Computer Vision (CS 419/619)

Image filtering in Frequency Domain

Dr. Puneet Gupta

Why talk about Fourier transforms?

- Convolution is point-wise multiplication in frequency space
  - Analyze which frequency components a particular filter lets through, e.g., *low-pass, high-pass* or *band-pass* filters
  - Leads to fast algorithms for convolution with large filters: Fast FFT
- Frequency space reveals structure at various scales
  - Noise is high-frequency
  - ”Average brightness” is low-frequency
- Useful to understand how we resize/resample images
  - Sampling causes information loss
  - What is lost exactly?
  - What can we recover?

Trigonometric Fourier Series

Any periodic function f(t) can be expressed as a weighted (infinite) sum of sine and cosine functions of increasing frequency:

If f(x) be a 2π-periodic function which is integrable on [−π, π] with at most a finite number of maxima and minima and at most a finite number of discontinuities in the interval.

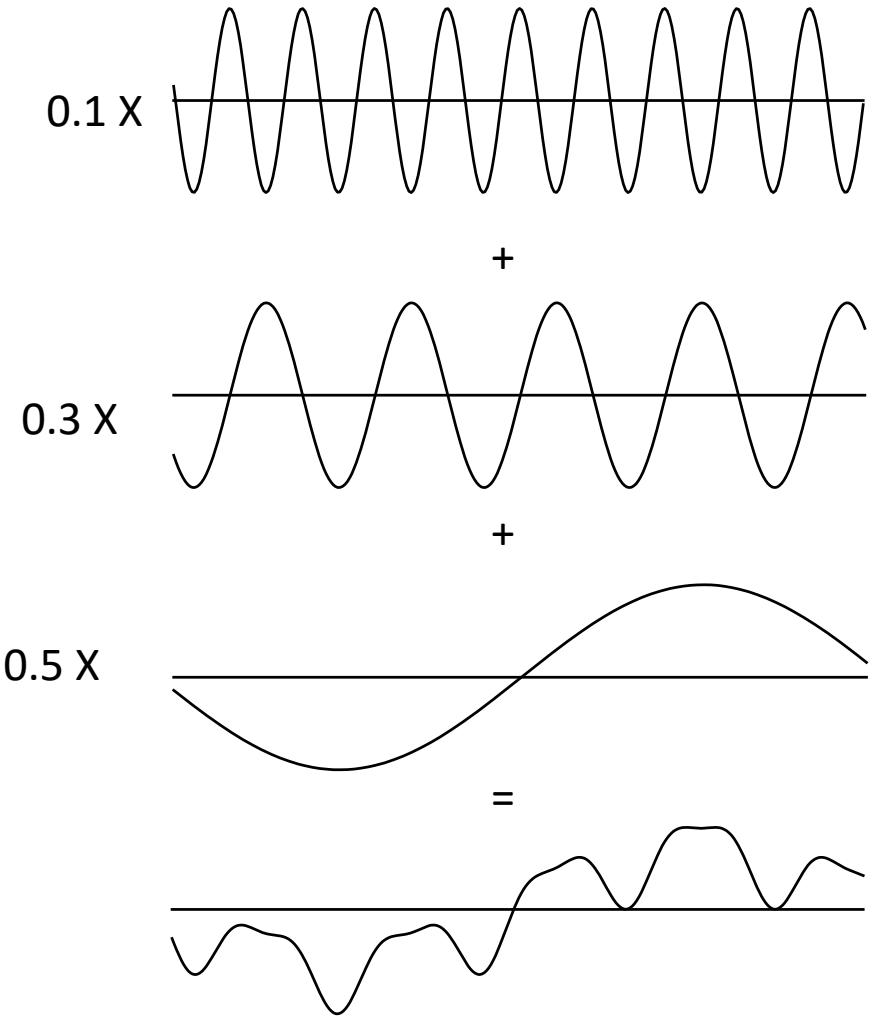
$$f(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} a_n \cos(nt) + \sum_{n=1}^{\infty} b_n \sin(nt)$$

basis functions

where $a_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(t)\cos(nt)dt$ $b_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(t)\sin(nt)dt$

coefficients of expansion

n=1,2,3,…



More explanation: https://www.ucl.ac.uk/~ucahmdl/LessonPlans/Lesson17.pdf

Exponential Fourier Series

$$e^{\pm j\theta} = cos(\theta) \pm jsin(\theta)$$

$$f(x) = \sum_{n=-\infty}^{\infty} c_n e^{-inx}$$

$$c_n = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(x)e^{inx}dx, \ where$$

$$c_n = \frac{1}{2}\left(a_n + ib_n\right), \quad n = 1, 2, \ldots$$

$$c_n = \frac{1}{2}\left(a_{-n} - ib_{-n}\right), \quad n = -1, -2, \ldots$$

$$c_0 = \frac{a_0}{2}$$

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty}\left(a_n \cos\frac{n\pi x}{L} + b_n \sin\frac{n\pi x}{L}\right) \ where \ x \in [-L, L]$$

$$a_n = \frac{1}{L}\int_{-L}^{L} f(x)\cos\frac{n\pi x}{L}dx, \quad n = 0, 1, \ldots,$$

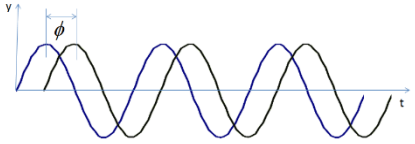
$$b_n = \frac{1}{L}\int_{-L}^{L} f(x)\sin\frac{n\pi x}{L}dx, \quad n = 1, 2, \ldots.$$

$$f(x) \sim \sum_{n=-\infty}^{\infty} c_n e^{-in\pi x/L}$$

$$where, \ c_n = \frac{1}{2L}\int_{-L}^{L} f(x)e^{in\pi x/L}dx$$

FT - Definitions

- In general, **F(u)** is a complex function: $F(u) = R(u) + jI(u)$
- Magnitude of FT (or spectrum): $|F(u)| = \sqrt{R^2(u) + I^2(u)}$ strength of different frequencies
- Phase of FT: $\phi(F(u)) = \tan^{-1}(\frac{I(u)}{R(u)})$ how sinusoidals line up (i.e., shift)



- Magnitude-Phase representation: $F(u) = |F(u)|e^{j\phi(u)}$
- Power of f(x): **P(u)**=|F(u)|^2= $R^2(u) + I^2(u)$

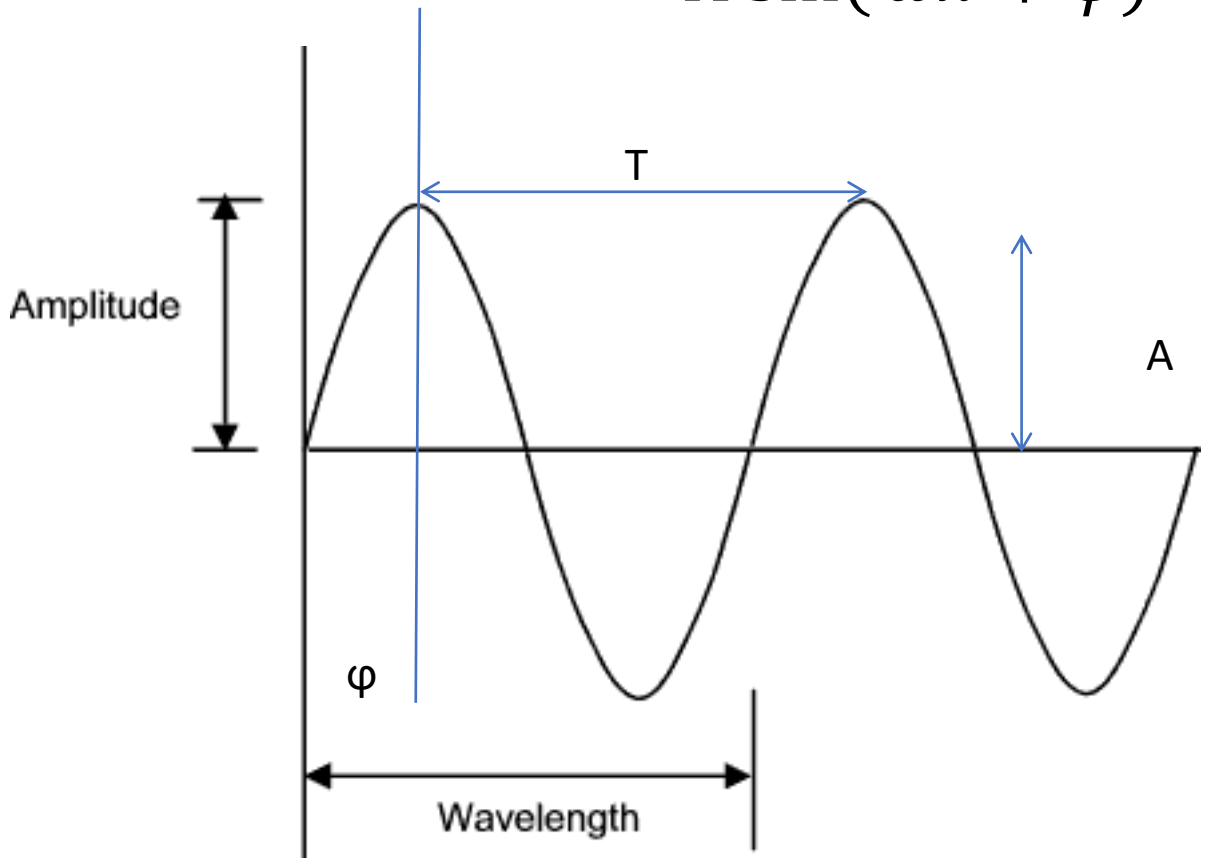
Demo: https://lpsa.swarthmore.edu/BackGround/phasor/phasor.html

Sine Wave

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

Why don't we represent the system with some other periodic signals?

- Sine wave is the only waveform that doesn't change shape when subject to a linear-time-invariant (LTI) system. Input sinusoids, output will be sinusoids.

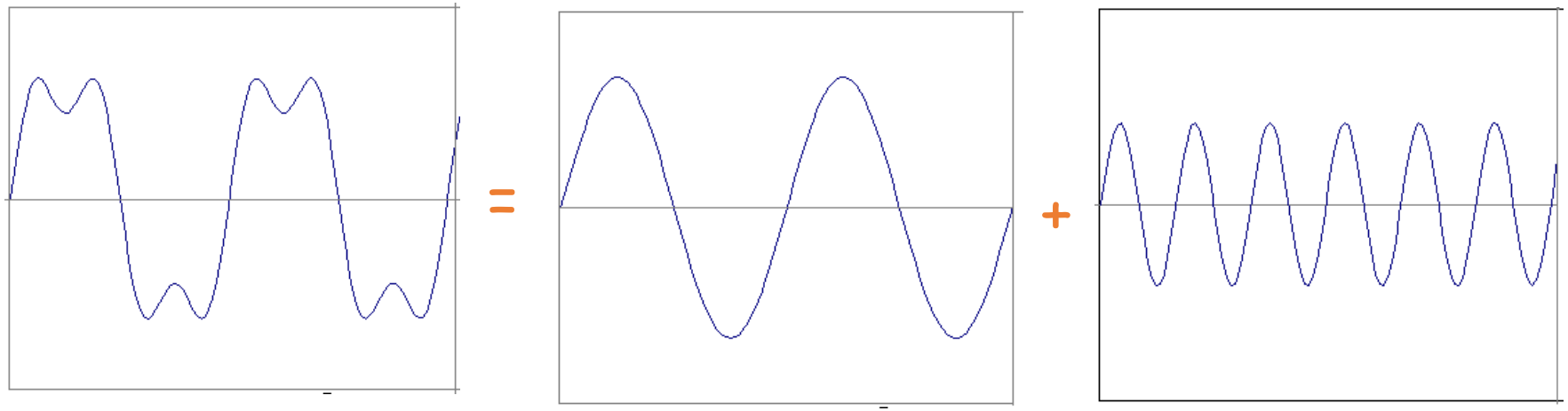


A: Amplitude
f: angular frequency, ω/2π (HZ)
Number of oscillations that occur each second of time
φ: phase

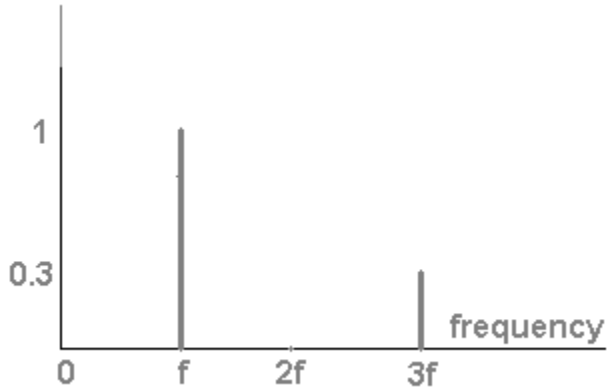
Demo: https://lpsa.swarthmore.edu/BackGround/phasor/phasor.html

Frequency Spectra

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



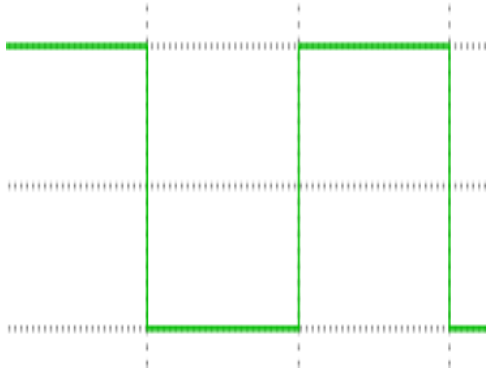
amplitude



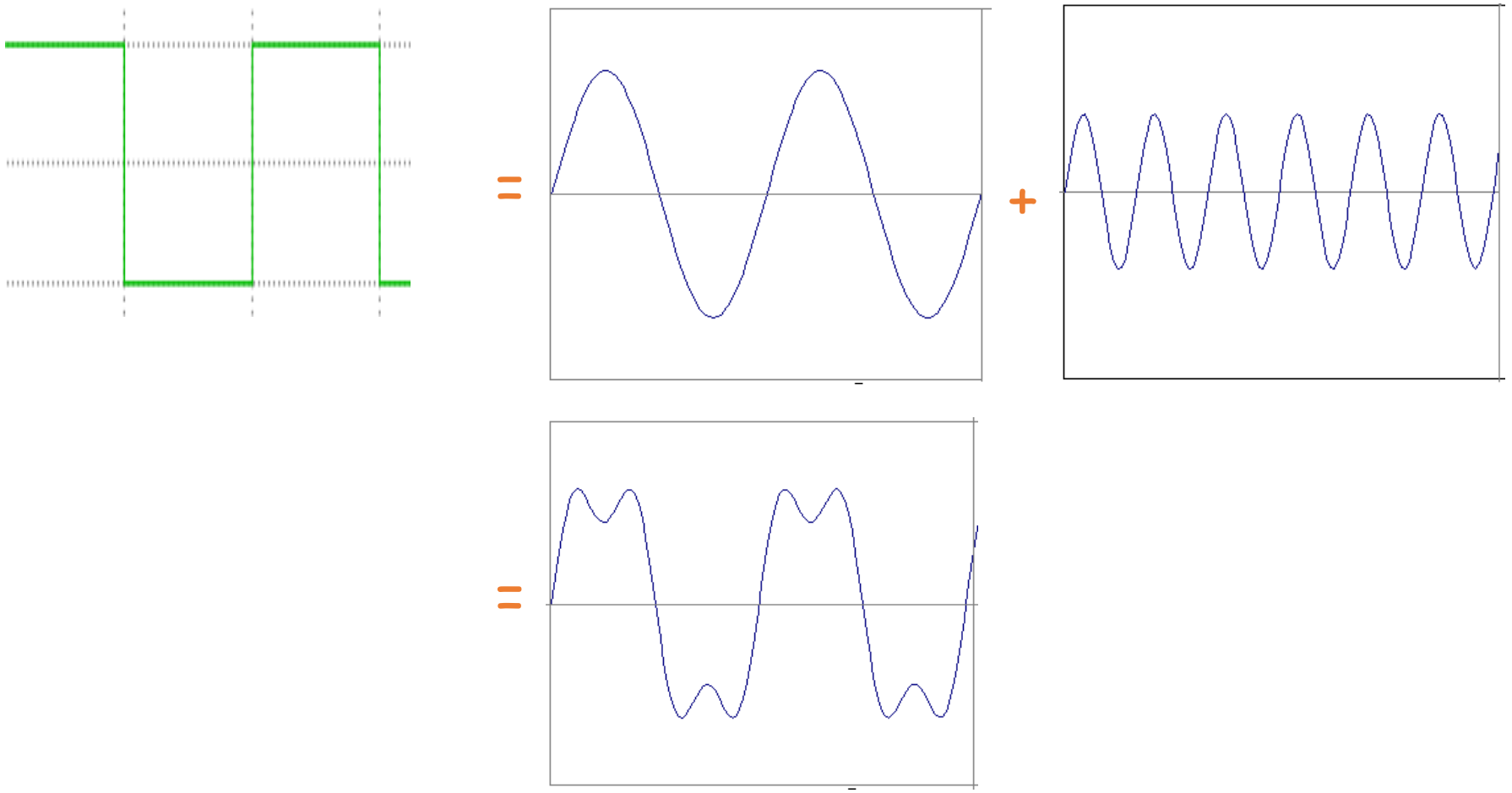
Slides: Efros

Frequency Spectra

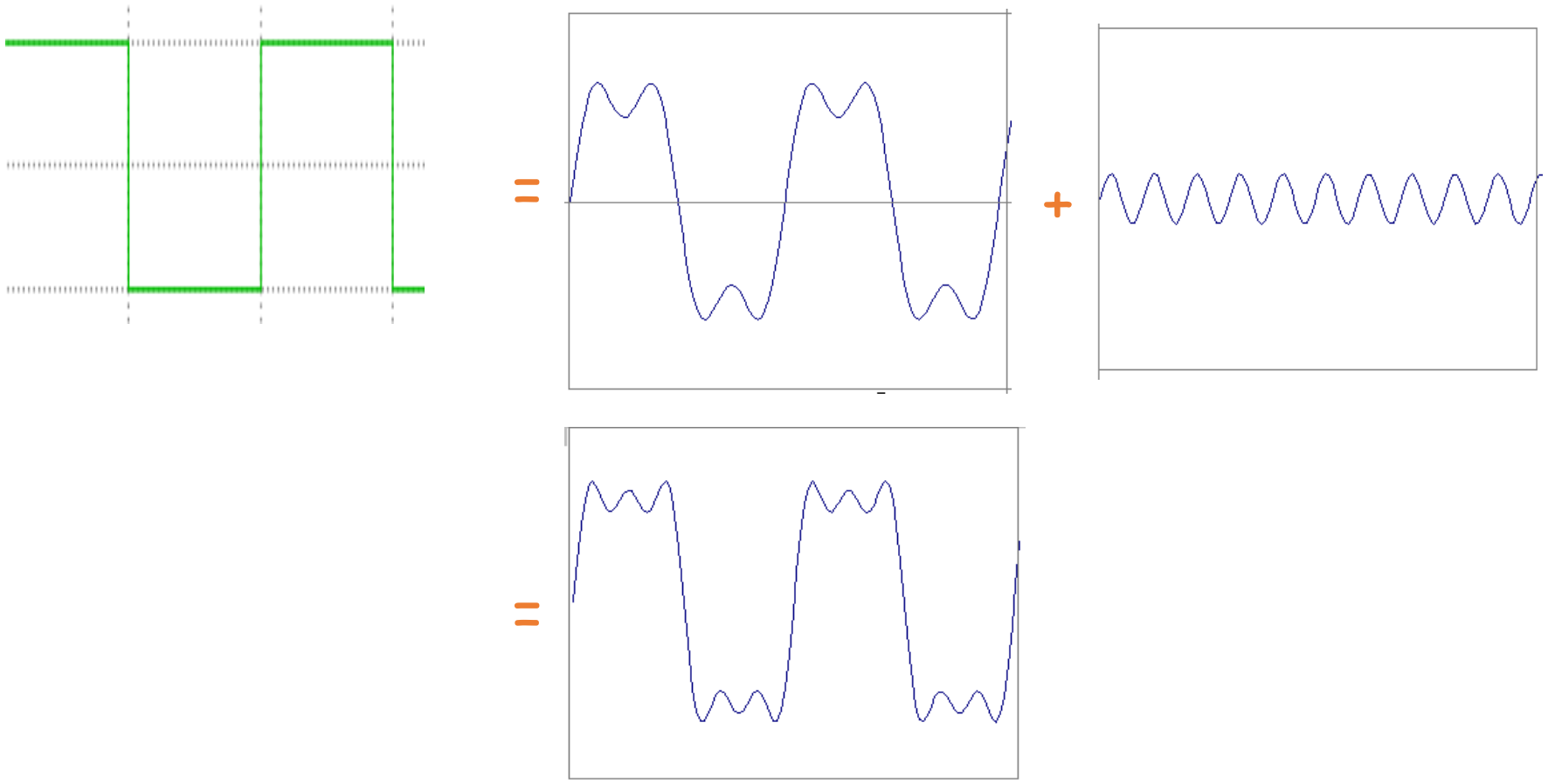
- Consider a square wave f(x)



