اصول پردازش تصویر

Principles of Image Processing

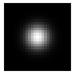
مصطفي كمالی تبریزی

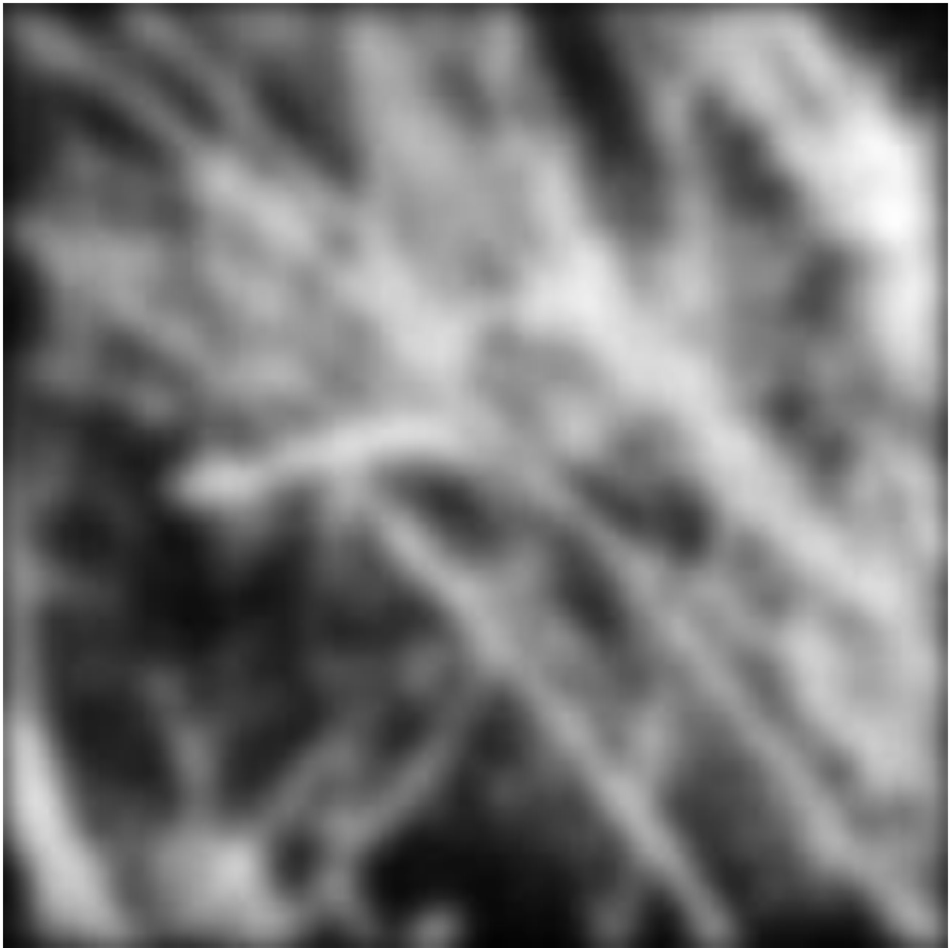
۵ آبان ۱۳۹۹

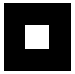
جلسه یازدهم

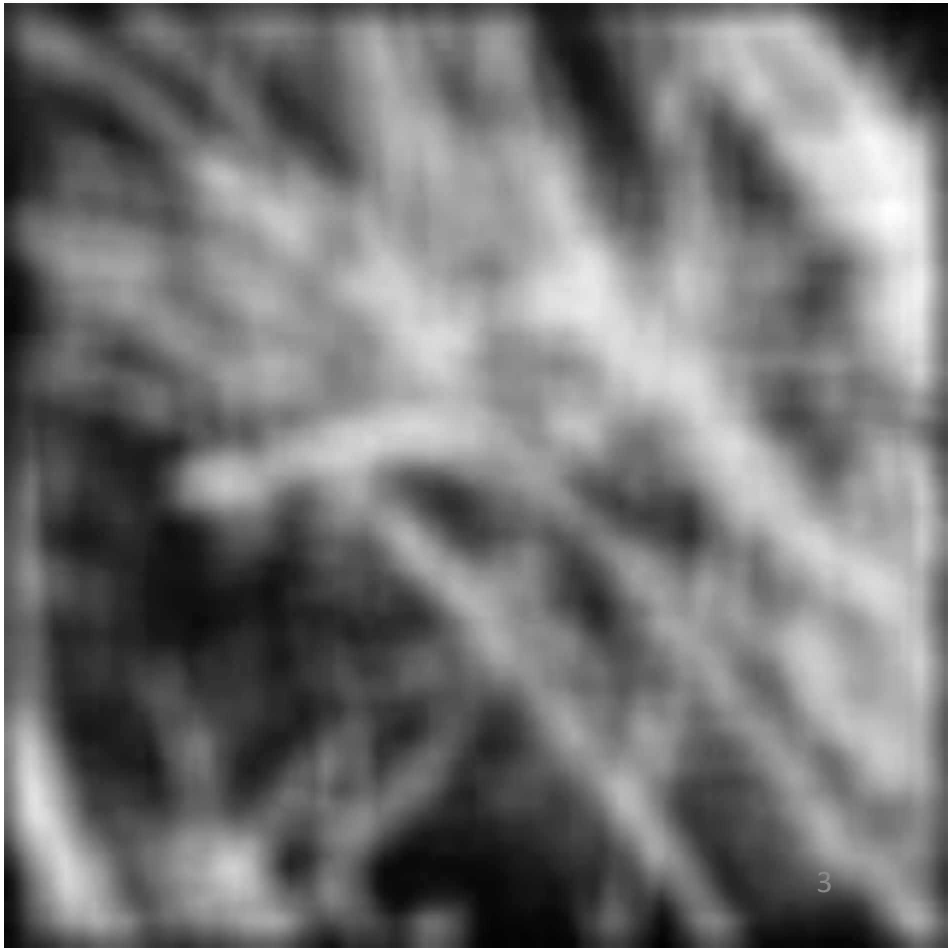
Fourier Transform

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

Gaussian 



Box filter 



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

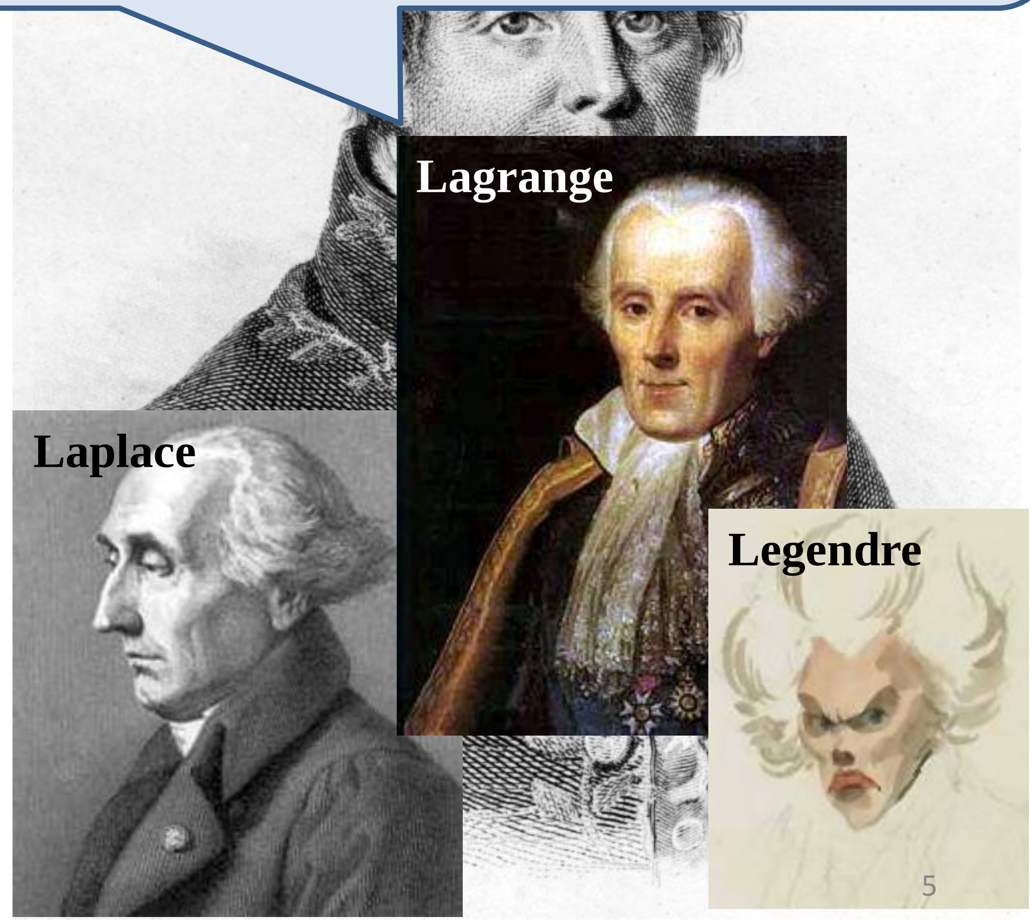
Image: http://www.flickr.com/photos/igorms/136916757/

Jean Baptiste Joseph Fourier (1768-1830)

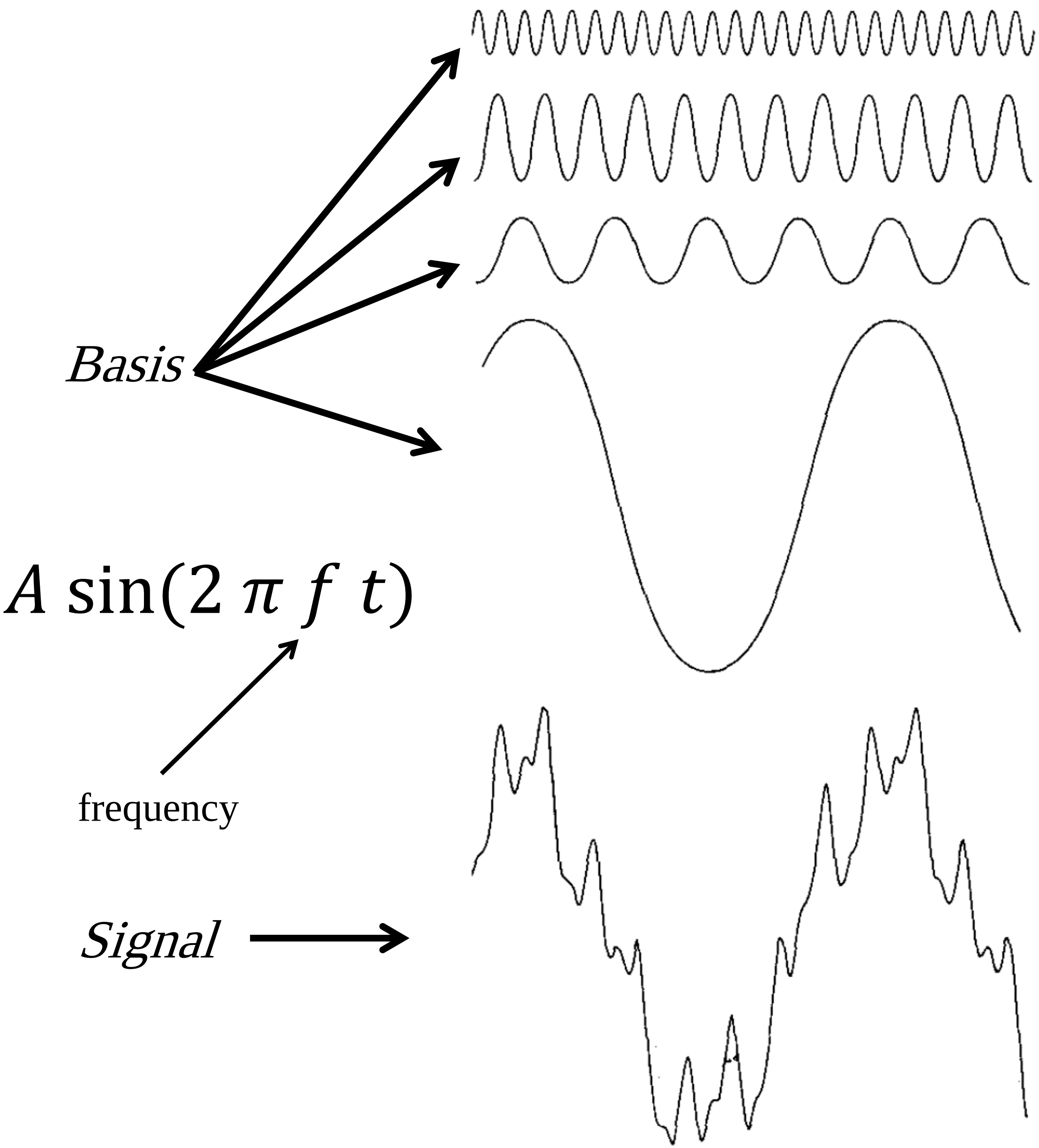
- had crazy idea (1807):
  Any *univariate function can be rewritten as a weighted sum of sines and cosines of different frequencies.*

...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.

- Don't believe it?
  - Neither did Lagrange, Laplace, Poisson and other big wigs
  - Not translated into English until 1878!
- But it's (mostly) true!
  - called Fourier Series
  - there are some subtle restrictions



Slide: Derek Hoiem


Basis
$A \sin(2 \pi f t)$
frequency
Signal

Slide: Alyosha Efros

A sum of sines

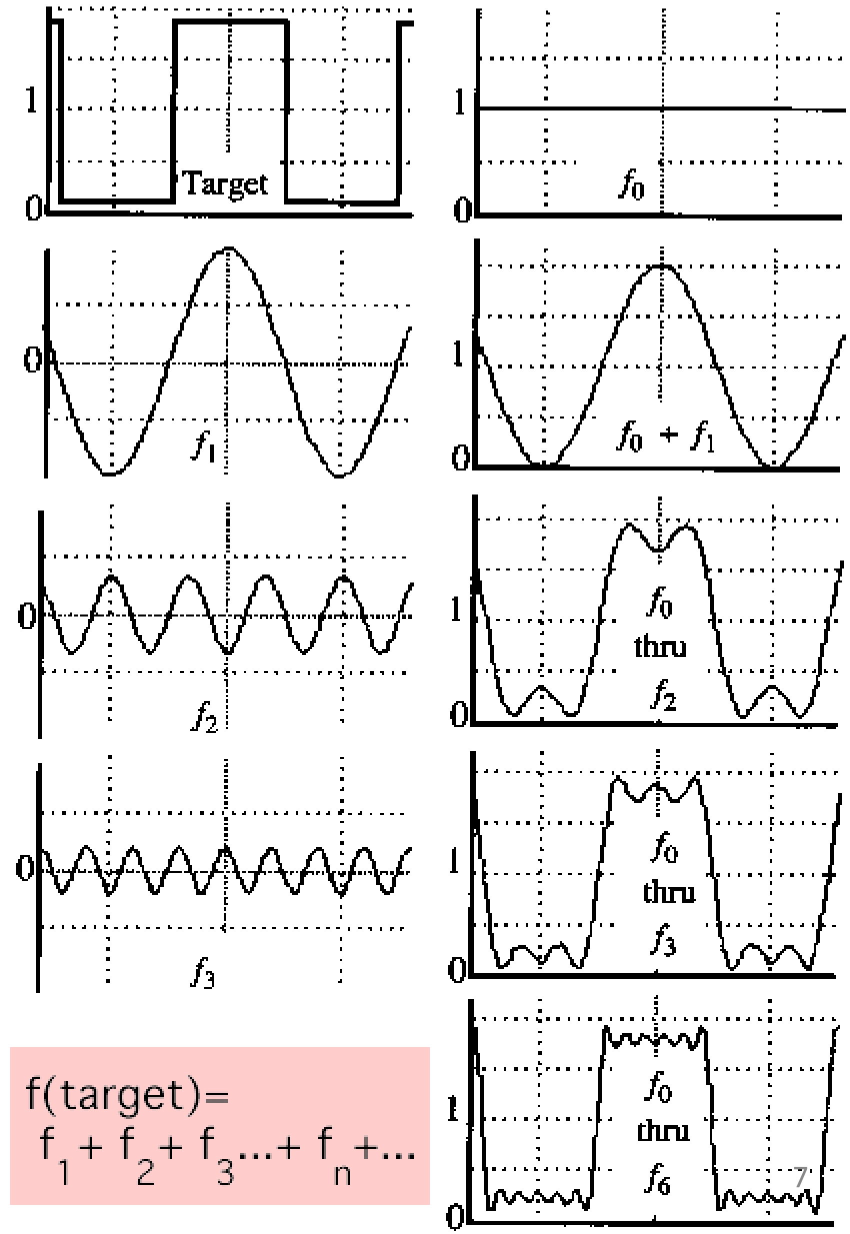
- Our building block:
- $A\ sin(wx + \phi)$

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

- How many degrees of freedom?

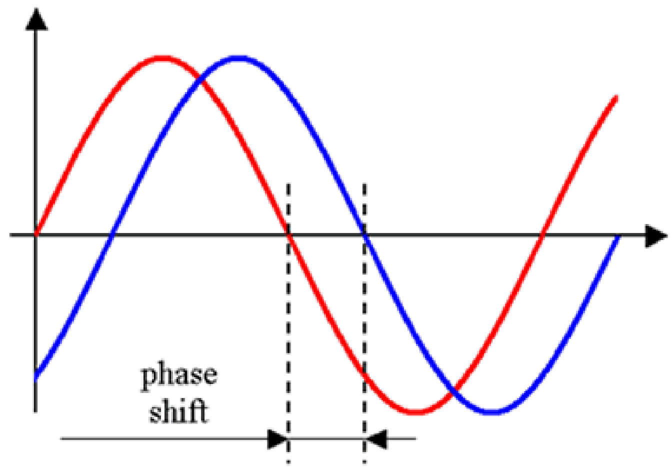
- What does each control?

- Which one encodes the coarse vs. fine structure of the signal?



