Filtering (in Frequency Domain)

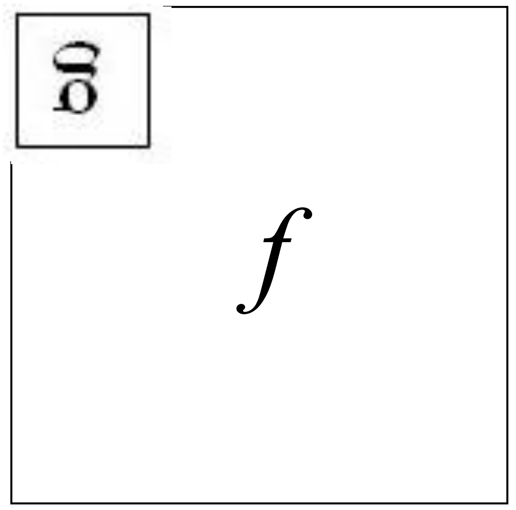
CS 543 / ECE 549 – Saurabh Gupta

Spring 2020, UIUC

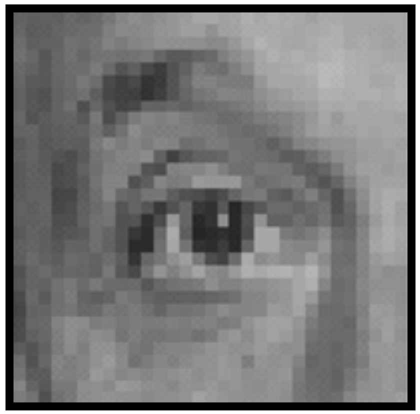
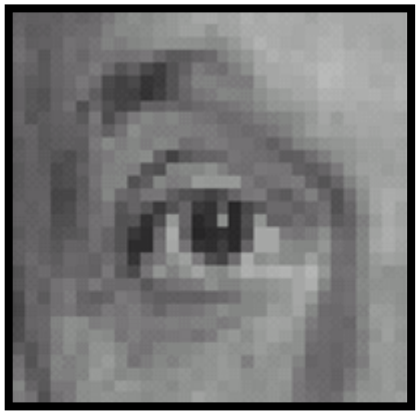
http://saurabhg.web.illinois.edu/teaching/ece549/sp2020/

Many slides from Derek Hoiem.

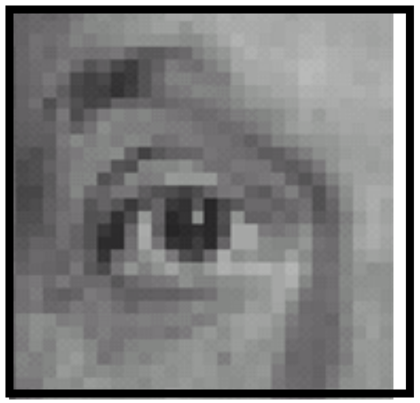
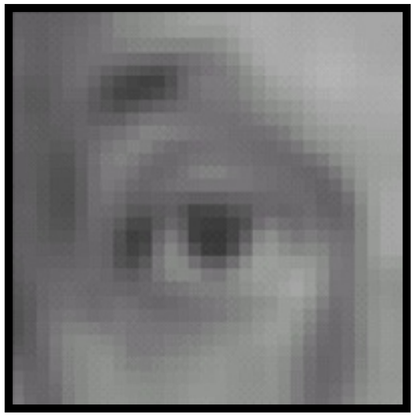
Recap



Convolution

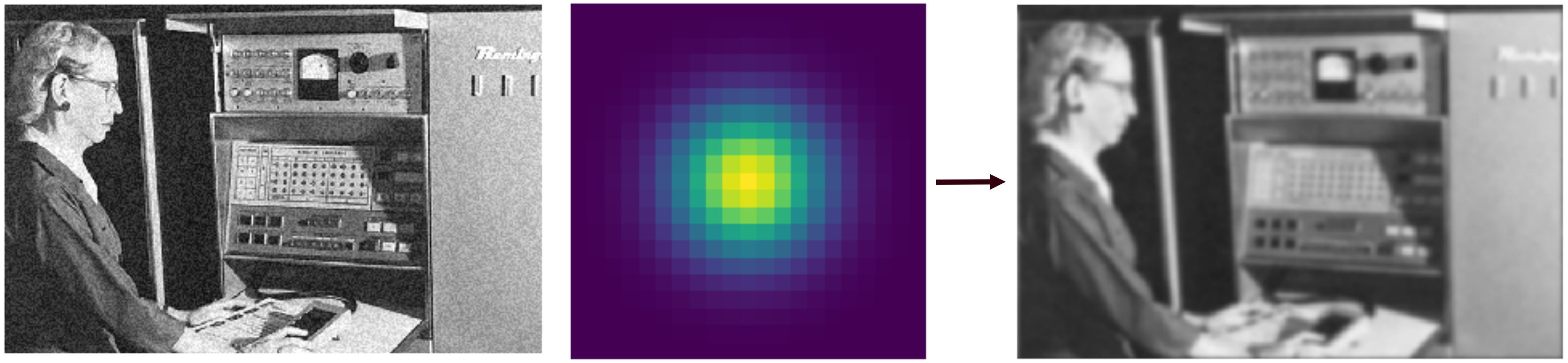



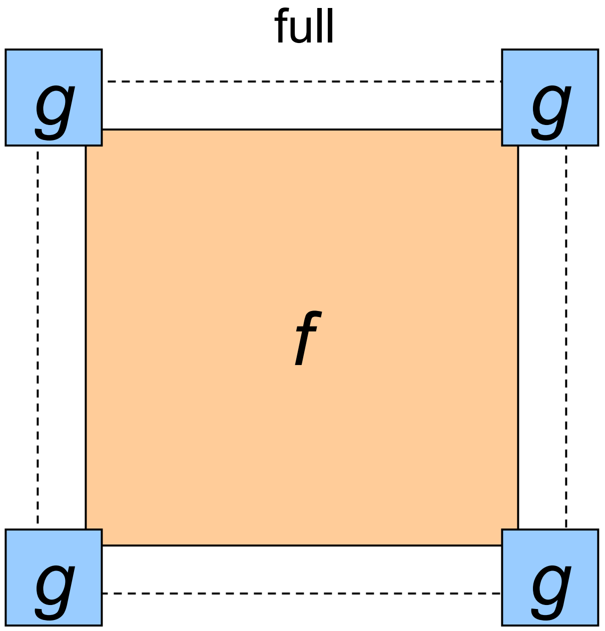
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \qquad \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

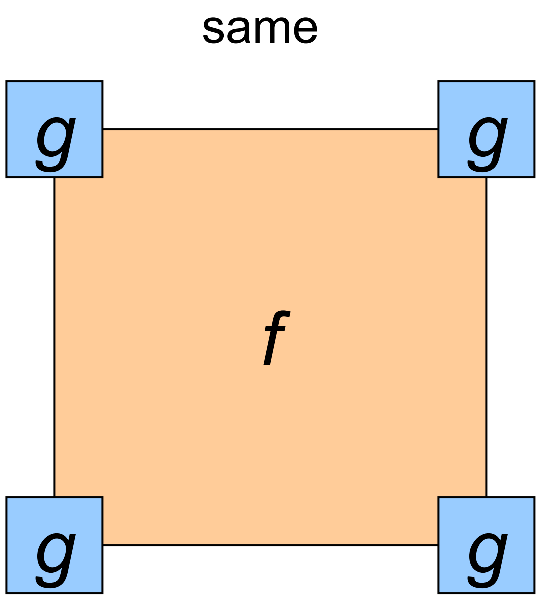



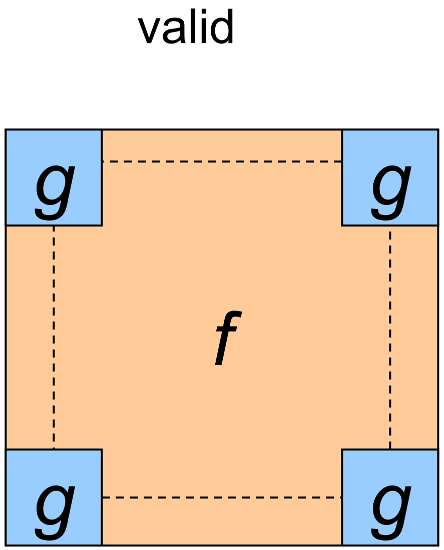
Source: F. Durand, D. Lowe

Recap



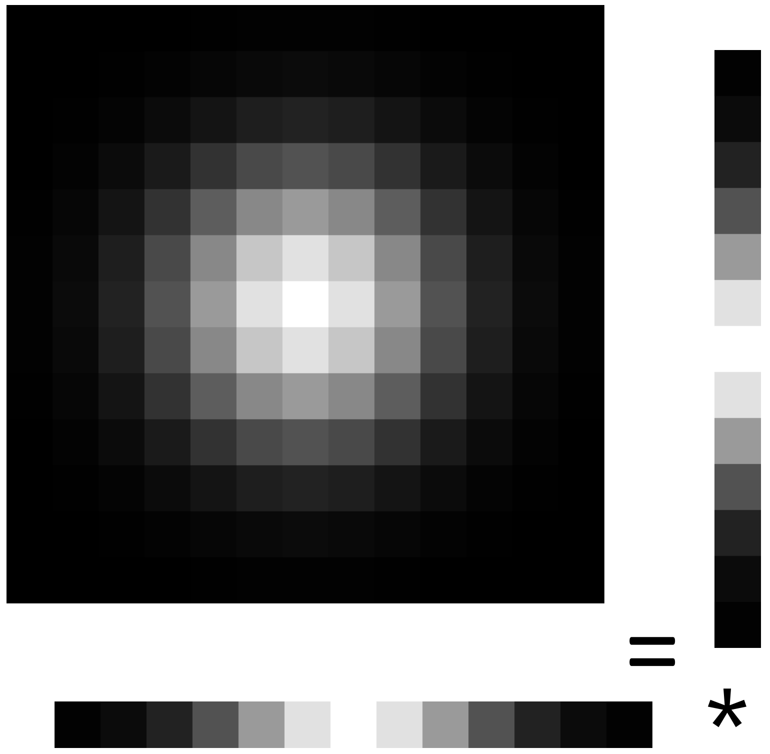




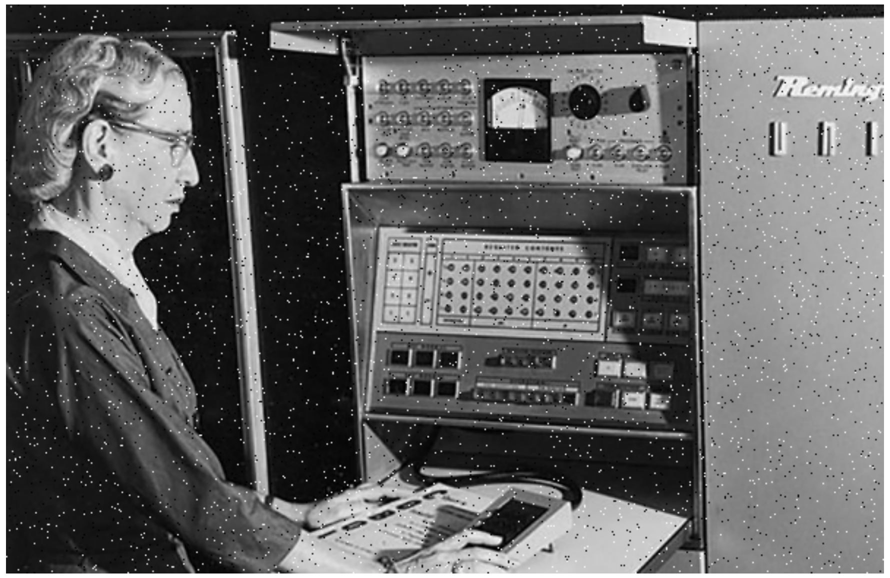


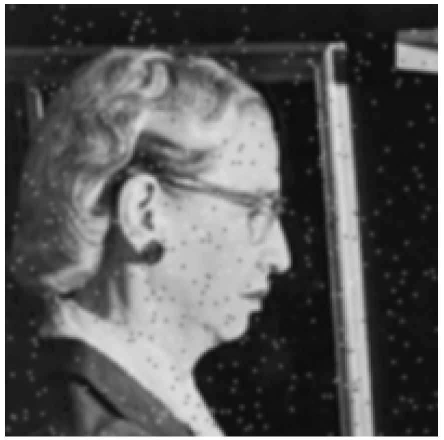
f/g Diagram Credit: D. Lowe

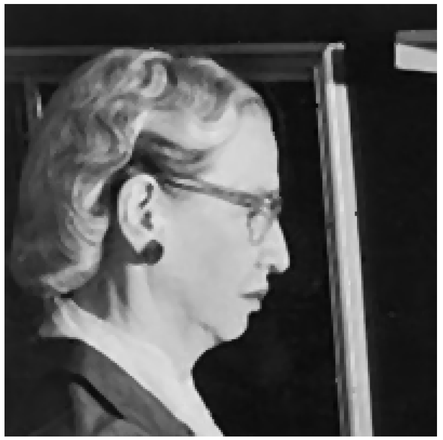
Recap



Separable Filters for Efficiency







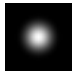
Gaussian Filtering Median Filtering

Today's Class

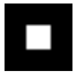
- Fourier transforms
- Filtering in frequency domain
- Sampling
- Image Pyramids

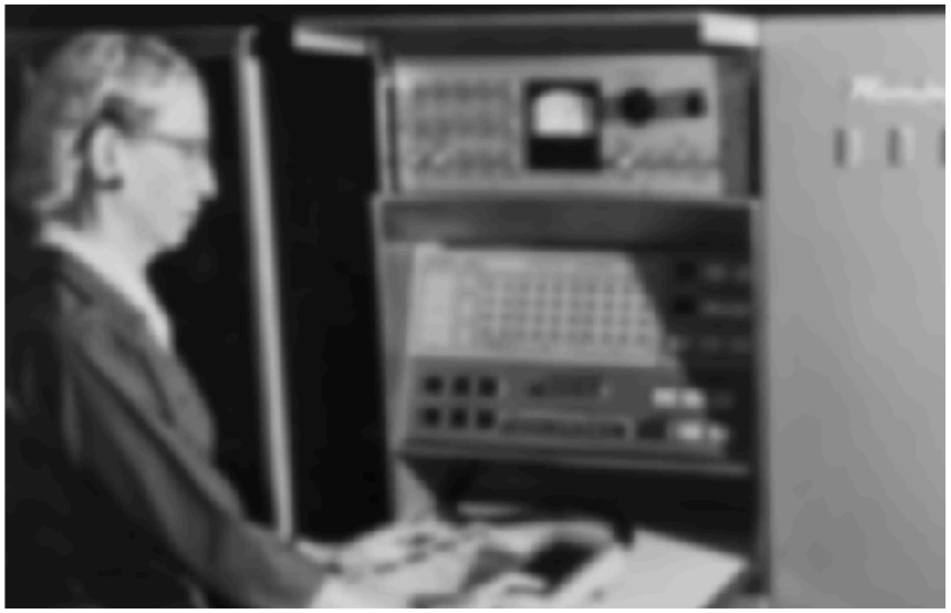
Why does the Gaussian give a nice smooth image, but the square filter give edgy artifacts?