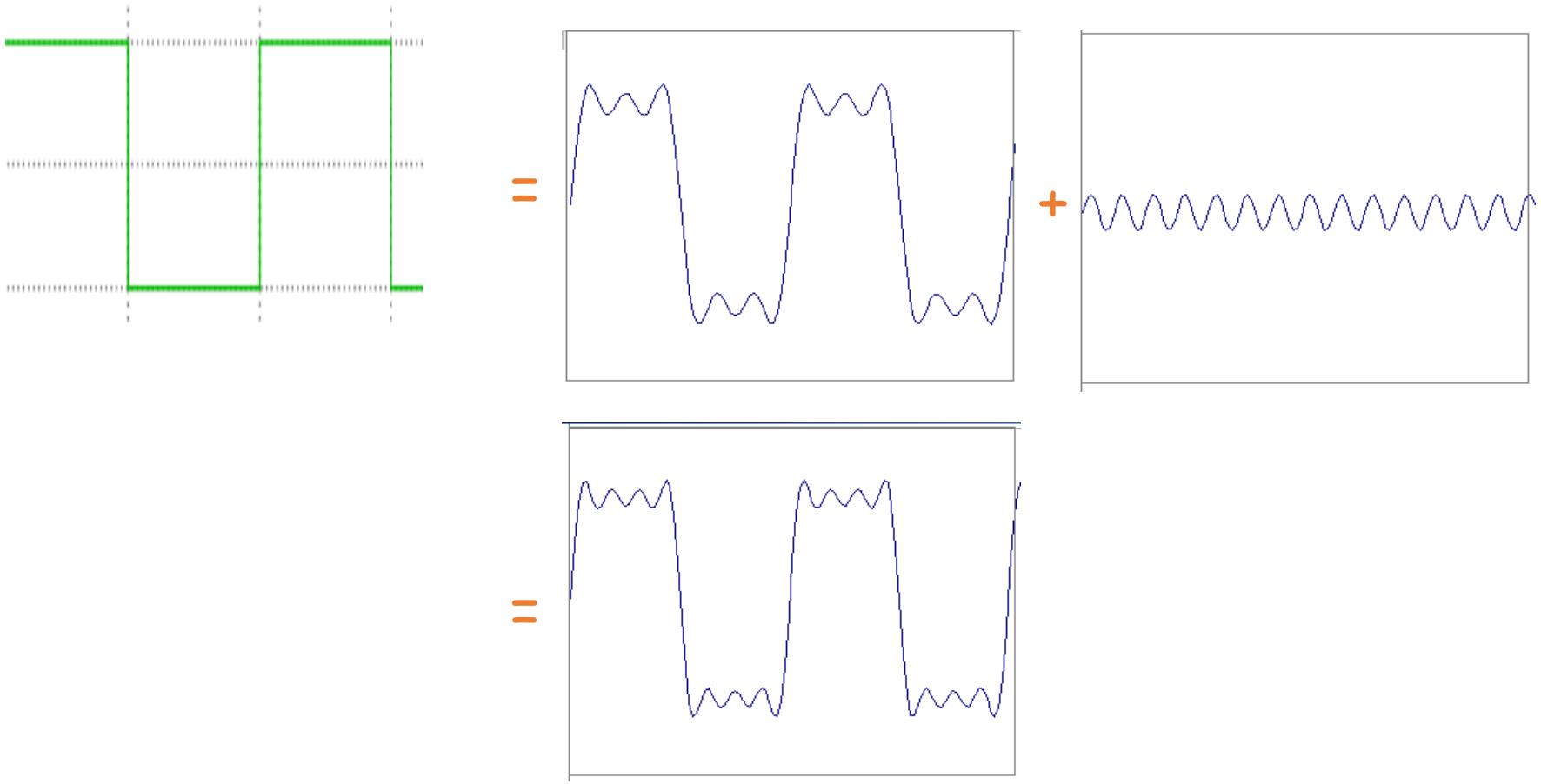
Frequency Spectra



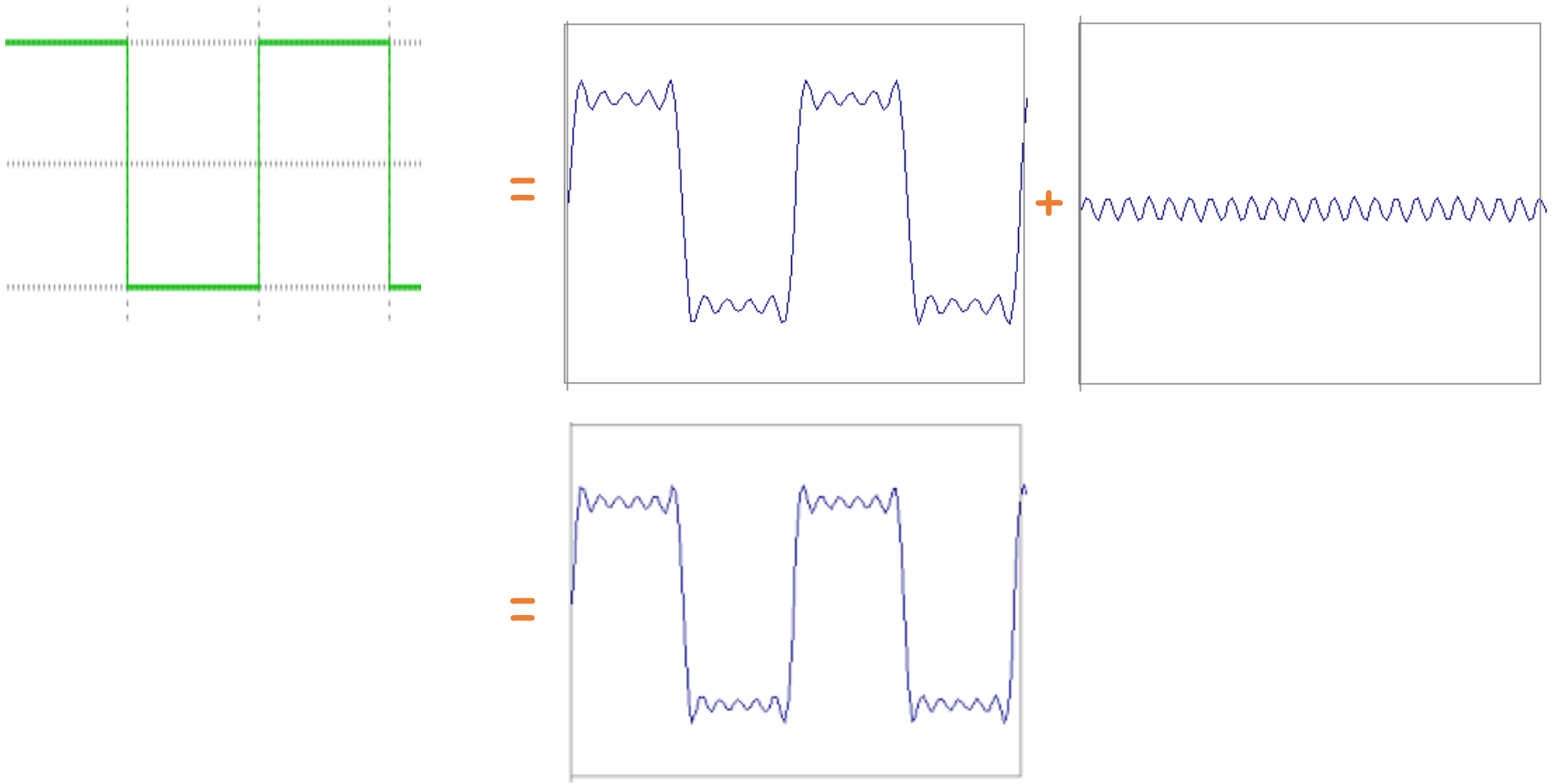
Frequency Spectra



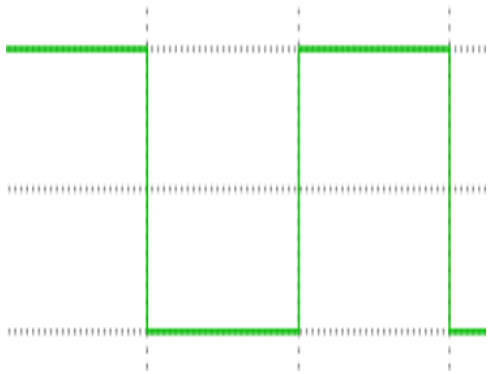
Frequency Spectra

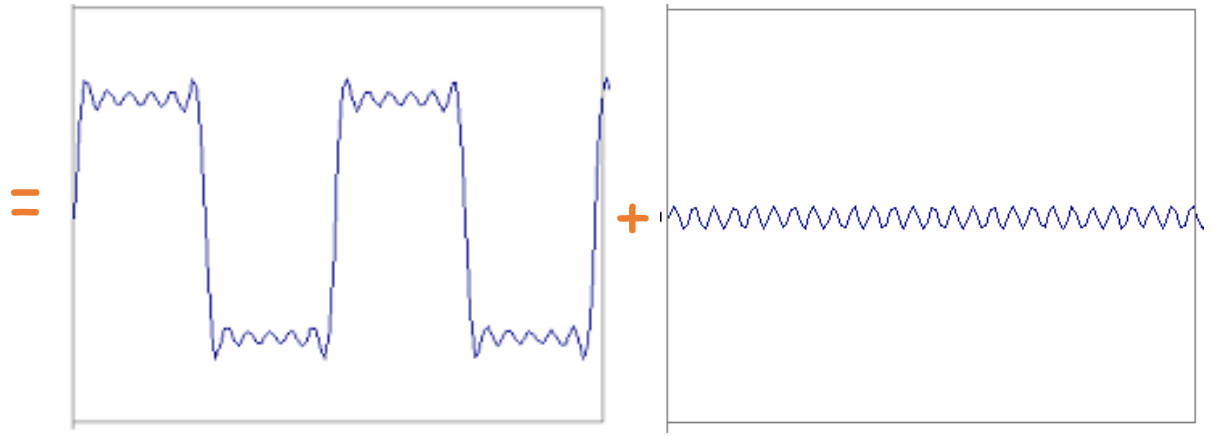


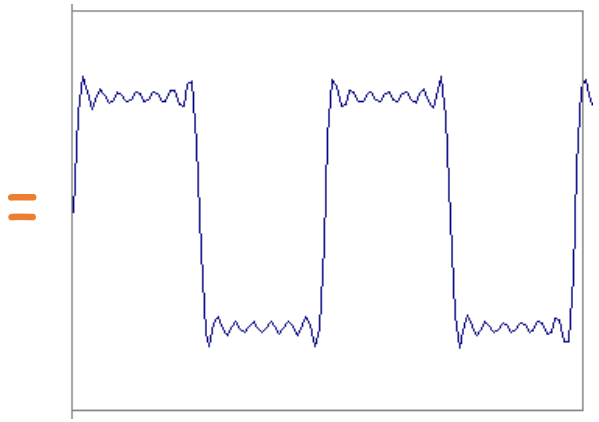
Frequency Spectra



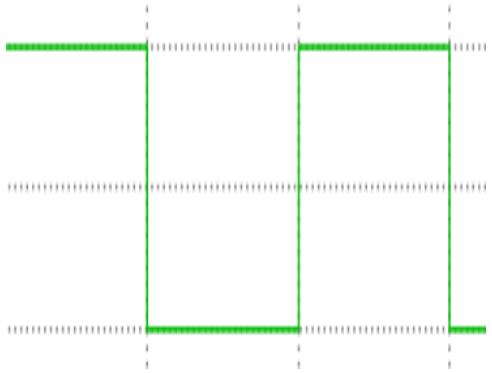
Frequency Spectra





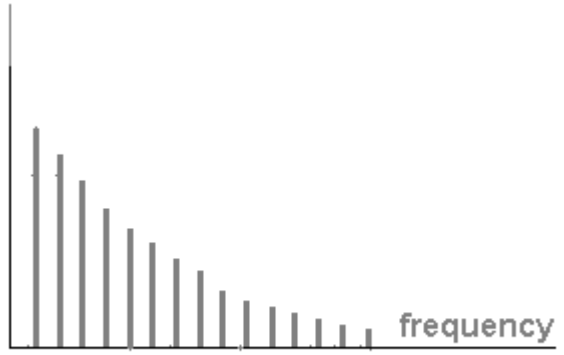


Frequency Spectra



=

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



Demo: https://meettechniek.info/additional/additive-synthesis.html

Fourier Transform

Fourier Transform

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt$$

Inverse Fourier transform

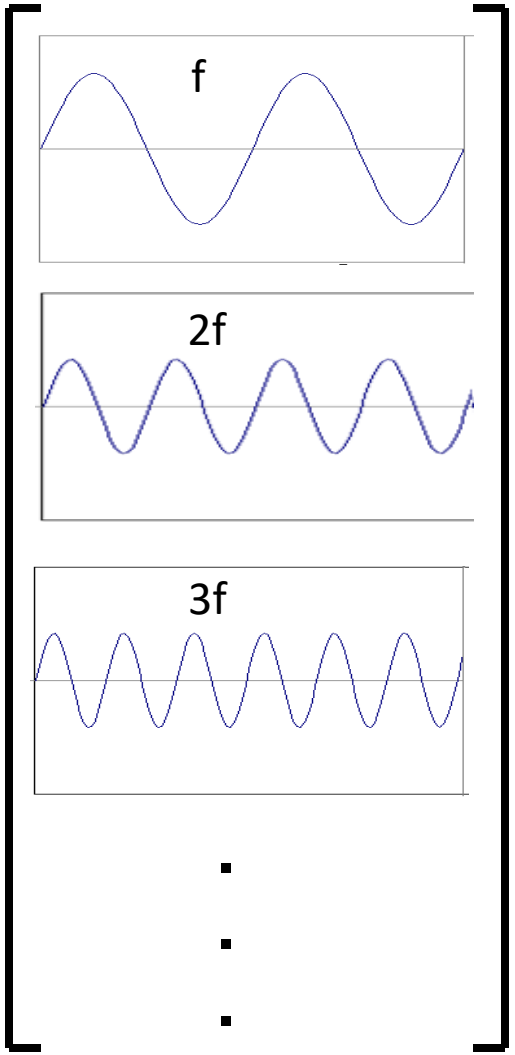
$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega) e^{i\omega t} dt$$

Time f(t)  Frequency f(w)

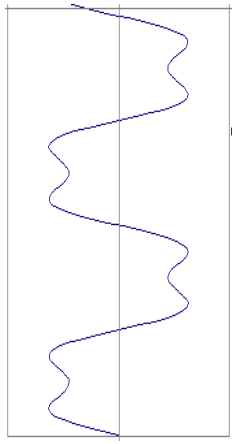
What is transformation? It is a mapping between two domains

FT: Just a change of basis

$$M * f(x) = F(\omega)$$



*



=

1

0

1/3

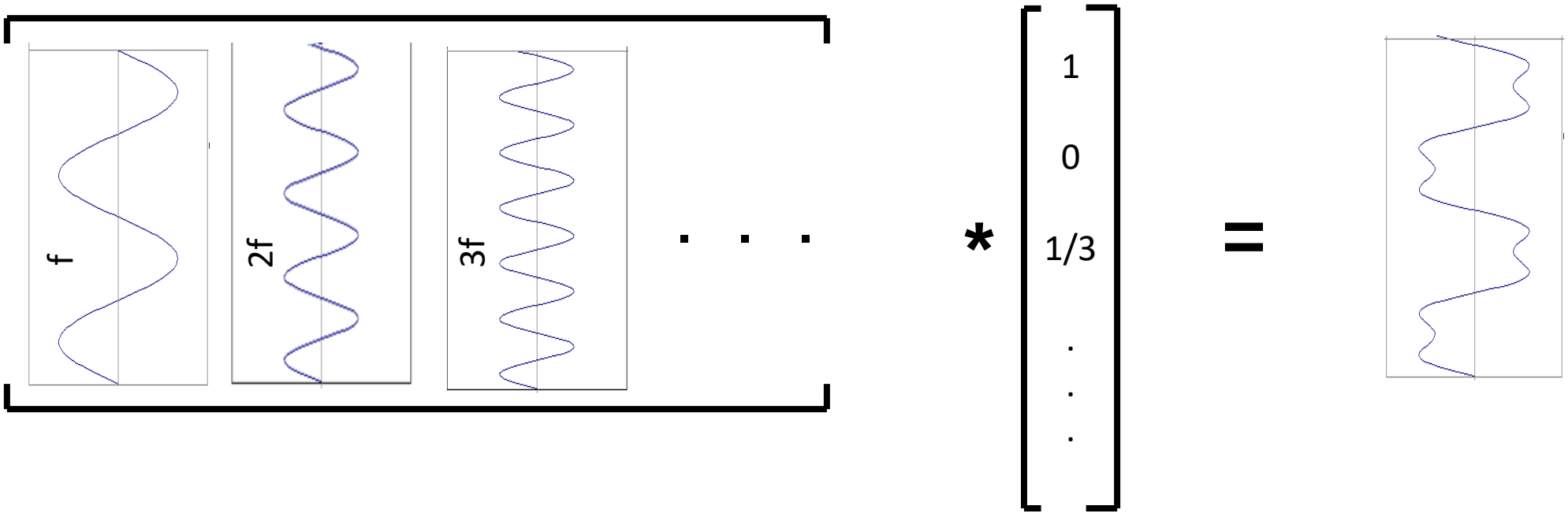
.

.

.

IFT: Just a change of basis

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



FT and DFT

The FT of continuous function, $f(t)$ is

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt$$

$$or, F(u) = \int_{-\infty}^{\infty} f(t) e^{-i2\pi ut} dt$$

$$e^{i\theta} = cos(\theta) + isin(\theta)$$

$$F(u).real = \int_{-\infty}^{\infty} f(t) cos(-2\pi ut) dt$$

$$F(u).imaginary = \int_{-\infty}^{\infty} f(t) sin(-2\pi ut) dt$$

The FT of continuous function, $f(x, y)$ is

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

$$e^{i\theta} = cos(\theta) + isin(\theta)$$

$$F(u).real = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) cos(-2\pi(ux + vy) dxdy$$

$$F(u).imaginary = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) sin(-2\pi(ux + vy) dxdy$$

The FT of discrete function, $f(x)$ with N value is

$$F(\omega) = \sum_{x=0,1,...,N-1} f(x) e^{-i\omega \frac{x}{N}}$$

$$or, F(u) = \sum_{x=0,1,...,N-1} f(x) e^{-i2\pi u \frac{x}{N}}$$

$$e^{i\theta} = cos(\theta) + isin(\theta)$$

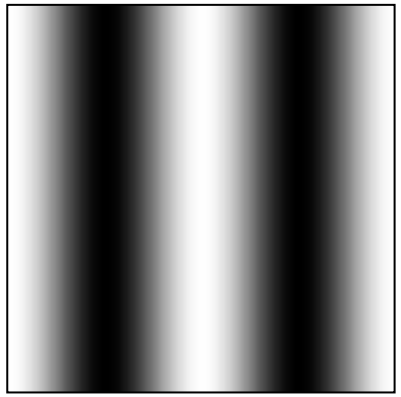
$$F(u).real = \sum_{x=0,1,...,N-1} f(x) cos(-2\pi u \frac{x}{N}) dt$$

$$F(u).imaginary = \sum_{x=0,1,...,N-1} f(x) sin(-2\pi u \frac{x}{N}) dt$$

Spatial Frequency in Image

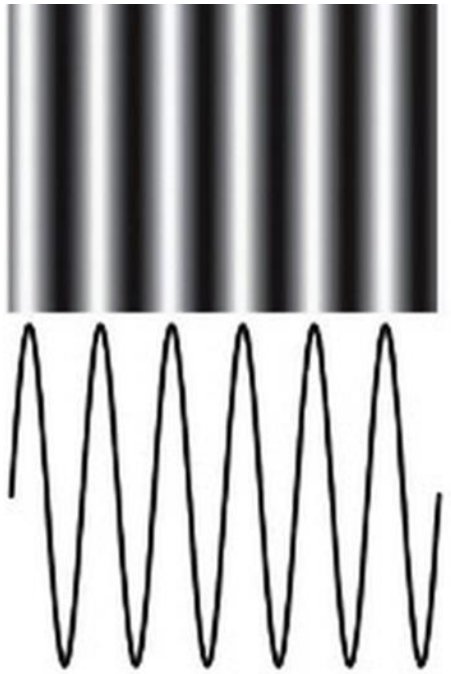
The spatial frequency is a measure of how often sinusoidal components (as determined by the Fourier transform) of the structure repeat per unit of distance.

