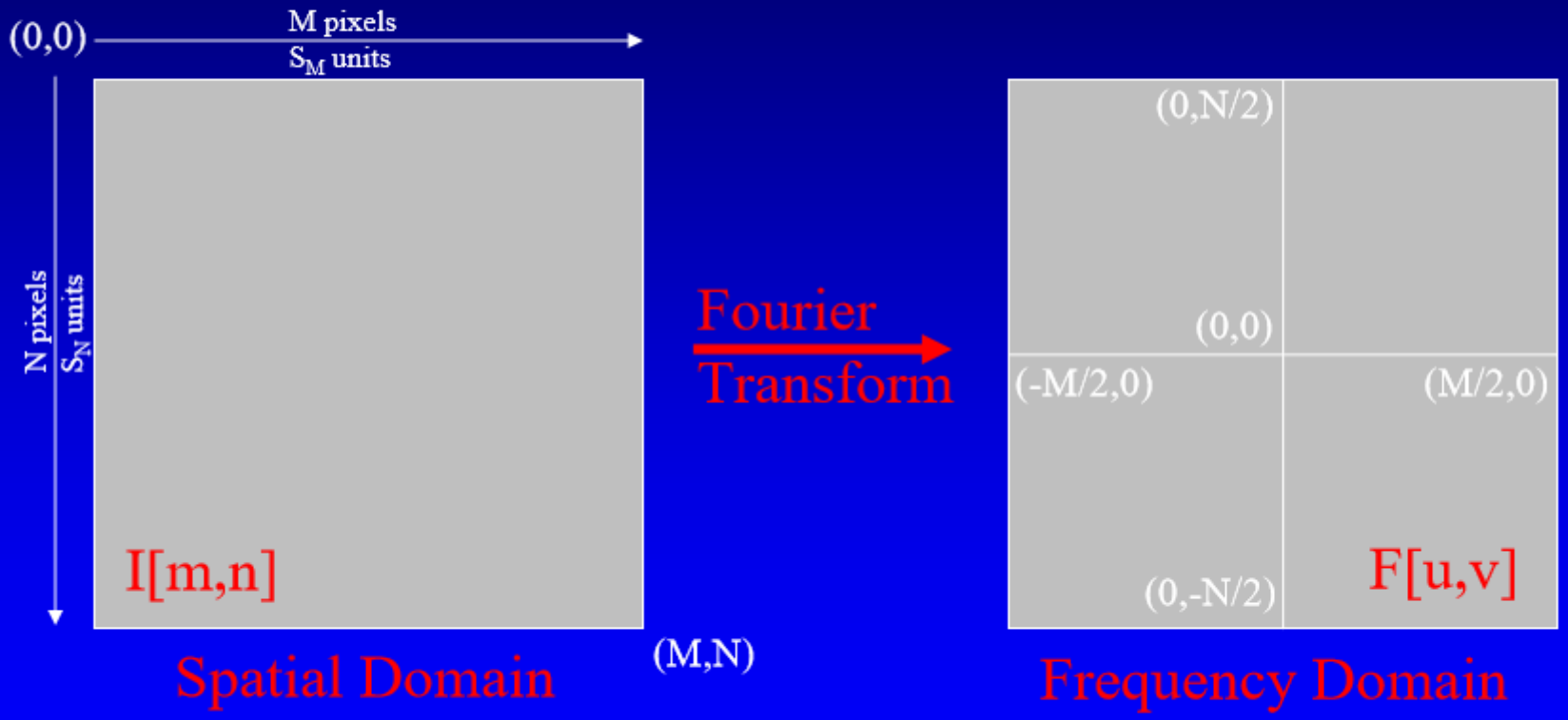
Intuitively, by correlating the signal with a set of waves of increasing frequency!

Notes in hidden slides.