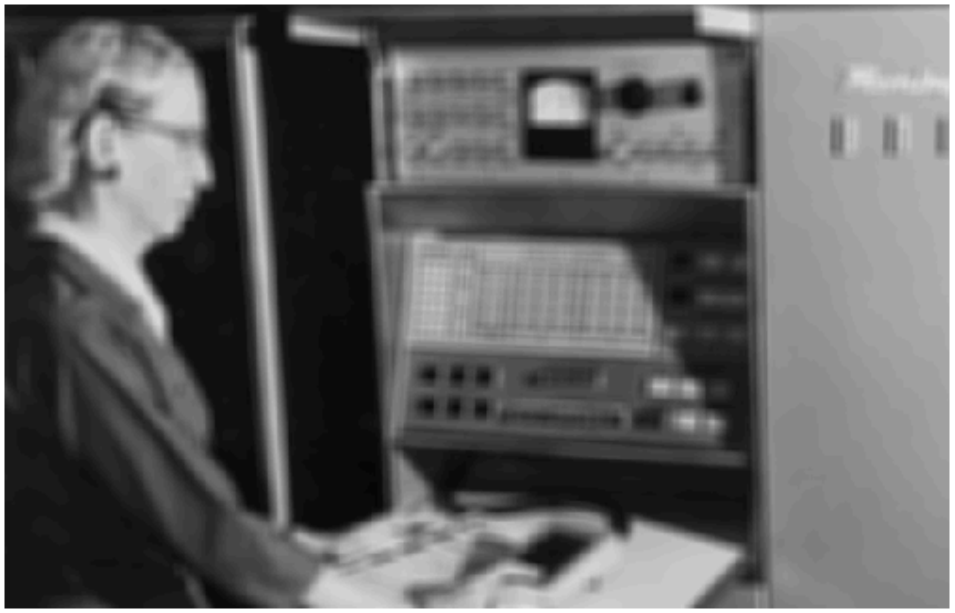
Gaussian



Box filter







Thinking in terms of frequency

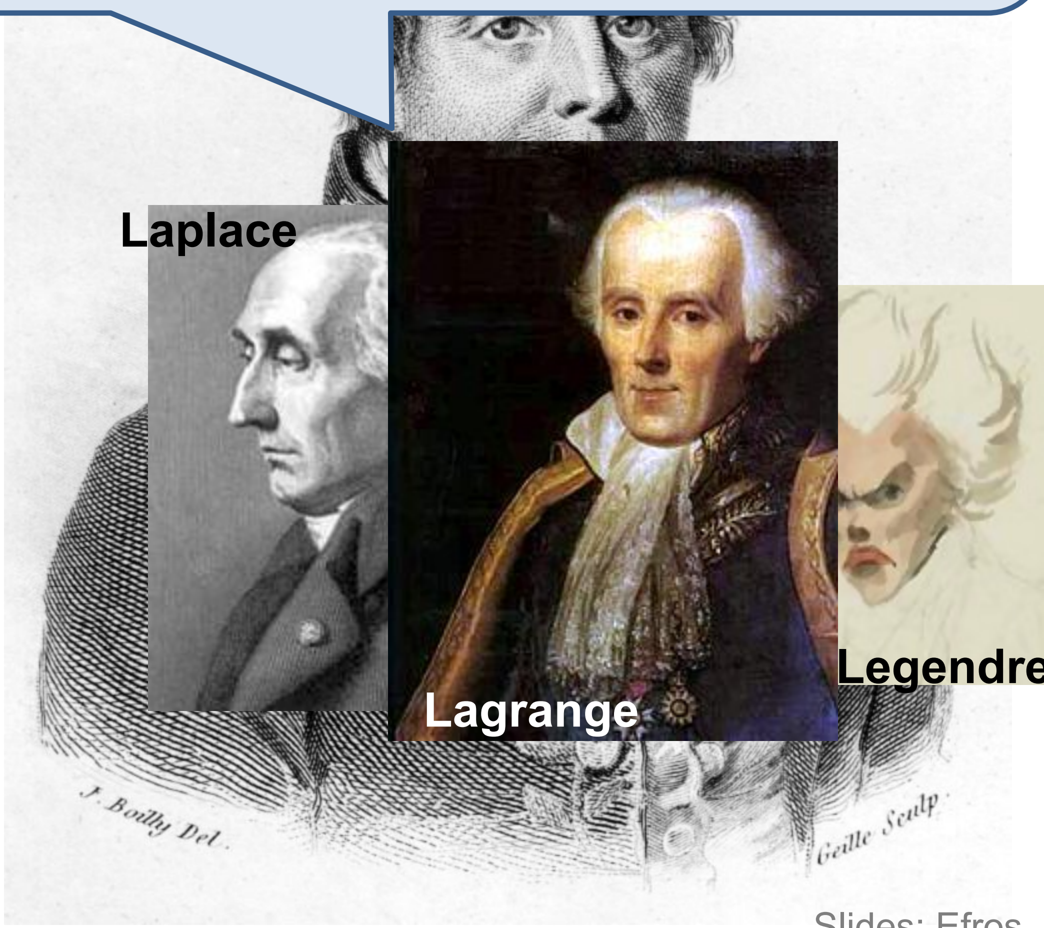
Jean Baptiste Joseph Fourier (1768-1830)

had crazy idea (1807):

> ***Any*** *univariate function can* *rewritten as a weighted sum* *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



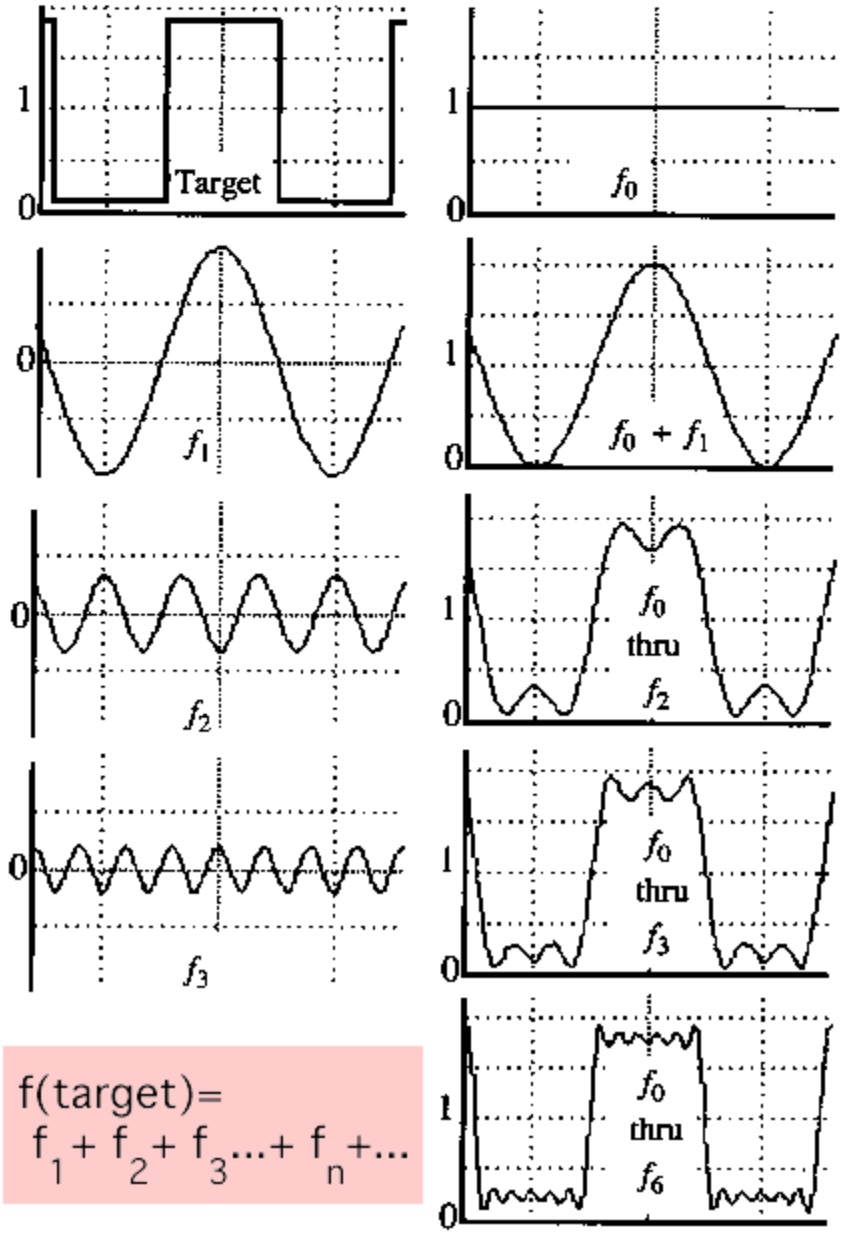
Slides: Efros

A sum of sines

Our building block:

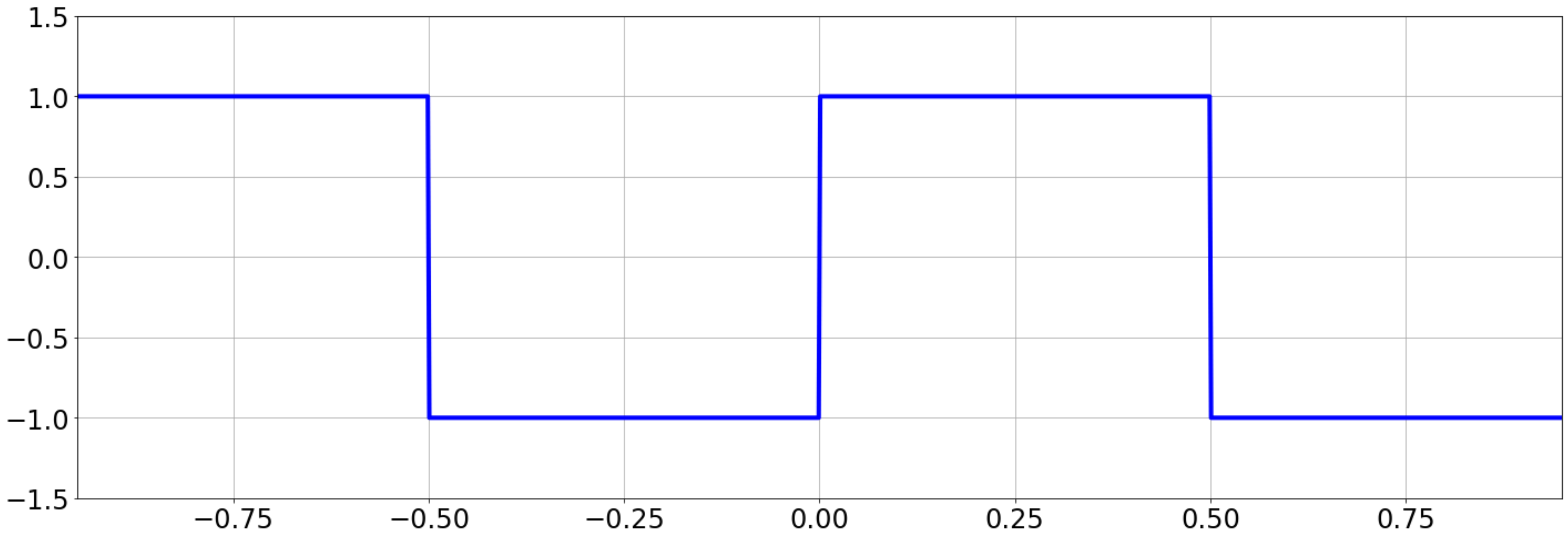
$$A\sin(\omega x+\phi)$$

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



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

Frequency Spectra

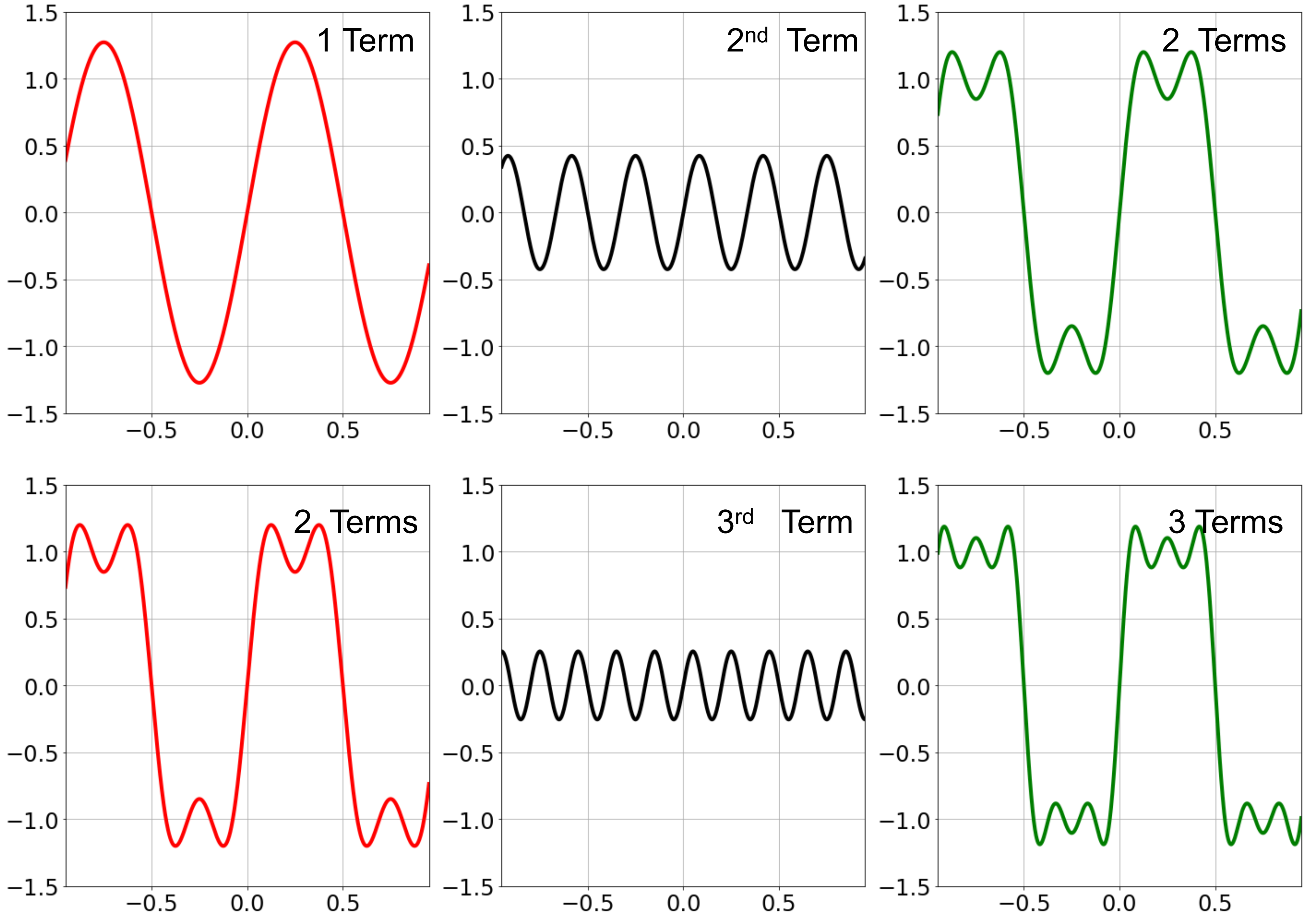


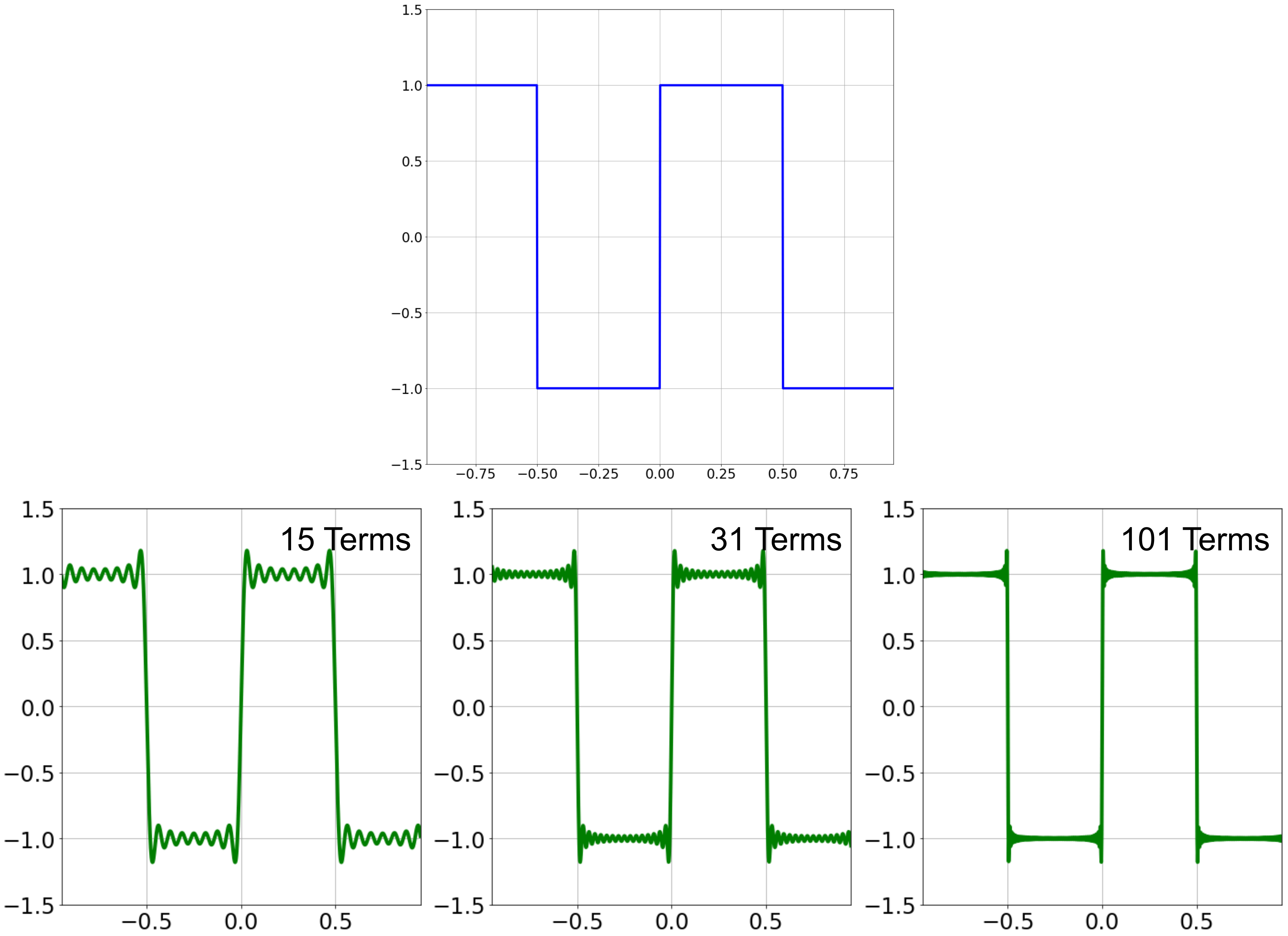
Square Wave:

$$f(x) = \begin{cases} 1, & \text{if frac}(x) < 0.5 \\ -1, & \text{otherwise} \end{cases}$$

Fourier Transform:

$$\frac{4}{\pi}\left(\frac{\sin(2\pi.1.x)}{1} + \frac{\sin(2\pi.3.x)}{3} + \frac{\sin(2\pi.5.x)}{5} + \ \dots\right)$$


1 Term
2nd Term
2 Terms
2 Terms
3rd Term
3 Terms
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−0.5
0.0
0.5


1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
15 Terms
31 Terms
101 Terms
−0.5
0.0
0.5

Fourier Transform

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

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

Computing the Fourier Transform

- $H(\omega) = \mathcal{F}[h(x)]$

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

- Discrete:
  - $H(k) = \frac{1}{N} \sum_{x=0}^{N-1} h(x) e^{-j2\pi kx/N}$

- Euler’s Formula:
  - $e^{jnx} = \cos(nx) + j \sin(nx)$

Properties of Fourier Transforms

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

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

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

See Szeliski Book (3.4)

The Convolution Theorem

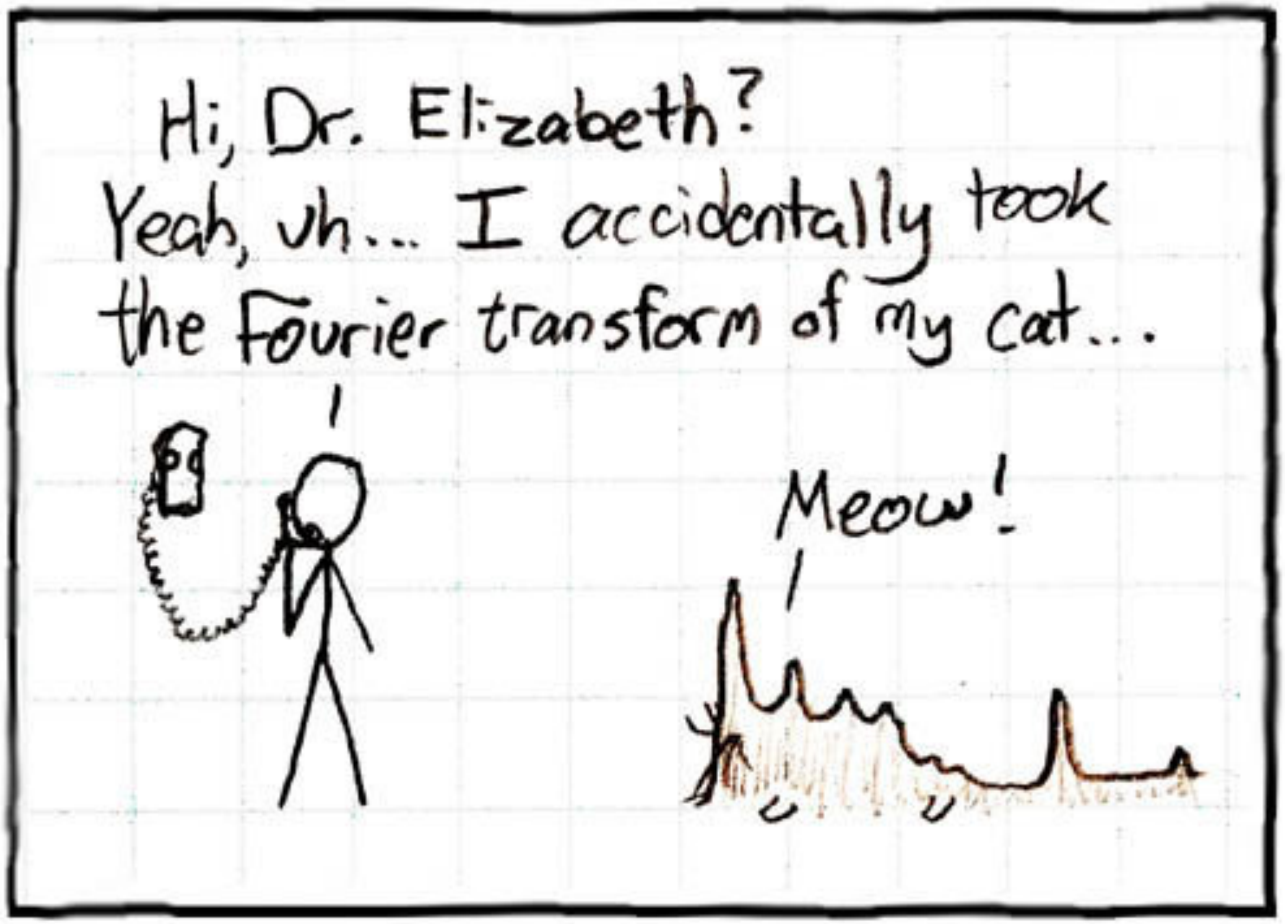
- The Fourier transform of the convolution of two functions is the product of their Fourier transforms
  - $\mathcal{F}[g * h] = \mathcal{F}[g]\mathcal{F}[h]$

- The inverse Fourier transform of the product of two Fourier transforms is the convolution of the two inverse Fourier transforms
  - $\mathcal{F}^{-1}[gh] = \mathcal{F}^{-1}[g] * \mathcal{F}^{-1}[h]$

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

Other signals

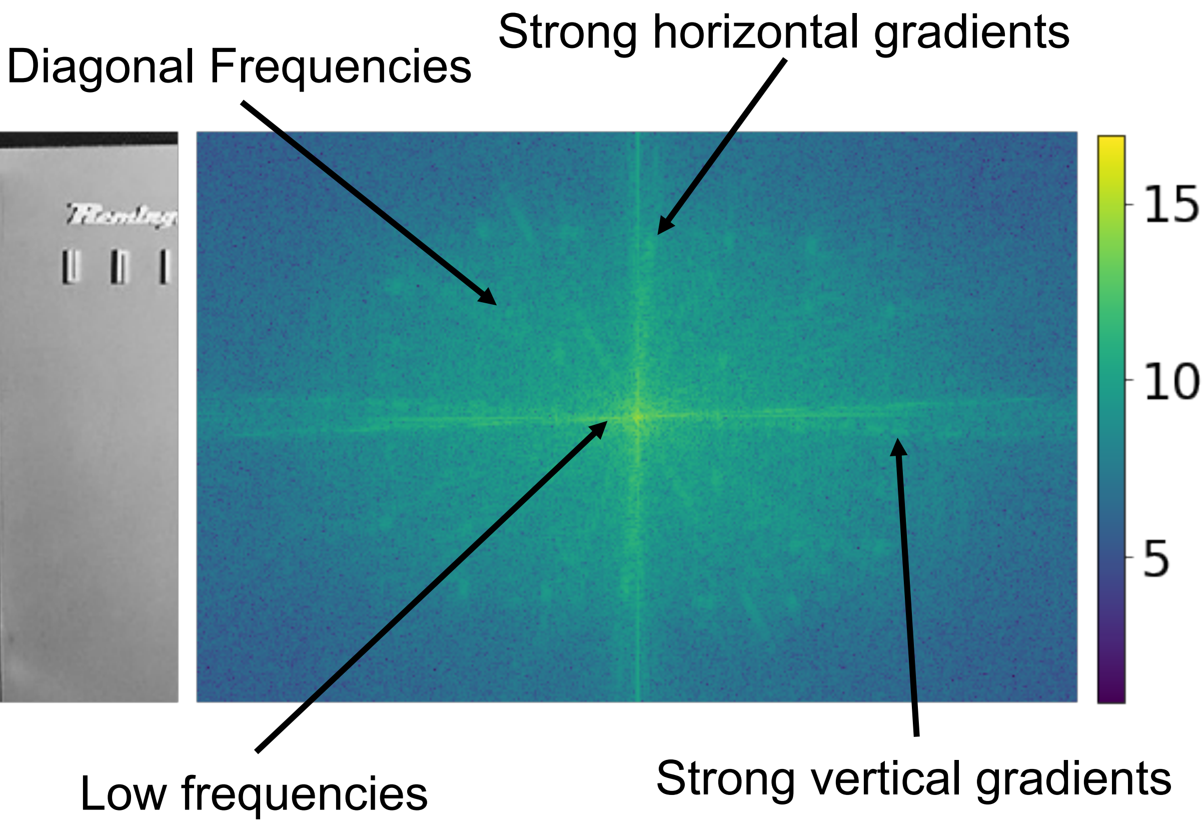
- We can also think of all kinds of other signals the same way


Hi, Dr. Elizabeth? Yeah, uh... I accidentally took the Fourier transform of my cat...
Meow!

xkcd.com

Images

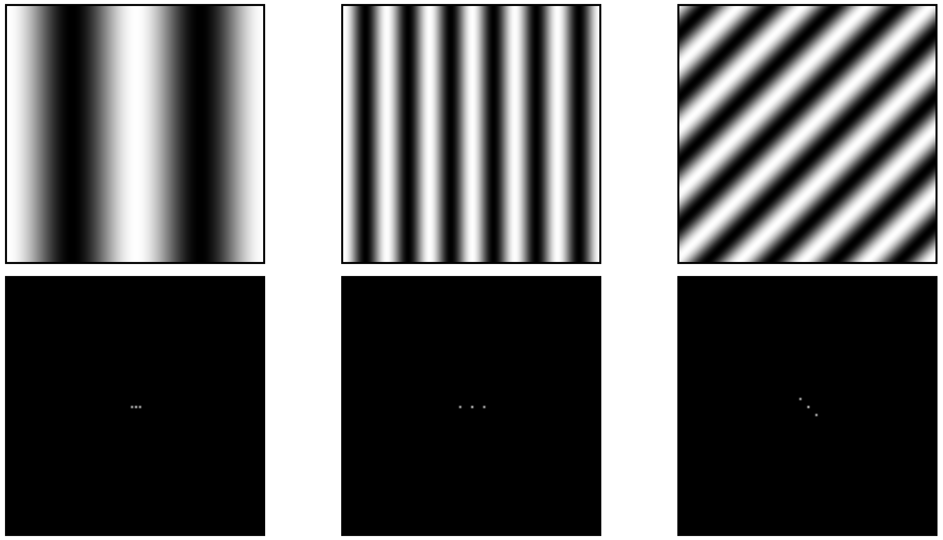
$$H(k,l) = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} h(m,n)e^{-j2\pi\left(\frac{k}{M}m+\frac{l}{N}n\right)}$$