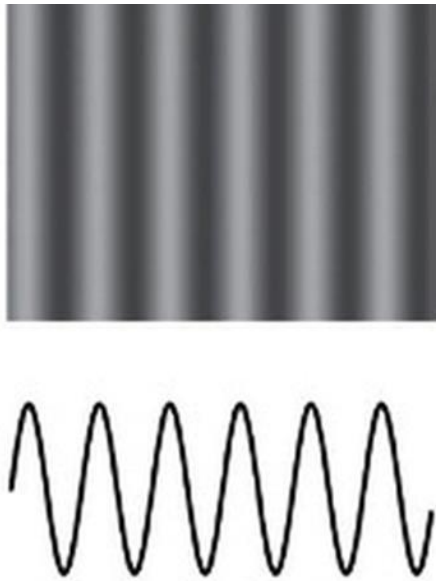
Spatial domain



Fourier domain

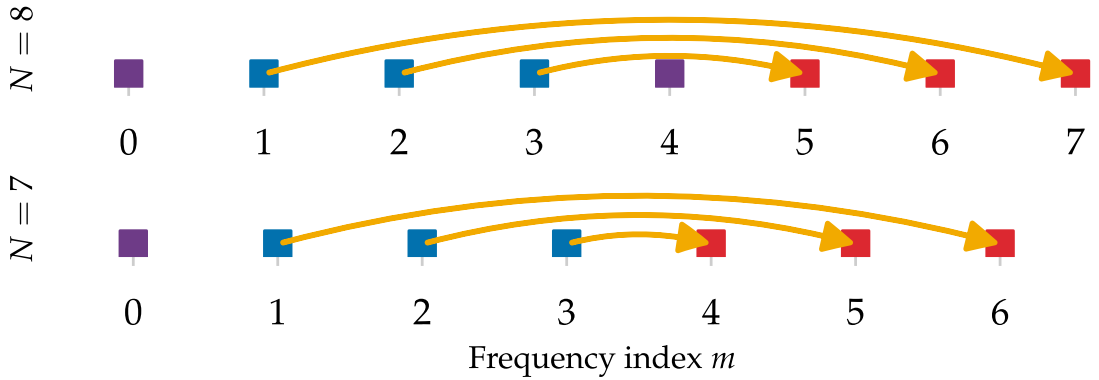
Think how it will look?





Demo: https://bigwww.epfl.ch/demo/ip/demos/FFT/

Conjugate symmetry



The DFT conjugate symmetry property relates X[m] with X[N-m] (arrows). Indices m=0 and m=N/2 (if N is even) are only related to themselves.
If the signal length is even (top, N=8), then the first M= 1+N/2 components are sufficient to determine all N components.
If the signal length is odd (bottom, N=7), then M= 1+ (N-1)/2 are sufficient.

Now give a try for previous example….

DFT component X[m] is the complex conjugate of component X[N−m] (and vice versa): their real parts are identical, and their imaginary parts are negatives of each-other.

$$\begin{aligned} e^{-\mathrm{j}\cdot\theta} &= \cos(-\theta) + \mathrm{j}\cdot\sin(-\theta) \\ &= \cos(\theta) - \mathrm{j}\cdot\sin(\theta) \\ &= \overline{e^{\mathrm{j}\cdot\theta}} \end{aligned}$$

$$\begin{aligned} \exp\left(-\mathrm{j}\cdot\omega\cdot\frac{N-m}{N}\right) &= \exp\left(+\mathrm{j}\cdot\omega\cdot\frac{m-N}{N}\right) && \text{cancelling negatives} \\ &= \exp\left(\mathrm{j}\cdot\omega\cdot\frac{m}{N}\right) && \text{add rotations } \omega\cdot N/N \\ &= \overline{\exp\left(-\mathrm{j}\cdot\omega\cdot\frac{m}{N}\right)} && \text{conjugation.} \end{aligned}$$

$$\begin{aligned} X[N-m] &= \sum_{n=0}^{N-1} x[n]\cdot\exp\left(-\mathrm{j}\cdot\omega\cdot\frac{N-m}{N}\right) \\ &= \sum_{n=0}^{N-1} x[n]\cdot\overline{\exp\left(-\mathrm{j}\cdot\omega\cdot\frac{m}{N}\right)} \\ &= \overline{\sum_{n=0}^{N-1} x[n]\cdot\exp\left(-\mathrm{j}\cdot\omega\cdot\frac{m}{N}\right)} \\ &= \overline{X[m]}. \end{aligned}$$

Frequency Spectra for 1D signals

I[m]

M pixels

(0,0) (M,0)

Spatial domain

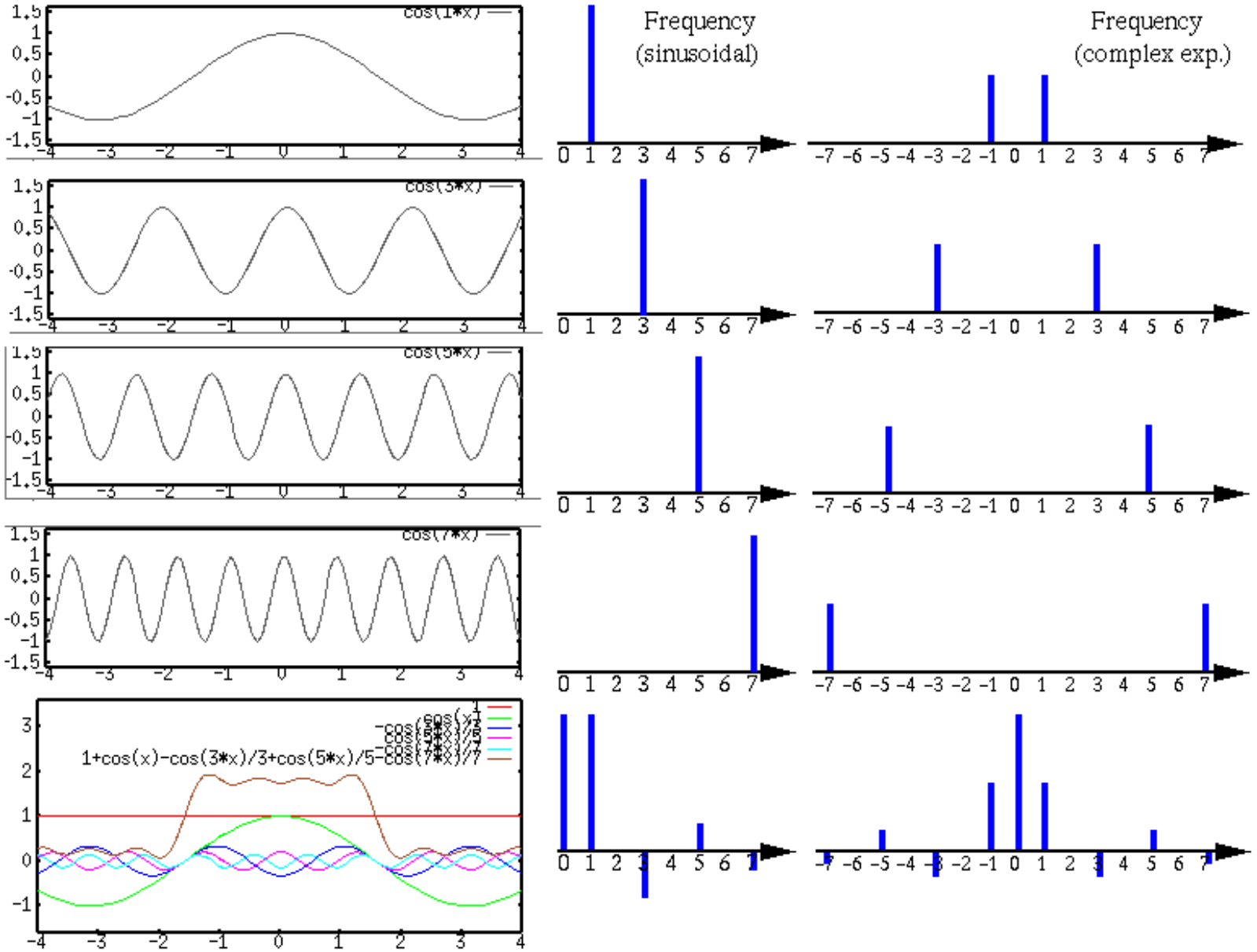
Transform Fourier

F[u]

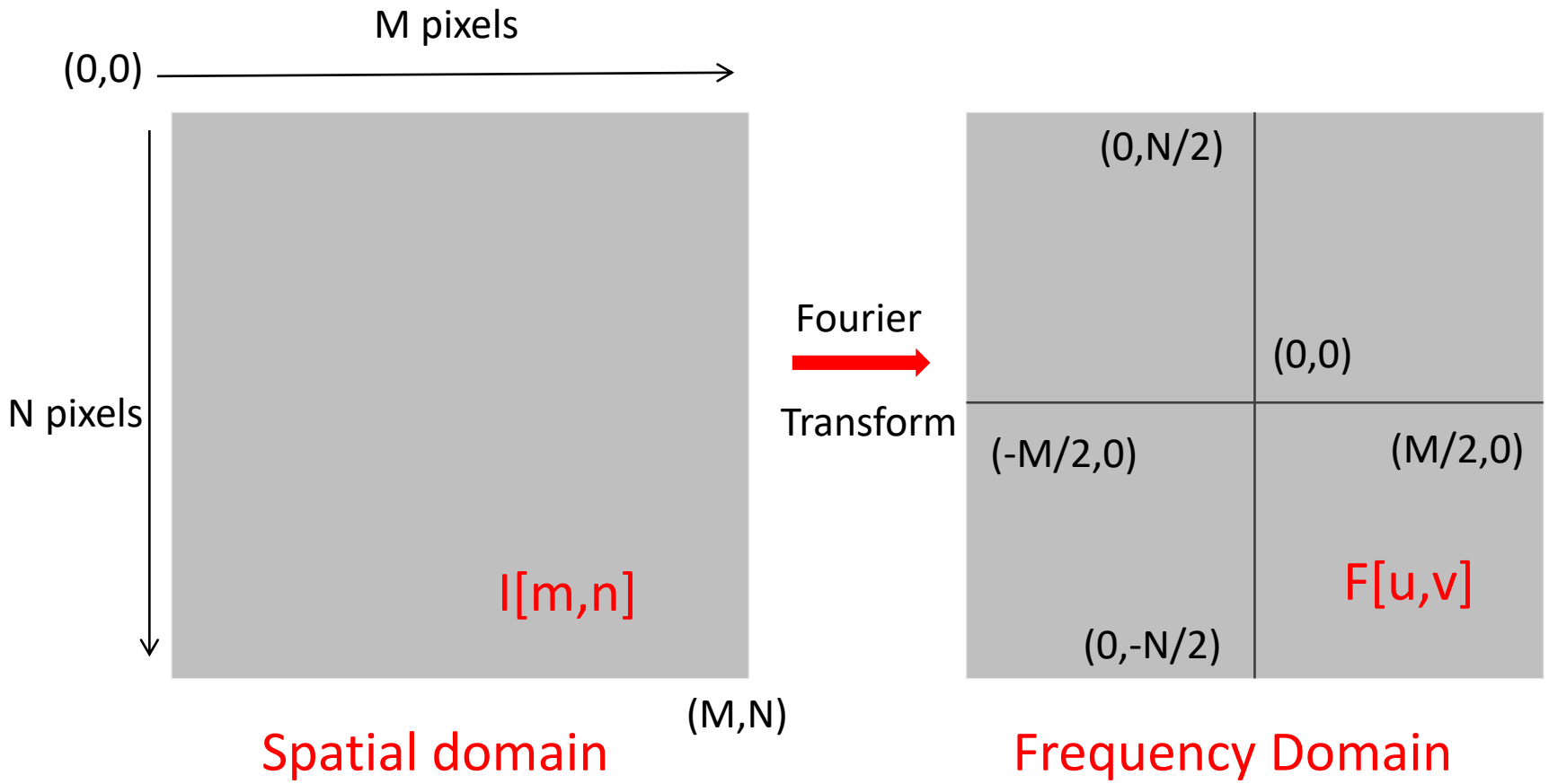
(0,0)

(-M/2,0) (M/2,0)