Plus: http://research.stowers-institute.org/efg/Report/FourierAnalysis.pdf

2D Discrete Fourier Transform

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

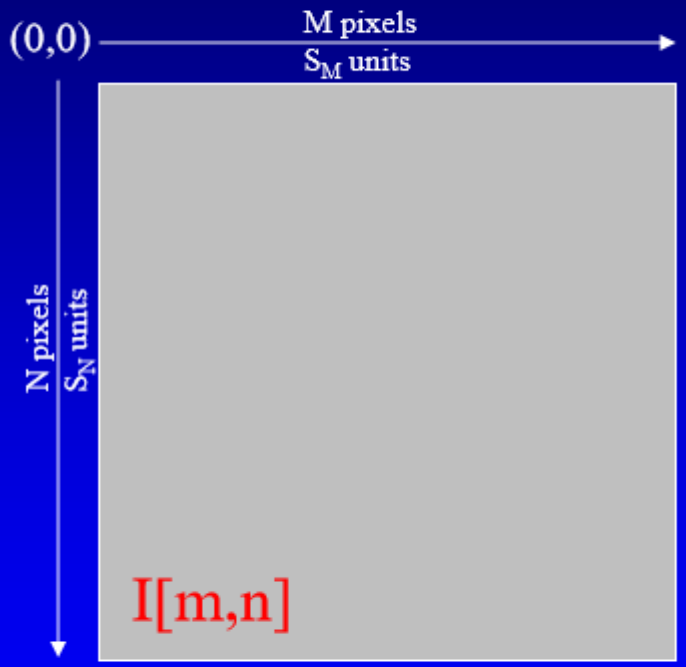


Source: Seul et al, *Practical Algorithms for Image Analysis*, 2000, p. 249, 262.

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

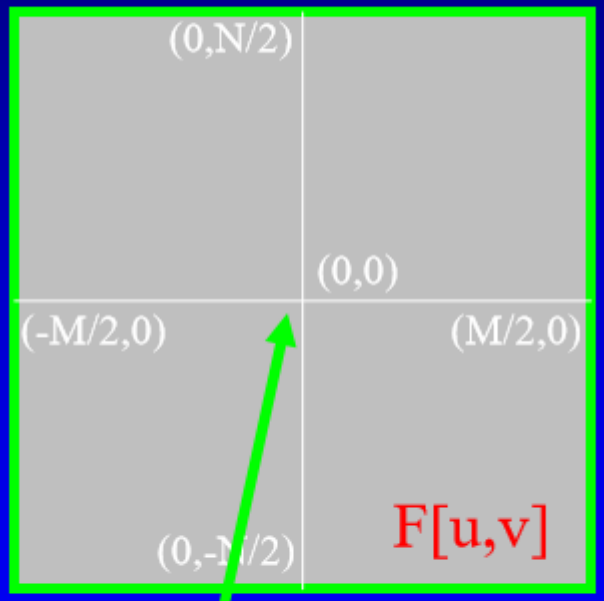
2D Discrete Fourier Transform

Edge represents highest frequency, smallest resolvable length (2 pixels)



Spatial Domain

Fourier Transform



Frequency Domain

Center represents lowest frequency, which represents average pixel value

Index cards before you leave

Fourier decomposition is tricky

I want to know what is confusing to you!

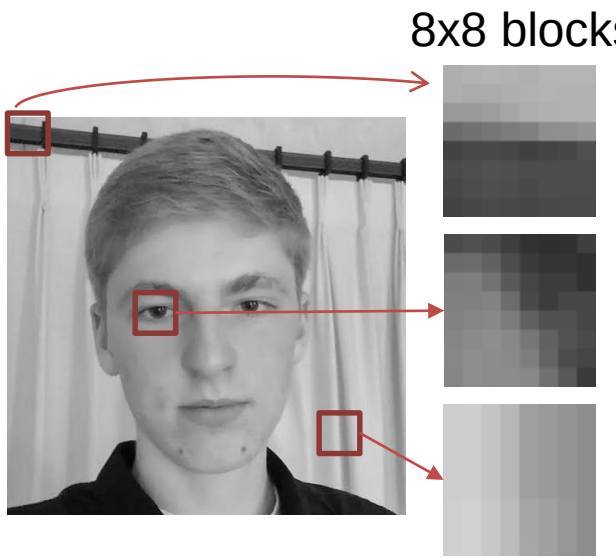
- Take 1 minute to talk to your partner about the lecture

- Write down on the card something that you'd like clarifying.