Fourier analysis in images

Intensity Image

Fourier Image

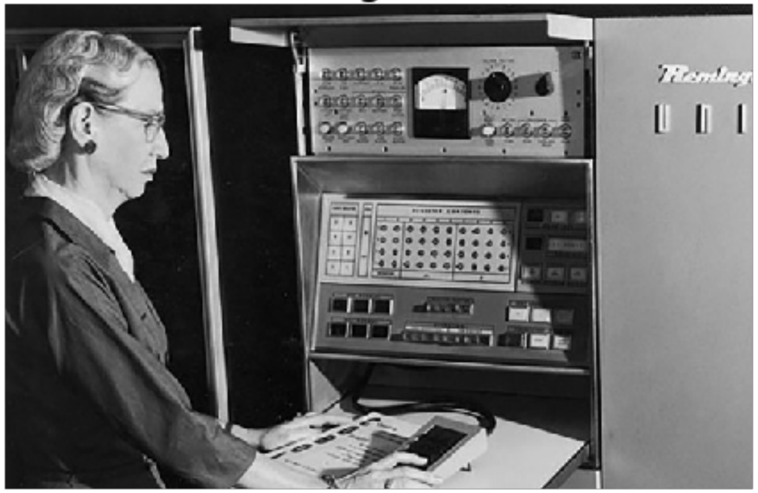


http://sharp.bu.edu/~slehar/fourier/fourier.html#filtering

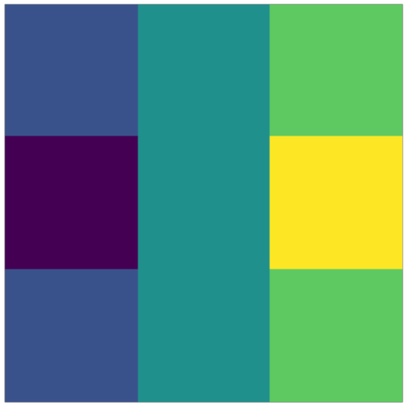
Filtering in spatial domain

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

Original

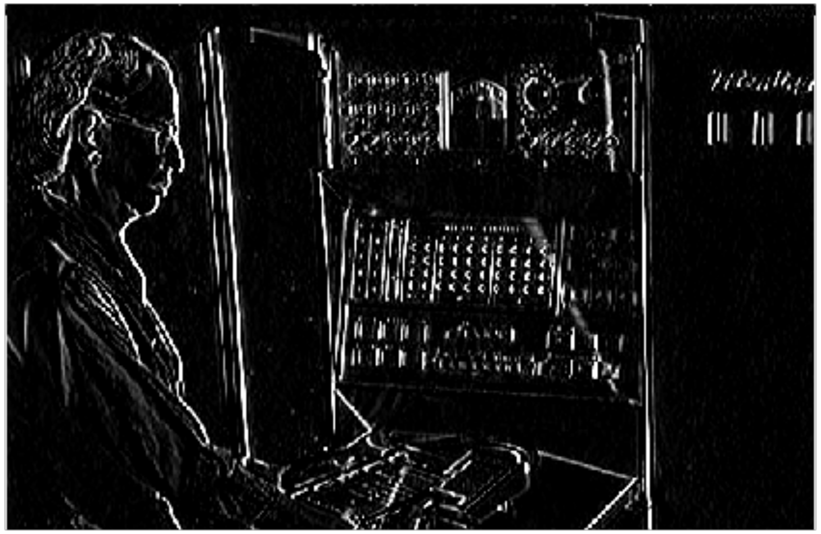


*

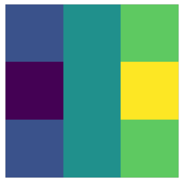


=

Filtered via Convolution

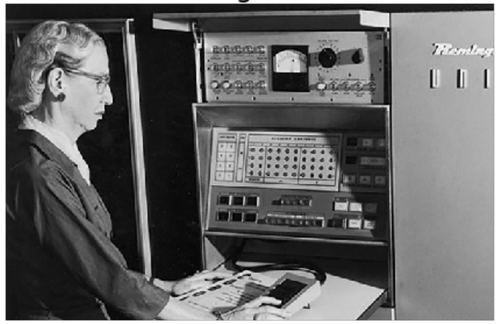


Filtering in frequency domain



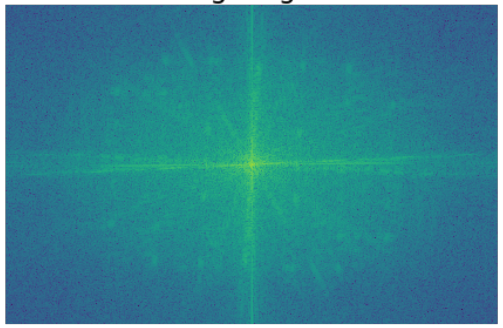
FFT

Original



FFT

FFT Log Magnitude



Filter FFT Magnitude

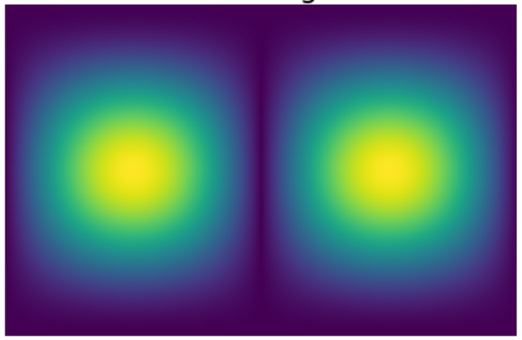
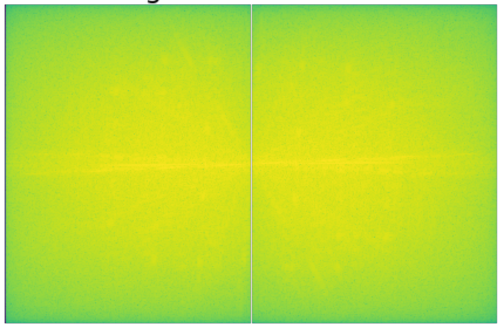
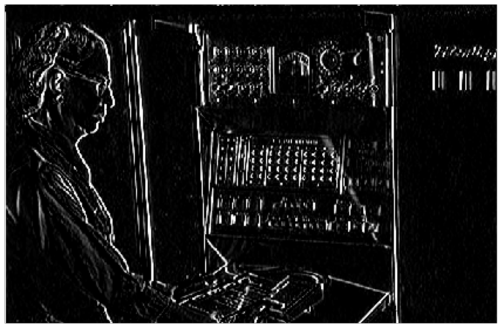


Image FFT .* Filter FFT



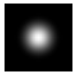
Inverse FFT

Filtered via FFT

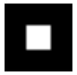


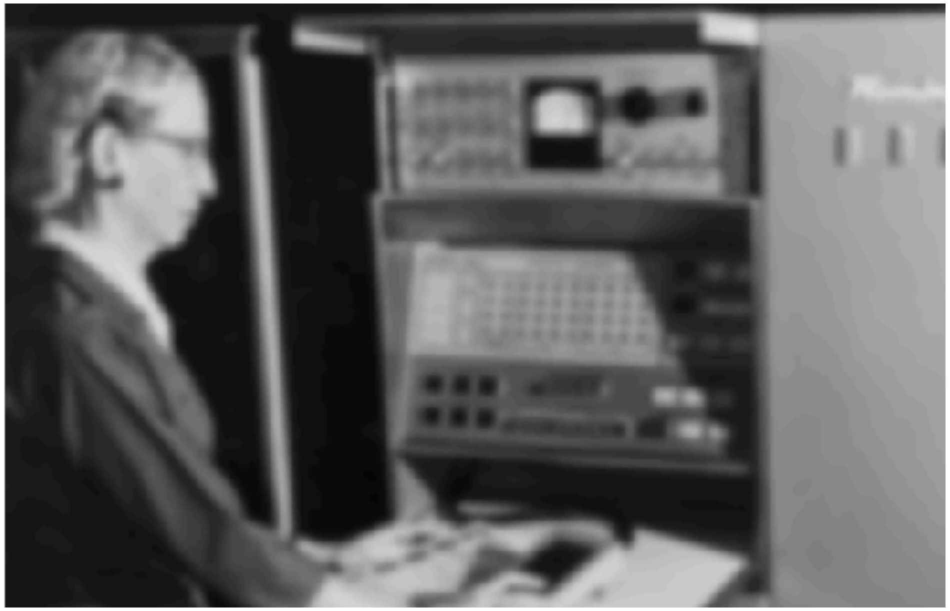
Why does the Gaussian give a nice smooth image, but the square filter give edgy artifacts?