Frequency Domain

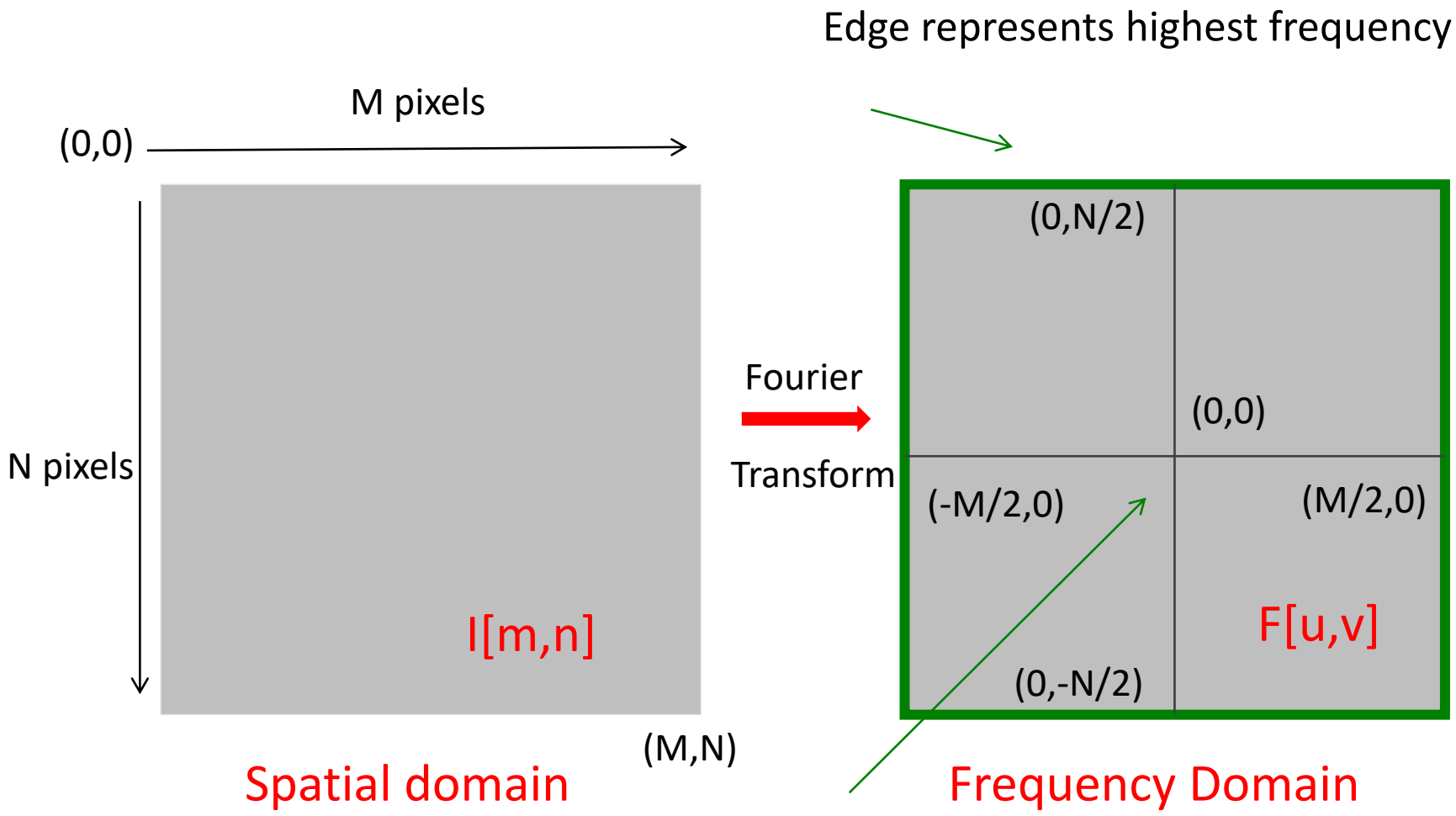


Fourier Transform of Images

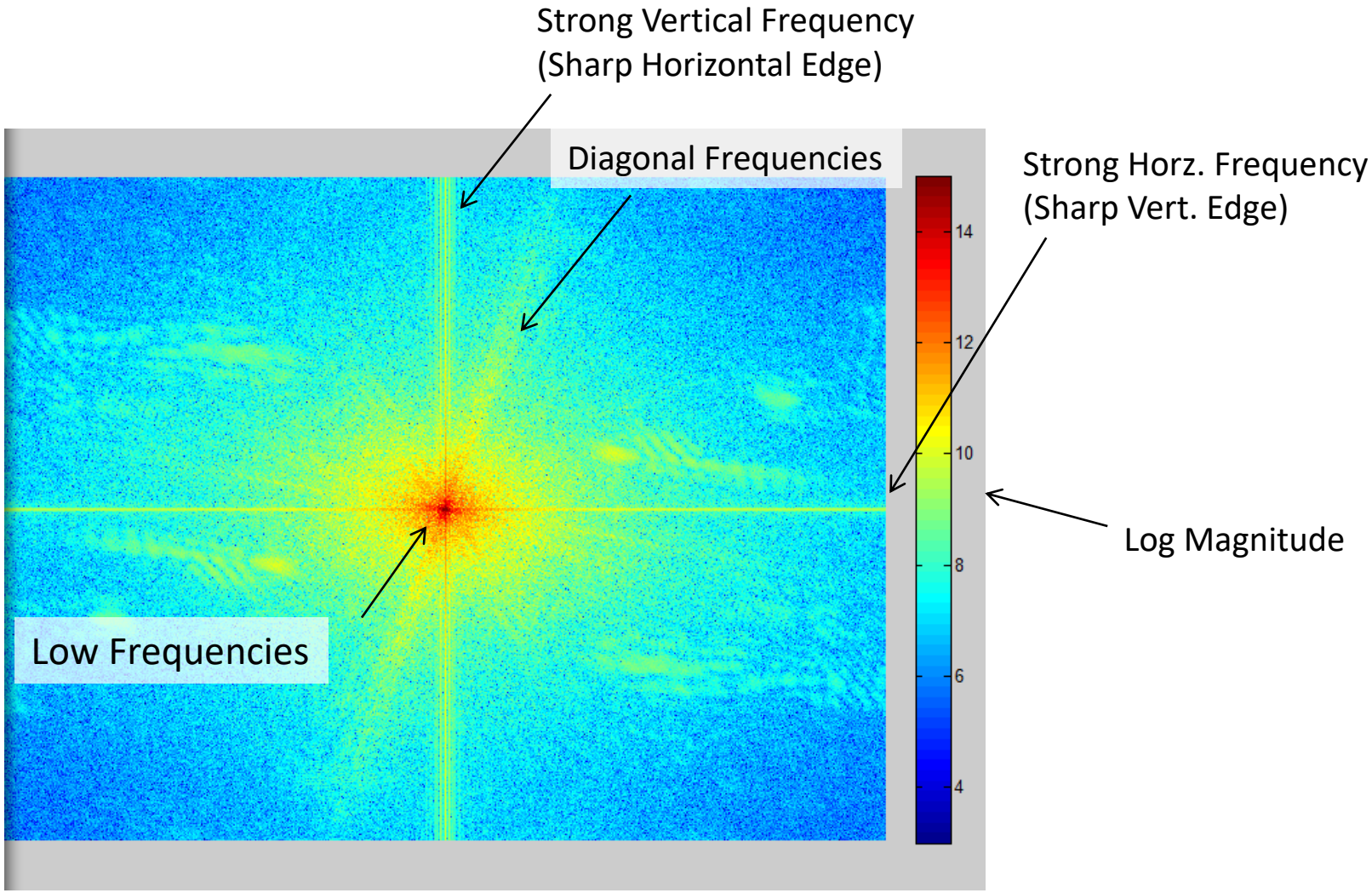


2D FFT can be composed as two discrete Fourier Transforms in 1 dimension

Fourier Transform of Images

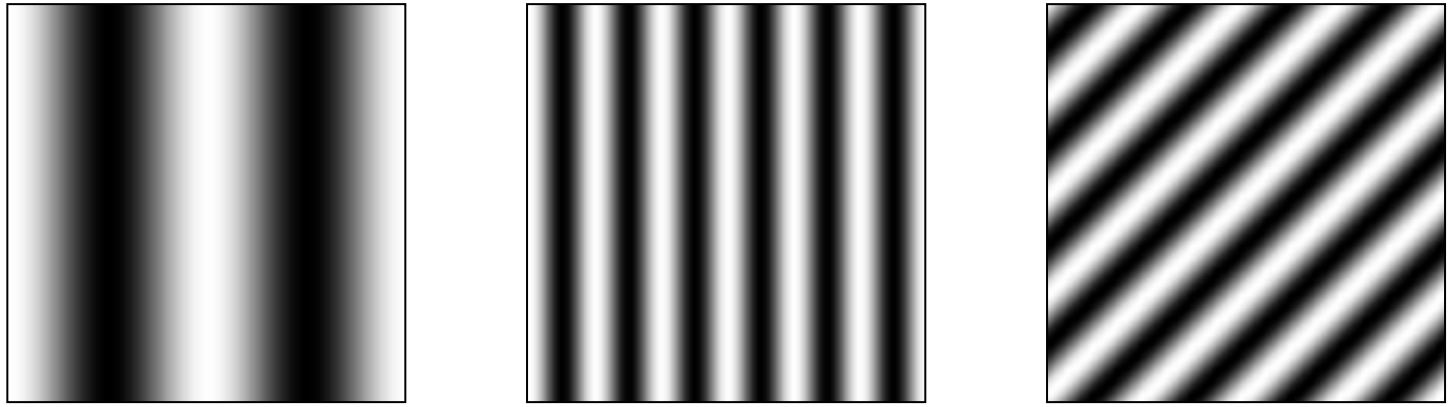


Fourier Transform of Image: Example

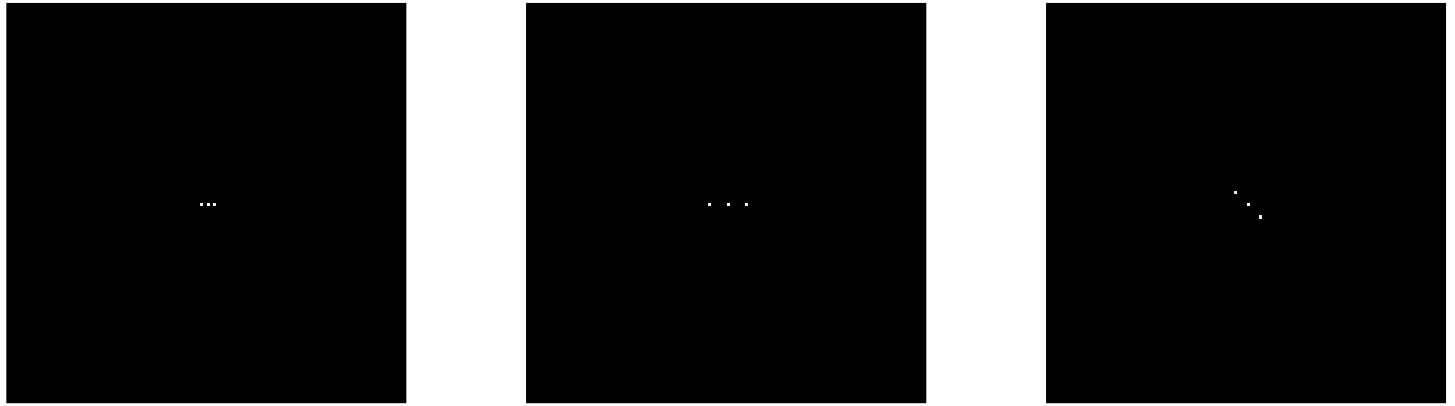


Fourier analysis in images

Intensity Image



Fourier Image

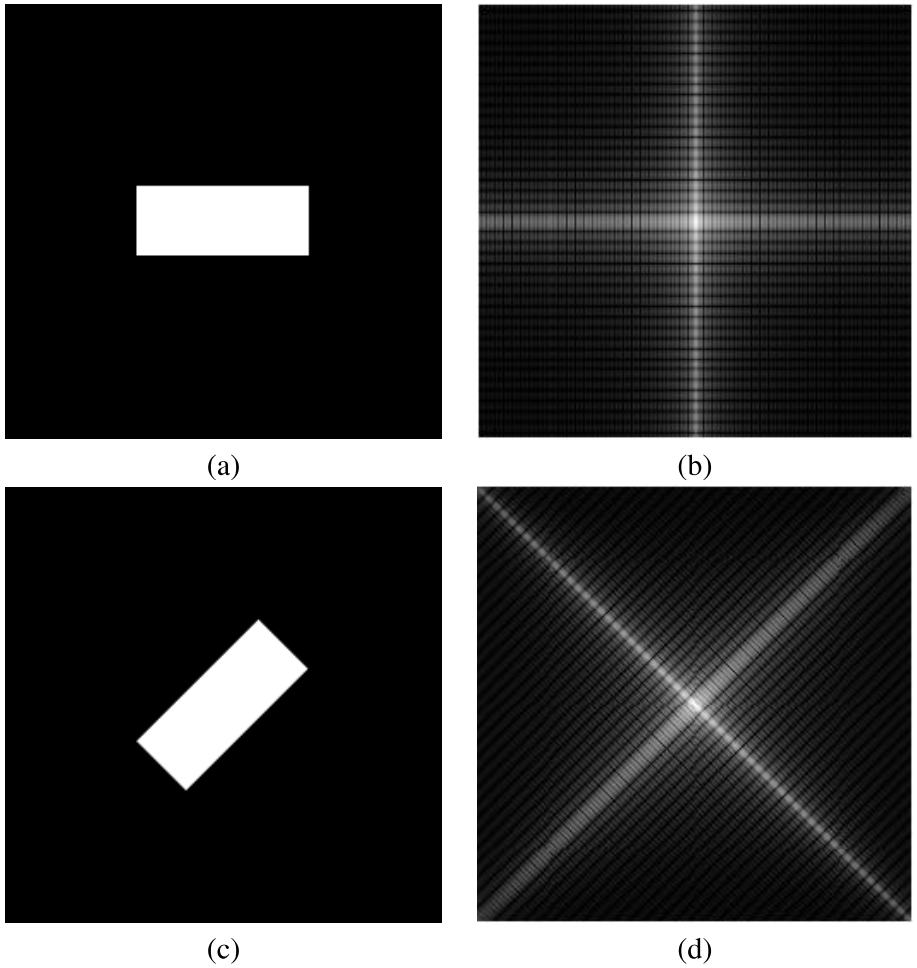


http://sharp.bu.edu/~slehar/fourier/fourier.html#filtering

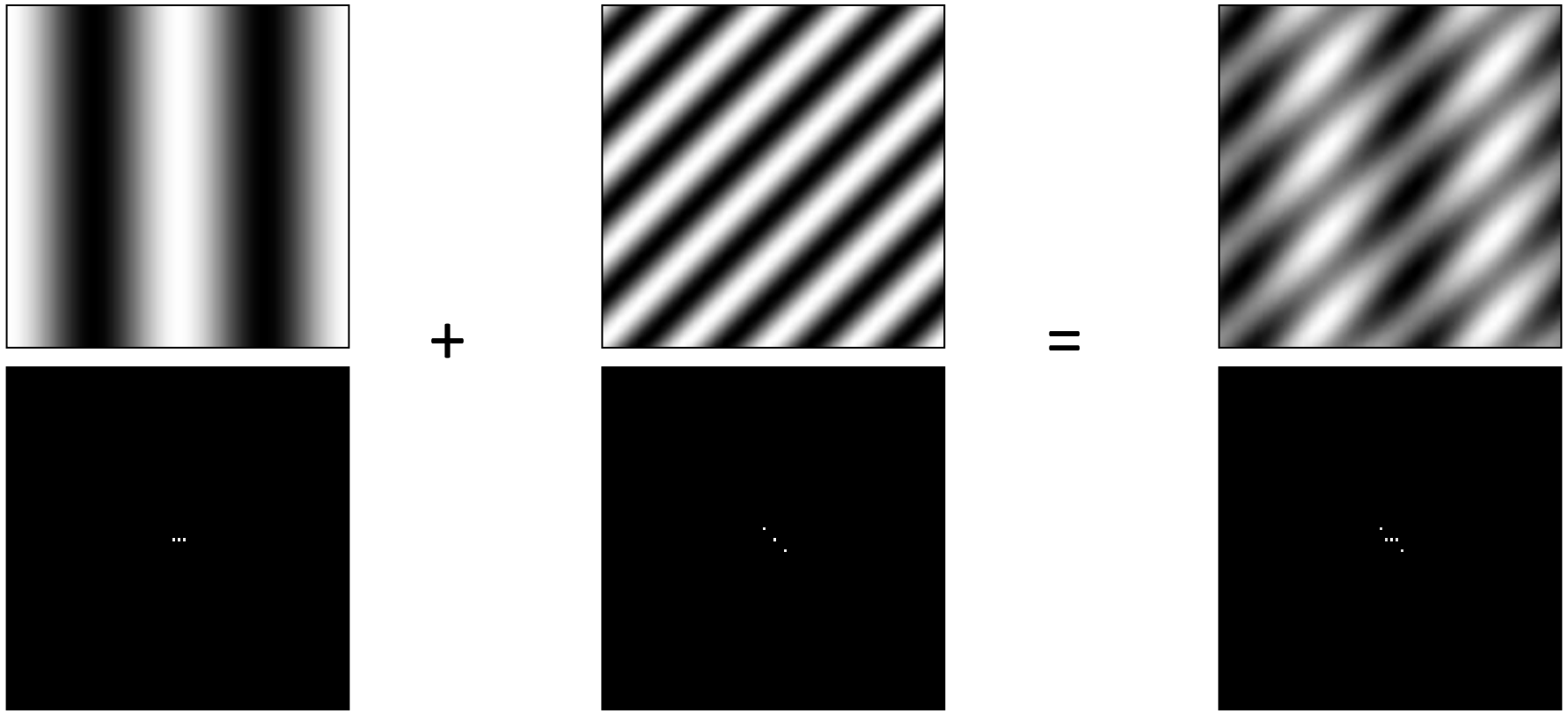
Fourier Transform: properties

Rotation

- If an image is rotated by a certain angle ϑ, its 2D FT will be rotated by the same angle.



Signals can be composed



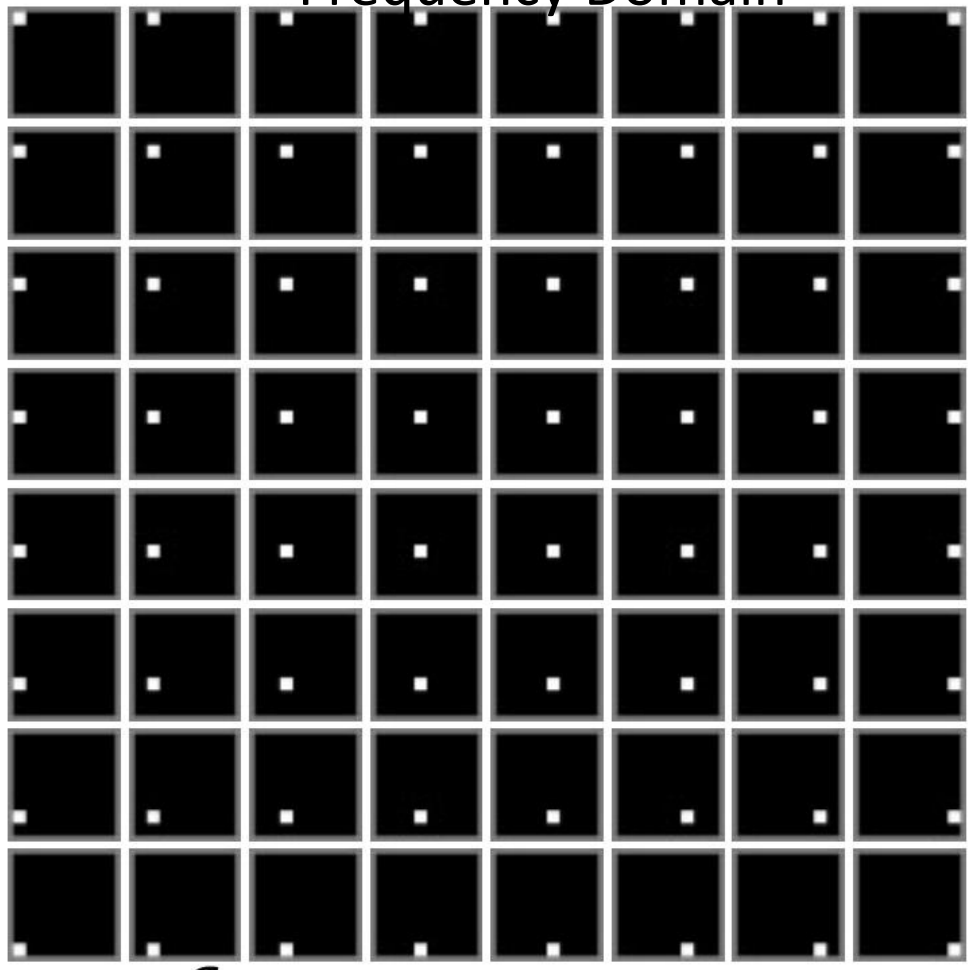
http://sharp.bu.edu/~slehar/fourier/fourier.html#filtering
More: http://www.cs.unm.edu/~brayer/vision/fourier.html

Fourier Bases

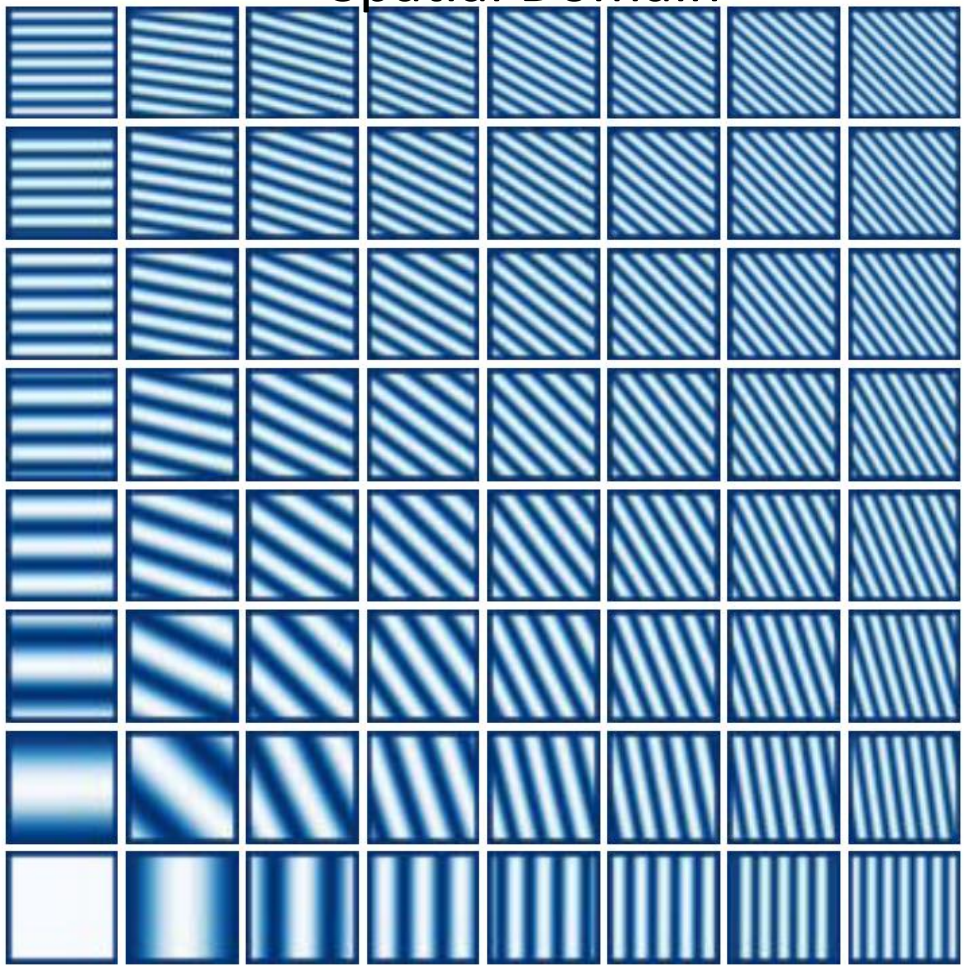
Teases away ‘fast vs. slow’ changes in the image.

Image as a sum of basis images

Frequency Domain



Spatial Domain



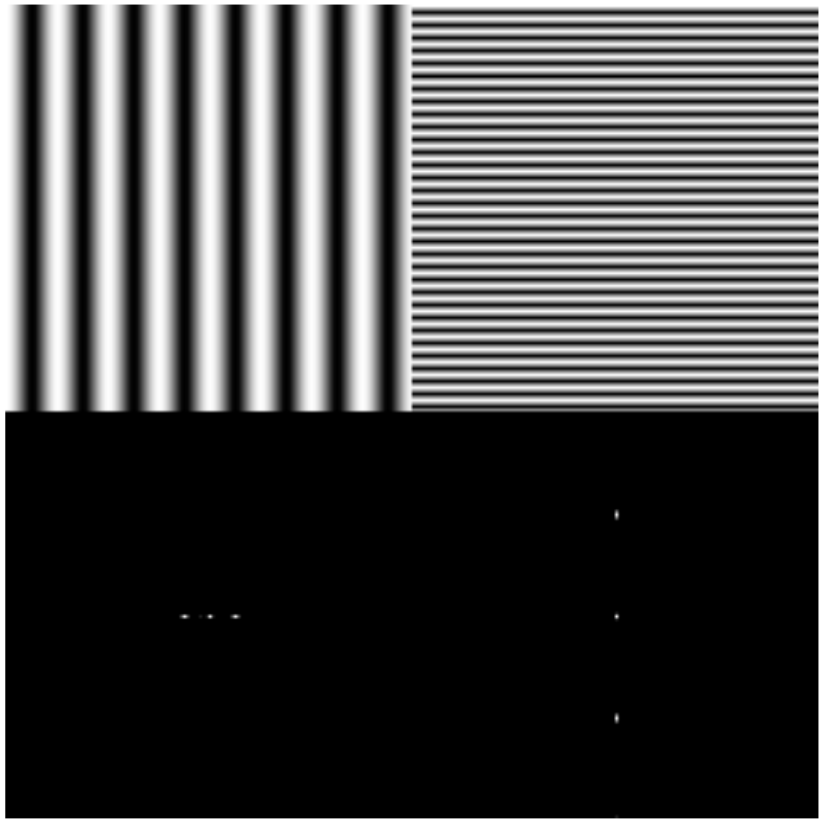
This change of basis is the Fourier Transform

Fourier analysis in images

The abs gives you the magnitude of the complex FFT output, and using a log scale gives you much more dynamic range when displaying the magnitudes using 8 bit grey scale pixels.

The logarithmic scale is most commonly used in spectrum analyzers and allows the user to view a wide range of amplitudes, making it easier to identify weaker signals.