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

Amplitude / Phase



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

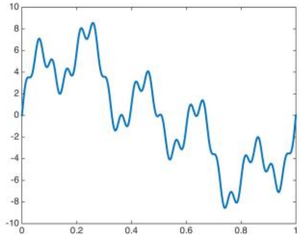
amplitude phase

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

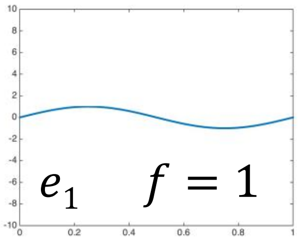
- Translate the image?
  - Amplitude unchanged
  - Adds a constant to the phase

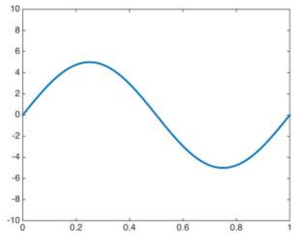
Slide: James Hays

$$e_f = \sin(2\pi f x)$$

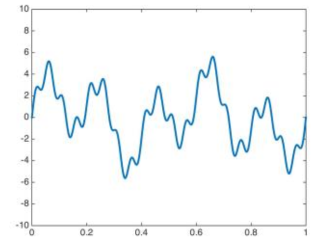


$$s = 5e_1 + 2e_2 + 3e_5 + e_{20}$$

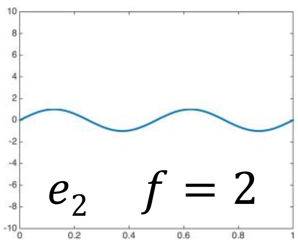


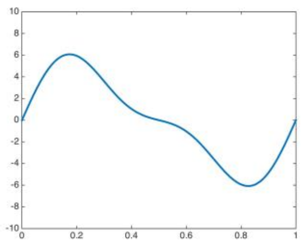


$$s_1 = 5e_1$$

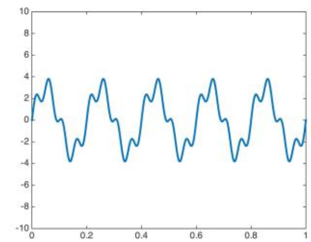


$$2e_2 + 3e_5 + e_{20}$$

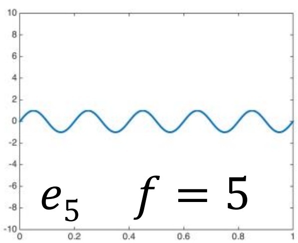


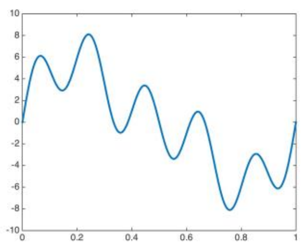


$$s_2 = 5e_1 + 2e_2$$

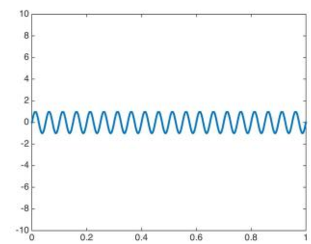


$$3e_5 + e_{20}$$

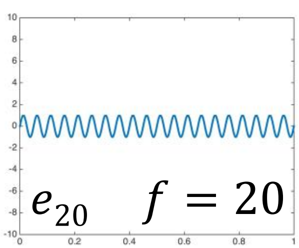


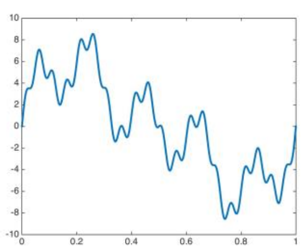


$$s_3 = 5e_1 + 2e_2 + 3e_5$$



$$e_{20}$$

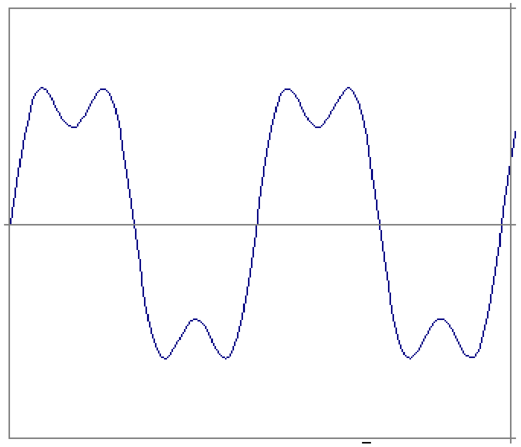




$$s_4 = 5e_1 + 2e_2 + 3e_5 + e_{20}$$

Time and Frequency

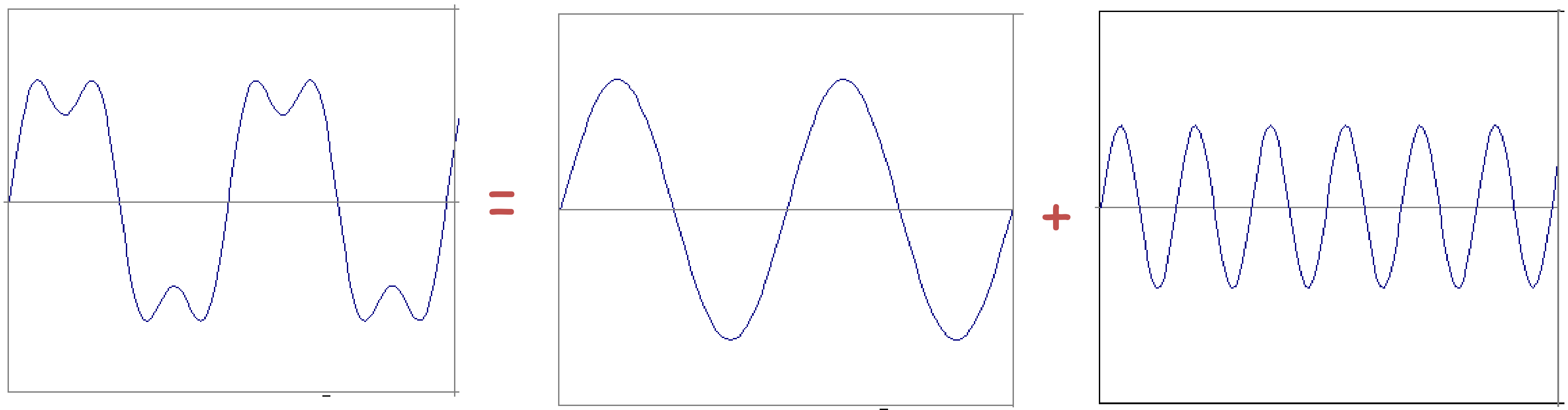
- example : $g(t) = \sin(2pft) + \frac{1}{3}\sin(2p(3f)t)$



Slide: Alyosha Efros

Time and Frequency

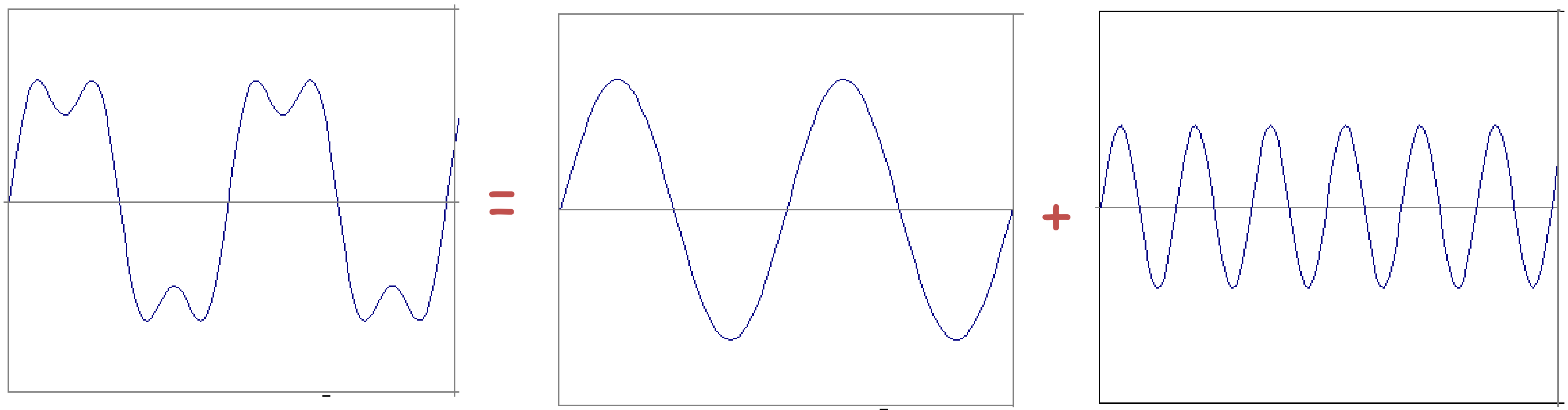
- example : $g(t) = \sin(2pft) + \frac{1}{3}\sin(2p(3f)t)$

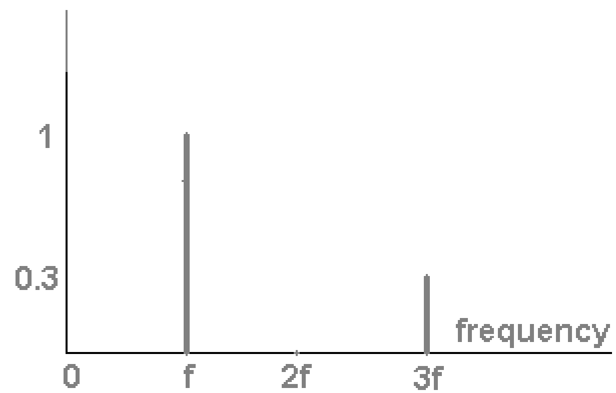


Slide: Alyosha Efros

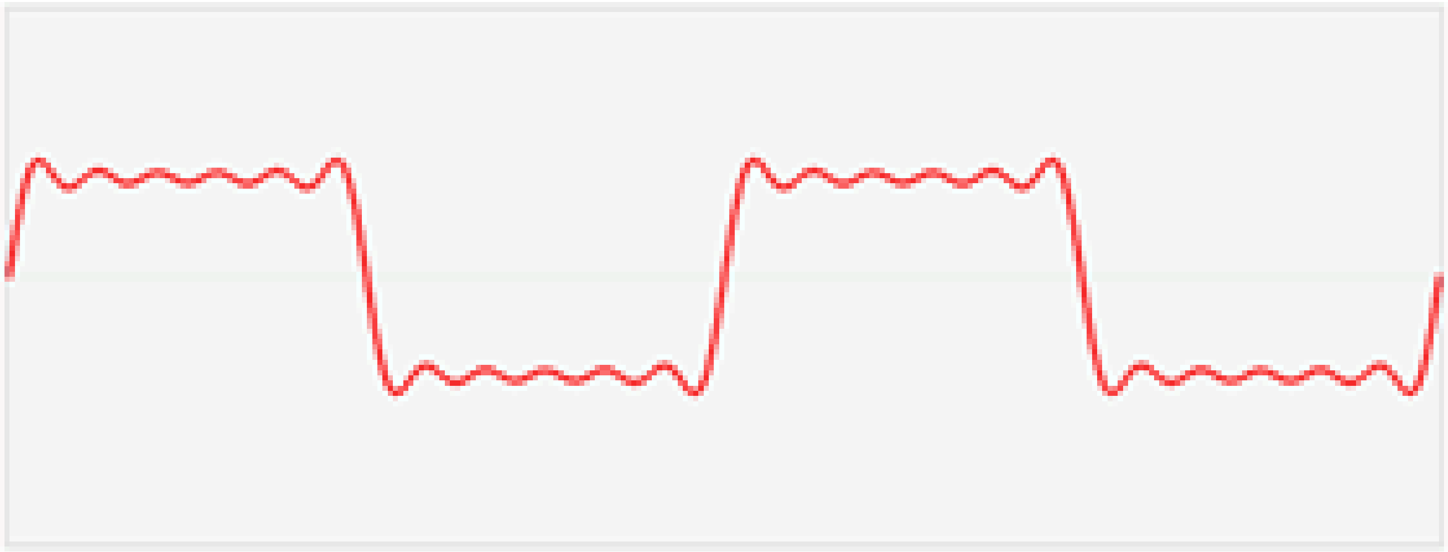
Frequency Spectra

- example : $g(t) = \sin(2pft) + \frac{1}{3}\sin(2p(3f)t)$





Slide: Alyosha Efros



Wikipedia – Fourier transform

Slide: James Tompkins

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

$$A\,e^{j\varphi} = A\cos(\varphi) + j\,A\sin(\varphi)$$

$$A = Magnitude$$

$$\varphi = Phase$$

$$j^2 = -1$$

Slide: James Hays

1D Discrete Fourier Transform (DFT)

Inverse Transform

$$f(x) = \sum_{u=-\frac{M}{2}+1}^{\frac{M}{2}} F(u) e^{j2\pi u \frac{x}{M}} \qquad x = 0, \ldots, M-1$$

Forward Transform

$$F(u) = \frac{1}{M} \sum_{x=0}^{M-1} f(x) e^{-j2\pi \frac{ux}{M}} \qquad u = -\frac{M}{2} + 1, \ldots, \frac{M}{2}$$

$$e^{j2\pi \frac{ux}{M}} = \cos\left(2\pi \frac{ux}{M}\right) + j \sin(2\pi \frac{ux}{M})$$

2D Discrete Fourier Transform (DFT)

Inverse

$$f(x,y) = \sum_{v=-\frac{N}{2}+1}^{\frac{N}{2}} \sum_{u=-\frac{M}{2}+1}^{\frac{M}{2}} F(u,v) e^{j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)} \qquad \begin{array}{l} x = 0, \dots, M-1 \\ y = 0, \dots, N-1 \end{array}$$

Forward

$$F(u,v) = \frac{1}{MN} \sum_{y=0}^{N-1} \sum_{x=0}^{M-1} f(x,y) e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)} \qquad \begin{array}{l} u = -\frac{M}{2}+1, \dots, \frac{M}{2} \\ v = -\frac{N}{2}+1, \dots, \frac{N}{2} \end{array}$$

Symmetric and Periodic

$$\boxed{e^{j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)} = \cos\left(2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)\right) + j\sin\left(2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)\right)}$$

2D Discrete Fourier Transform (DFT)

$$F(u,v) = \underbrace{R(u,v)}_{\text{Real Term}} + j\,\underbrace{I(u,v)}_{\text{Imaginary Term}}$$

Magnitude $\rightarrow$ $$|F(u,v)| = \sqrt{R(u,v)^2 + I(u,v)^2}$$

Phase $\rightarrow$ $$\phi(u,v) = \arctan\left(\frac{I(u,v)}{R(u,v)}\right)$$

Power $\rightarrow$ $$P(u,v) = |F(u,v)|^2$$

Properties of Fourier Transforms

- Linearity: $\mathcal{F}(\alpha\, f + g) = \alpha\, \mathcal{F}(f) + \mathcal{F}(g)$

- 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

Basic Properties

$$F(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(x)e^{-j2\pi\frac{ux}{M}}$$

$$F(0) = \frac{1}{M}\sum_{x=0}^{M-1} f(x) = mean\{f(x)\}$$

$$\boxed{\begin{matrix}\cos(2\pi ux)\\ \sin(2\pi ux)\\ 0 \le 2\pi u \le 2\pi\end{matrix}} \Longleftrightarrow \boxed{\begin{matrix}\cos(2\pi ux)\\ \sin(2\pi ux)\\ -\pi \le 2\pi u \le \pi\end{matrix}}$$

$$u = -\frac{M}{2} + 1, \dots, \frac{M}{2}$$

2D DFT

In Forward transformation, interested in ***magnitude***

$$|F(u,v)| = \sqrt{R(u,v)^2 + I(u,v)^2}$$

In Inverse transformation, interested in ***Real Part!***

Fast Fourier Transform (FFT)   O(n log(n))