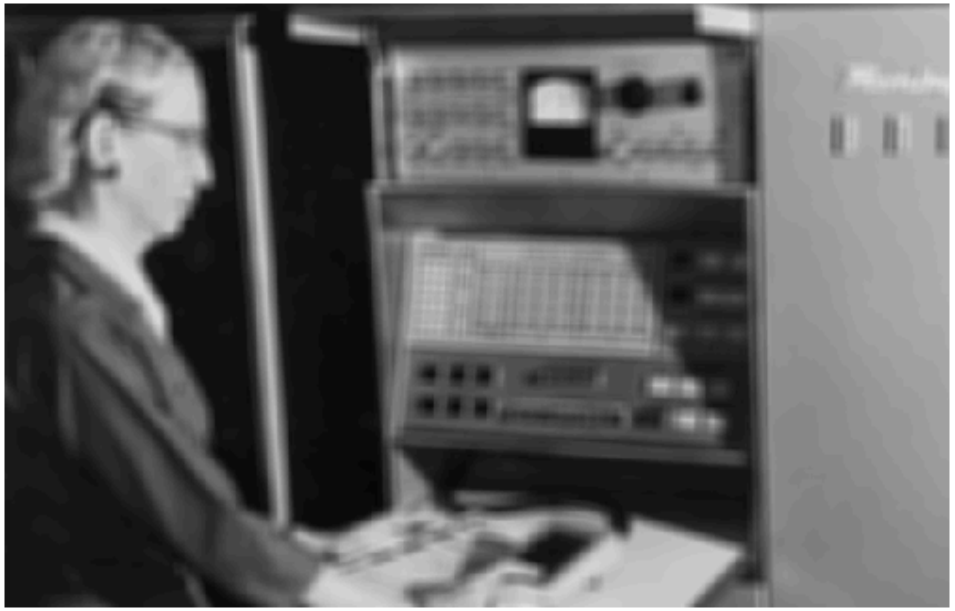
Gaussian



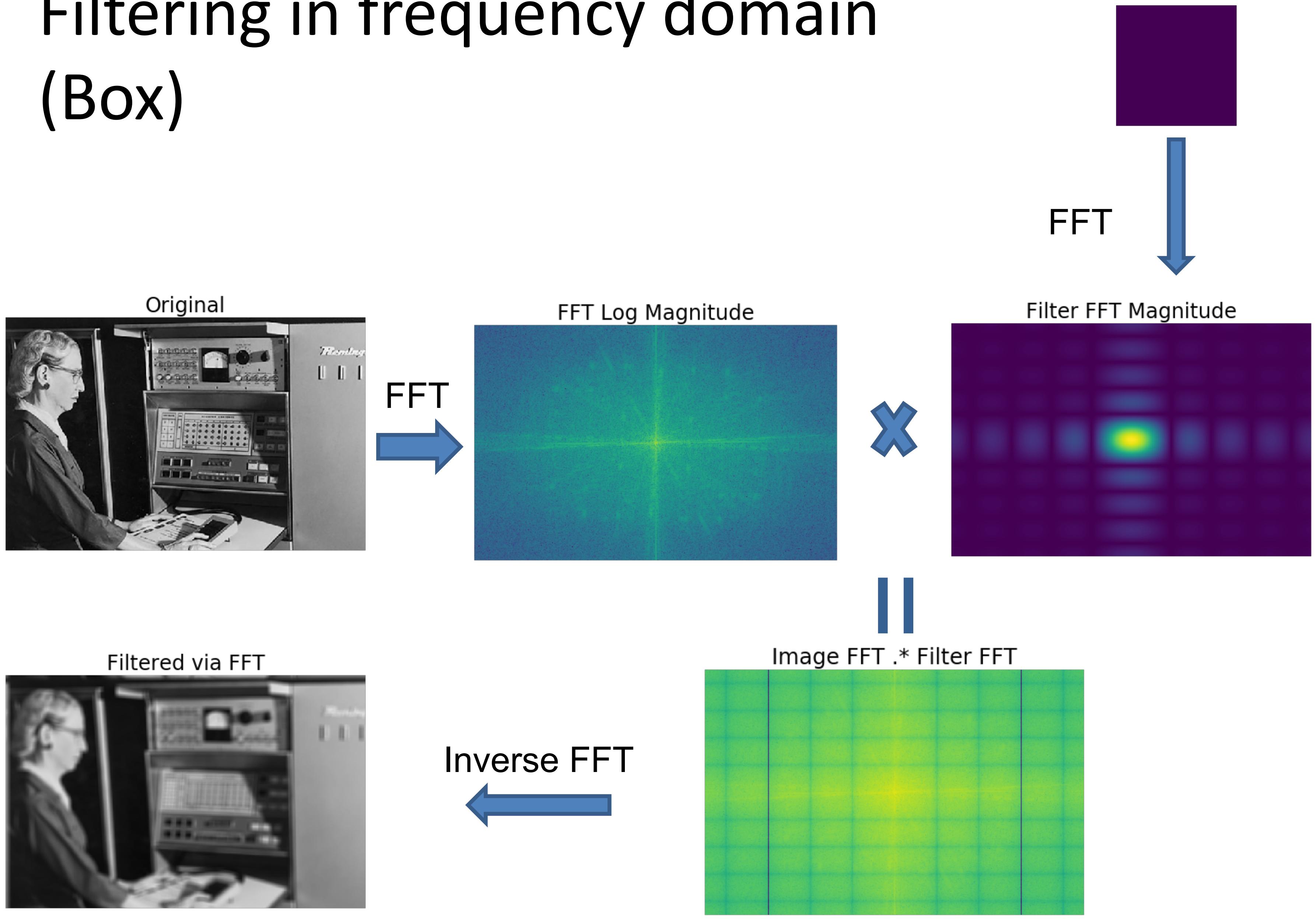
Box filter



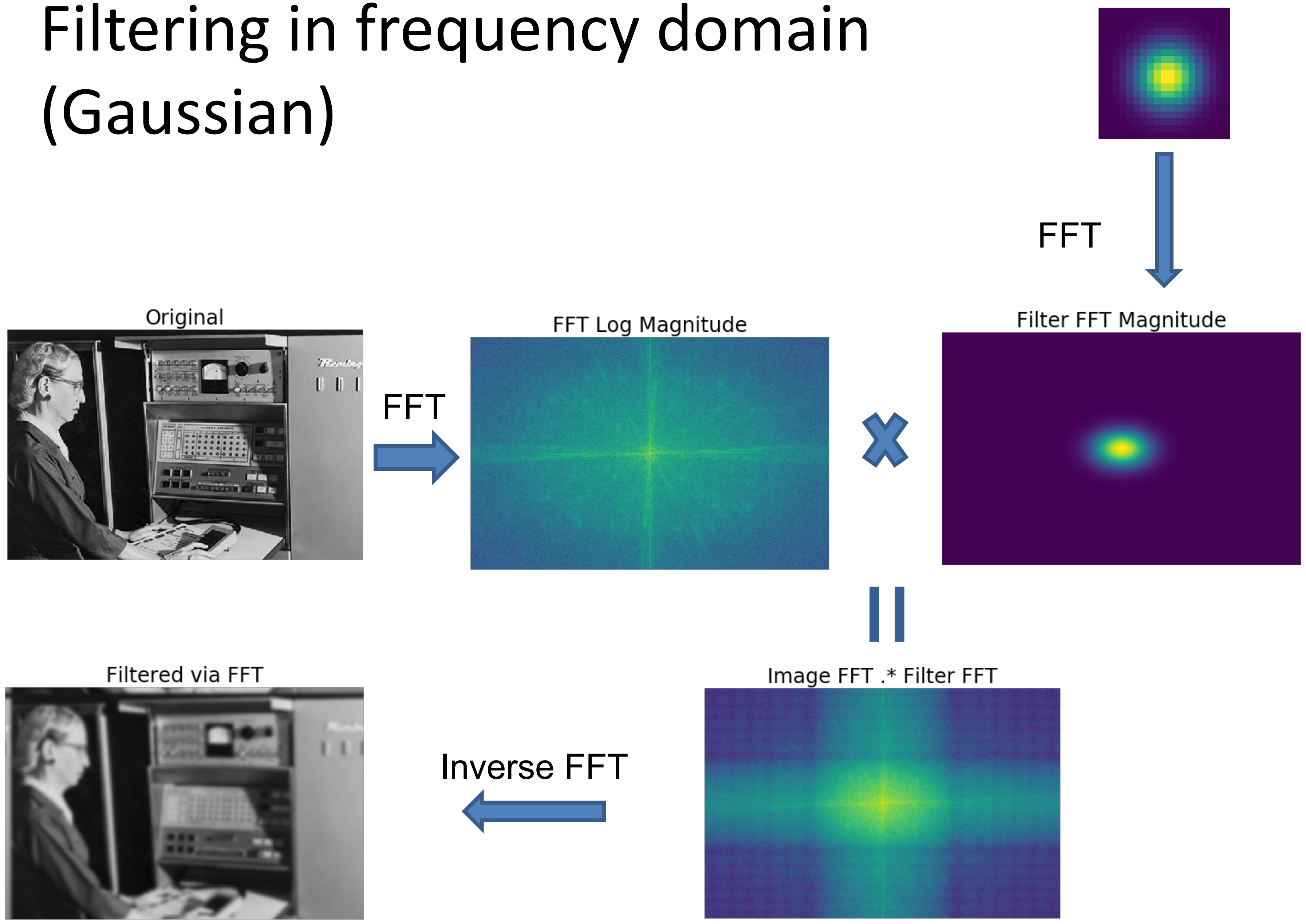




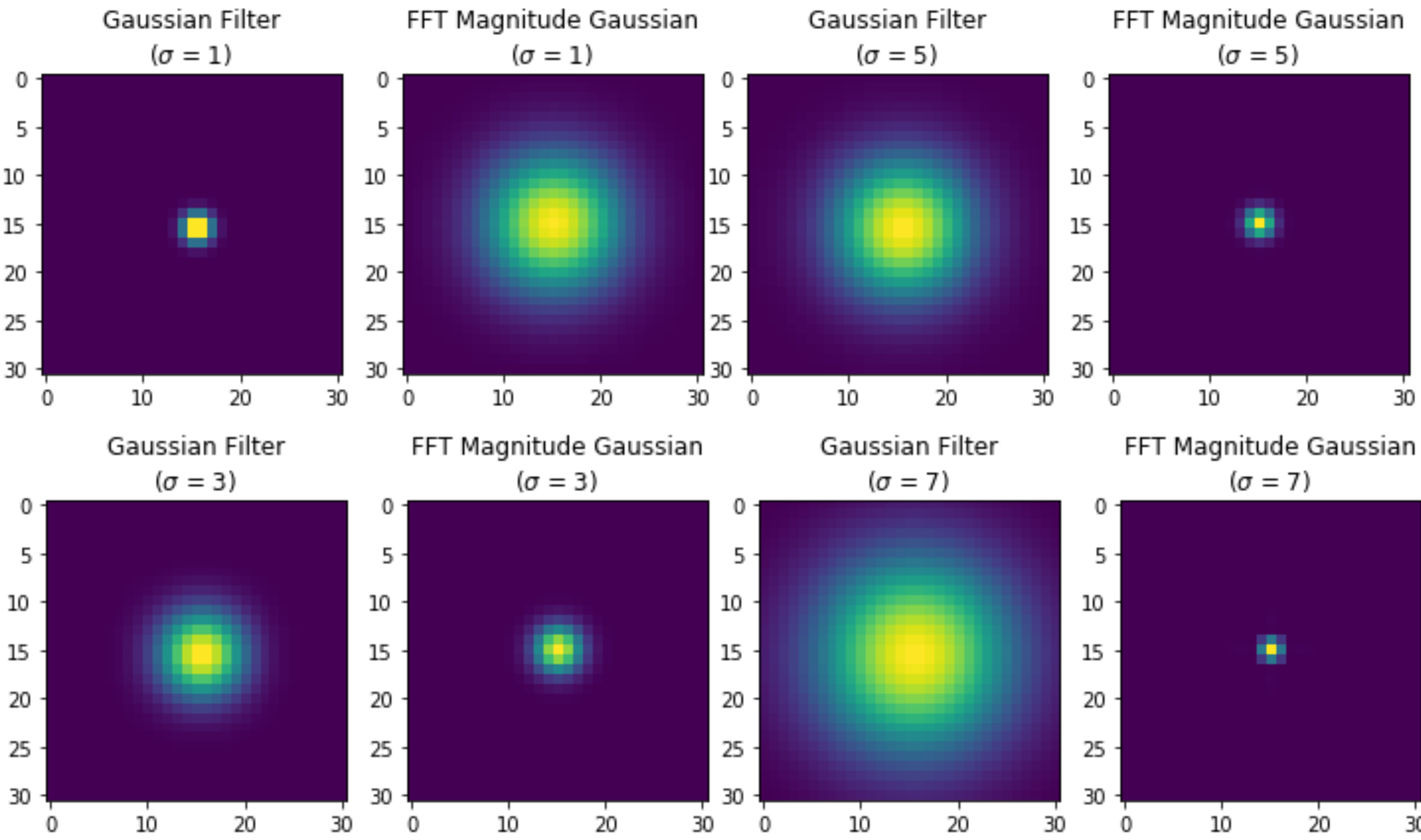
Filtering in frequency domain (Box)



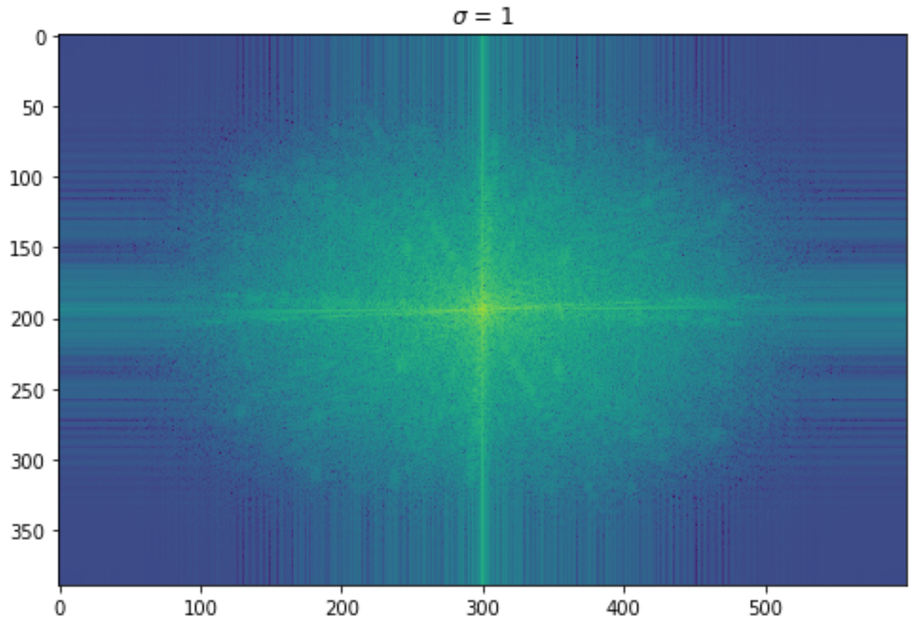
Filtering in frequency domain (Gaussian)

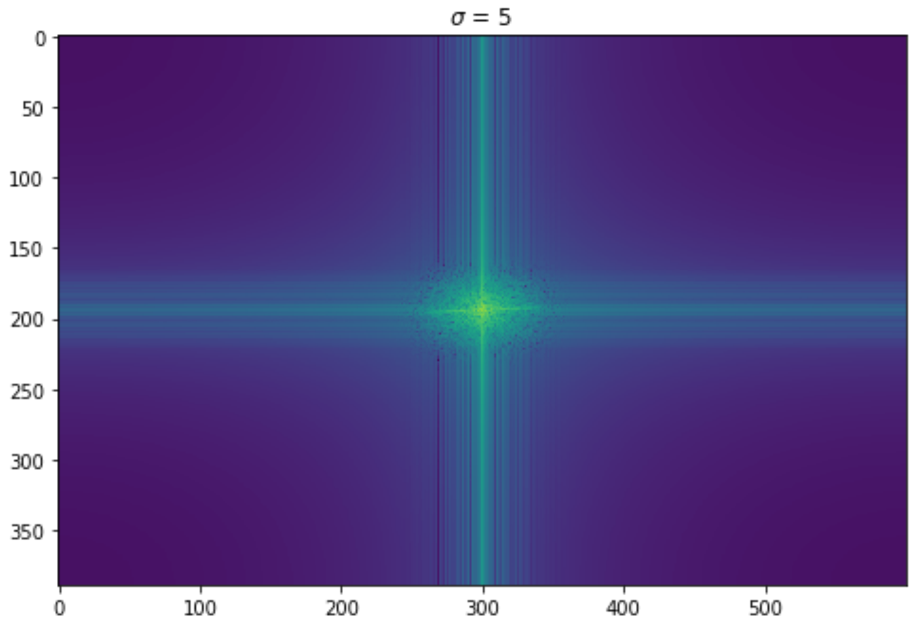


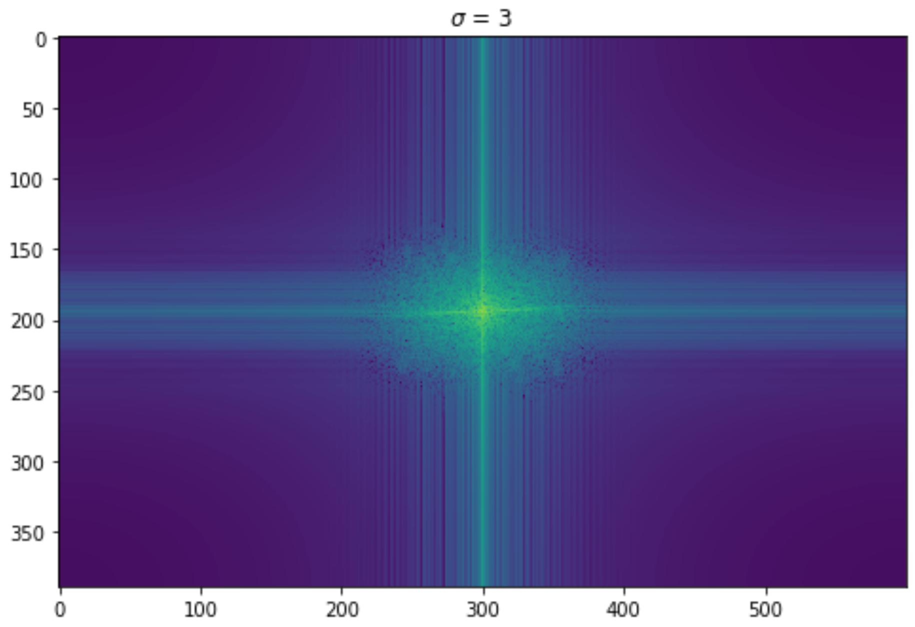
Filtering in frequency domain (Gaussian)

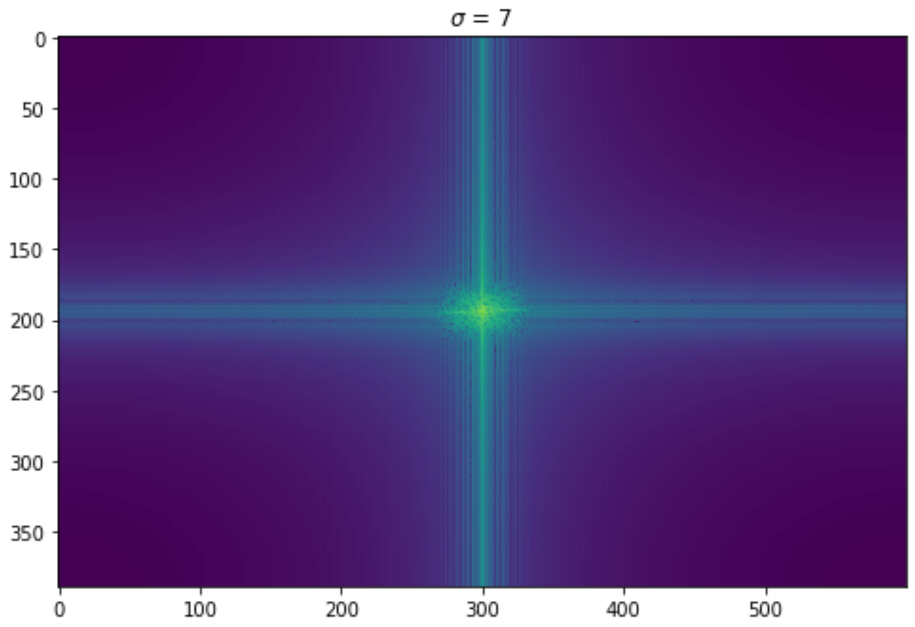


Filtering in frequency domain (Gaussian)