If you linearly stretch the values from max=white to min=black, you will see only one pixel white, and the rest will be black or nearly black. The log transform allows viewing values with such large difference in magnitude



http://www.csce.uark.edu/~jgauch/5683/notes/ch04a.pdf

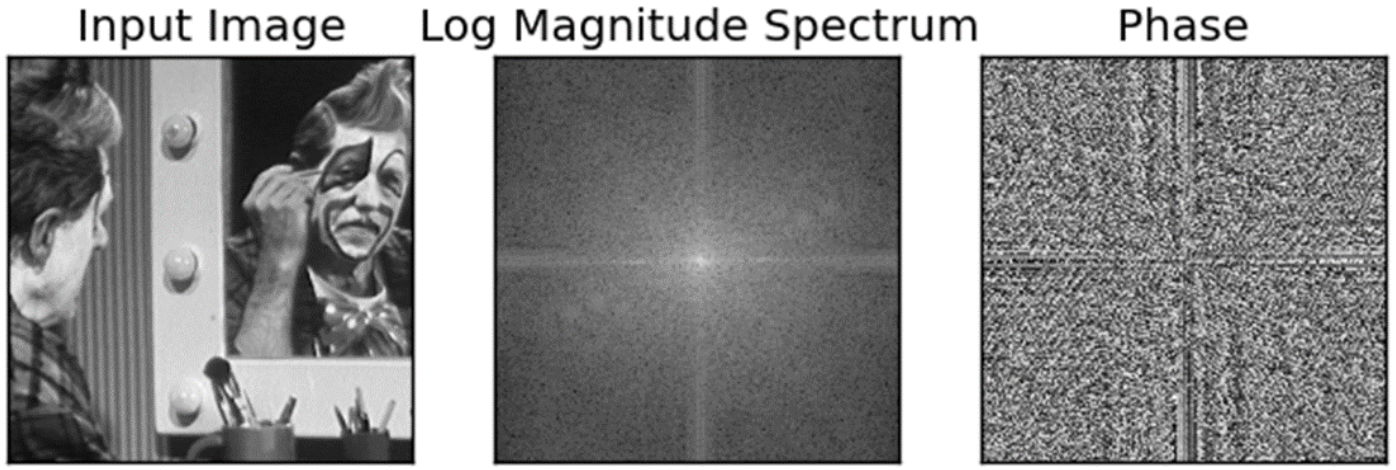
Phase and Magnitude

FT stores these at each frequency:

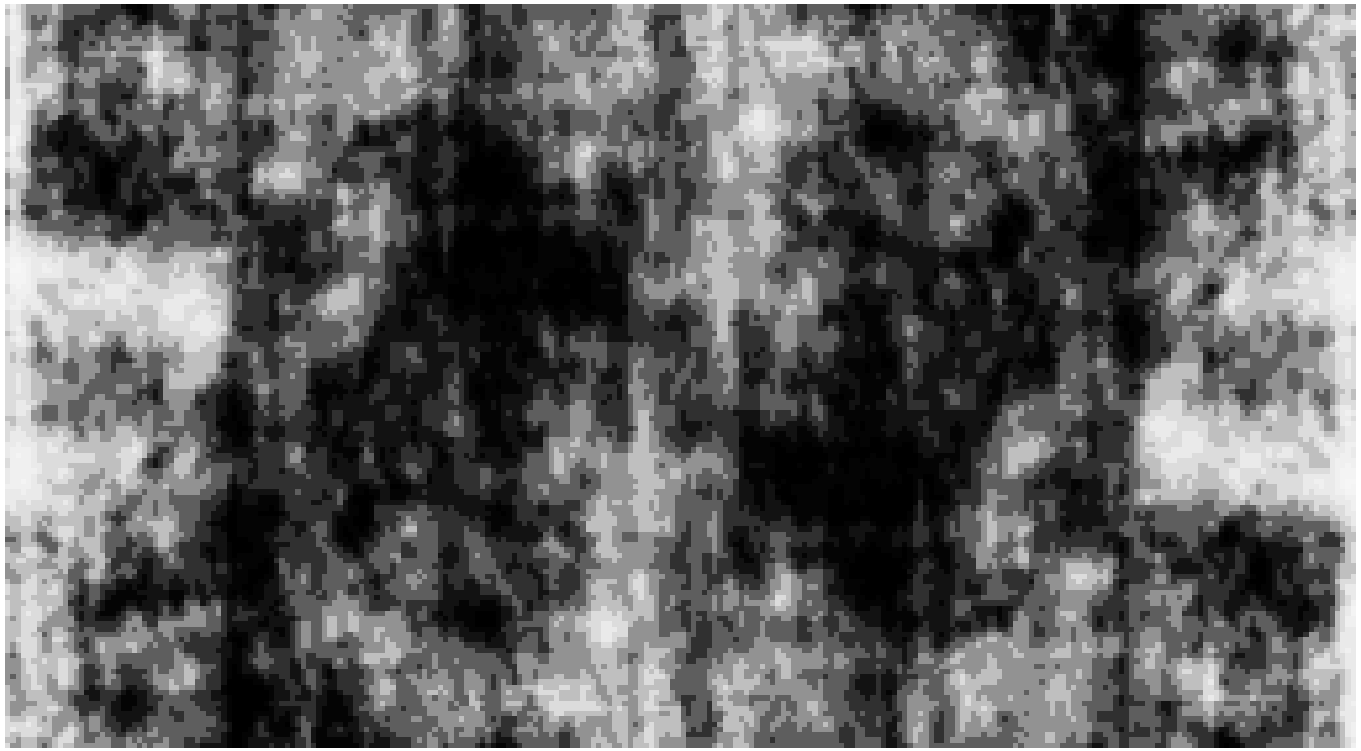
- Magnitude (M) encodes how much signal there is at a particular frequency
- Phase (φ) encodes spatial information (indirectly)

$$M = \pm\sqrt{R(\omega)^2 + I(\omega)^2}$$

$$\varphi = \tan^{-1}\frac{I(\omega)}{R(\omega)}$$



Same frequency, abstract nonsense

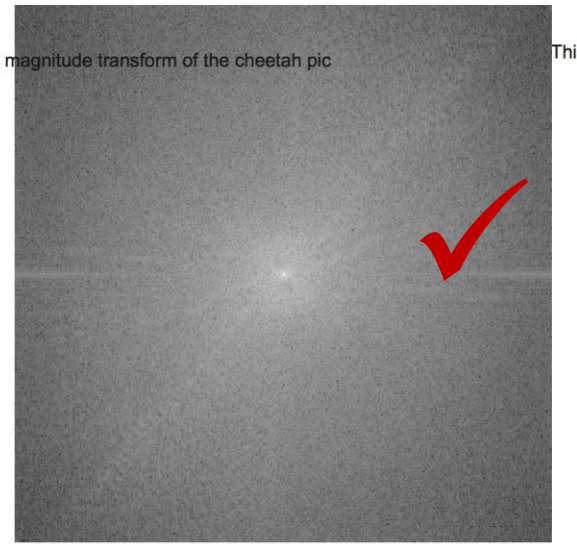


What happens if you reverse Fourier without the phase?

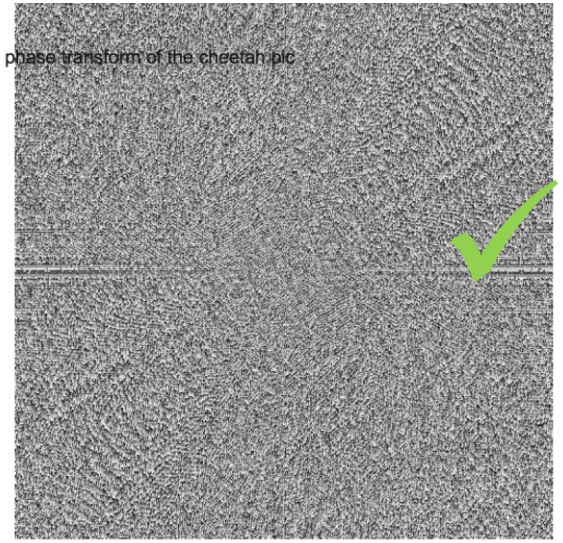
- Open an image
- Compute magnitude and phase of image
- Reverse the ONLY magnitude back
- Image histogram equalize the reversed FFT image
- Display the image
- What happens?

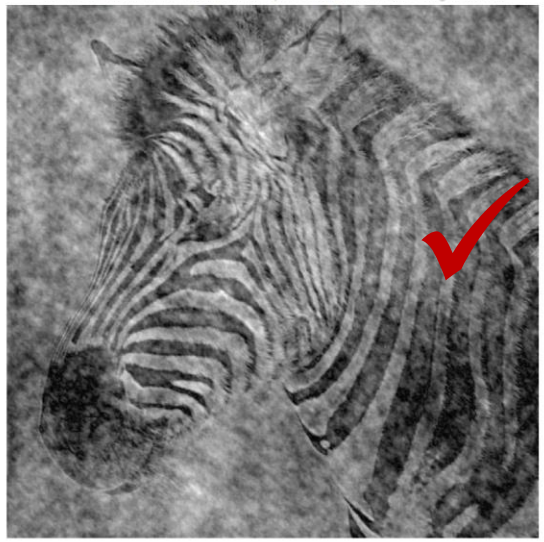
Phase and Magnitude

This is the magnitude transform of the cheetah pic

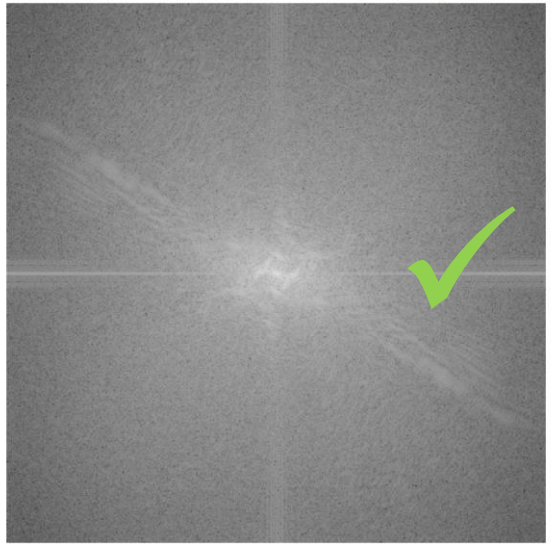


This is the phase transform of the cheetah pic

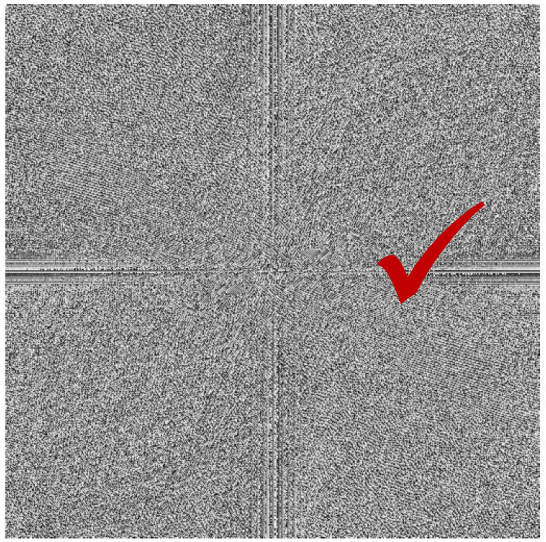


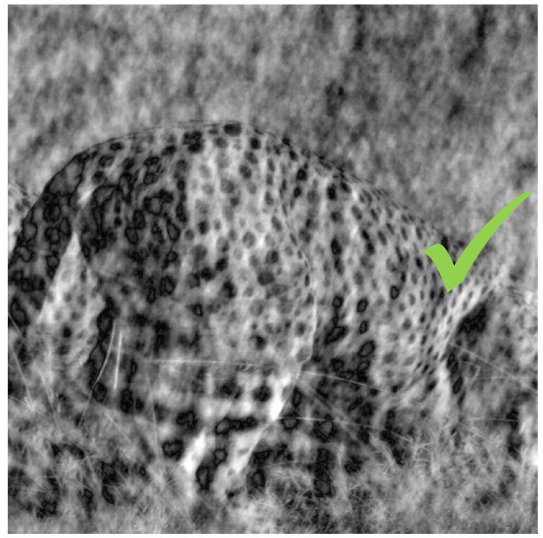


This is the magnitude transform of the zebra pic

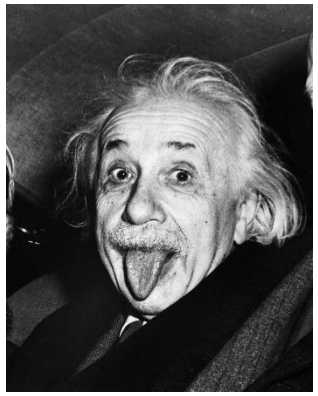


This is the phase transform of the zebra pic



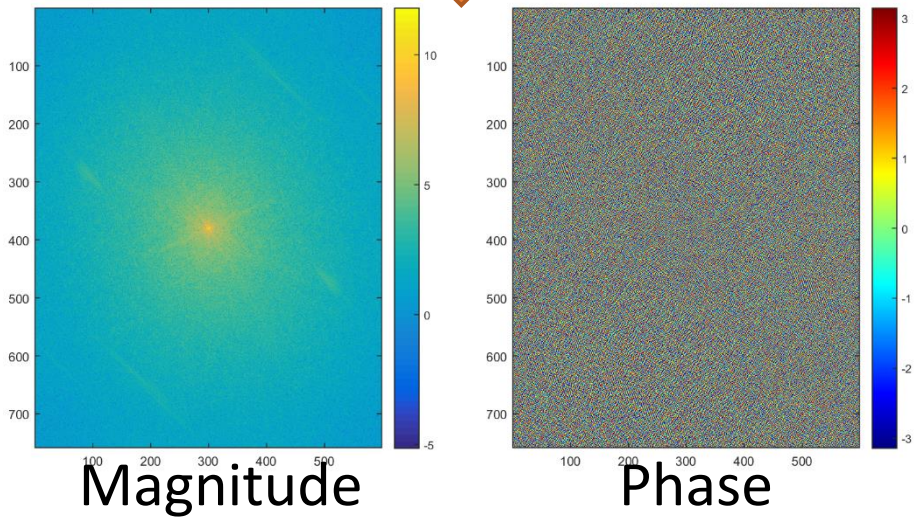


Phase vs. Magnitude



Intensity image

FFT

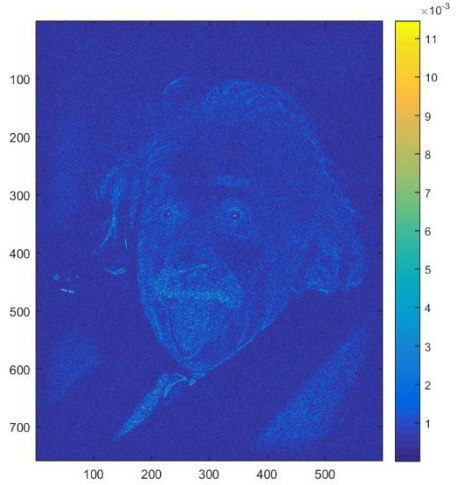


Magnitude

Phase

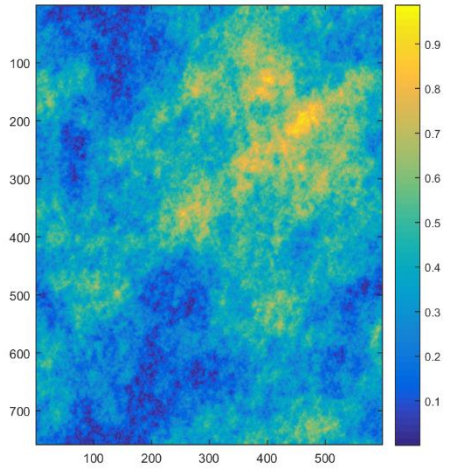
Use random magnitude

Inverse FFT



Use random phase

Inverse FFT



Frequency Filtering: Main Steps

1. Take the FT of f(x): $F(f(x))$

2. Manipulate frequencies by applying some filter: $D(F(f(x)))$

3. Convert back to spatial domain: $\hat{f}(x)=F^{-1}(D(F(f(x))))$

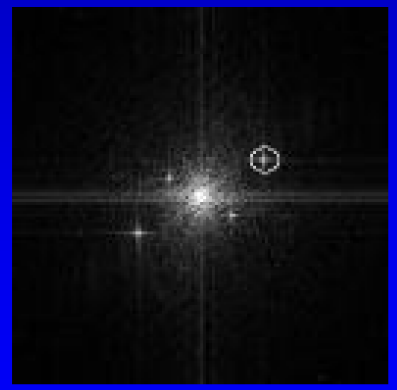
Let's look at some examples….

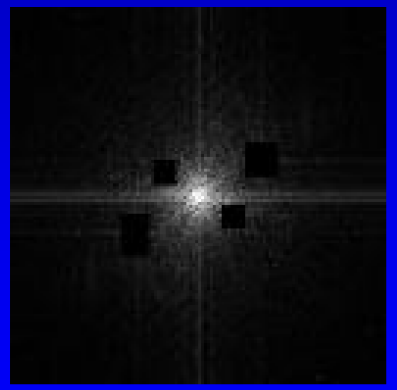
Demo: https://bigwww.epfl.ch/demo/ip/demos/FFT-filtering/

Noise Removal

FFT

Inverse FFT





Noise Pattern Stands Out as Four Spikes

Four Noise Spikes Removed

Edit FFT

Source: www.mediacy.com/apps/fft.htm, Image Pro Plus FFT Example. Last seen online in 2004.

Fourier Transform and Convolution

Let $g = f * h$

Then $$G(u) = \int_{-\infty}^{\infty} g(x) e^{-i2\pi ux} dx$$

$$= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(\tau) h(x-\tau) e^{-i2\pi ux} d\tau dx$$

$$= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \left[f(\tau) e^{-i2\pi u\tau} d\tau \right] \left[h(x-\tau) e^{-i2\pi u(x-\tau)} dx \right]$$

$$= \int_{-\infty}^{\infty} \left[f(\tau) e^{-i2\pi u\tau} d\tau \right] \int_{-\infty}^{\infty} \left[h(x') e^{-i2\pi ux'} dx' \right]$$

$$= F(u) H(u)$$

Convolution in spatial domain

$\Leftrightarrow$ Multiplication in frequency domain

Fourier Transform and Convolution

Spatial Domain *(x)* | Frequency Domain *(u)*

$$g = f * h \quad \longleftrightarrow \quad G = FH$$

$$g = fh \quad \longleftrightarrow \quad G = F * H$$

So, we can find *g(x)* by Fourier transform

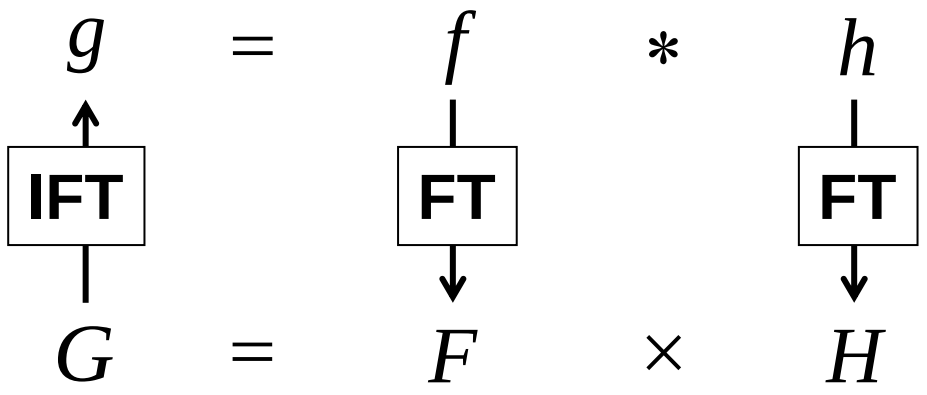
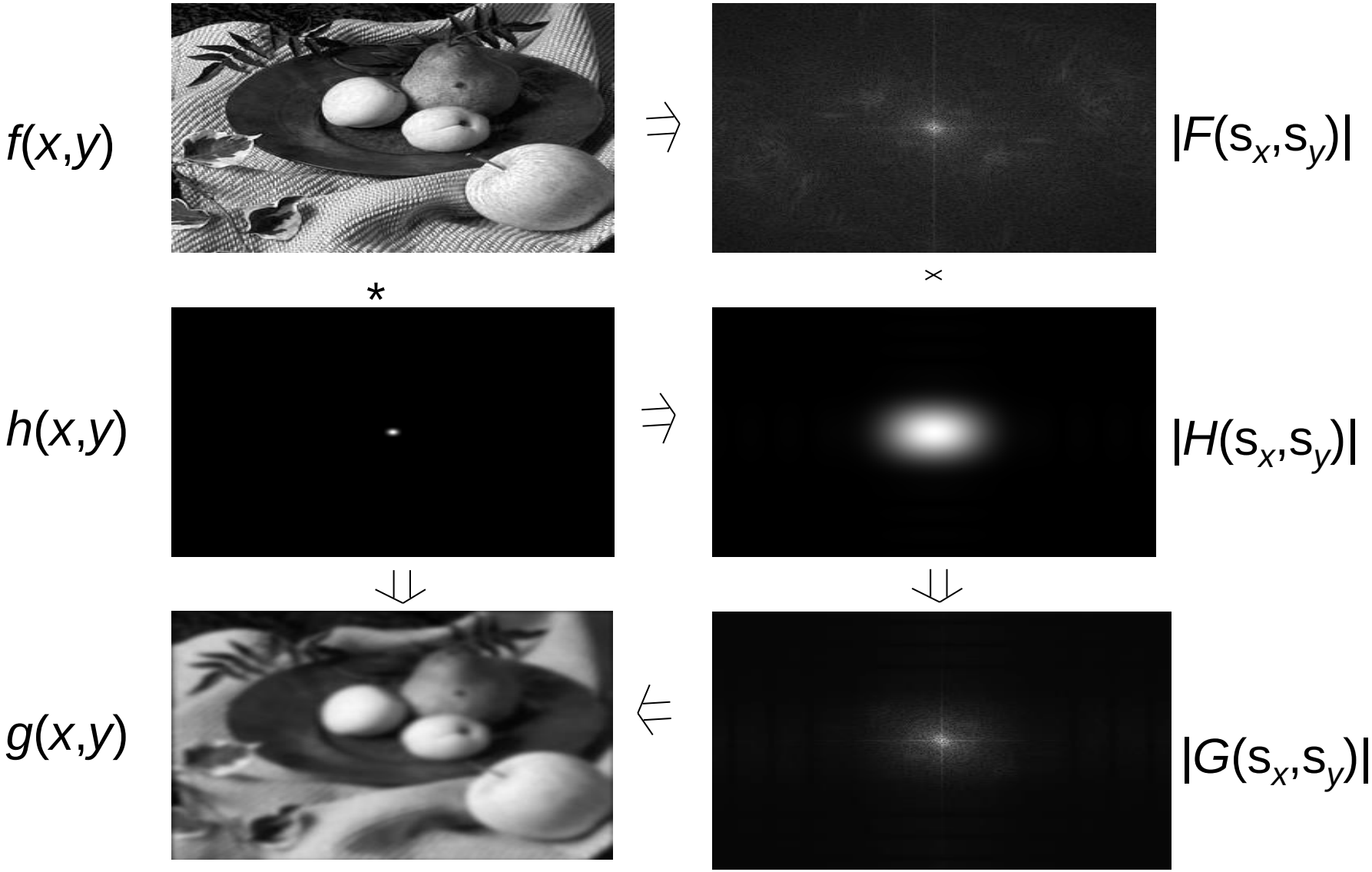


Image Processing using FT and Convolution

Convolution is Multiplication in Fourier Domain

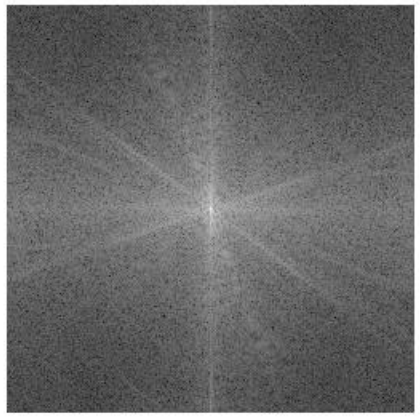


Example of Low pass filtering

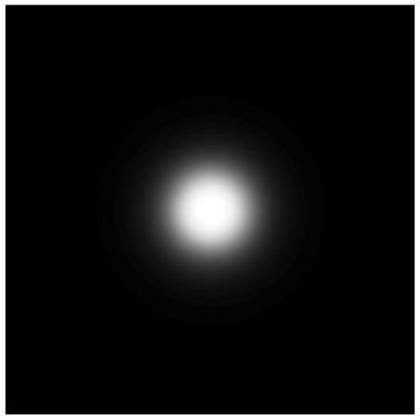
Image Processing using FT and Convolution

Original image

FFT of original image



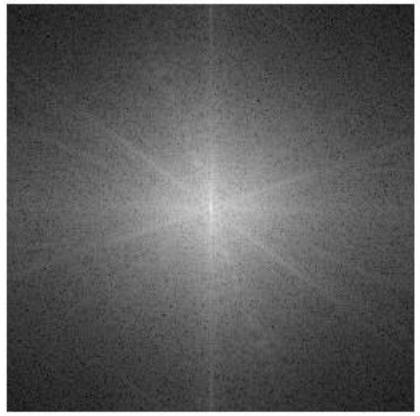
Low-pass filter



Let the low frequencies pass and eliminating the high frequencies.