Basis

*Image Size = 1024*1024*

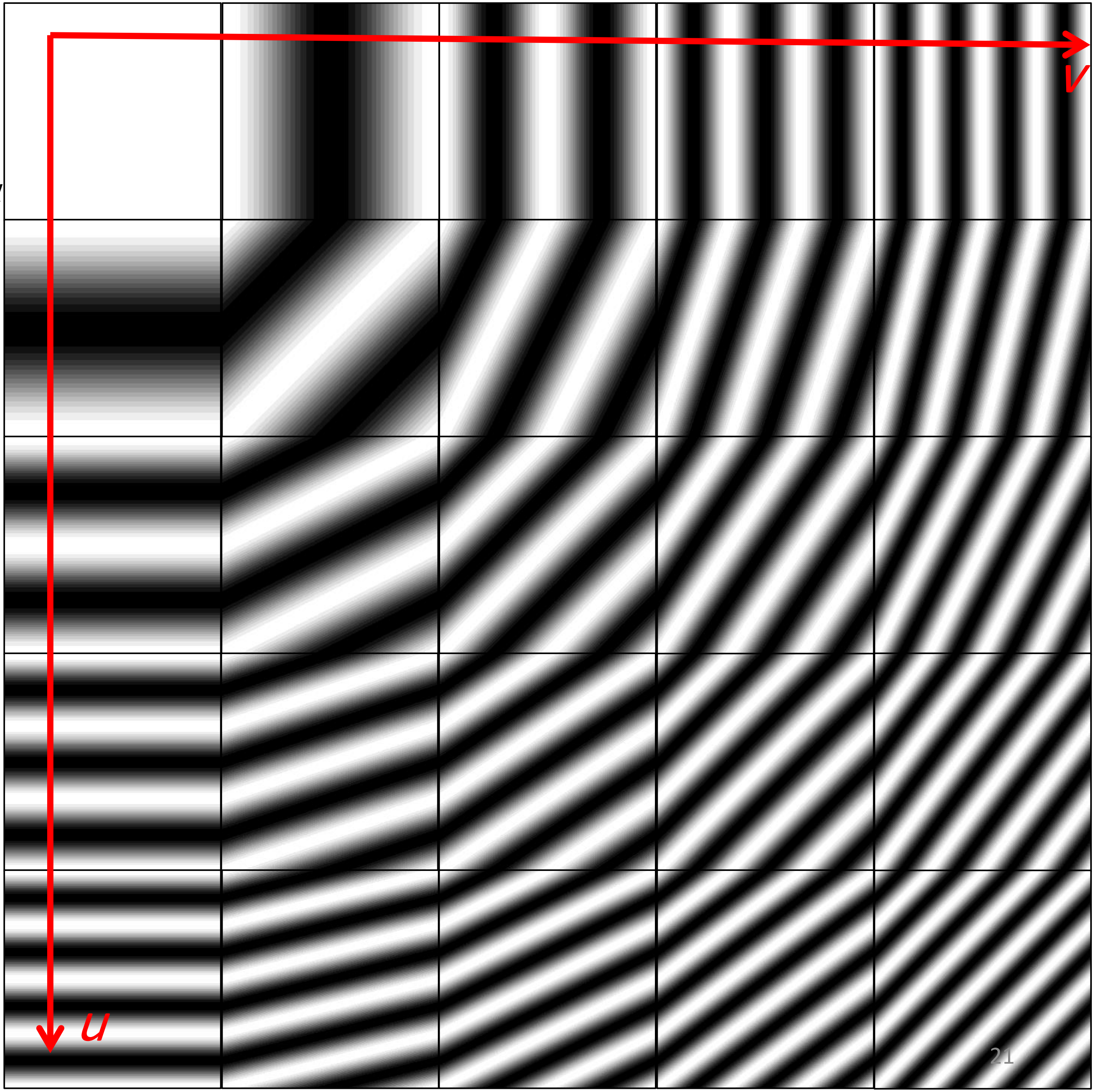
Cosine Functions

$$\cos\left(2\pi(ux + vy)\right)$$
$$0 \leq x, y \leq 1$$

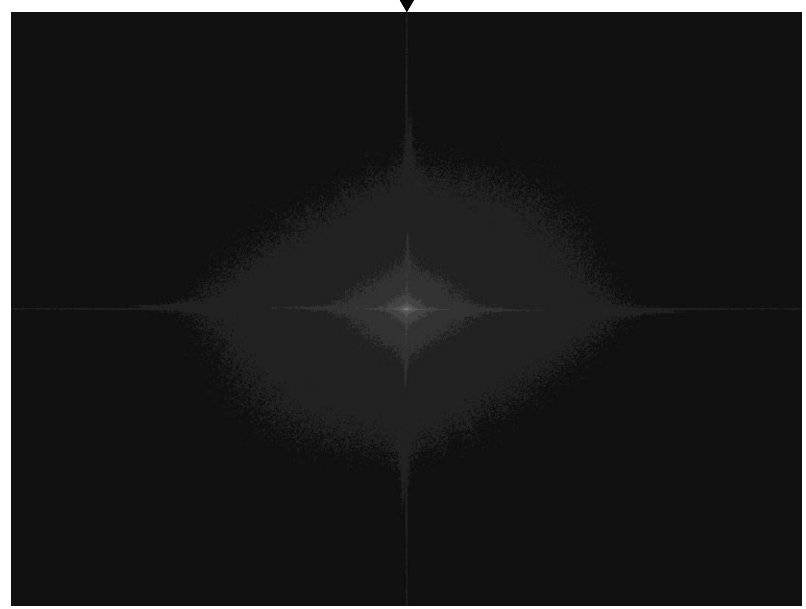
u = 0, 1, 2, 3, 4
v = 0, 1, 2, 3, 4

(u,v) magnitude
→ Frequency

(u,v) direction
→ Orientation

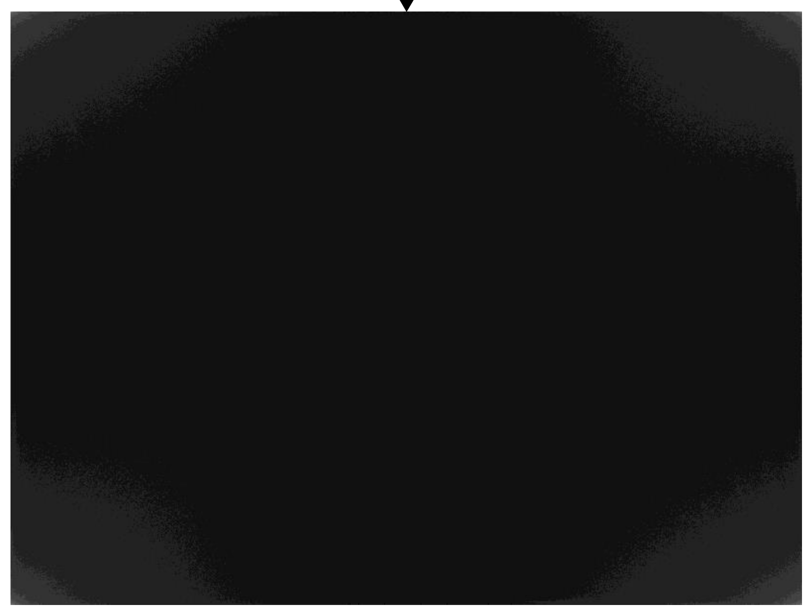


What we use

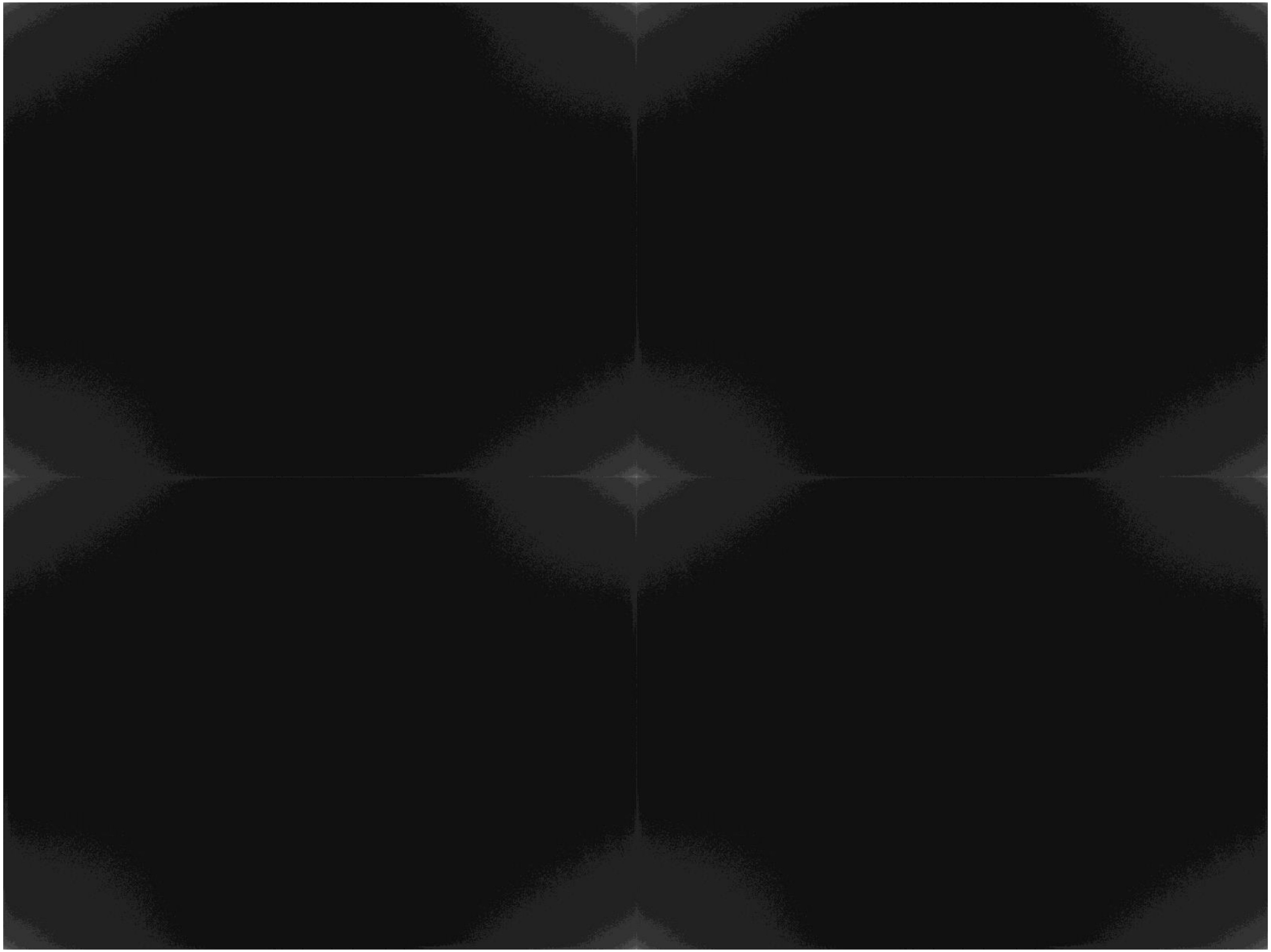


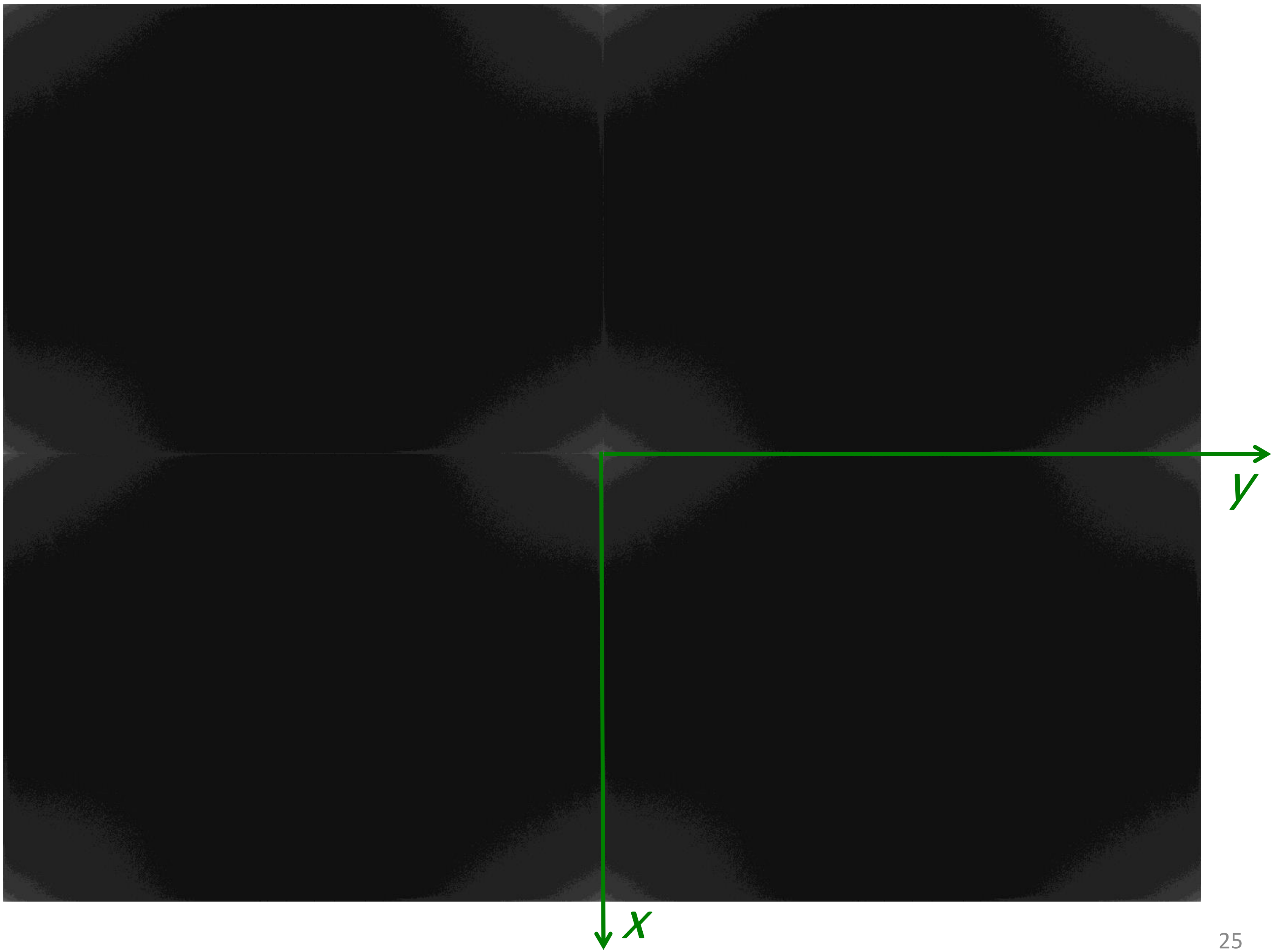
Symmetric and Periodic

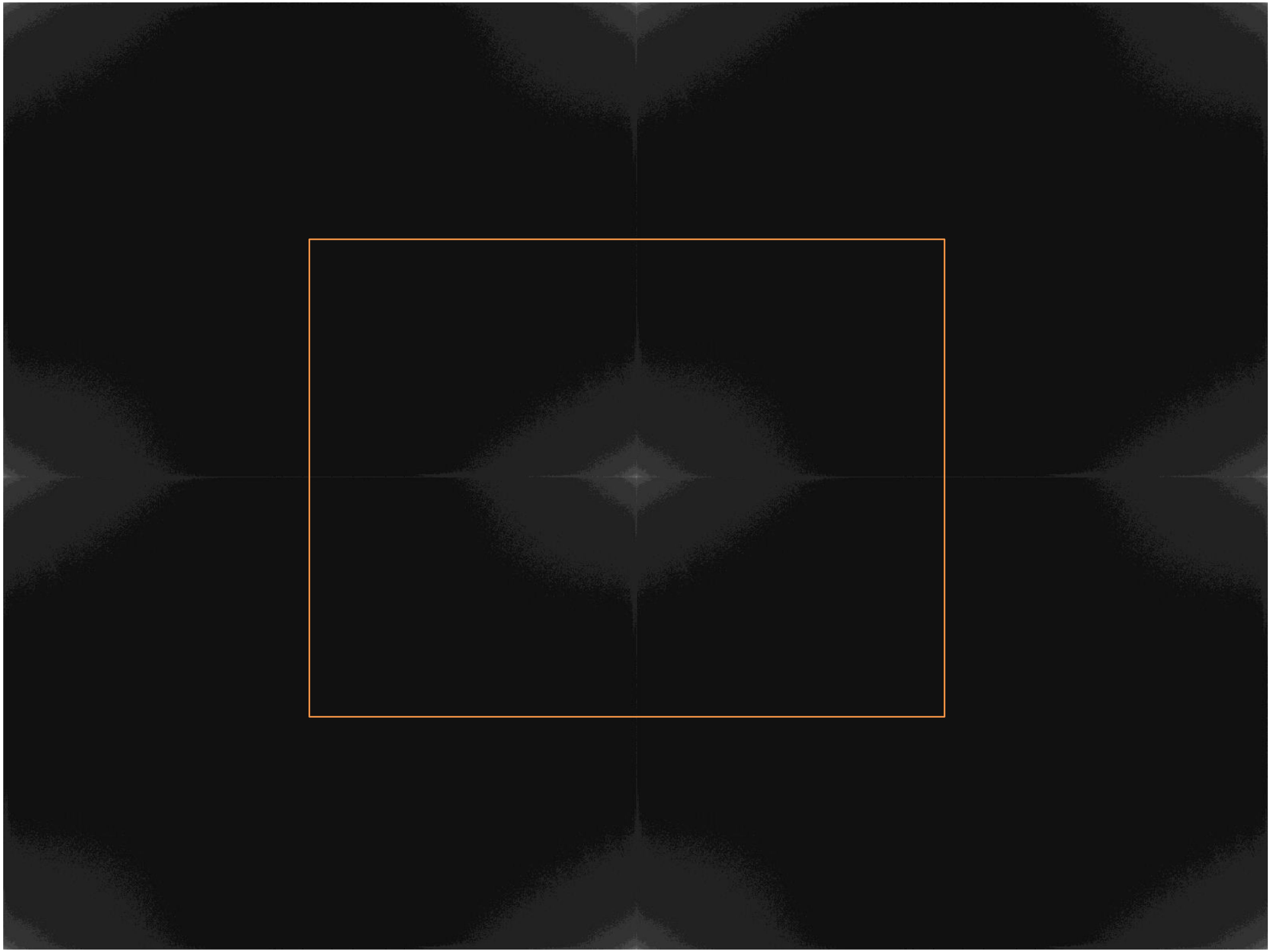
What FFT Obtains

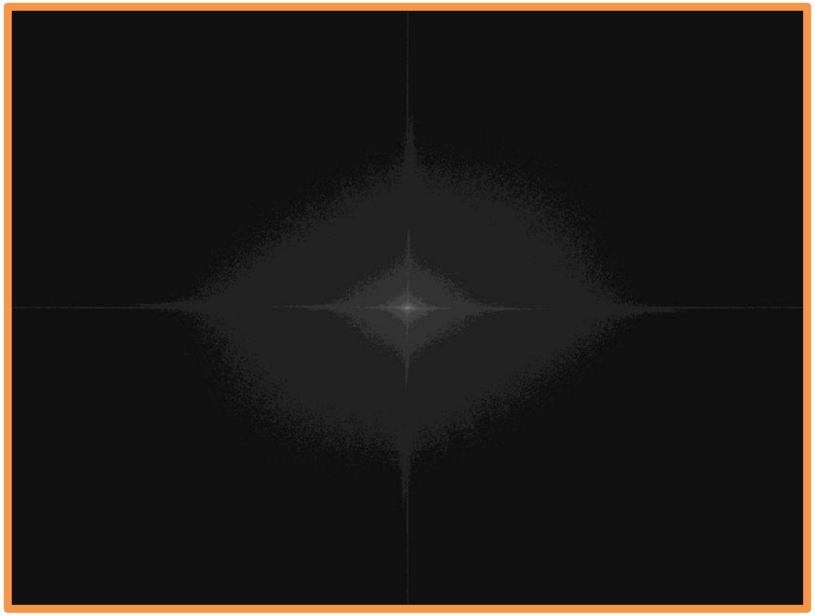


Symmetric and Periodic