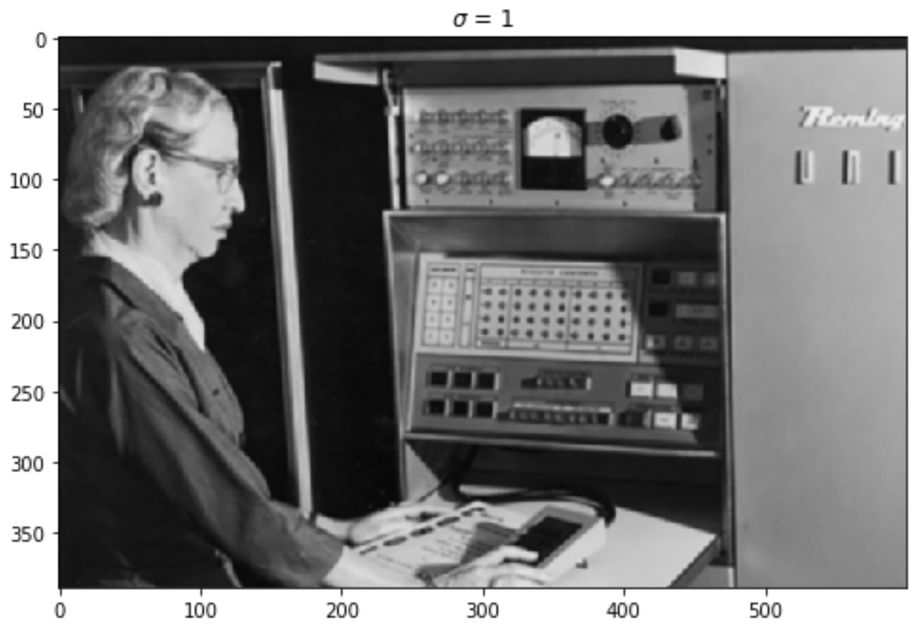


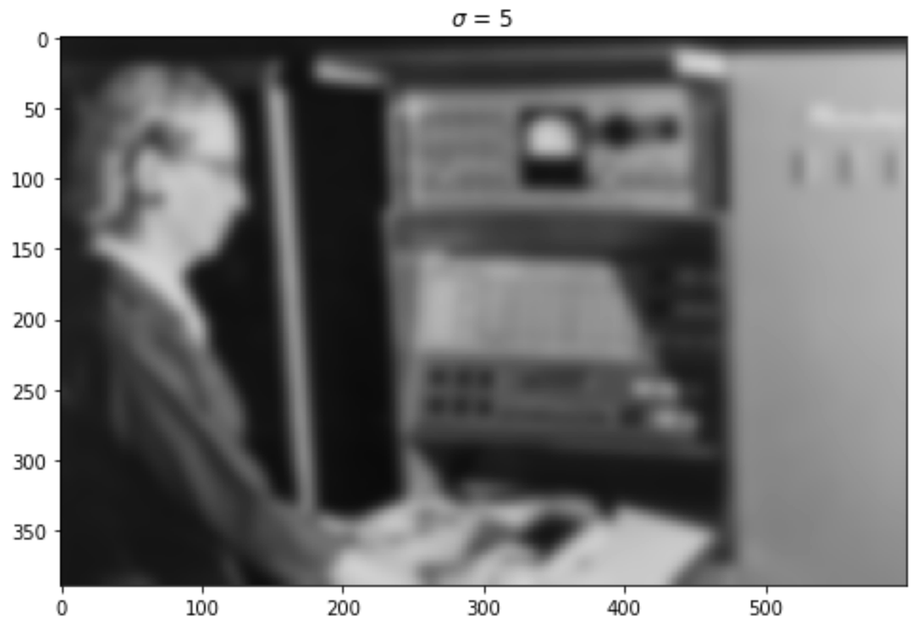


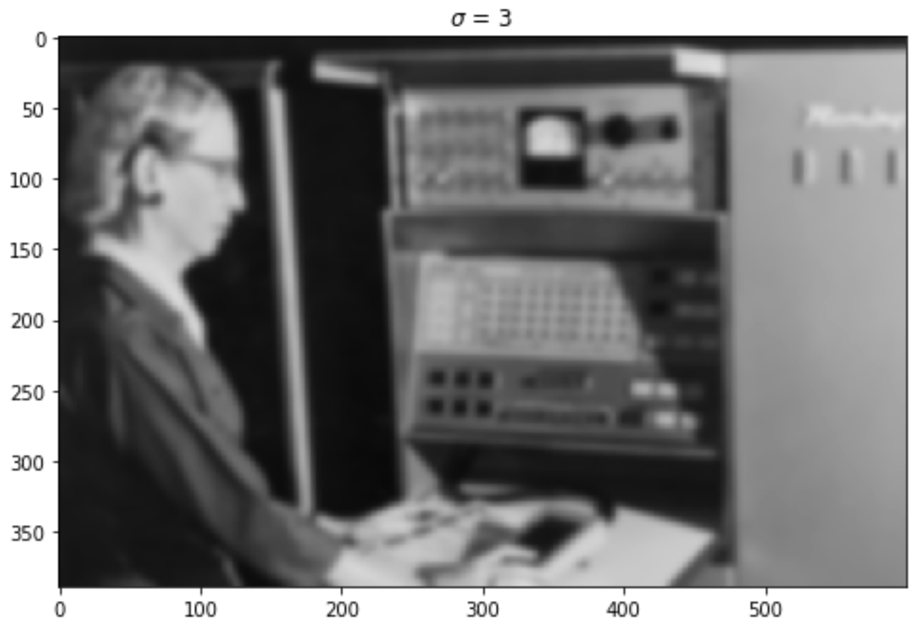


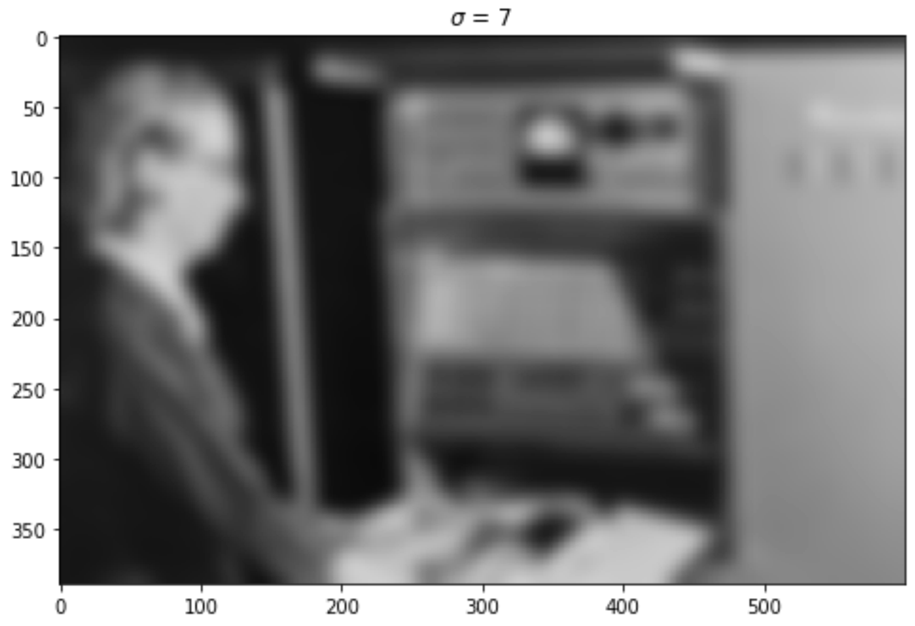


Filtering in frequency domain (Gaussian)









Sampling

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

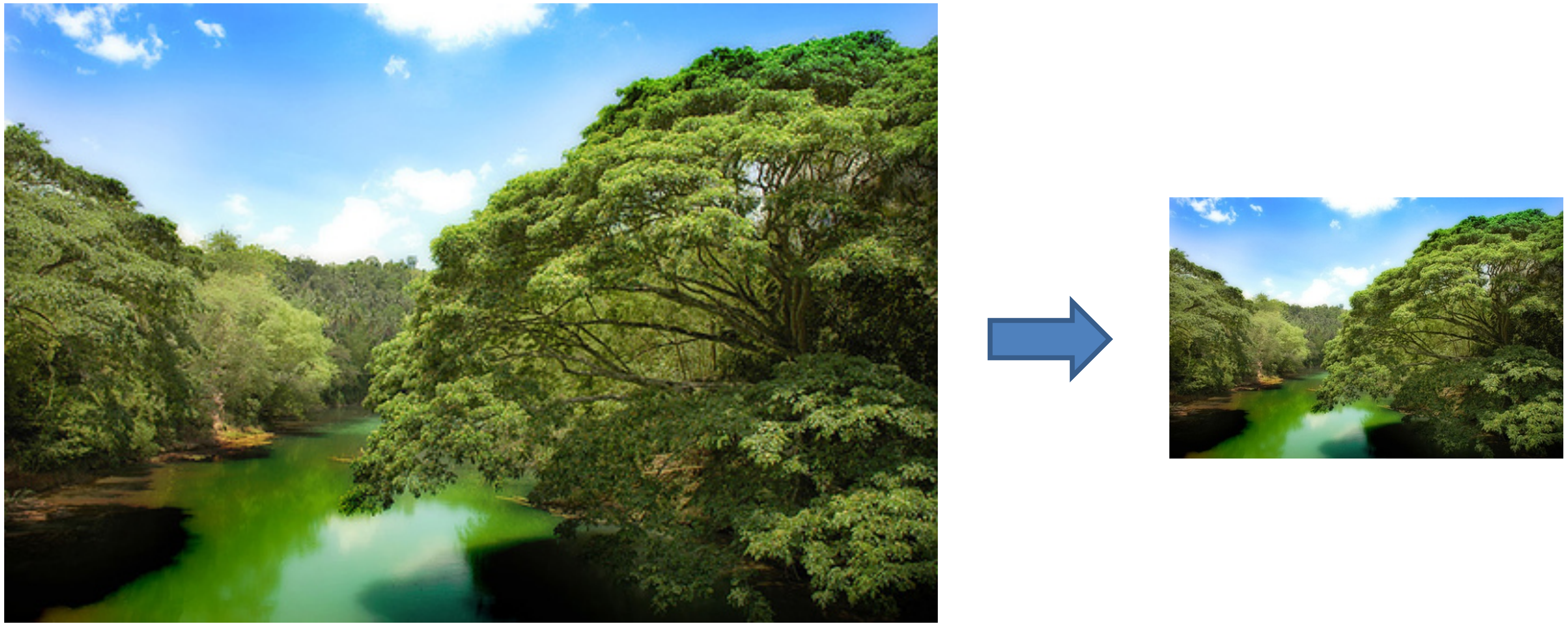
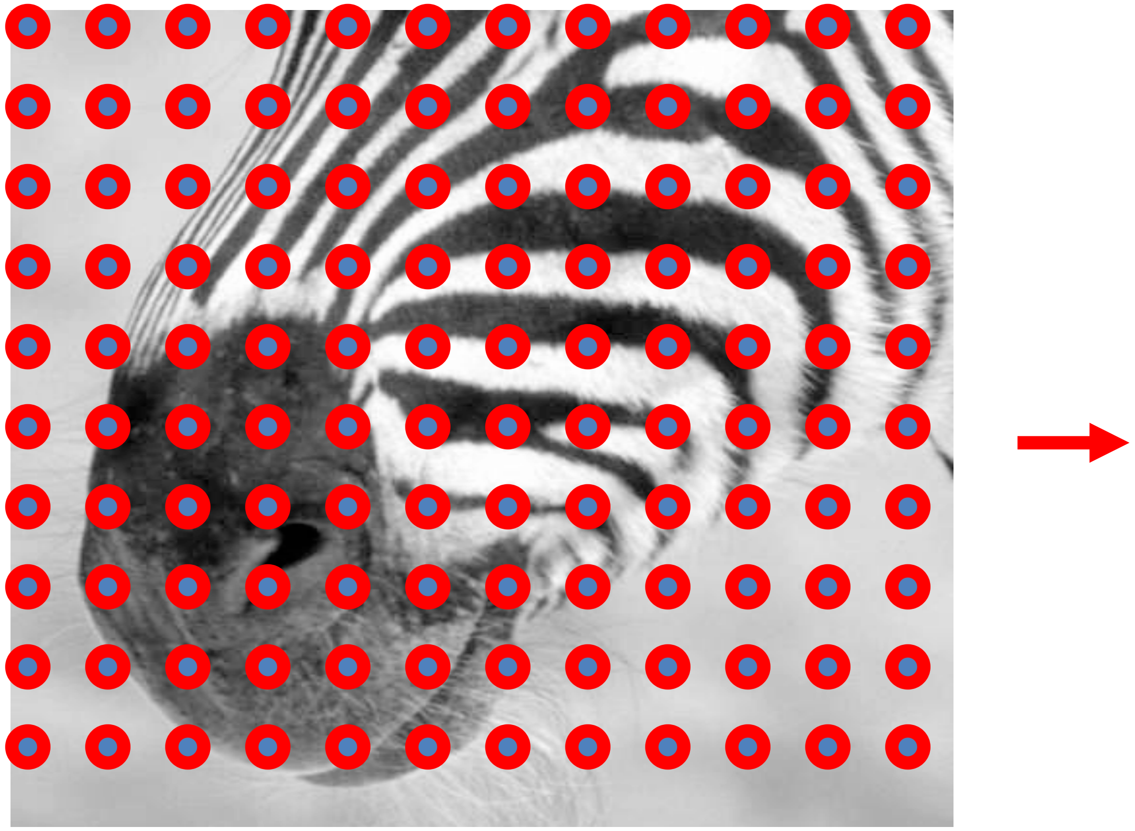


Image: http://www.flickr.com/photos/igorms/136916757/

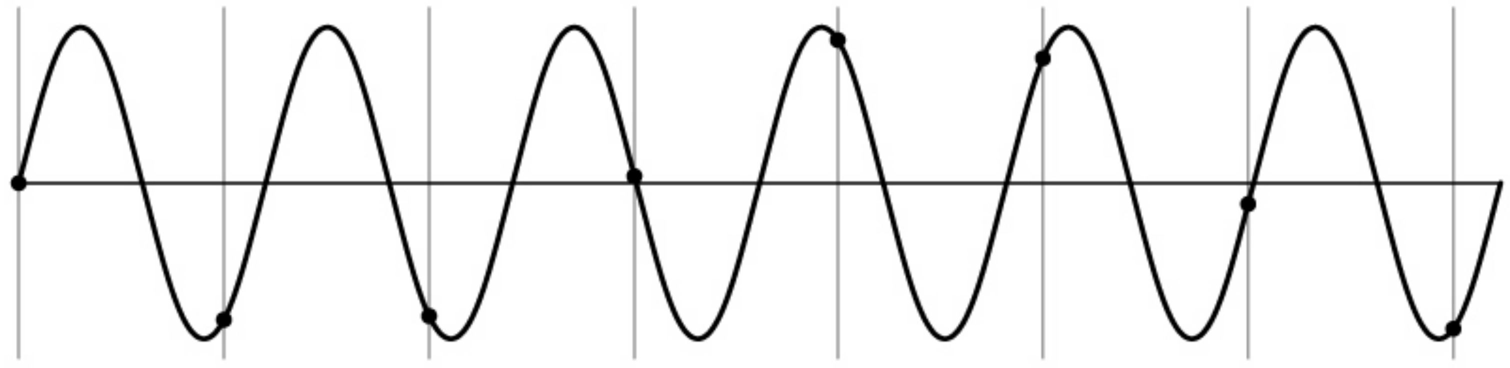
Subsampling by a factor of 2



Throw away every other row and column to create a 1/2 size image

Aliasing problem

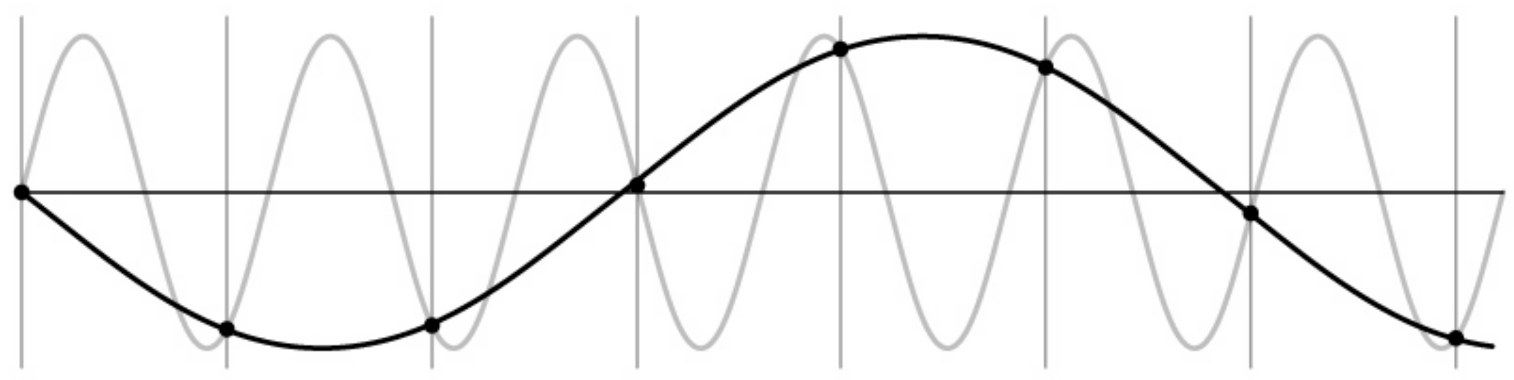
- 1D example (sinewave):