Low-pass image

FFT of low-pass image



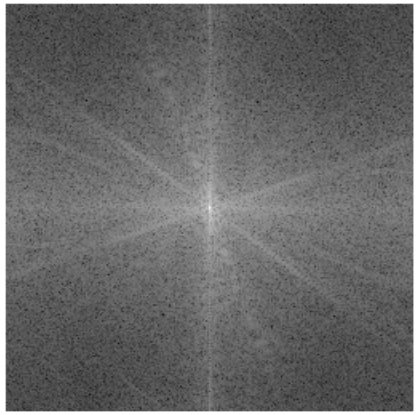
Generates image with overall shading, but not much detail

Another example of Low pass filtering

Image Processing using FT and Convolution

Original image

FFT of original image

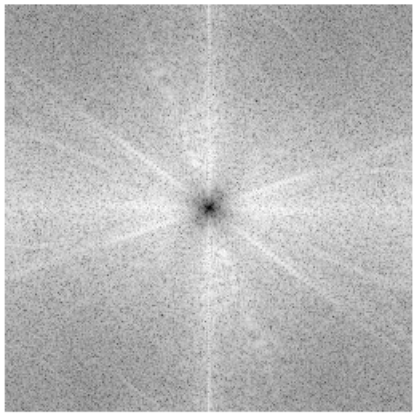


High-pass filter

Lets through the high frequencies (the detail), but eliminates the low frequencies (the overall shape). It acts like an edge enhancer.

High-pass image

FFT of high-pass image

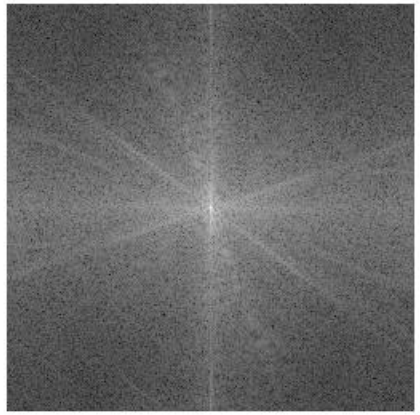


An example of High pass filtering

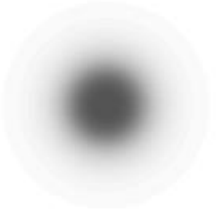
Image Processing using FT and Convolution

Original image

FFT of original image

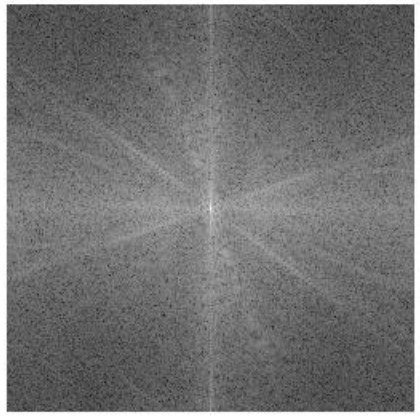


High-boost filter



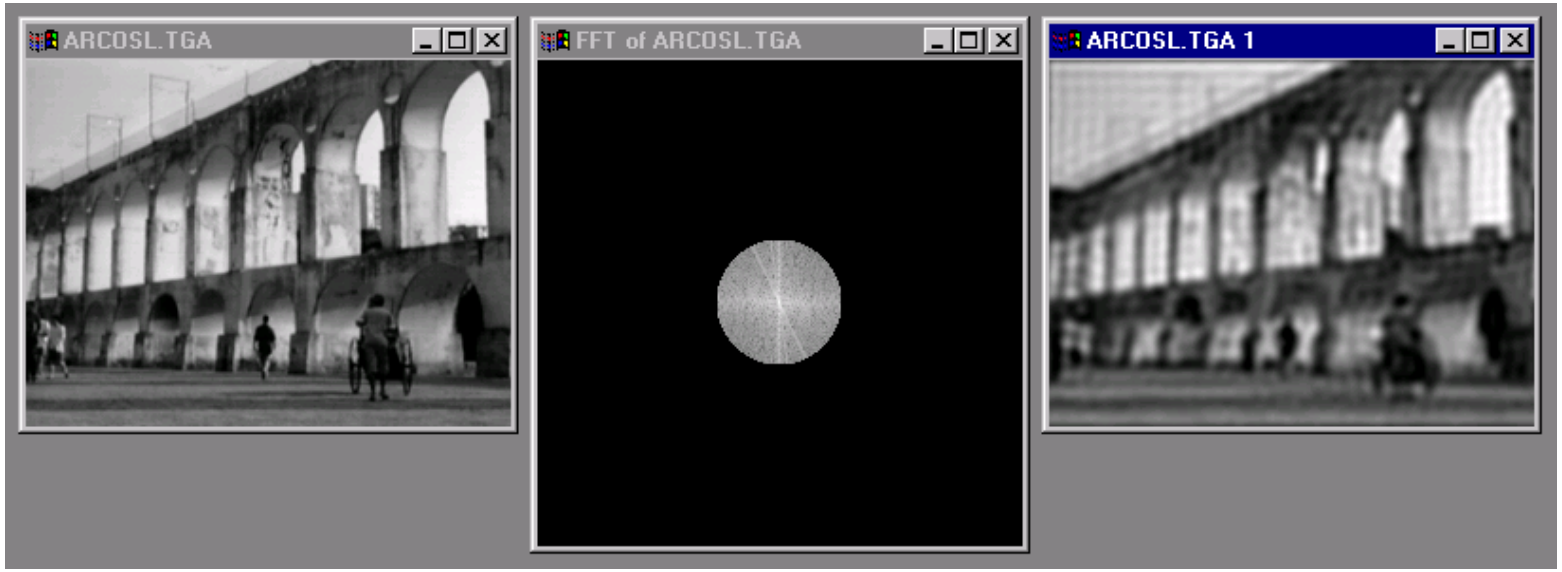
High boosted image

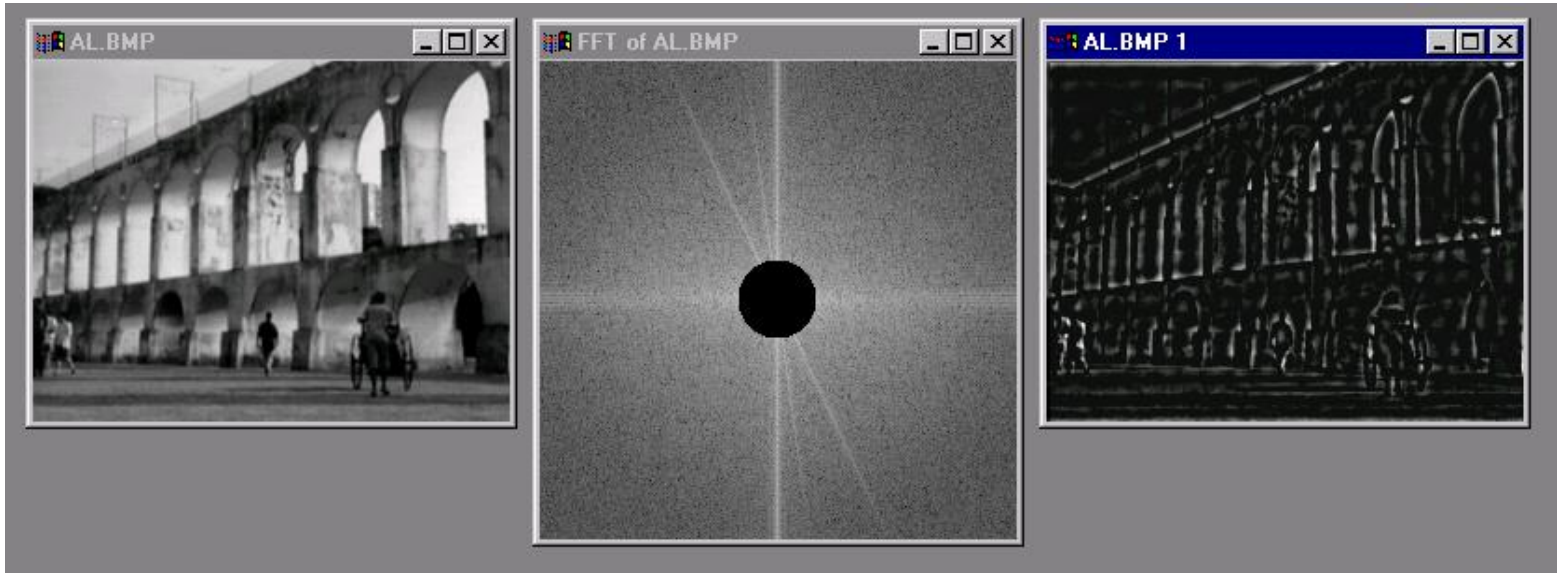
FFT of high boosted image



An example of boosting high frequencies

Information in frequencies of FT





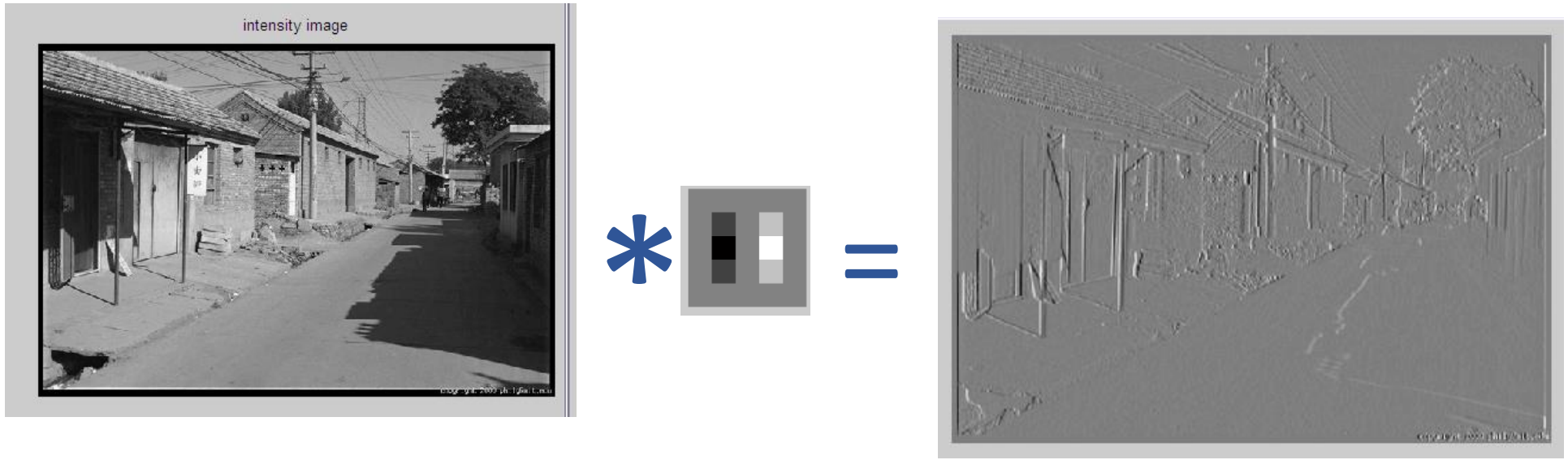
Information in frequencies of FT

Filtering

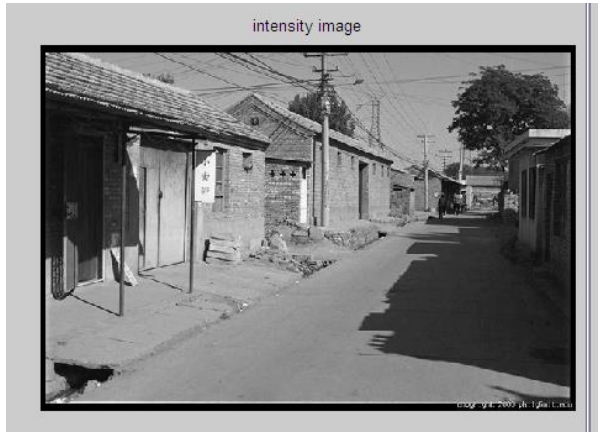
- We pixel-wise multiply the DFT (Discrete Fourier Transform) of the input image by the DFT of the filter
- Frequencies where the magnitude of the response of the filter are new zero (black in the images) will be removed.