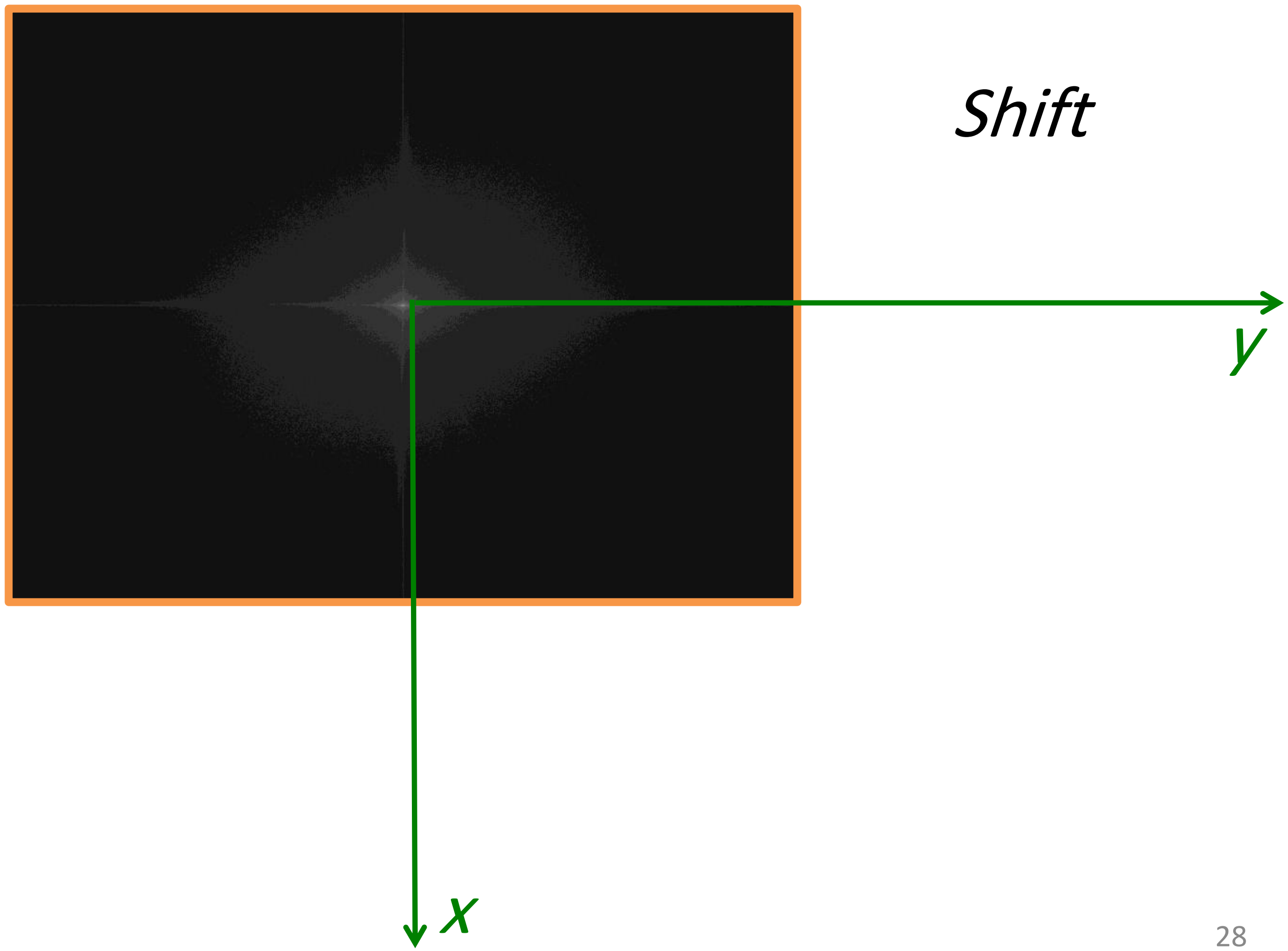



y
x





Shift



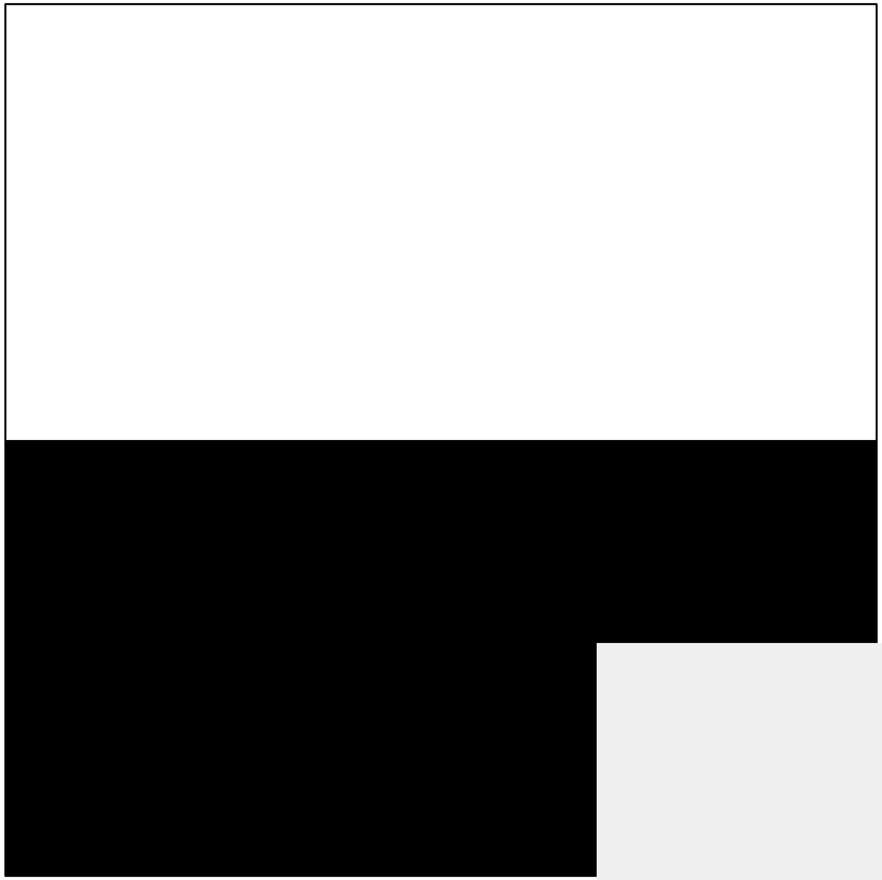
FFT in Python

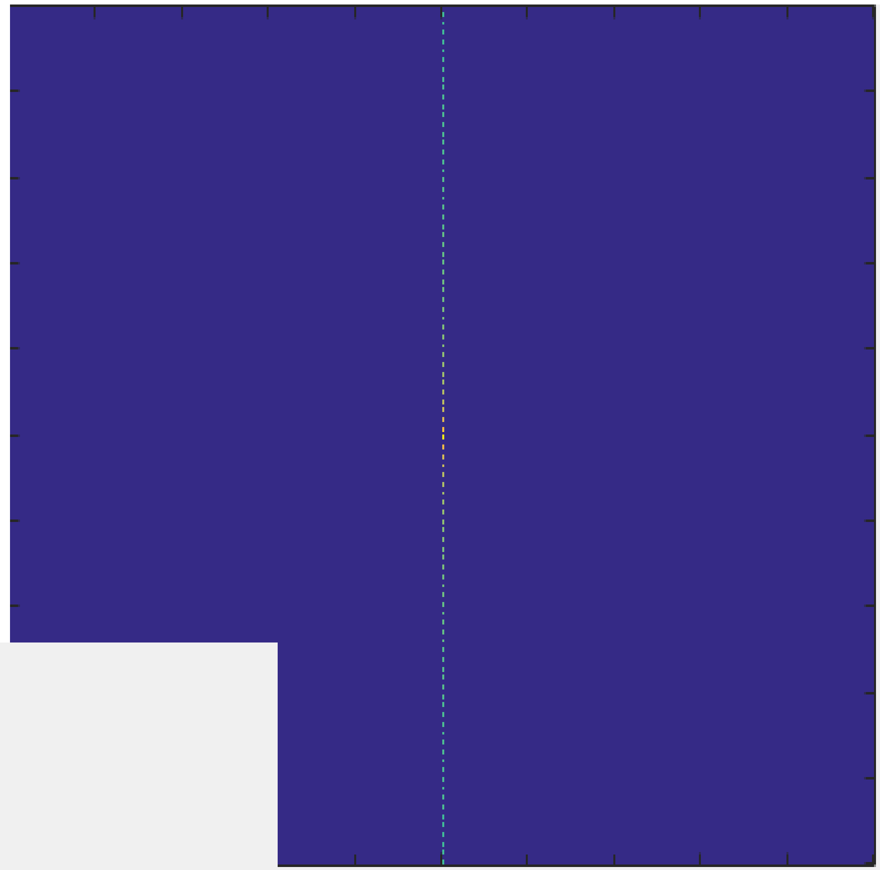
```
import matplotlib.pyplot as plt
import numpy as np

im = plt.imread("…")
im_fft = np.fft.fft2(im)
shifted_image = np.fft.fftshift(im_fft)

amplitude_image = np.abs(shifted_image)
log_amplitude_image = np.log(amplitude_image)
plt.imshow(log_amplitude_image, cmap='gray')

fil_im_fft = shifted_image * filter_in_fft
fil_im_ishifted = np.fft.ifftshift(fil_im_fft)
fil_im = np.fft.ifft2(fil_im_ishifted)
fil_im = np.real(fil_im)
```