Source: S. Marschner

Aliasing problem

- 1D example (sinewave):



Source: S. Marschner

Aliasing problem

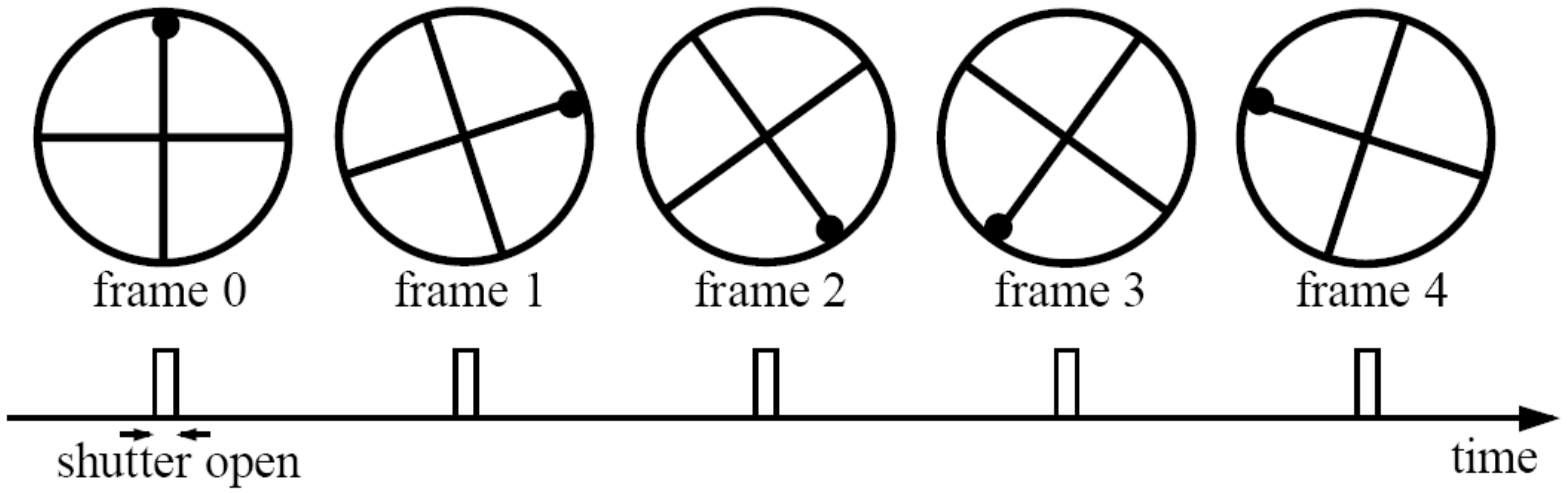
- Sub-sampling may be dangerous....
- Characteristic errors may appear:
  - "Wagon wheels rolling the wrong way in movies" See
  - "Checkerboards disintegrate in ray tracing"
  - "Striped shirts look funny on color television"

Source: D. Forsyth

Aliasing in video

Imagine a spoked wheel moving to the right (rotating clockwise). Mark wheel with dot so we can see what's happening.

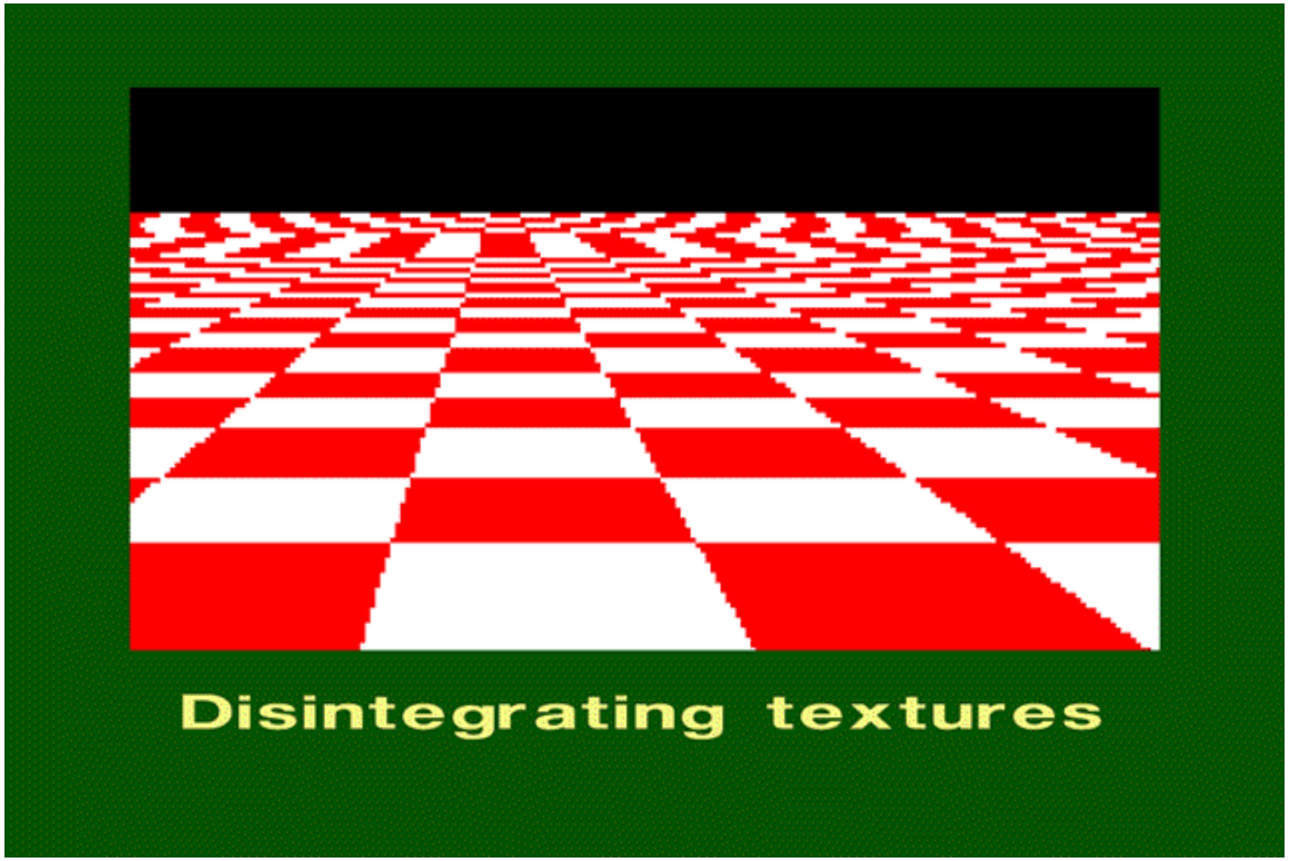
If camera shutter is only open for a fraction of a frame time (frame time = 1/30 sec. for video, 1/24 sec. for film):



Without dot, wheel appears to be rotating slowly backwards! (counterclockwise)

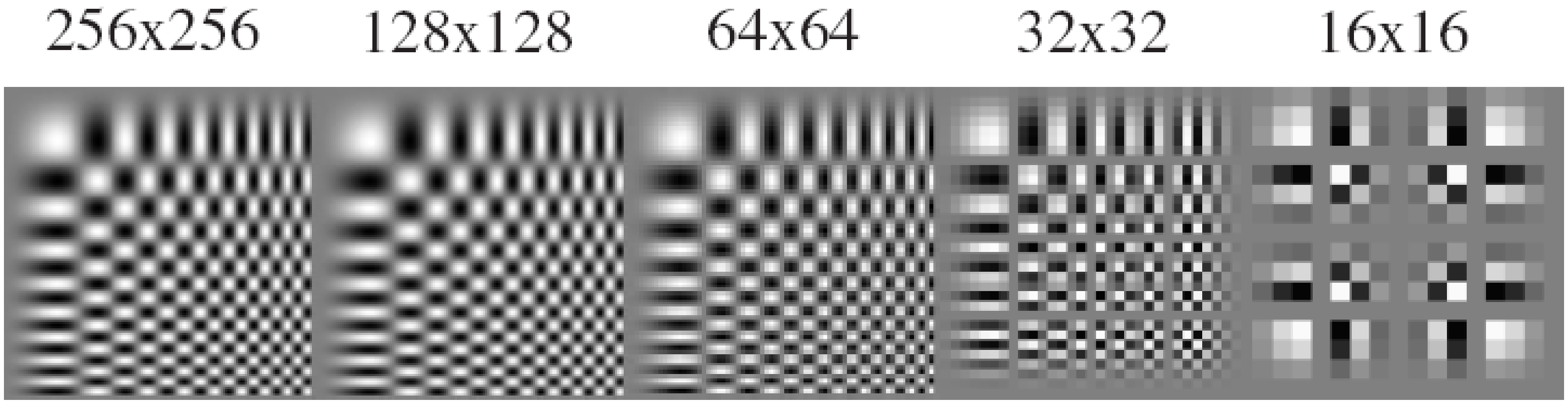
Slide by Steve Seitz

Aliasing in graphics



Source: A. Efros

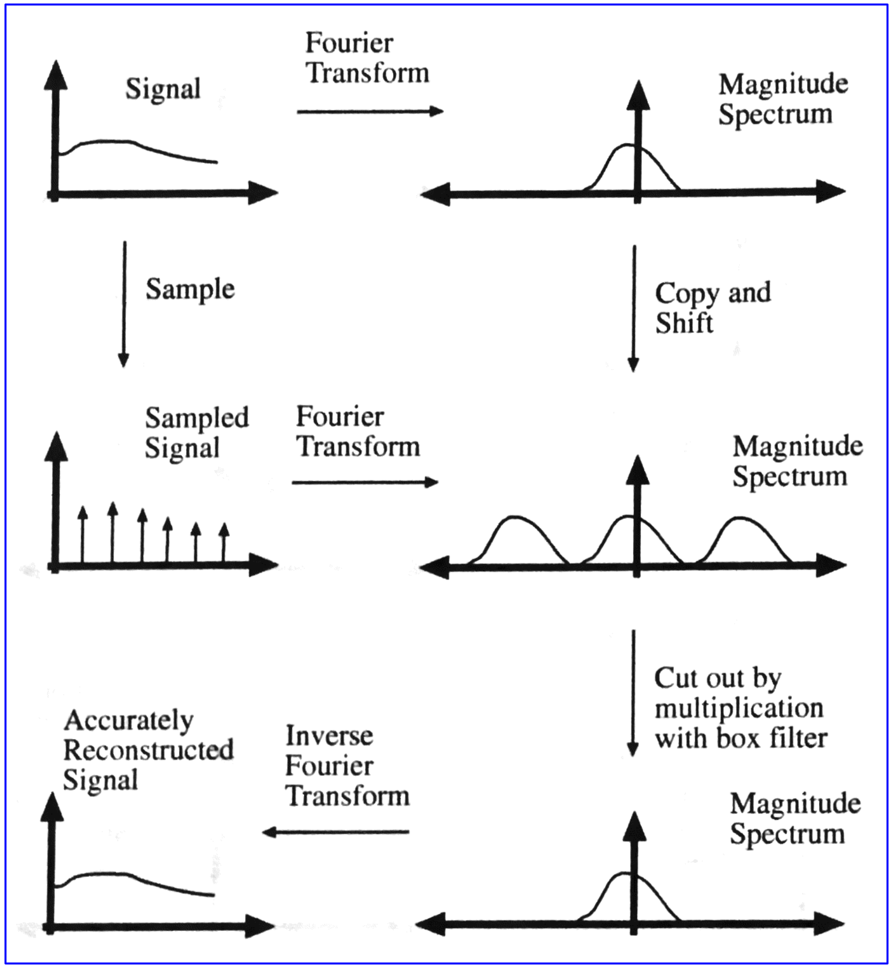
Sampling and aliasing



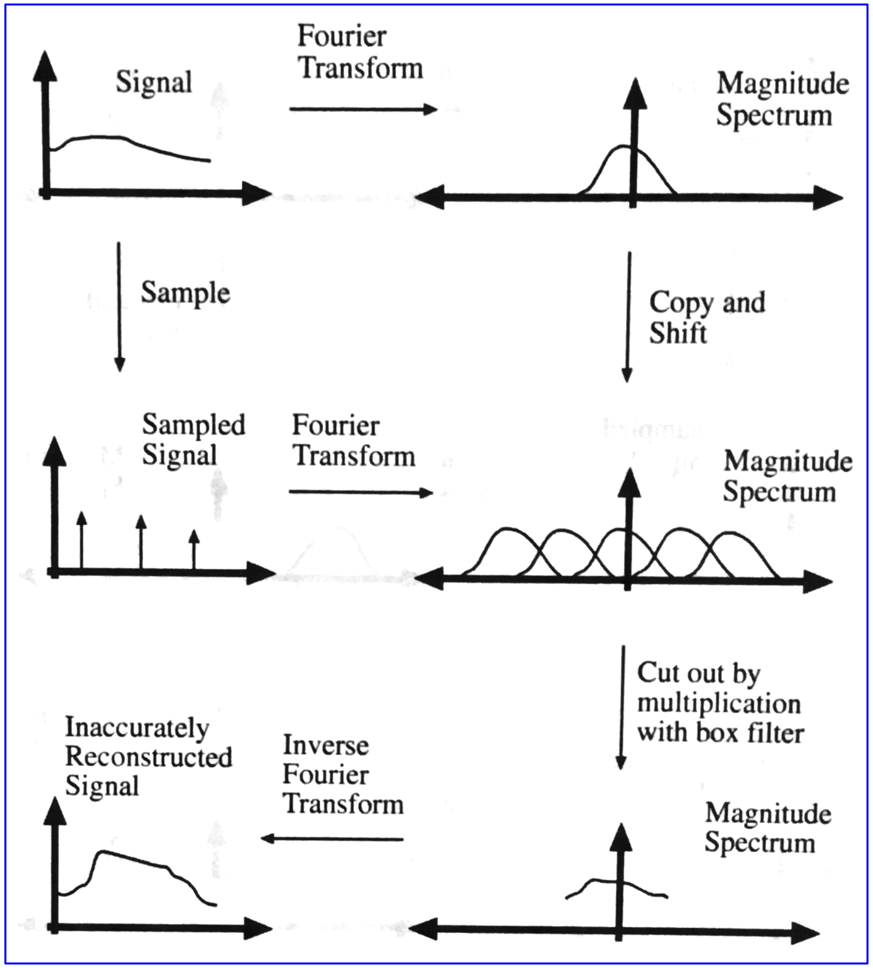
Slide from Derek Hoiem.