Thinking in Frequency - Compression

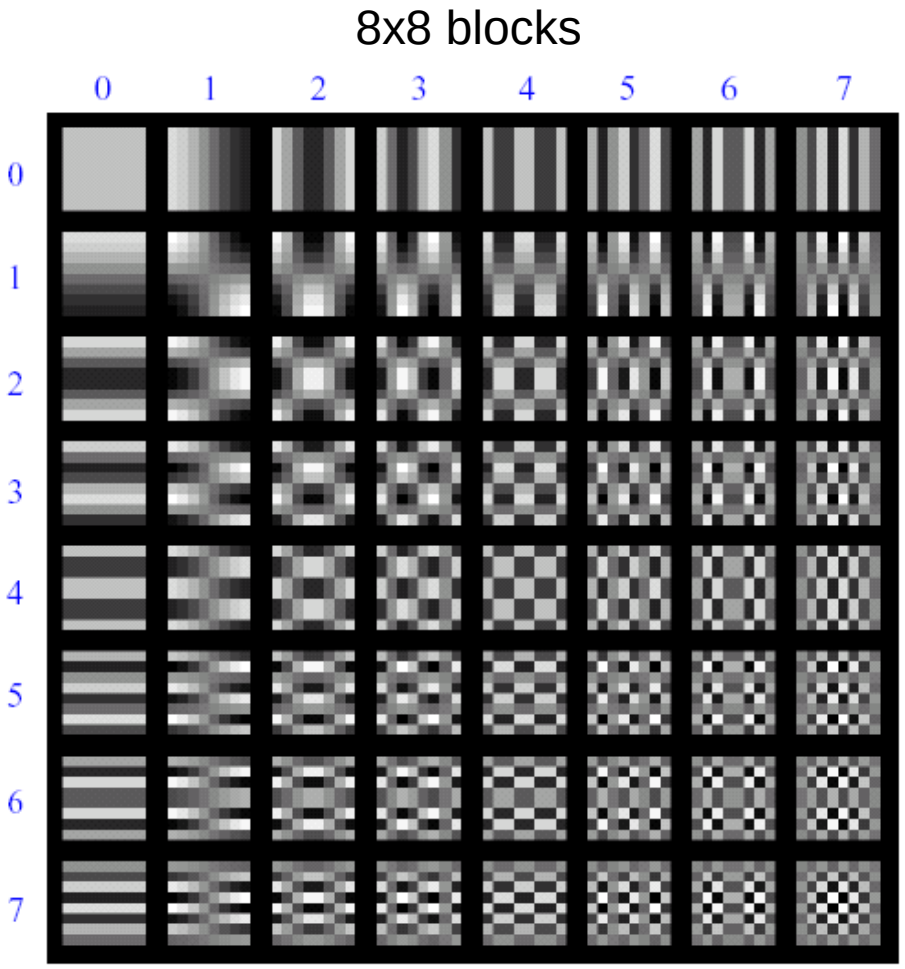
How is it that a 4MP image can be compressed to a few hundred KB without a noticeable change?

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

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

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

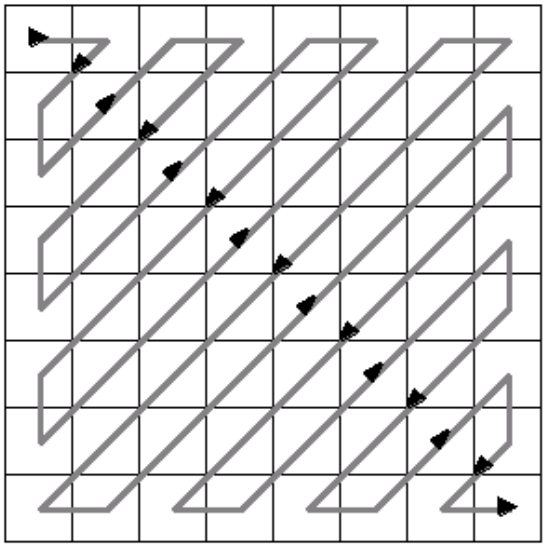
JPEG Encoding

- Entropy coding (Huffman-variant)