Filtering in spatial domain

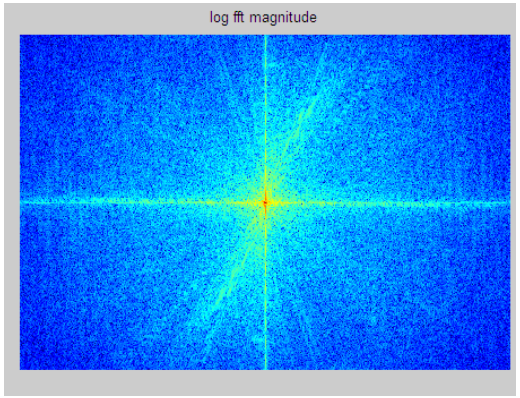
1	0	-1
2	0	-2
1	0	-1



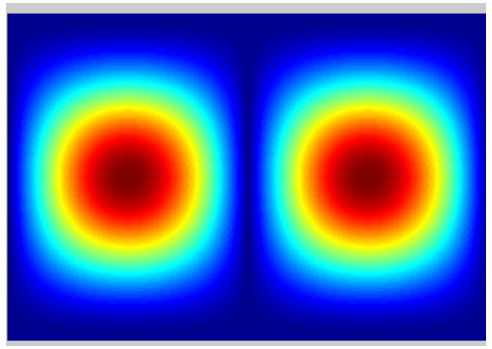
Filtering in frequency domain

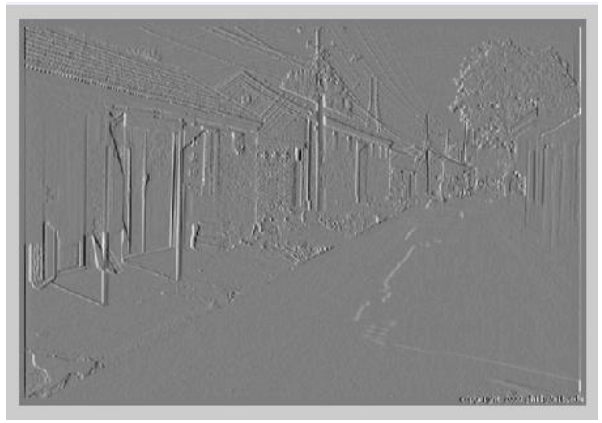


FFT

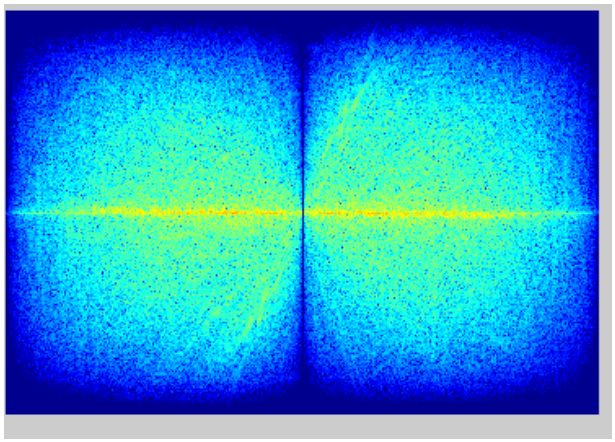


FFT





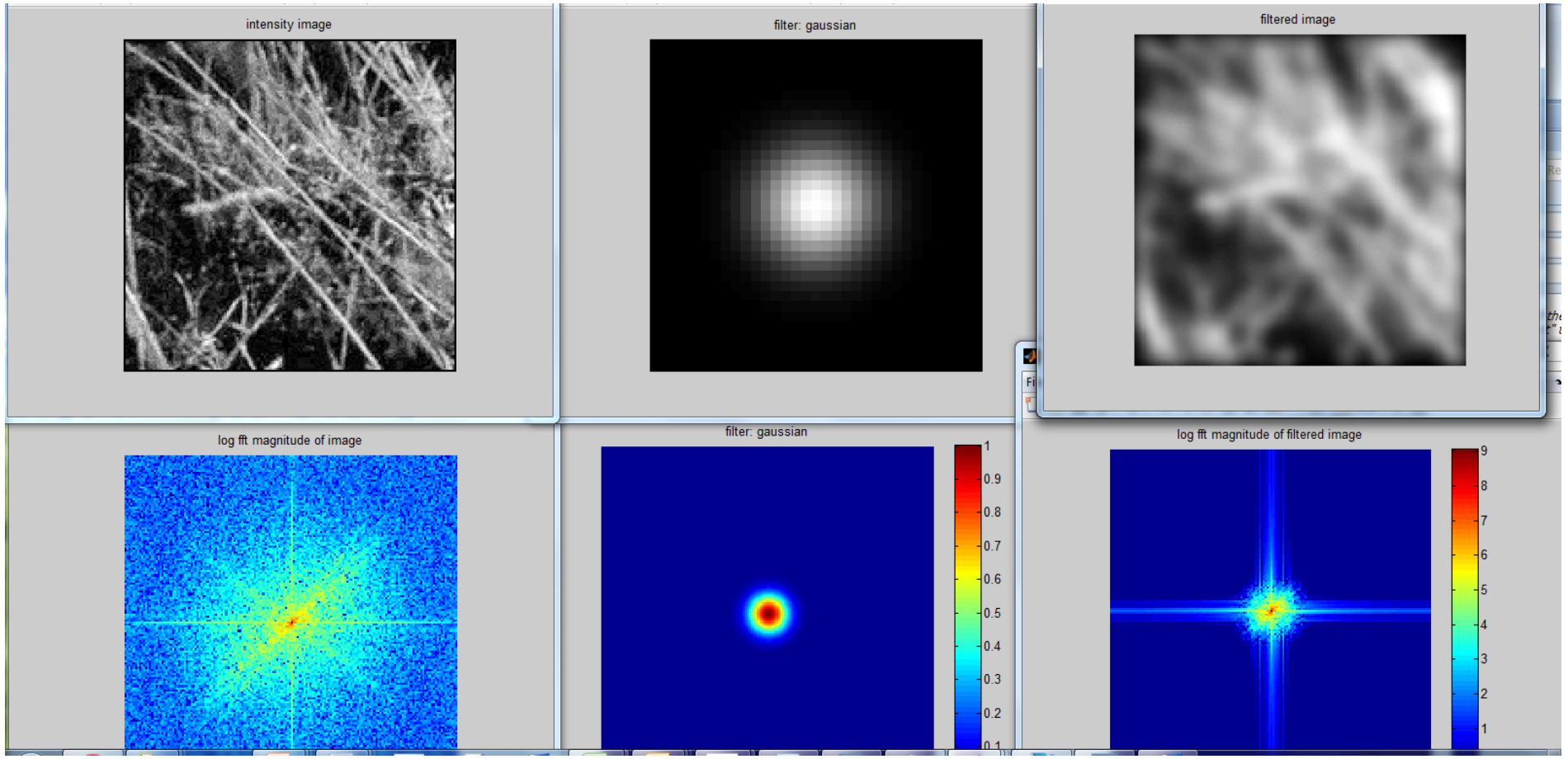
Inverse FFT



Modified from Slide: Hoiem

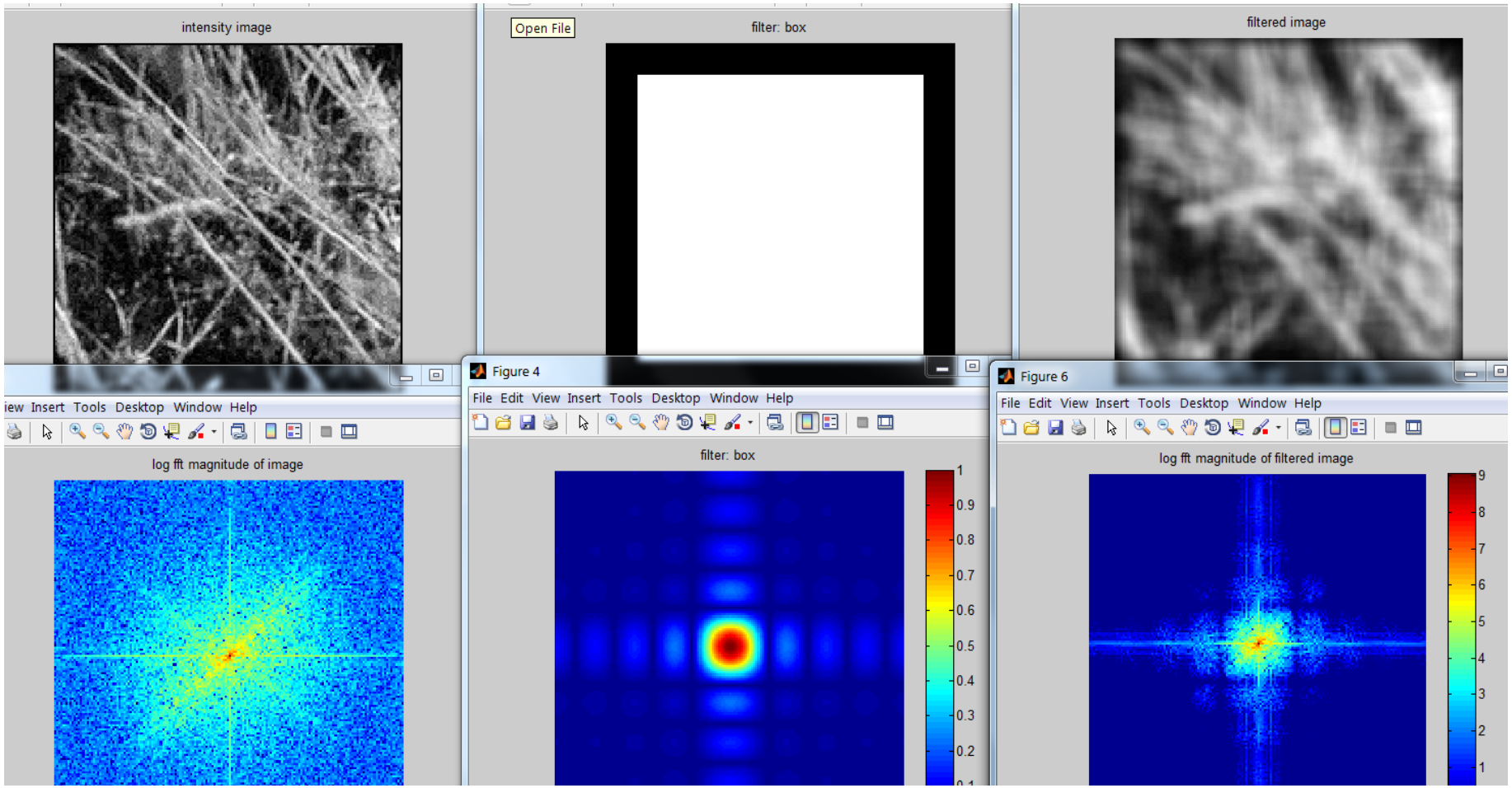
Filtering in frequency domain

Gaussian



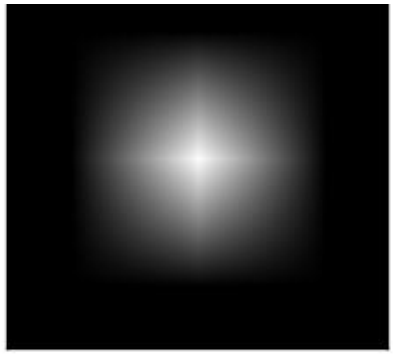
Filtering in frequency domain

Box Filter

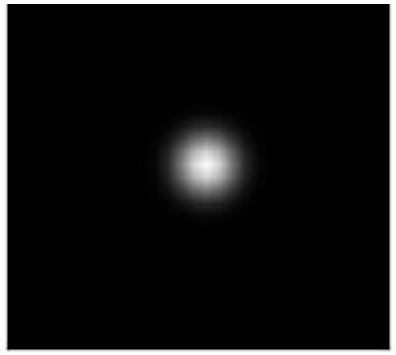


Filtering in frequency domain

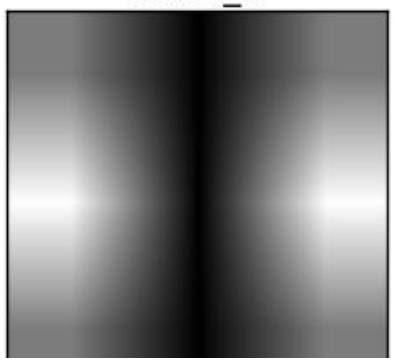
Mean filter



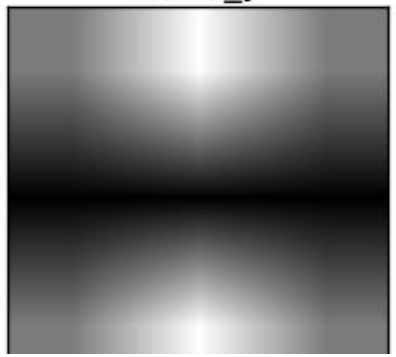
Gaussian Filter



sobel_x

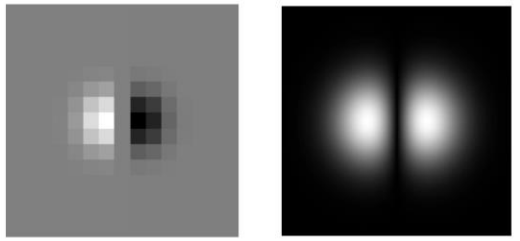


sobel_y



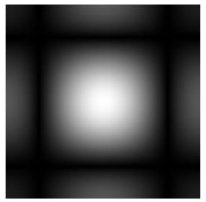
Vocabulary

Band-pass Filter:



Low Pass Filter:

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9



High-pass Filter:

Low-pass and high-pass filtering

Remove high frequencies with a *low pass* filter

- A blur filter!
- *Could also be called a 'high cut' filter (but few do)*

Counterpart: *high pass* filter

- Only keeps the high frequencies
- Edge detection (Sobel) is an example high pass filter

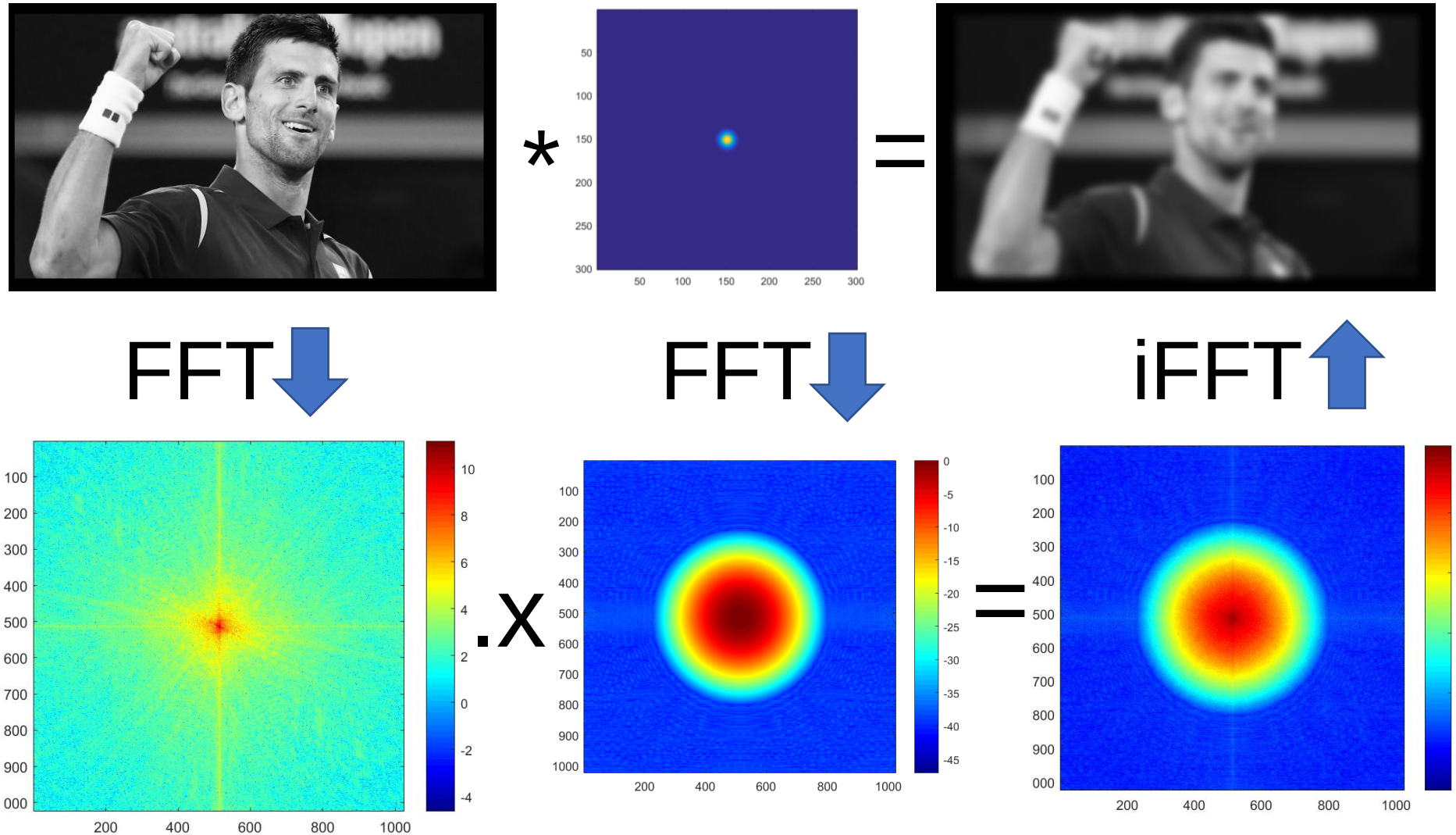
Related: *band pass / band stop* (or *notch)* filter

- Only affect *specific* frequencies
- How could we construct one with linear filters?

Is convolution invertible?

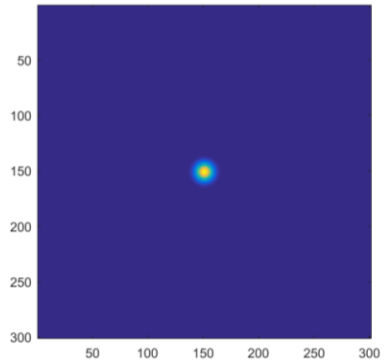
- If convolution is just multiplication in the Fourier domain, isn't deconvolution just division?
- Sometimes, it clearly is invertible (e.g. a convolution with an identity filter)
- In one case, it clearly isn't invertible (e.g. convolution with an all zero filter)
- What about for common filters like a Gaussian?

Convolution

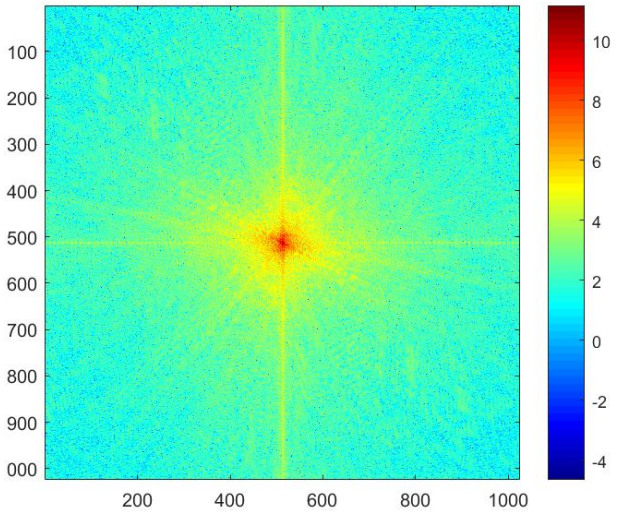


Hays

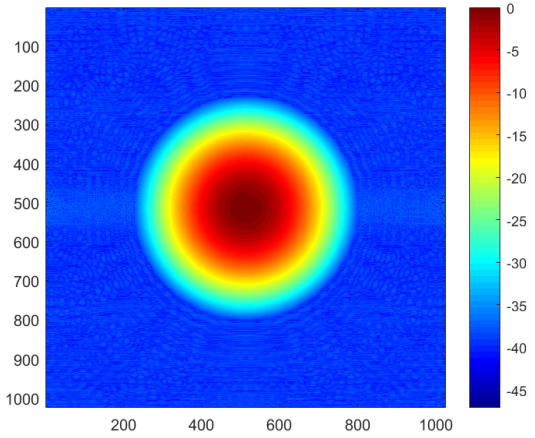
Deconvolution?