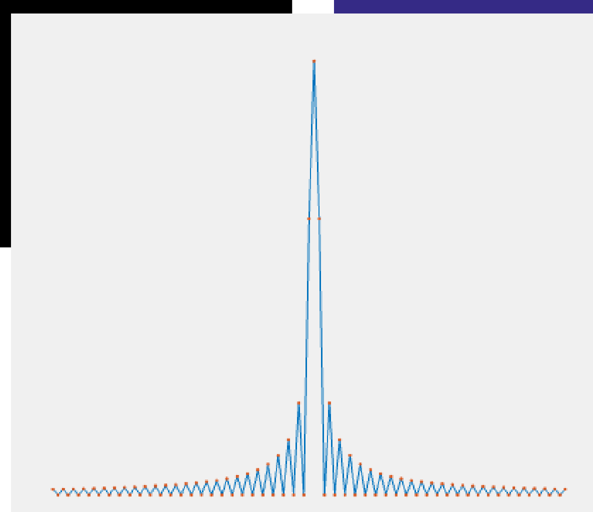
Slide Credit: Derek Hoiem

Spatial Domain

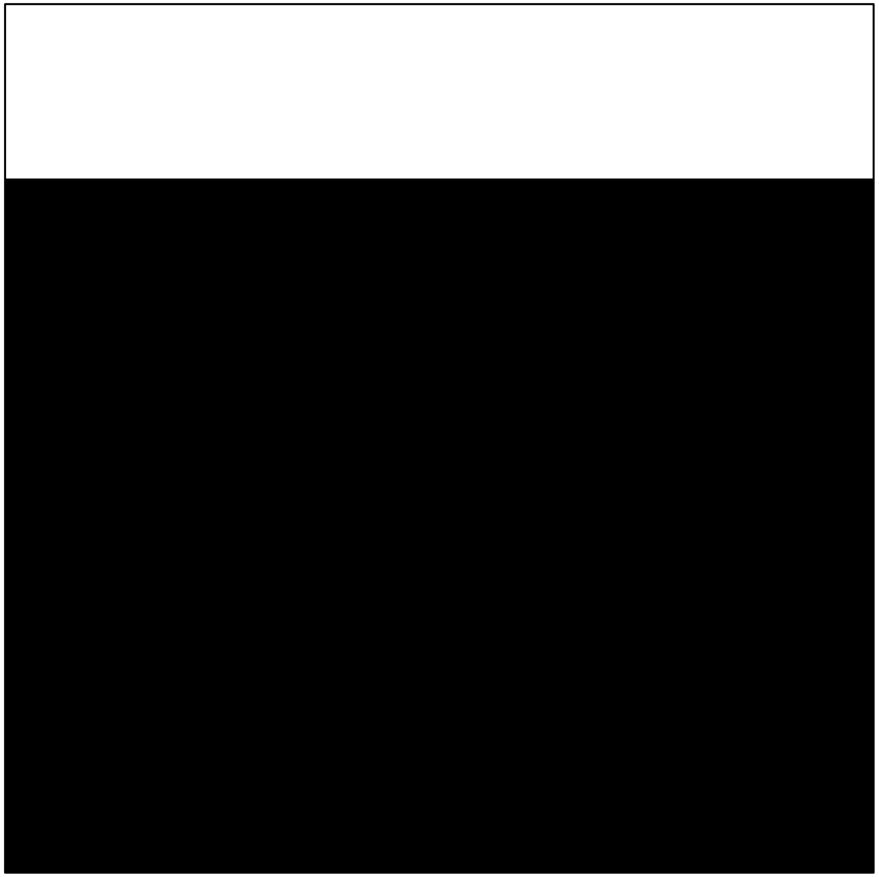
Frequency Domain

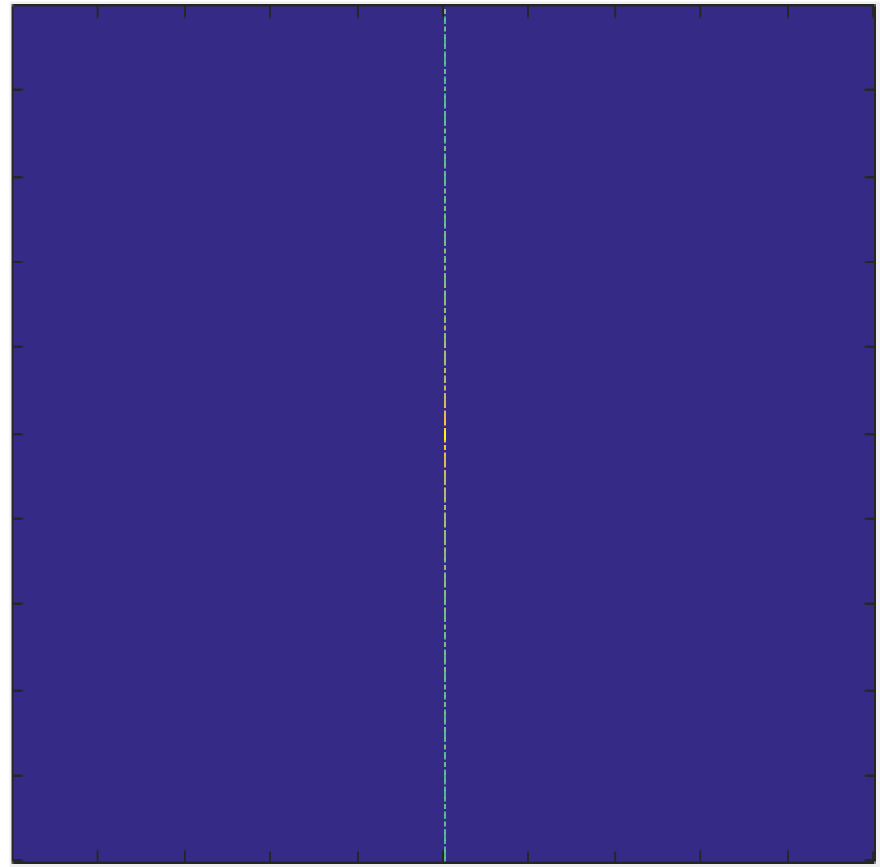






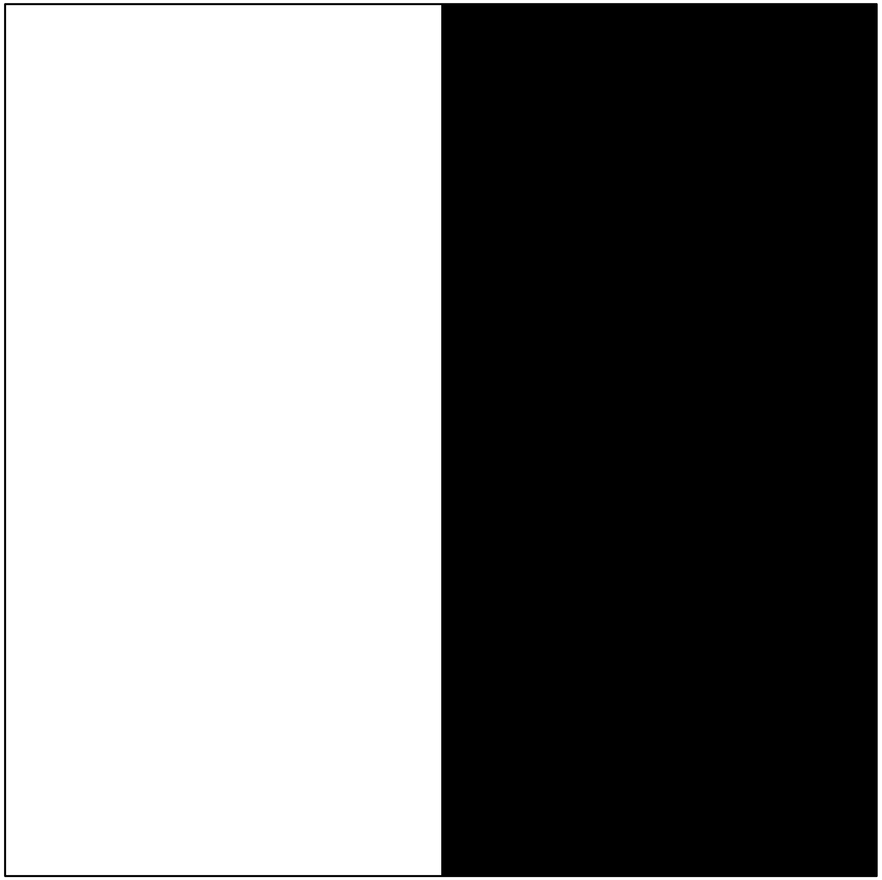
Spatial Domain | *Frequency Domain*

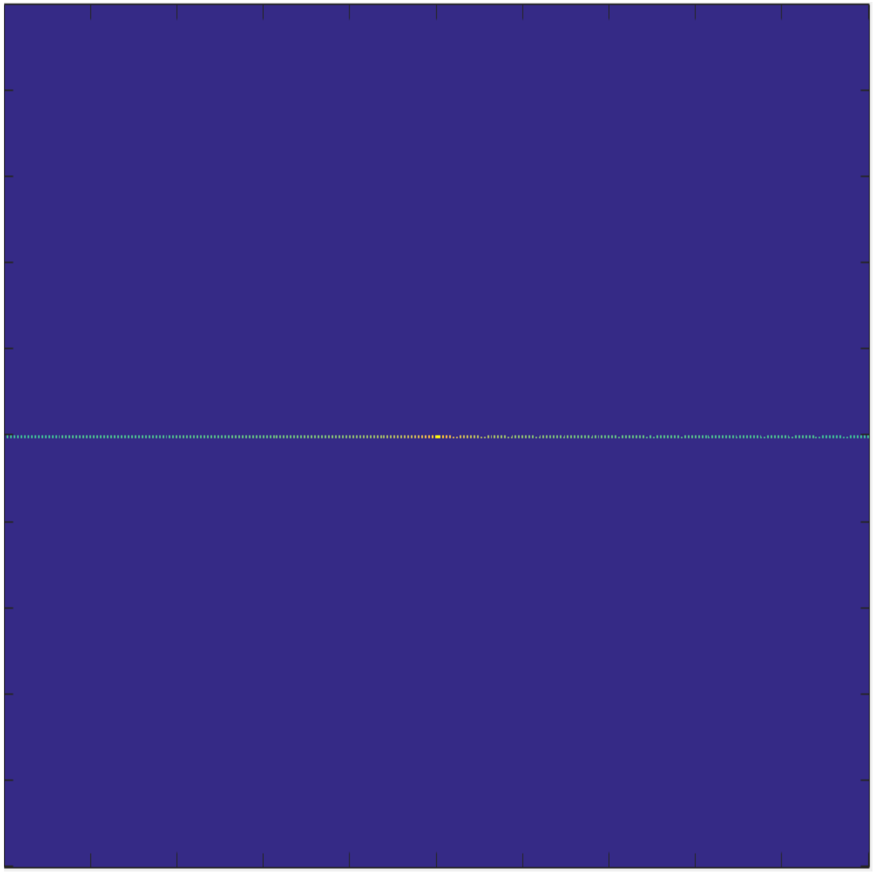




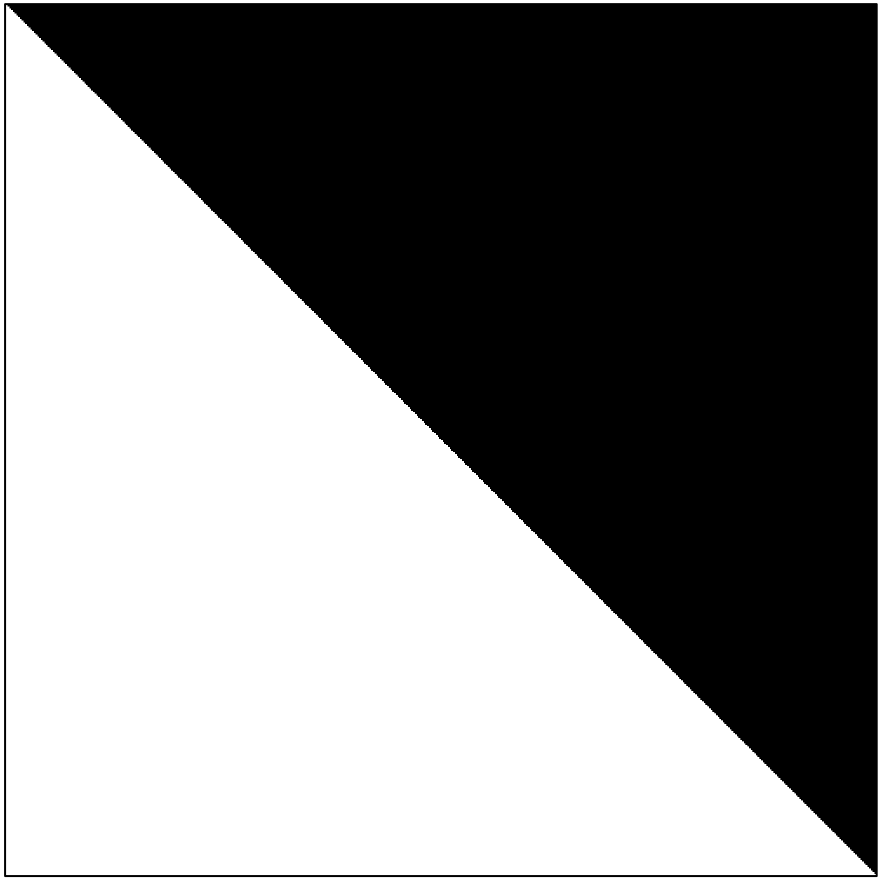
Spatial Domain

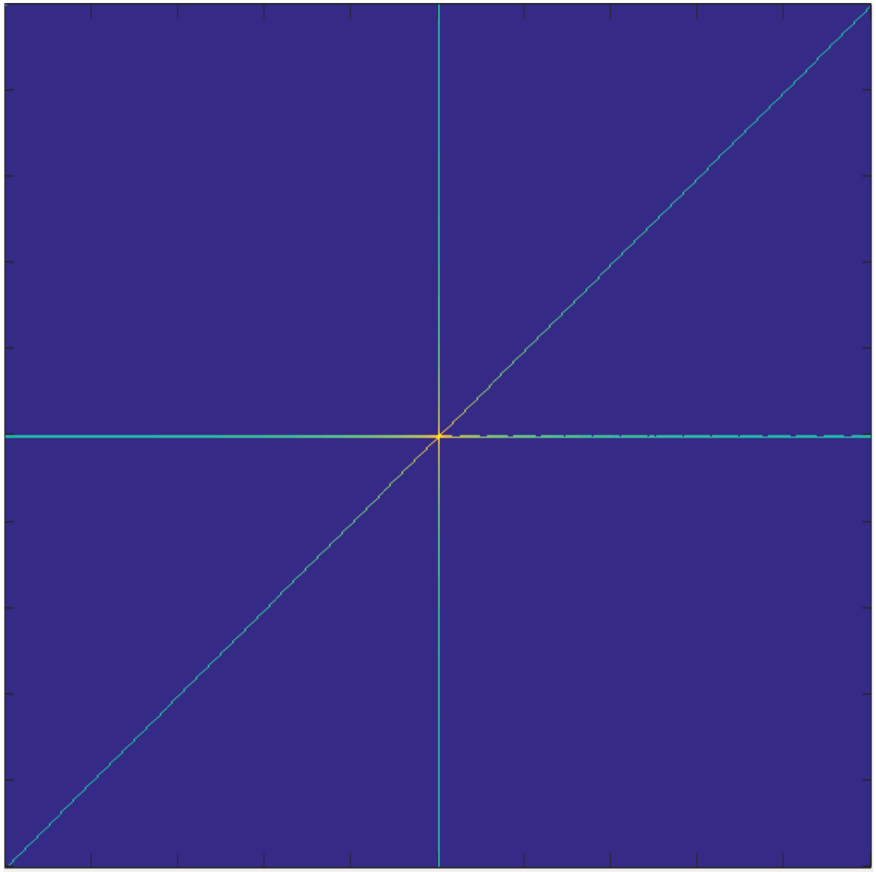
Frequency Domain



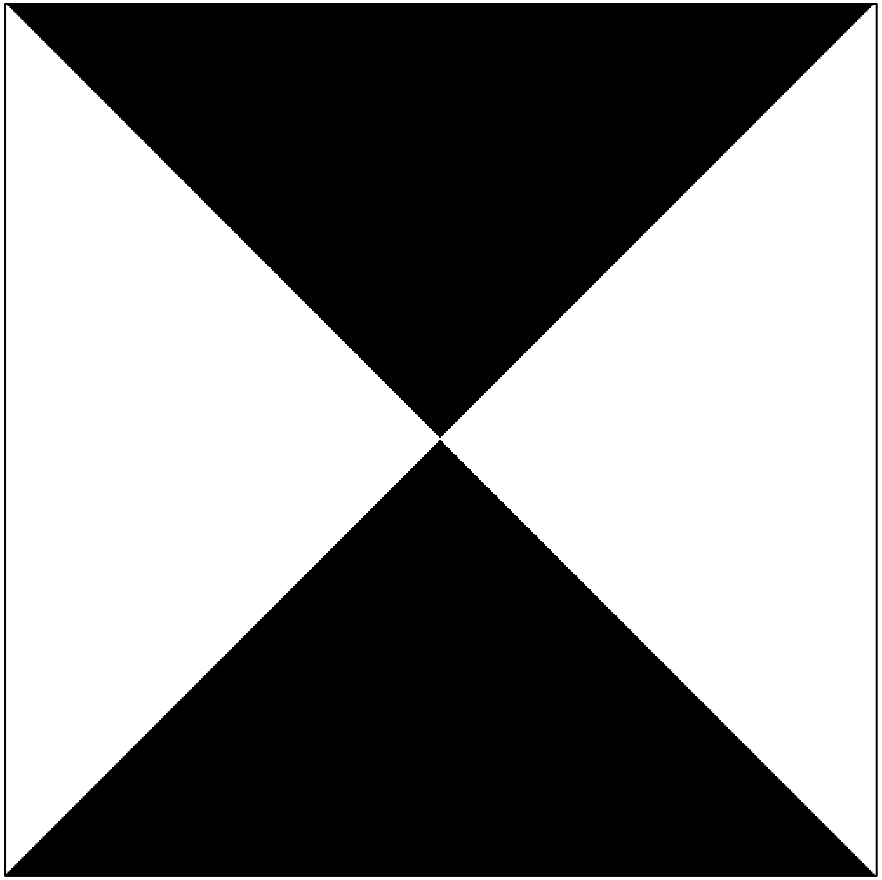


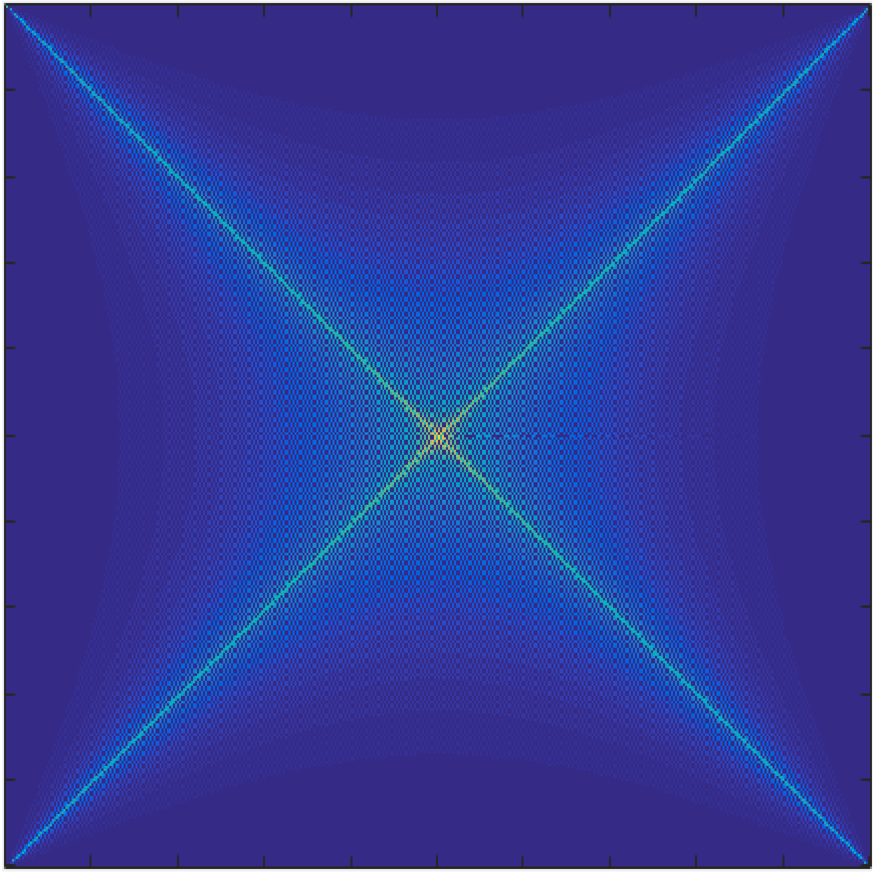
Spatial Domain *Frequency Domain*



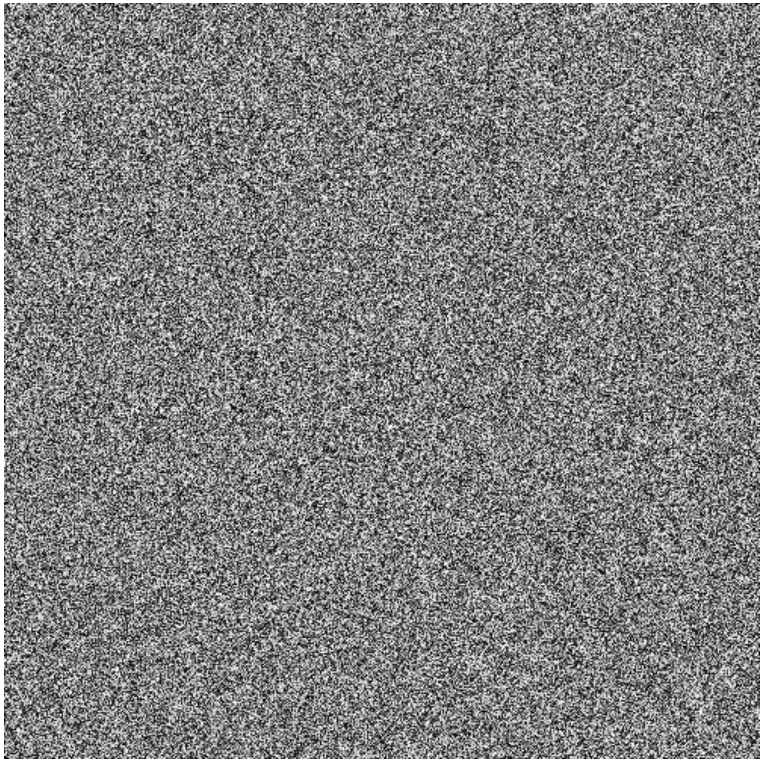


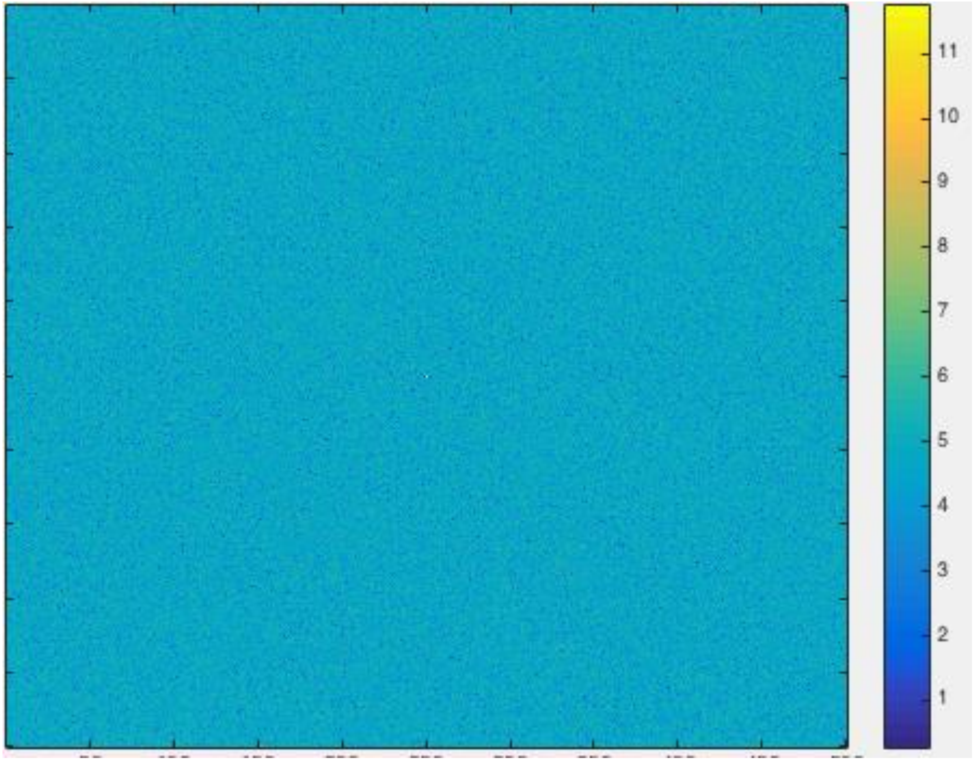
Spatial Domain *Frequency Domain*





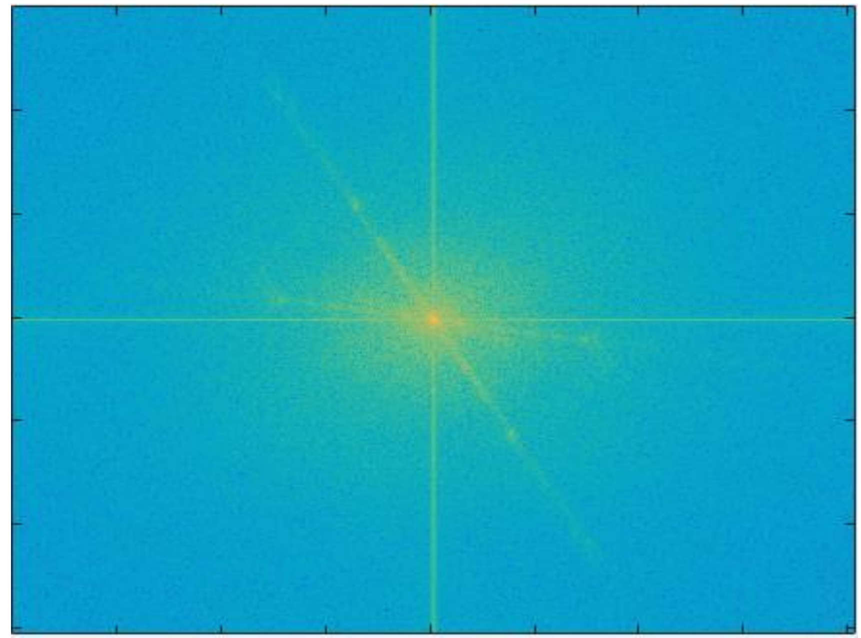
Spatial Domain *Frequency Domain*





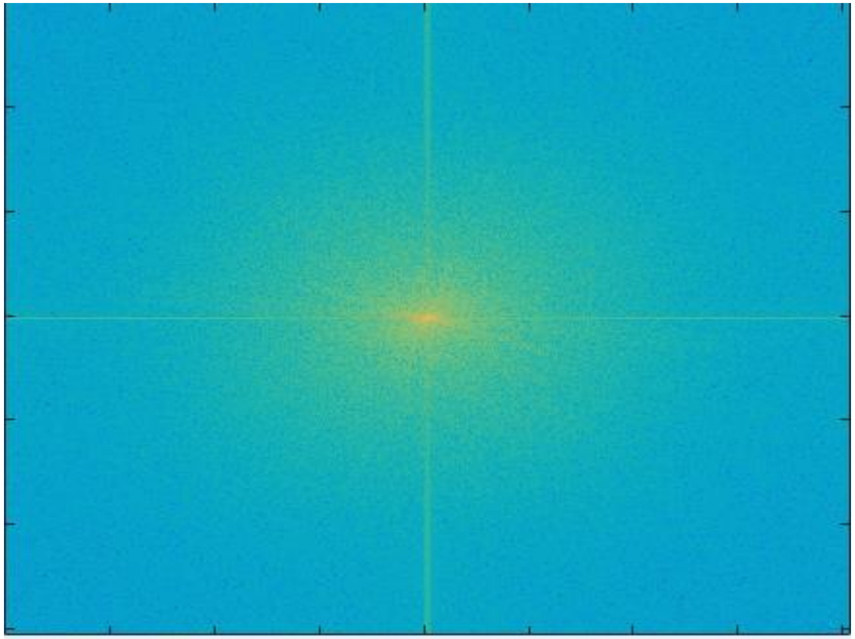
Spatial Domain

Frequency Domain



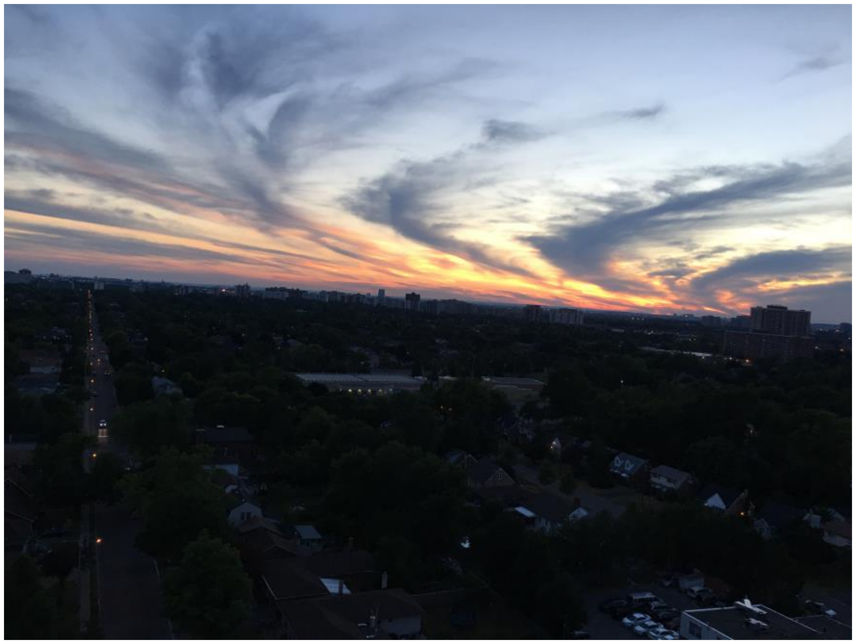
Spatial Domain