Quantized values

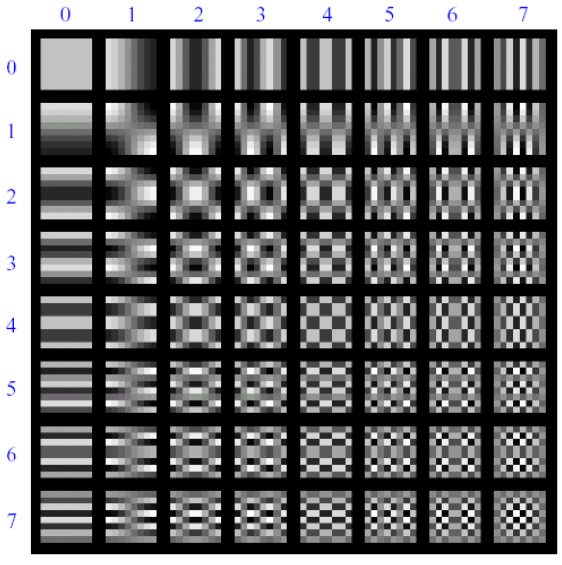
$$B = \begin{bmatrix} -26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\ 0 & -2 & -4 & 1 & 1 & 0 & 0 & 0 \\ -3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\ -3 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Linearize *B* like this.



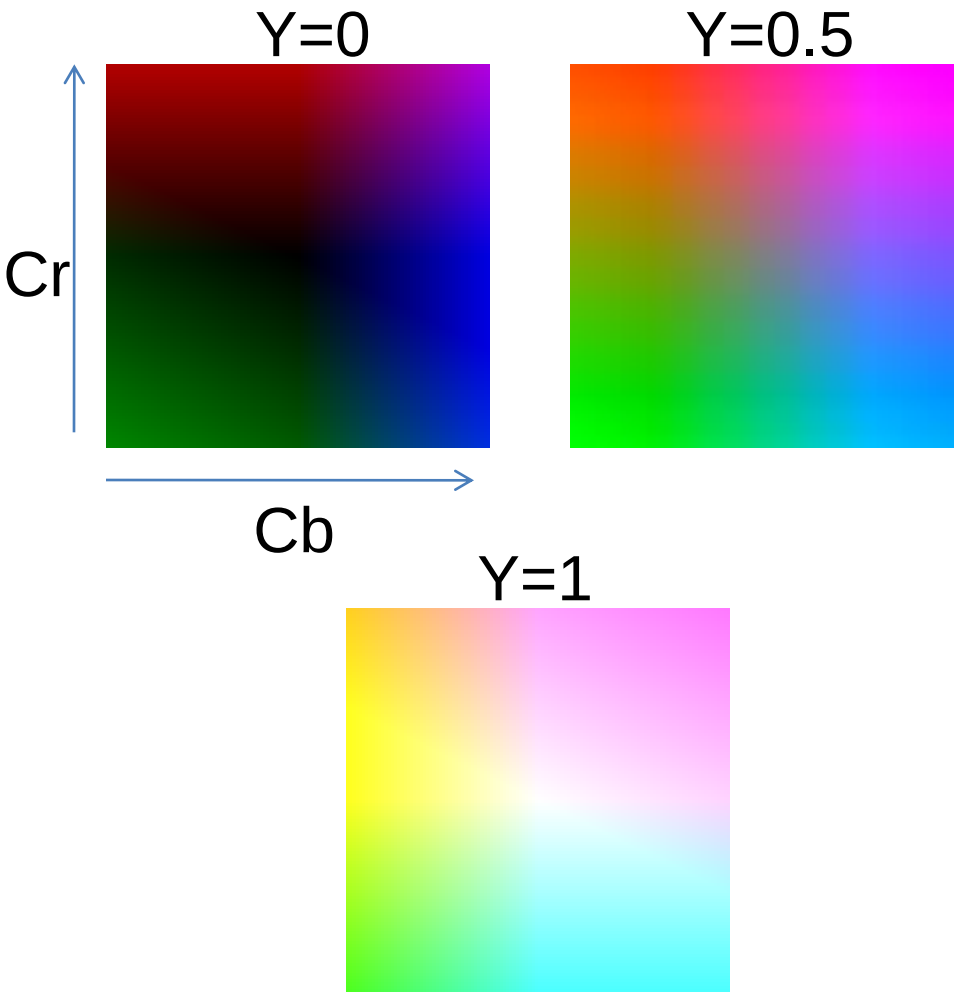
Helps compression:

- 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