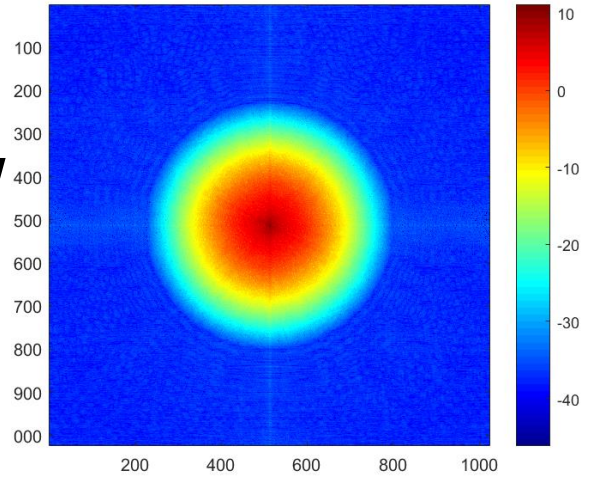
iFFT ↑ FFT ↓ FFT ↓



=

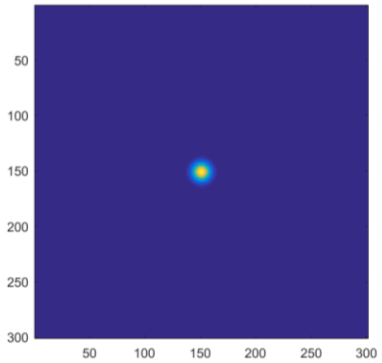


./



But under more realistic conditions

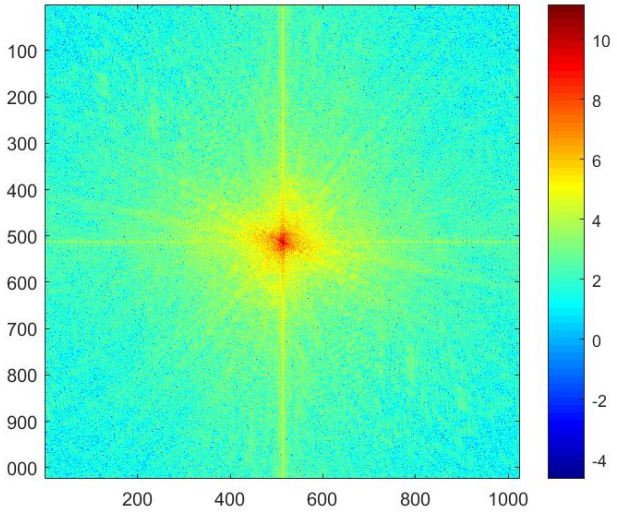
Random noise, .000001 magnitude



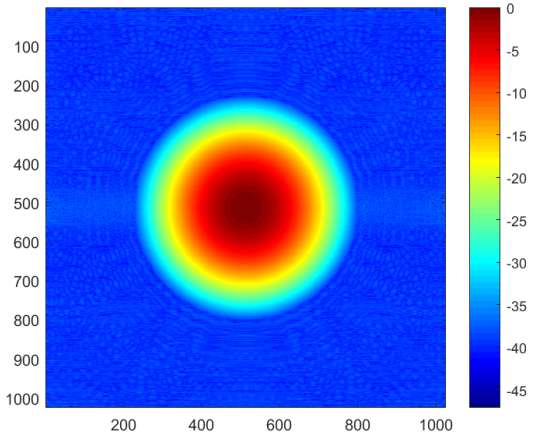
iFFT

FFT

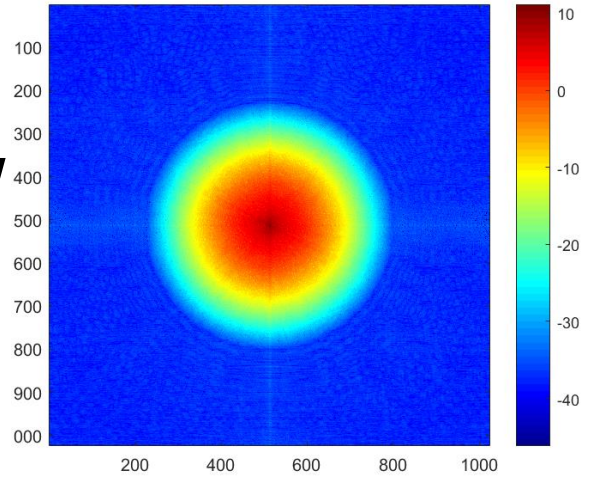
FFT



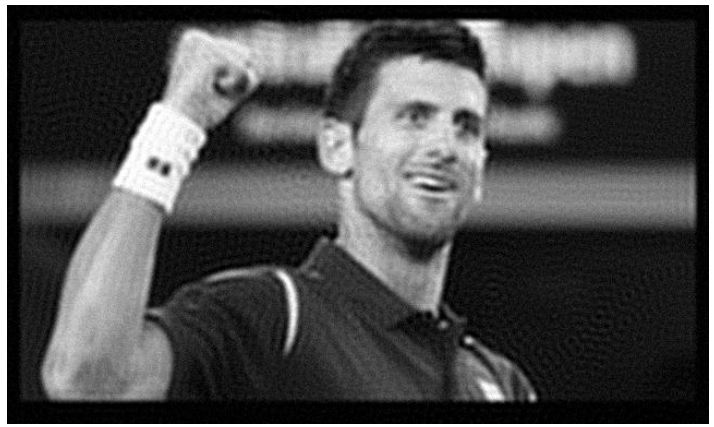
=

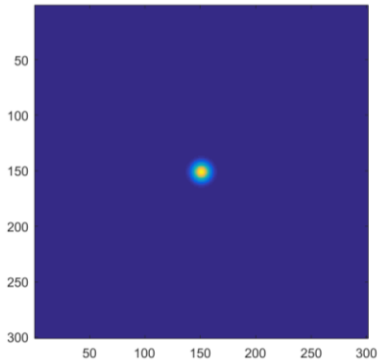


./



But under more realistic conditions



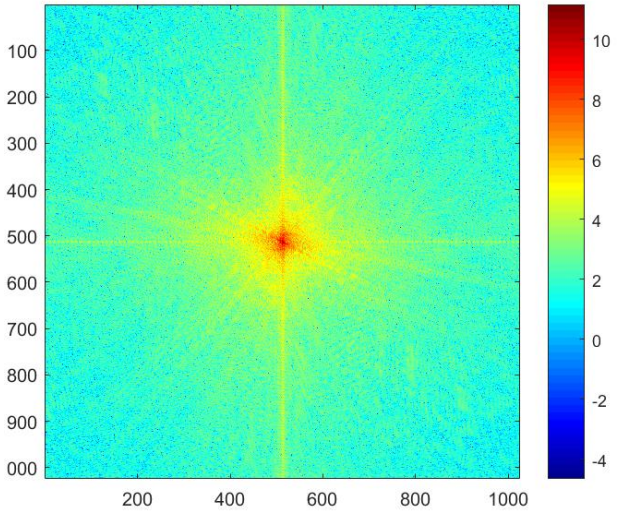


Random noise, .0001 magnitude

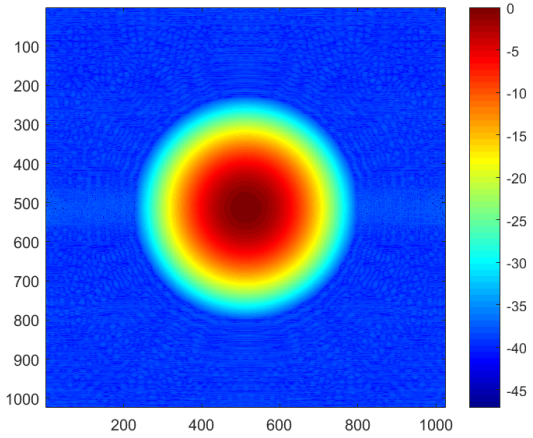
iFFT ↑

FFT ↓

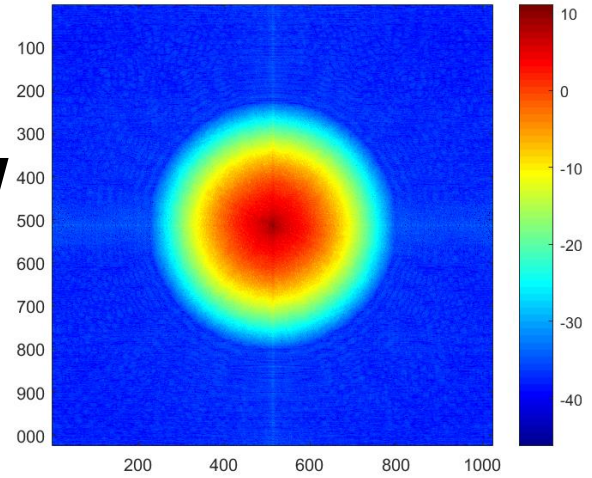
FFT ↓



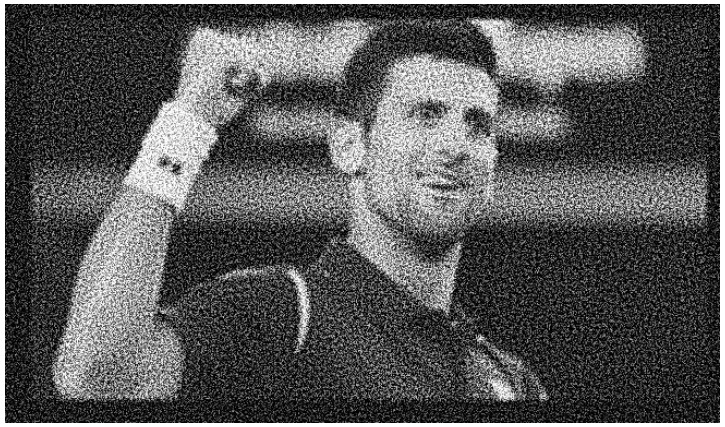
=

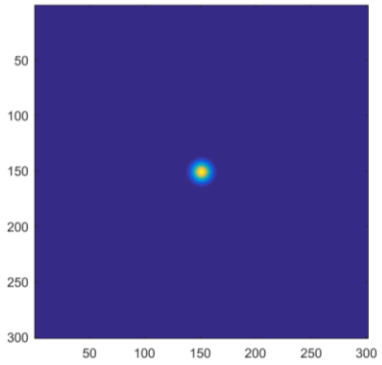


./



But under more realistic conditions



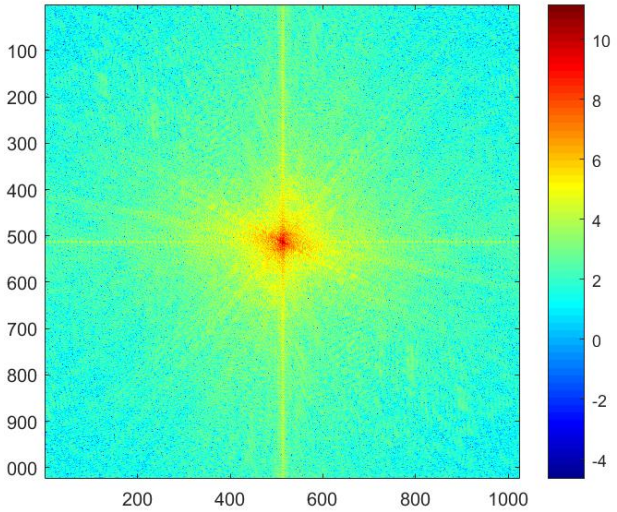


Random noise, .001 magnitude

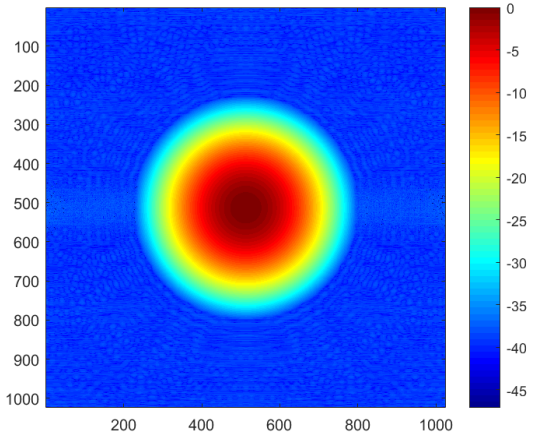
iFFT

FFT

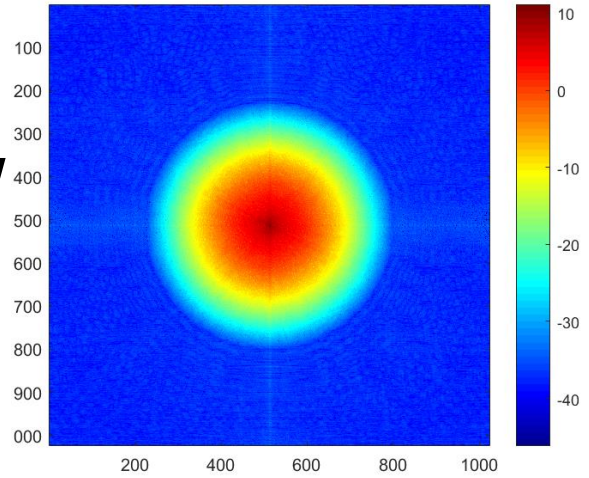
FFT



=

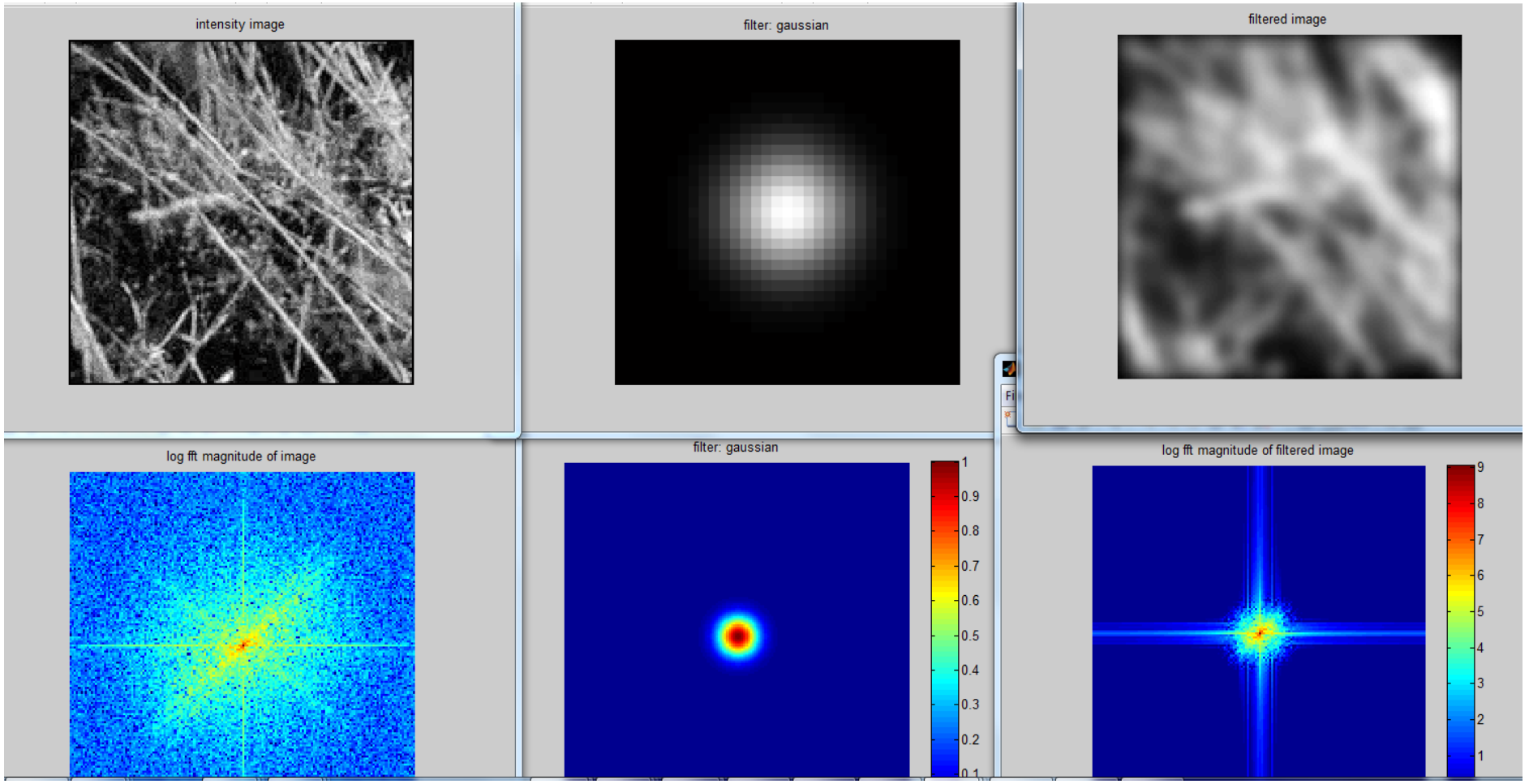


./

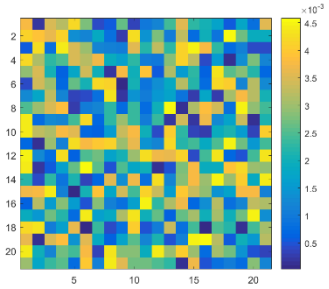


But you can't invert multiplication by 0!

- A Gaussian is only zero at infinity…



With a random filter…

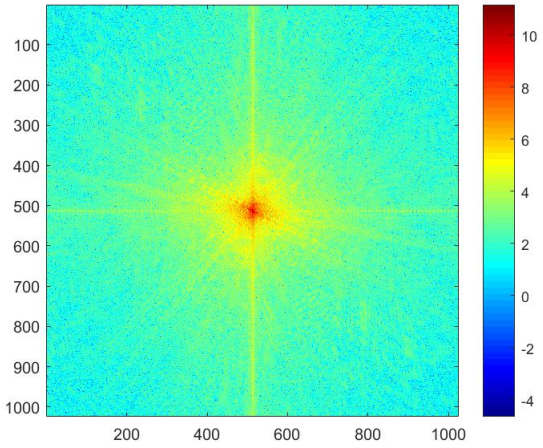


Random noise, .001 magnitude

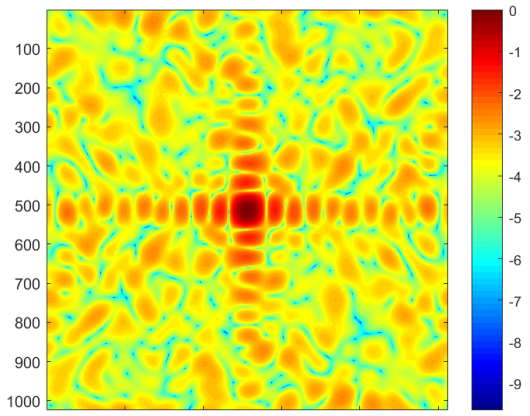
iFFT

FFT

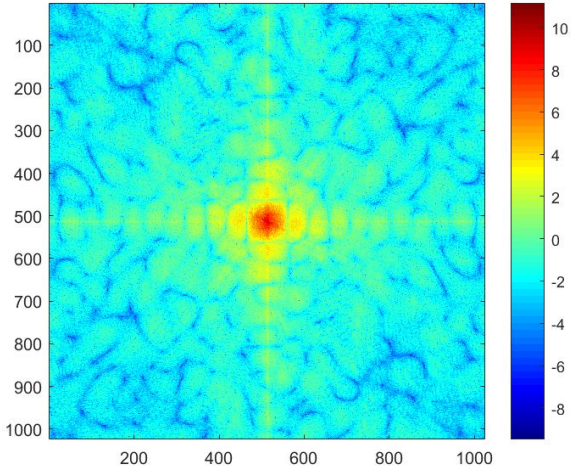
FFT



=



./



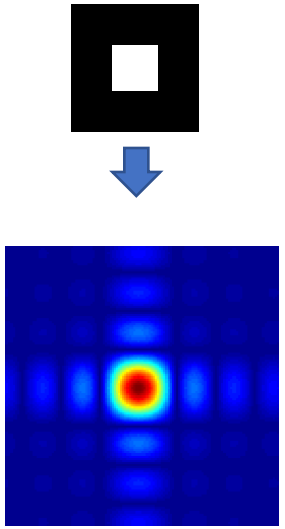
Deconvolution is hard.

- 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.

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

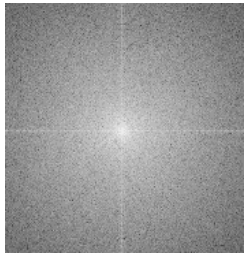
- N logN vs. N^2 for convolution/correlation

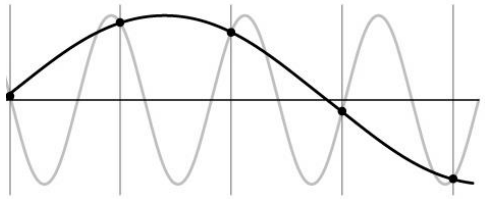
Images are mostly smooth

- 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())

Hybrid Image

- Hybrid image =

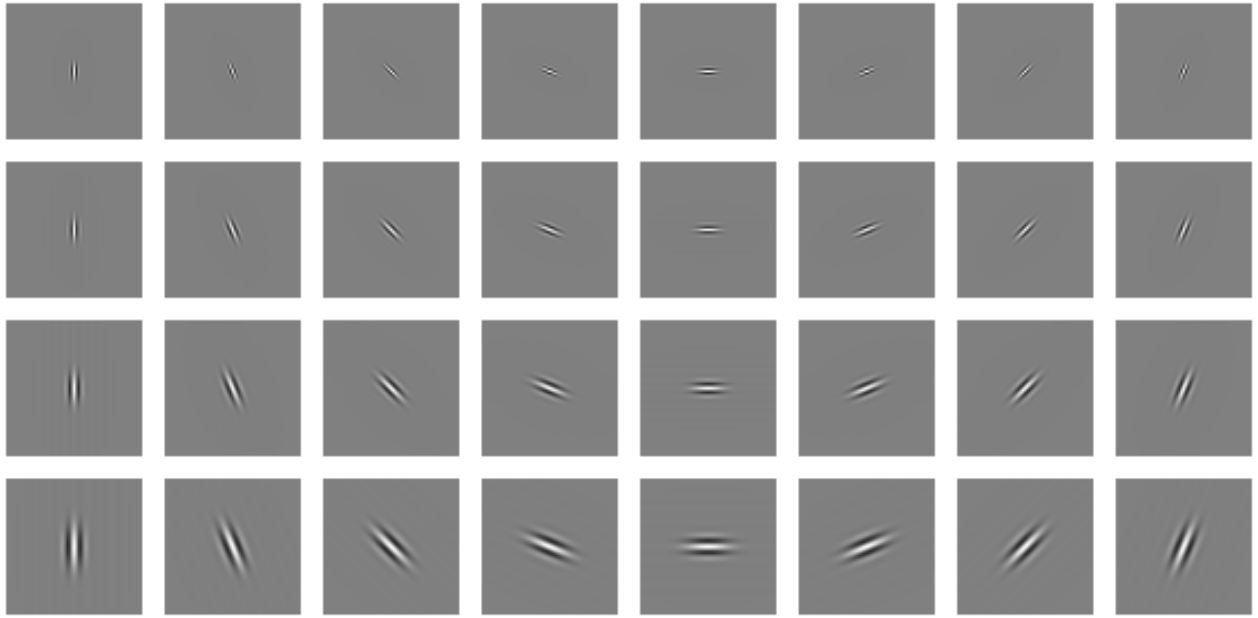
Low-Freq(Image A) + Hi-Freq(Image B)

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

?

Clues from Human Perception

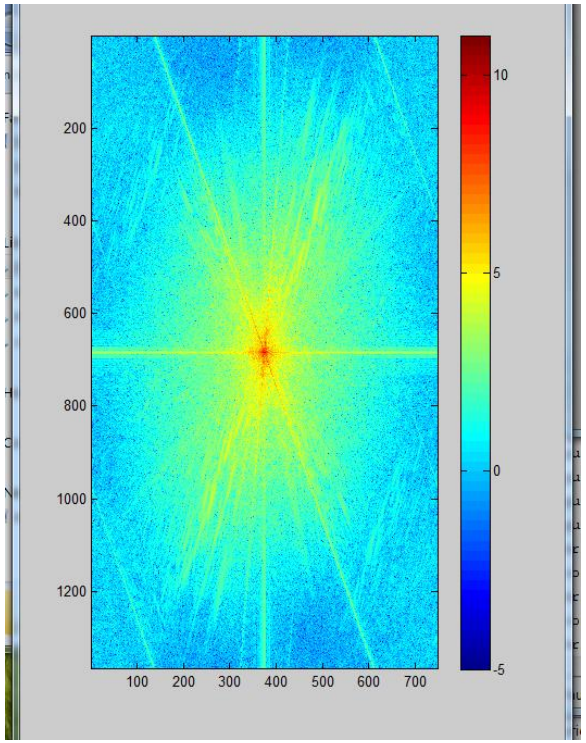
- Early processing in humans filters for various orientations and scales of frequency
- Perceptual cues in the mid-high frequencies dominate perception
- When we see an image from far away, we are effectively subsampling it



Early Visual Processing: Multi-scale edge and blob filters

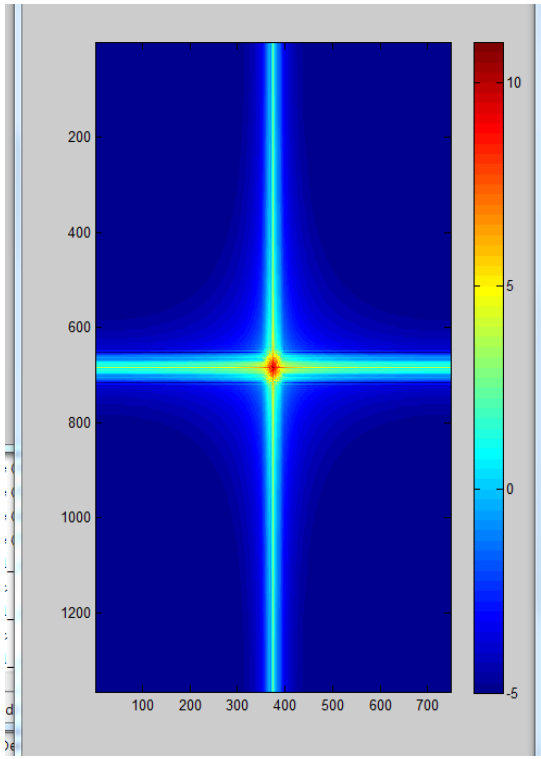
Hybrid Image in FFT

Hybrid Image



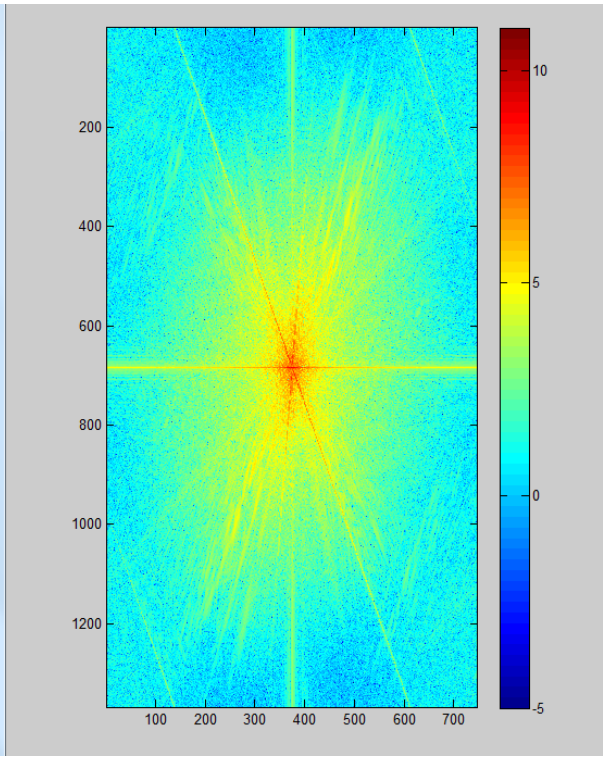
=

Low-passed Image



+

High-passed Image



Three views of image filtering

- Image filters in spatial domain
  - Filter is a mathematical operation on values of each patch
  - Smoothing, sharpening, measuring texture
- Image filters in the frequency domain
  - Filtering is a way to modify the frequencies of images
  - Denoising, sampling, image compression
- Templates and Image Pyramids
  - Filtering is a way to match a template to the image
  - Detection, coarse-to-fine registration