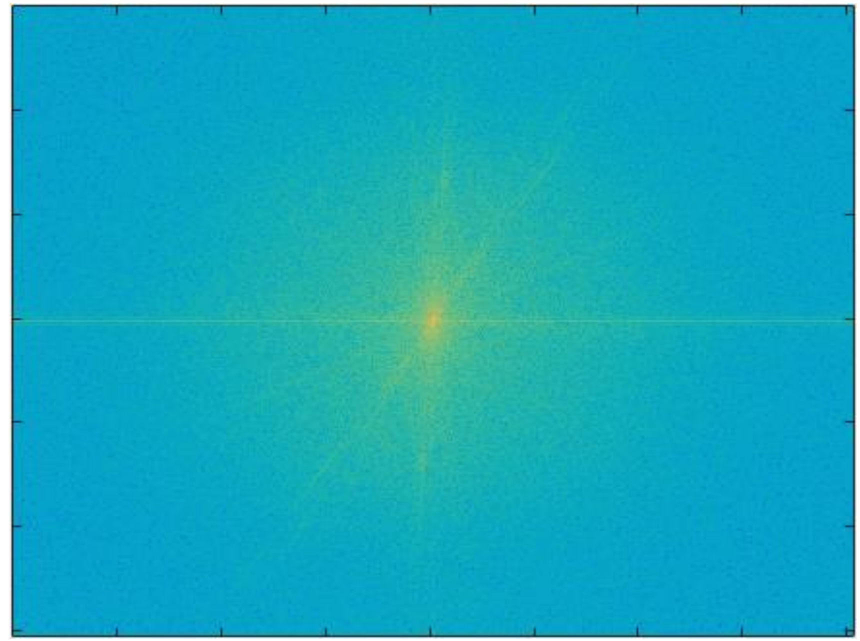
Frequency Domain



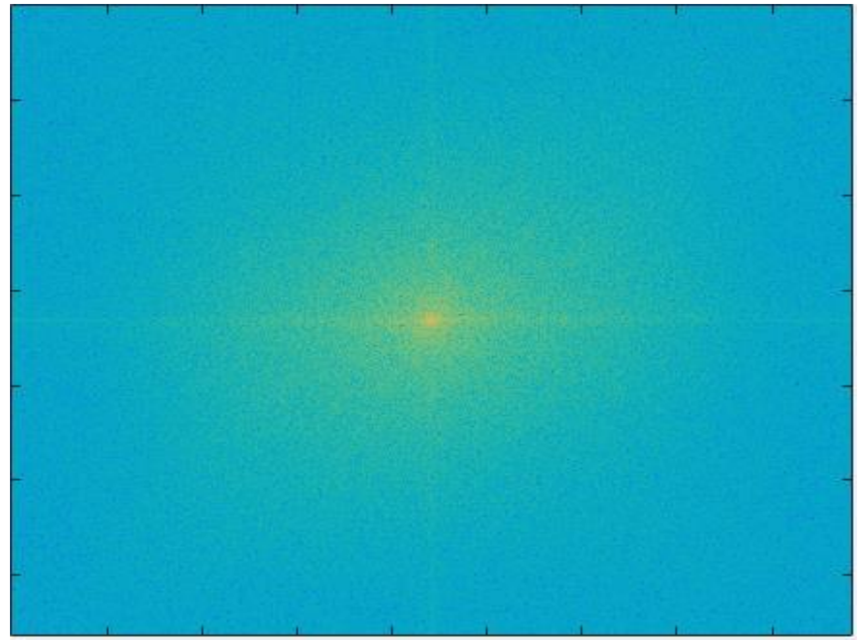
Spatial Domain

Frequency Domain

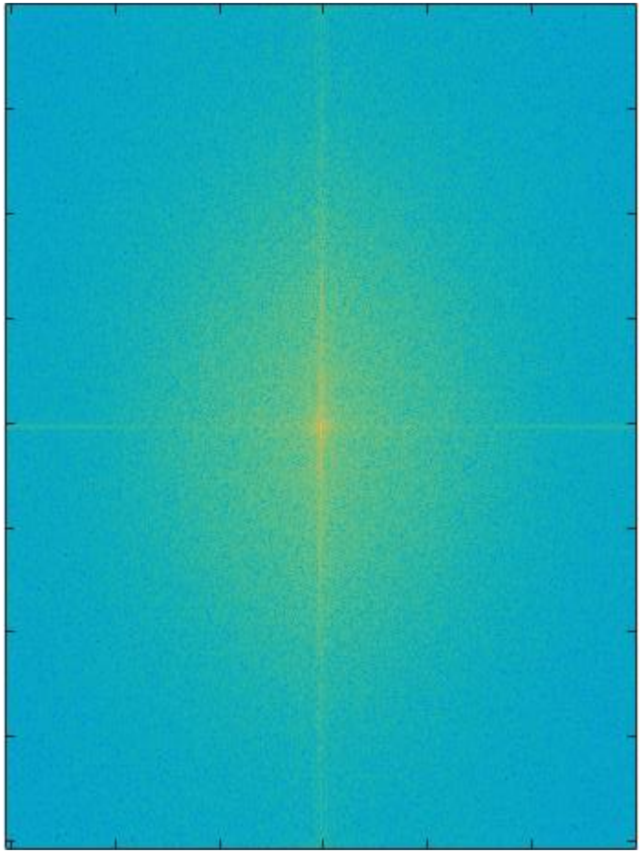




Spatial Domain *Frequency Domain*



Spatial Domain *Frequency Domain*



Filtering in the Frequency Domain

Convolution Theorem

$$f * g \leftrightarrow F.G$$

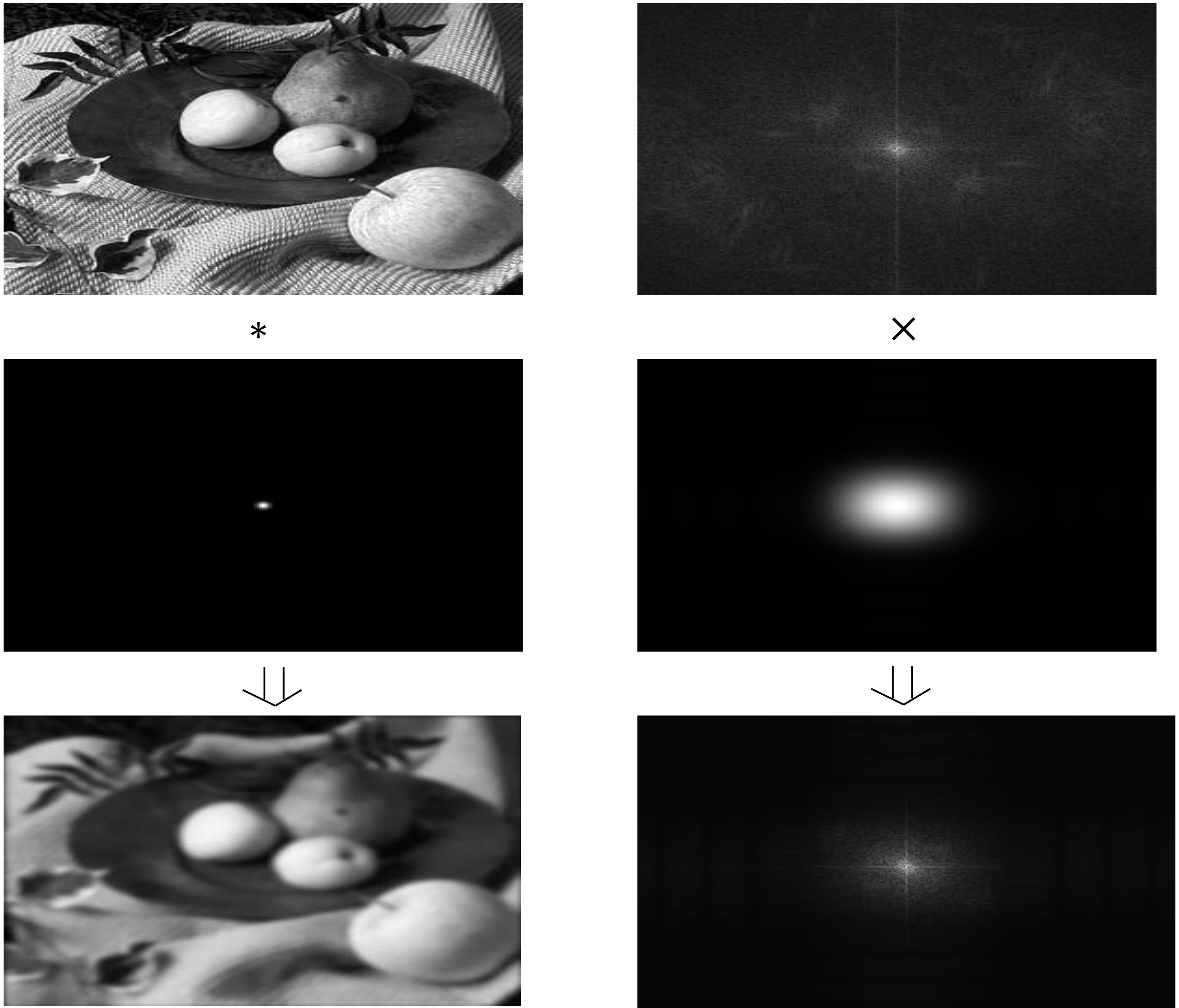
$$\mathcal{F}(f * g) = \mathcal{F}(f).\mathcal{F}(g)$$

$$f * g = \mathcal{F}^{-1}\big(\mathcal{F}(f).\mathcal{F}(g)\big)$$

$$\mathcal{F}(f.g) = \mathcal{F}(f) * \mathcal{F}(g)$$

$$f.g = \mathcal{F}^{-1}\big(\mathcal{F}(f) * \mathcal{F}(g)\big)$$

2D Convolution Theorem Example

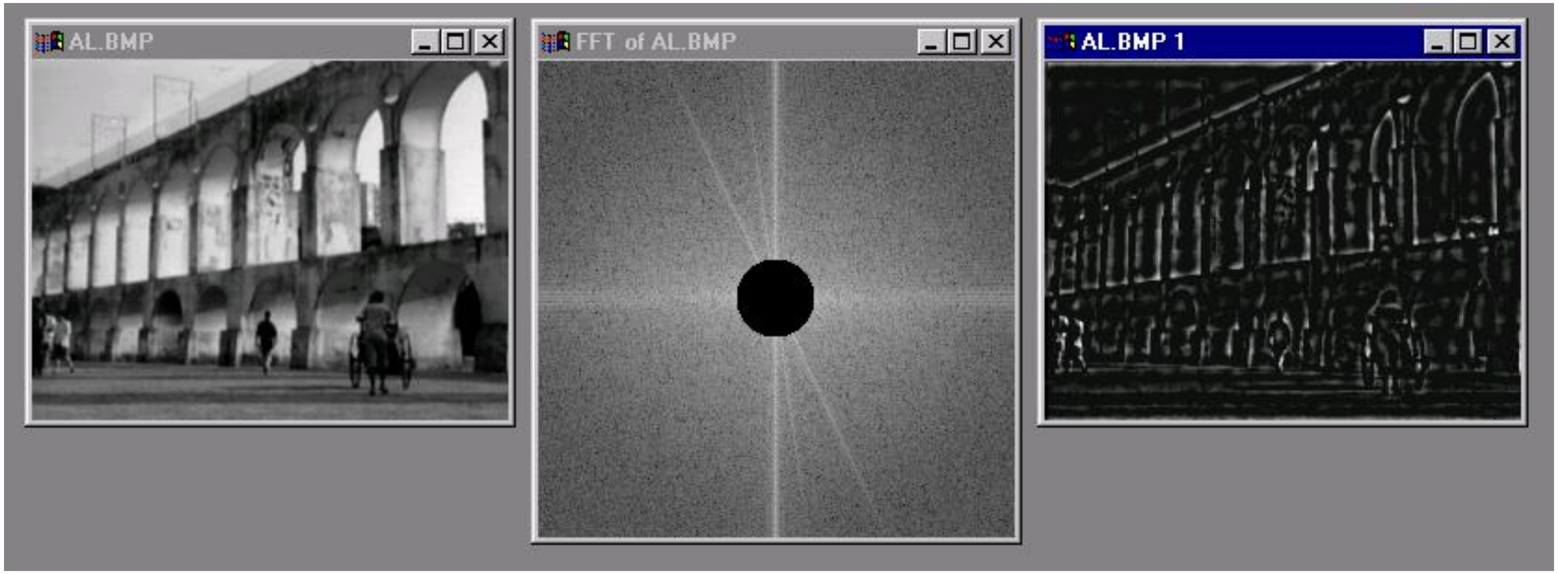


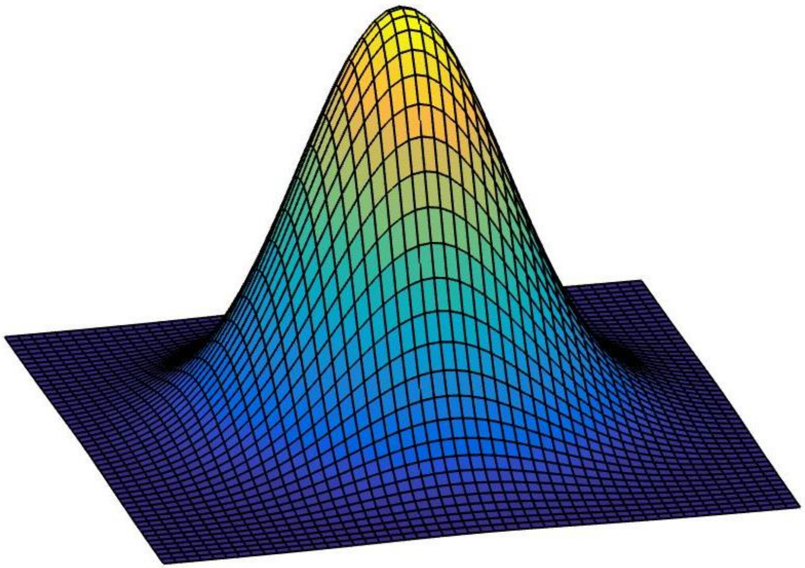
$f(x,y)$ $\quad * \quad$ $h(x,y)$ $\quad \Downarrow \quad$ $g(x,y)$

$\|F(u,v)\|$ $\quad \times \quad$ $\|H(u,v)\|$ $\quad \Downarrow \quad$ $\|G(u,v)\|$

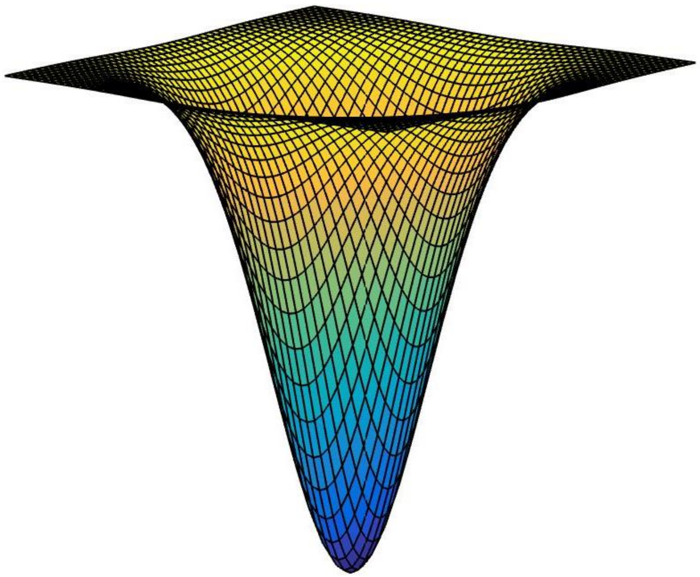
Low Pass Filtering

High Pass Filtering

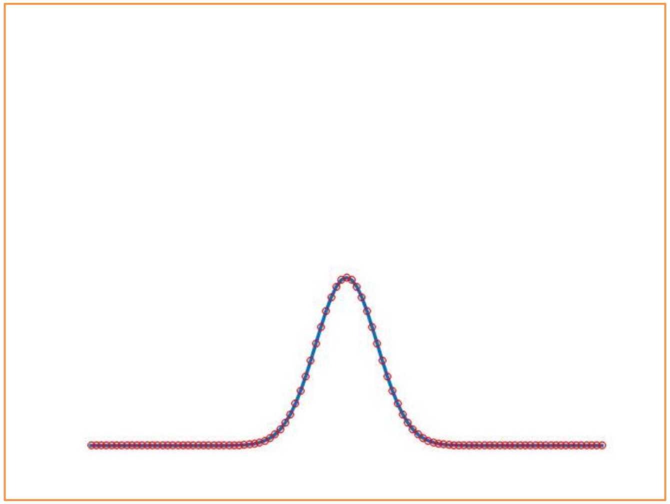
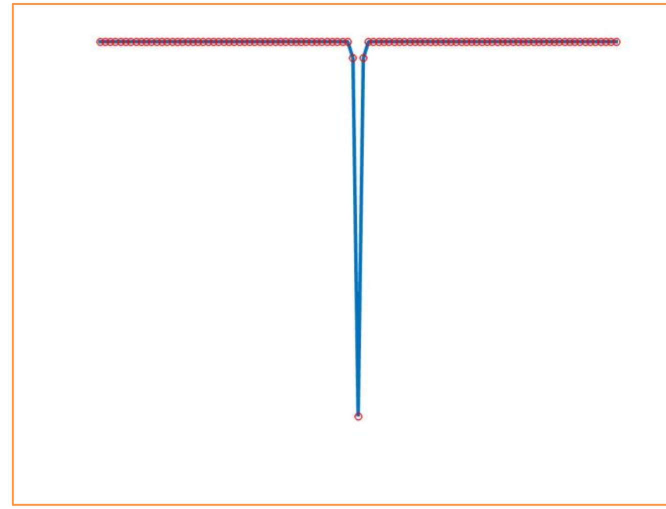
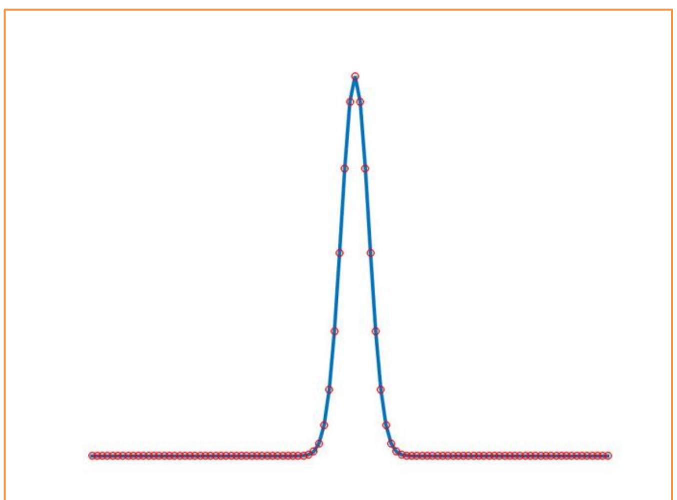
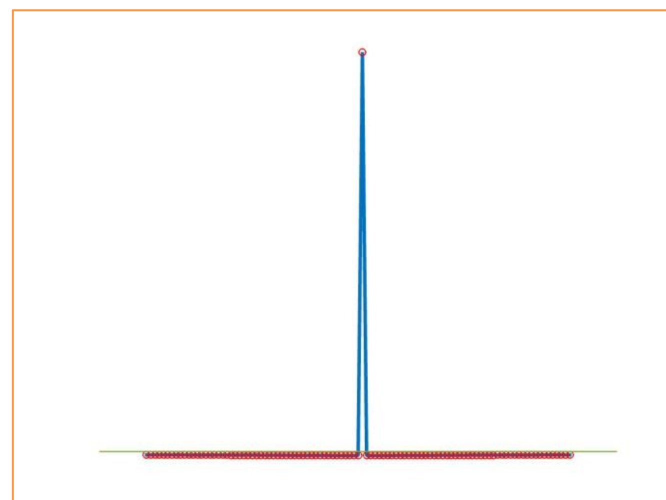




Lowpass Filter

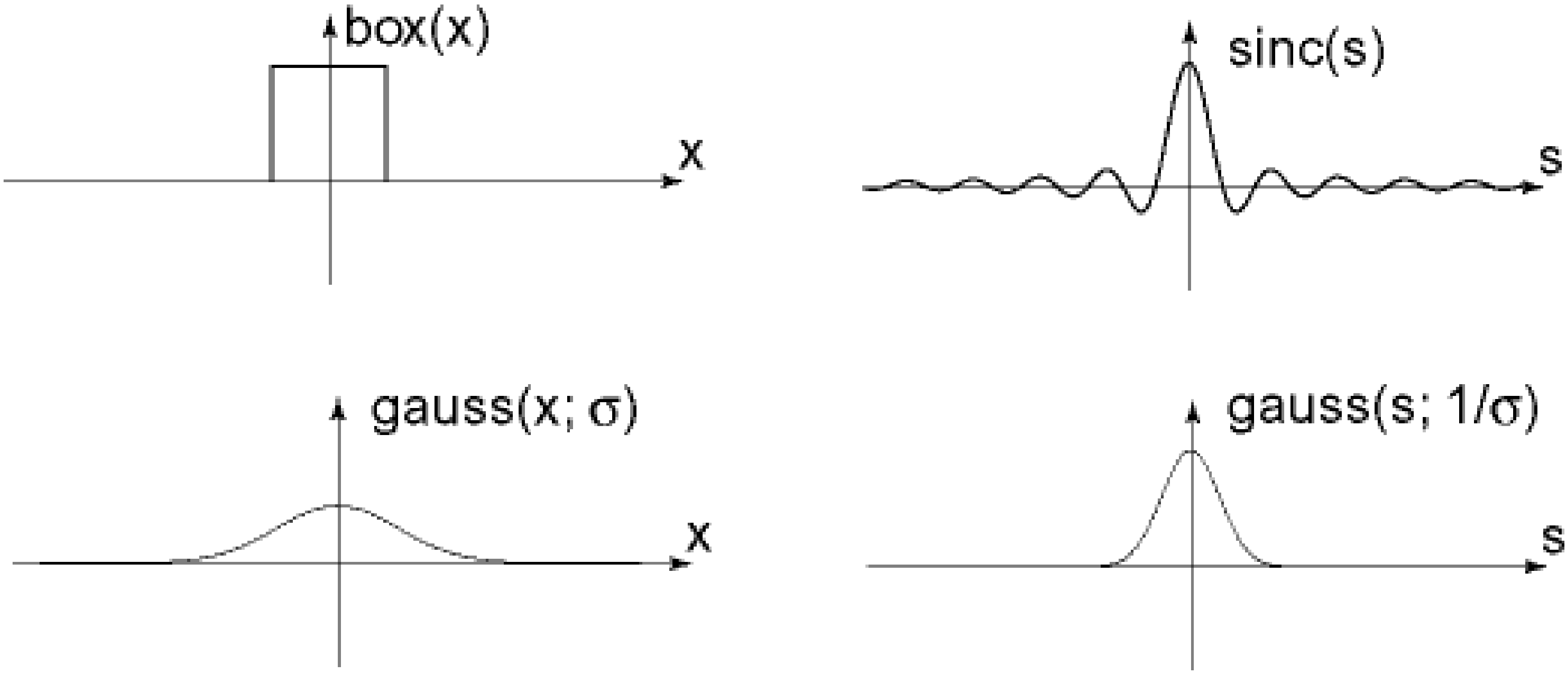


Highpass Filter

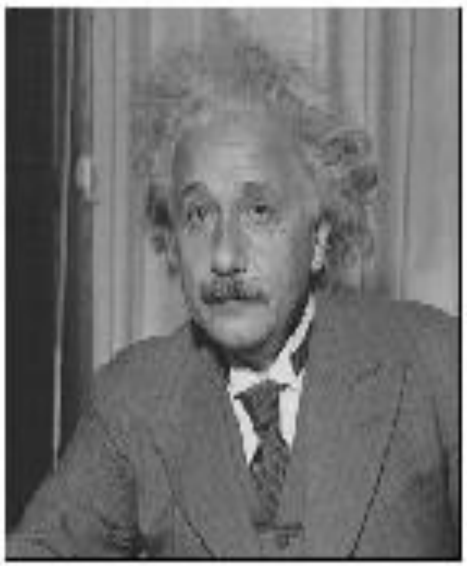
	Lowpass	***Highpass***
Frequency	$H(u) = A\, e^{-\frac{u^2}{2\sigma^2}}$ 	
Spatial	$h(u) = \sqrt{2\pi}\sigma A\, e^{-\pi^2\sigma^2x^2}$ 	

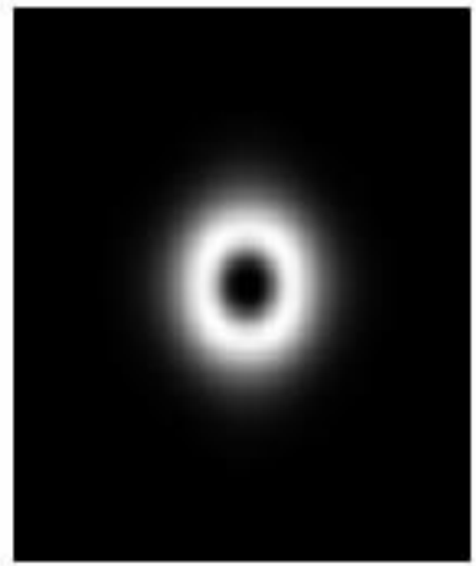
Fourier Transform pairs

Spatial domain | Frequency domain



Band Pass Filtering





Ideal Lowpass Filter

$$H(u,v) = \begin{cases} 0 & D(u,v) > D_0 \\ 1 & D(u,v) \le D_0 \end{cases} \qquad D(u,v) = \sqrt{u^2 + v^2}$$

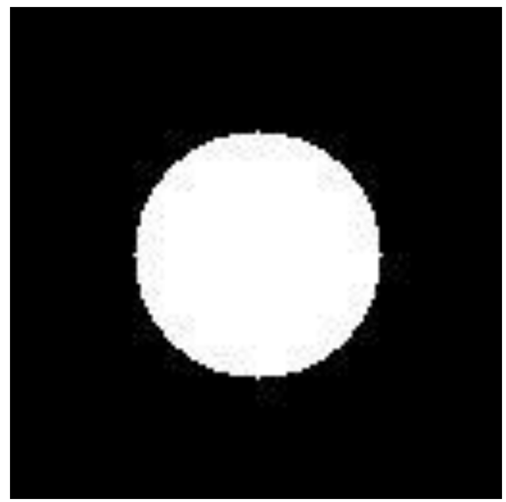
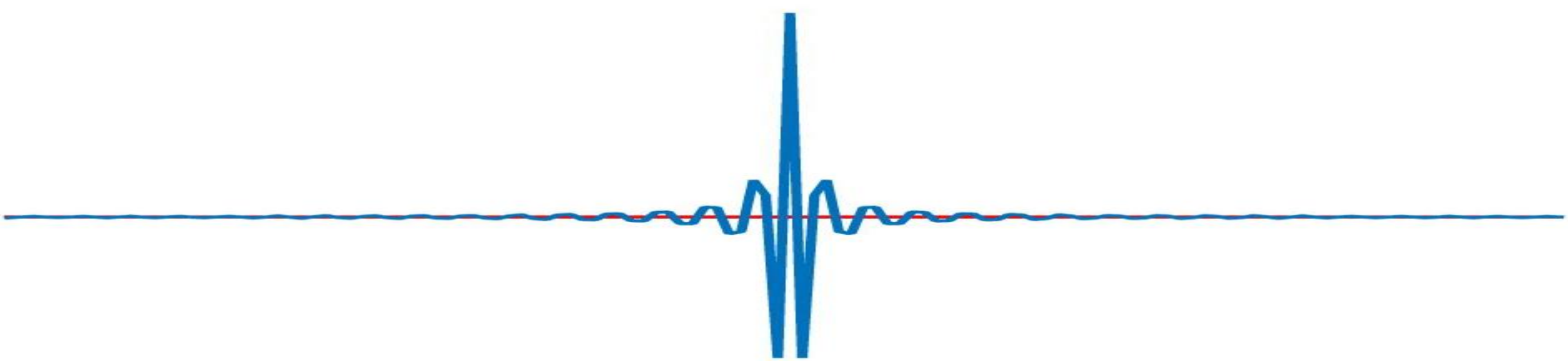


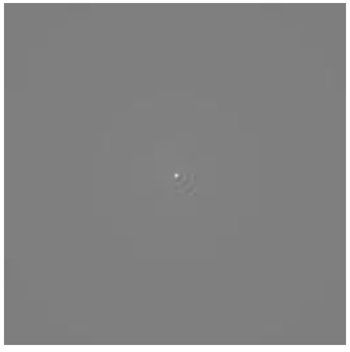
*Image Size = 128*128*
Radius = 32



Butterworth Lowpass Filter

$$H(u,v) = \frac{1}{1 + \left[D(u,v) / D_0 \right]^n}$$

Image Size = 128
Radius = 32
N = 0.02



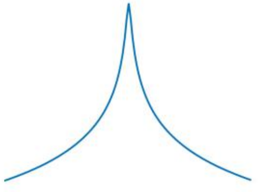
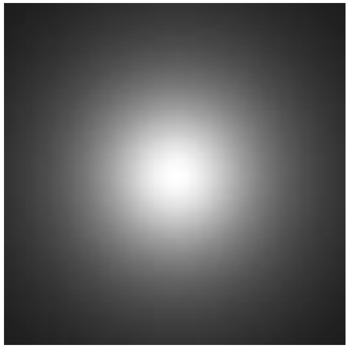


Image Size = 128
Radius = 32
N = 2



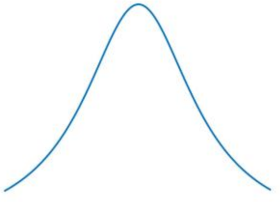
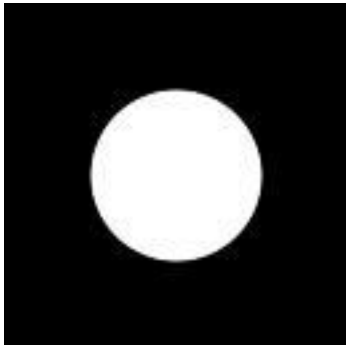
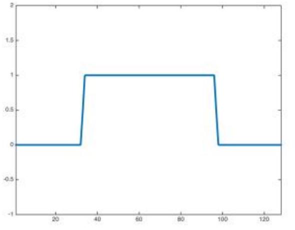


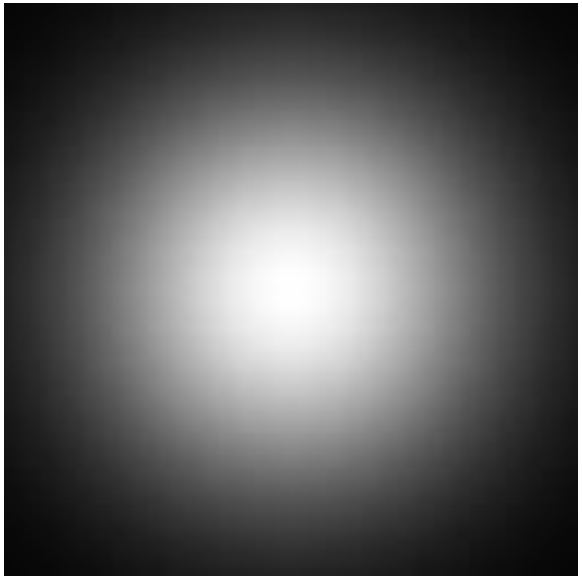
Image Size = 128
Radius = 32
N = 200

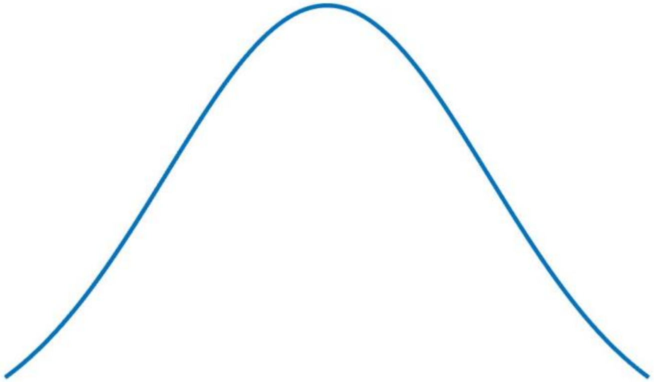




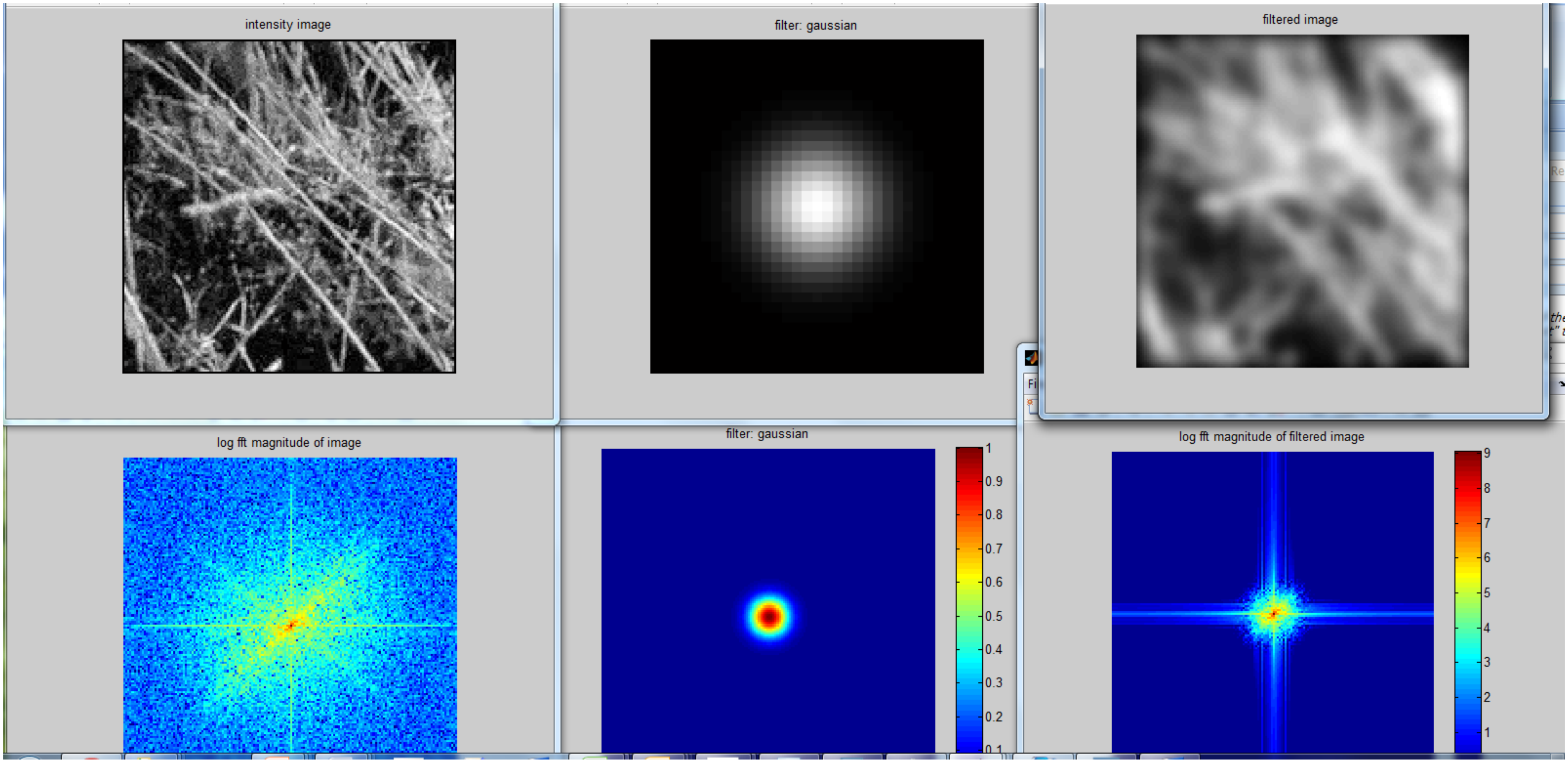
Gaussian Lowpass Filter

$$H(u,v) = e^{-D^2(u,v)/2D_0^2}$$

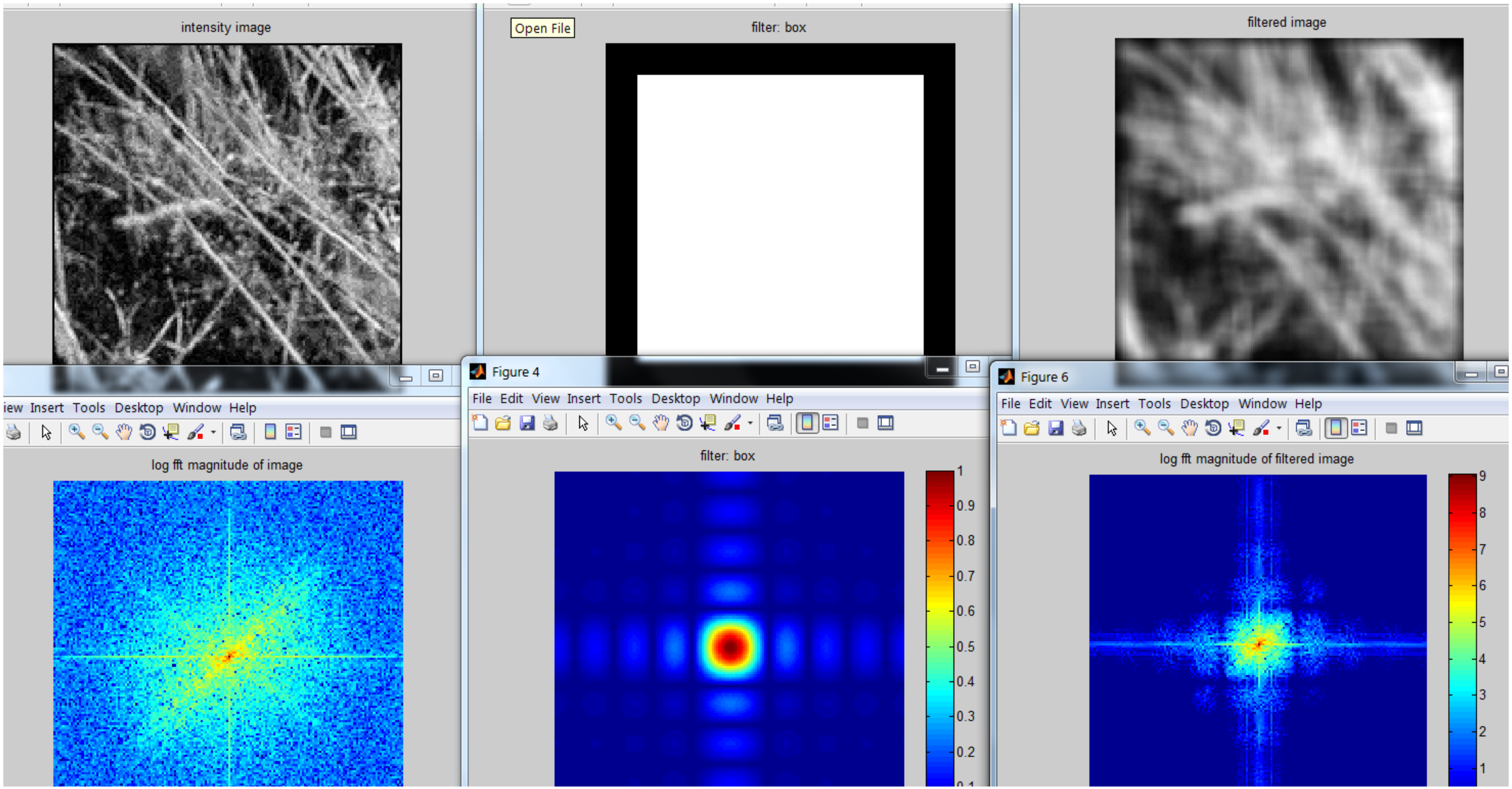




Gaussian Lowpass Filter



Box Filter


intensity image
Open File
filter: box
filtered image
Figure 4
Figure 6
File Edit View Insert Tools Desktop Window Help
log fft magnitude of image
filter: box
log fft magnitude of filtered image
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
9
8
7
6
5
4
3
2
1

Highpass Filters for Sharpening

Ideal: $H(u,v) = \begin{cases} 1 & D(u,v) > D_0 \\ 0 & D(u,v) \leq D_0 \end{cases}$

Butterworth: $H(u,v) = \dfrac{1}{1 + \left[D_0 / D(u,v) \right]^n}$

Gaussian: $H(u,v) = 1 - e^{-D^2(u,v)/2D_0^2}$

Laplacian in the Frequency Domain

$$\mathcal{F}\left(\frac{d^n f(x)}{dx^n}\right) = (ju)^n F(u)$$

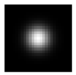
$$\mathcal{F}\left(\frac{d^2 f(x,y)}{dx^2} + \frac{d^2 f(x,y)}{dy^2}\right) =$$

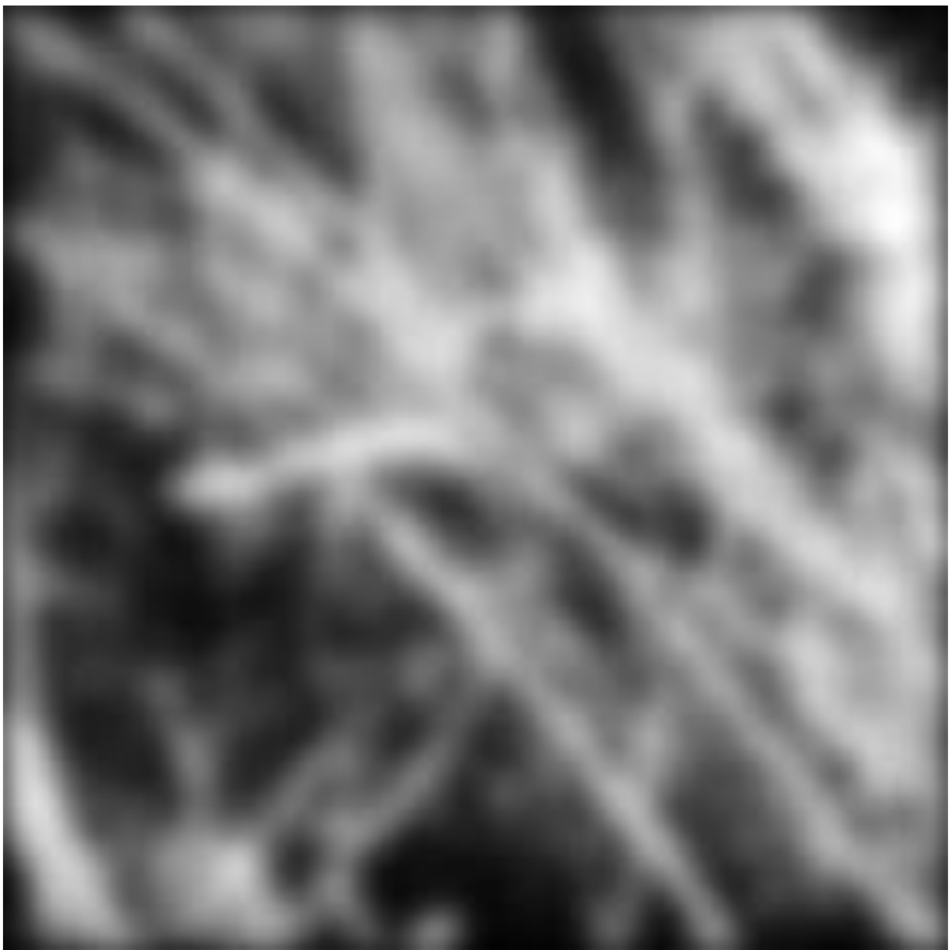
$$(ju)^2 F(u,v) + (jv)^2 F(u,v) =$$

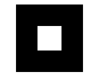
$$-(u^2 + v^2)F(u,v)$$

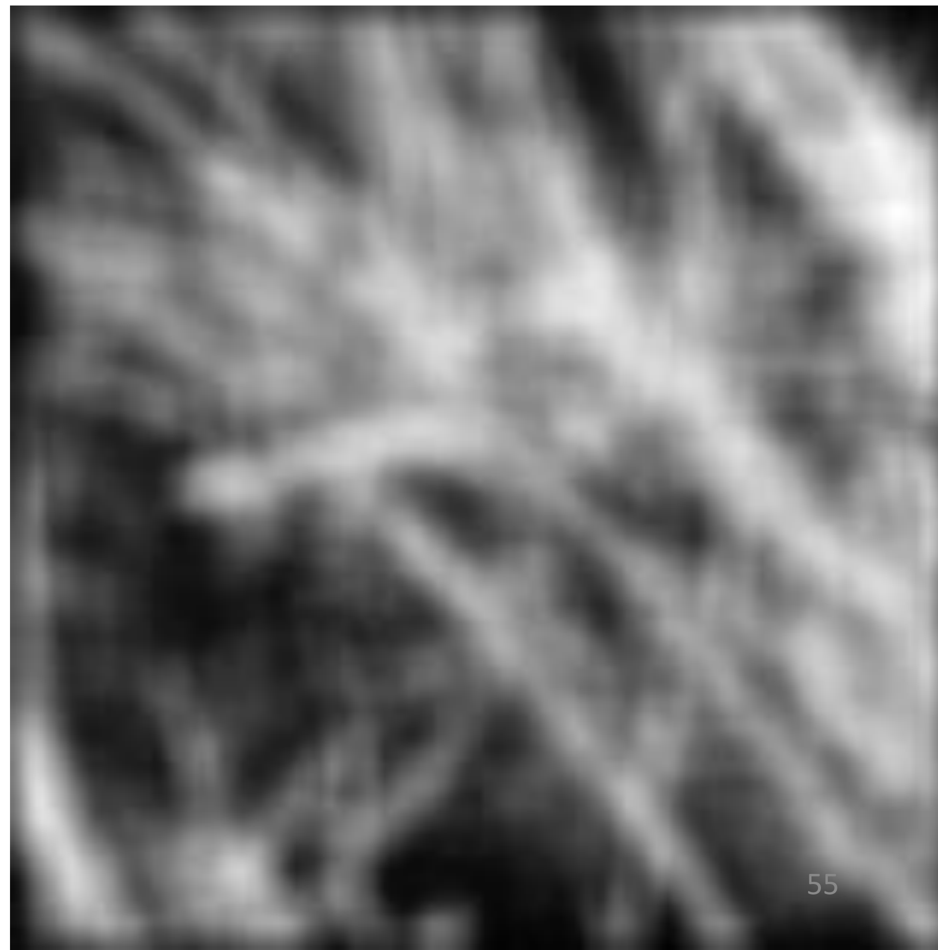
$$\mathcal{F}\left(\nabla^2 f(x,y)\right) = -(u^2 + v^2)F(u,v)$$

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

Gaussian 



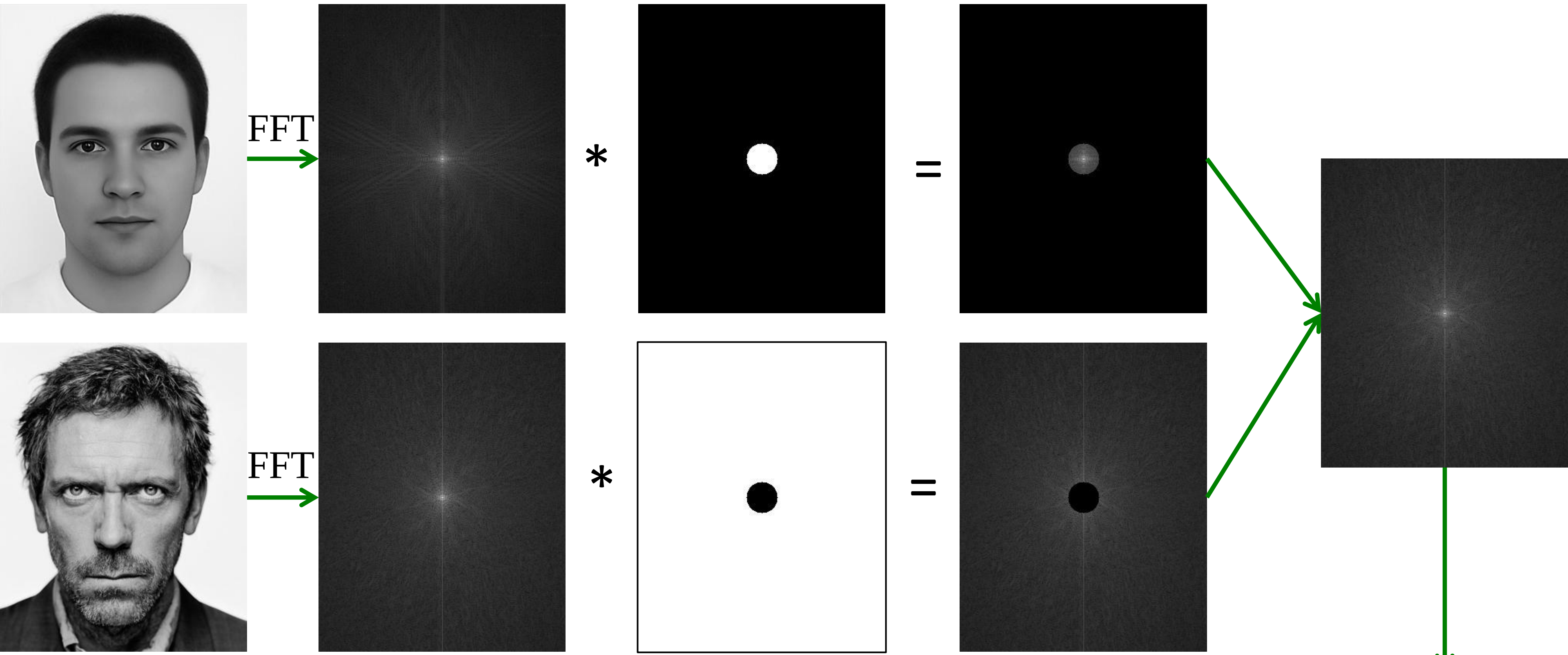
Box filter 



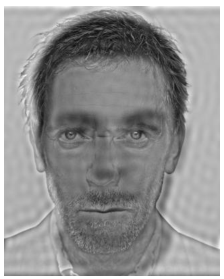
Why does a lower resolution image still make sense to us? What do we lose?

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

Hybrid Images

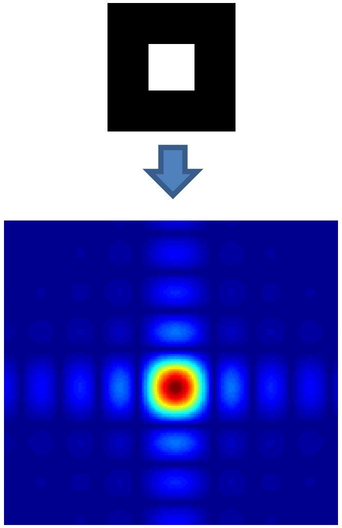


A. Oliva, A. Torralba, P.G. Schyns,
"Hybrid Images," SIGGRAPH 2006

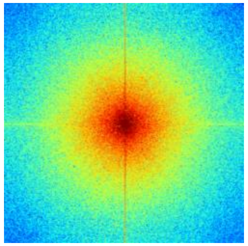


Things to Remember

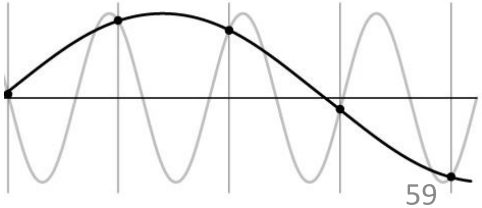
- Sometimes it makes sense to think of images and filtering in the frequency domain
  - Fourier analysis



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



- Images are mostly smooth
  - Basis for compression



- Remember to low-pass before sampling

Slide: Derek Hoiem

References

- Fourier Transform
  Gonzalez, chapter 4
  Szeliski, section 3.4