Aliasing in Frequency Domain

No Aliasing



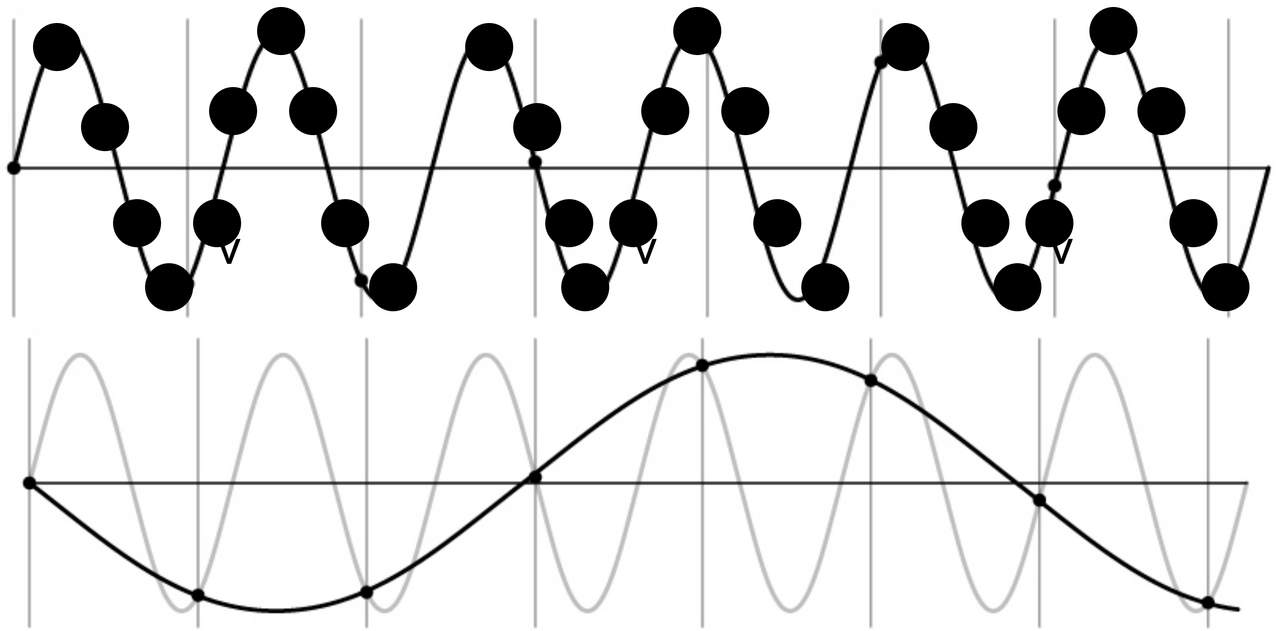
Aliasing



Source: Forsyth and Ponce

Nyquist-Shannon Sampling Theorem

- When sampling a signal at discrete intervals, the sampling frequency must be $\geq 2 \times f_{max}$
- f_{max} = max frequency of the input signal
- This will allows to reconstruct the original perfectly from the sampled version



Slide from Derek Hoiem.

Anti-aliasing

Solutions:

- Sample more often

- Get rid of all frequencies that are greater than half the new sampling frequency
  - Will lose information
  - But it’s better than aliasing
  - Apply a smoothing filter

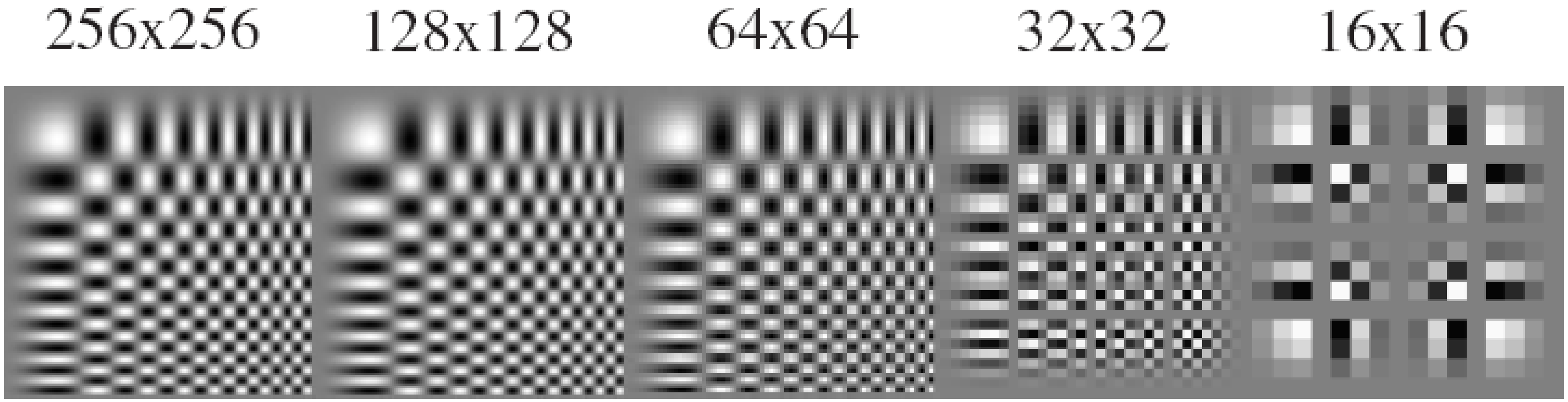
Slide from Derek Hoiem.

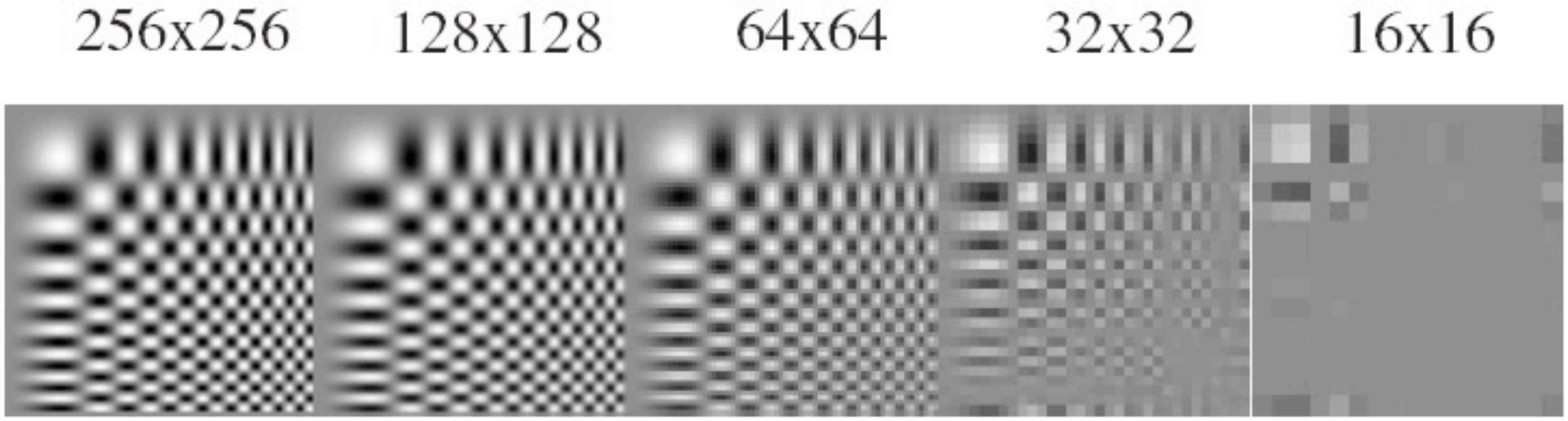
Algorithm for downsampling by factor of 2

1. Start with image(h, w)
2. Apply low-pass filter
3. Sample every other pixel

Slide from Derek Hoiem.

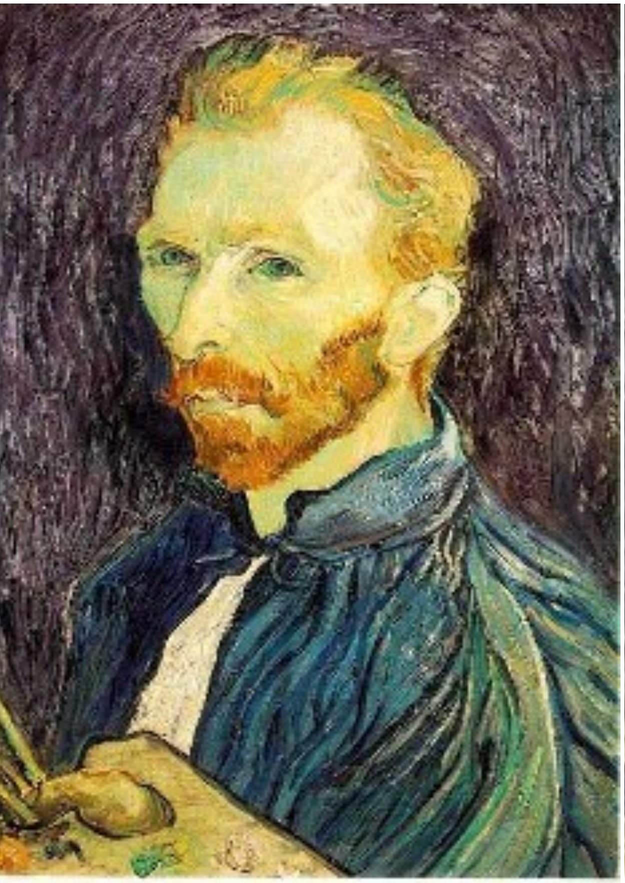
Anti-aliasing

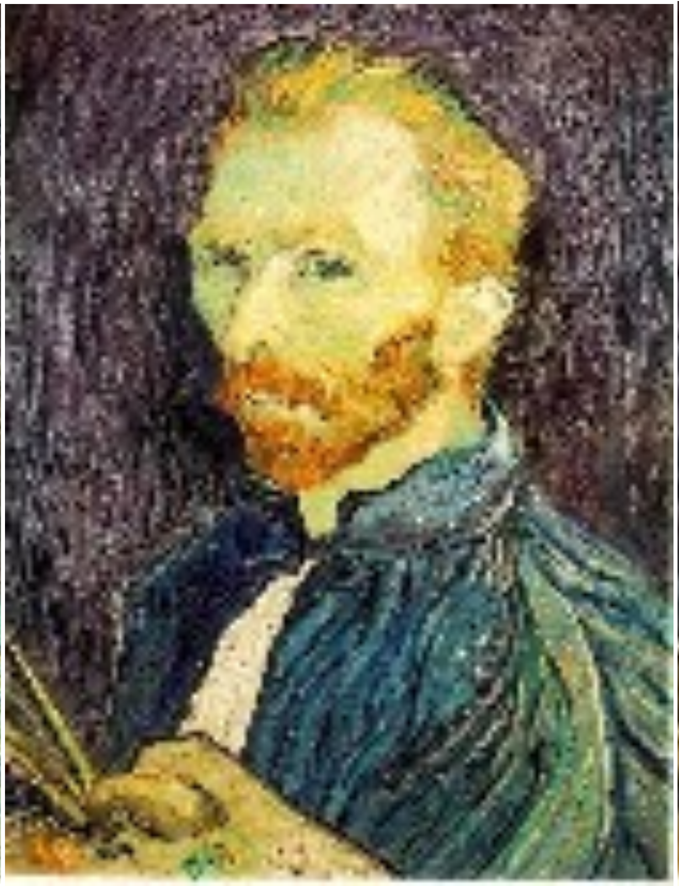


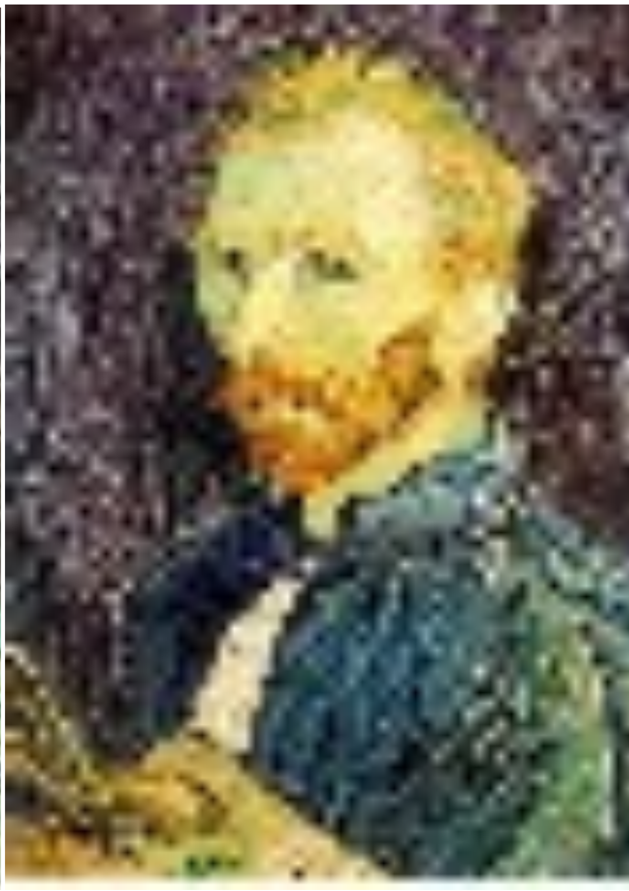


Forsyth and Ponce 2002

Subsampling without pre-filtering

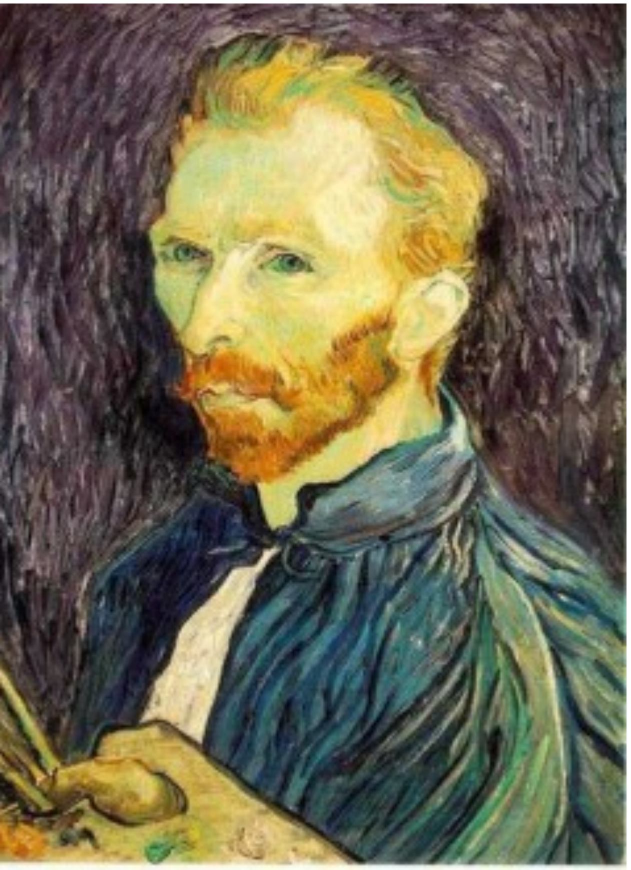

1/2

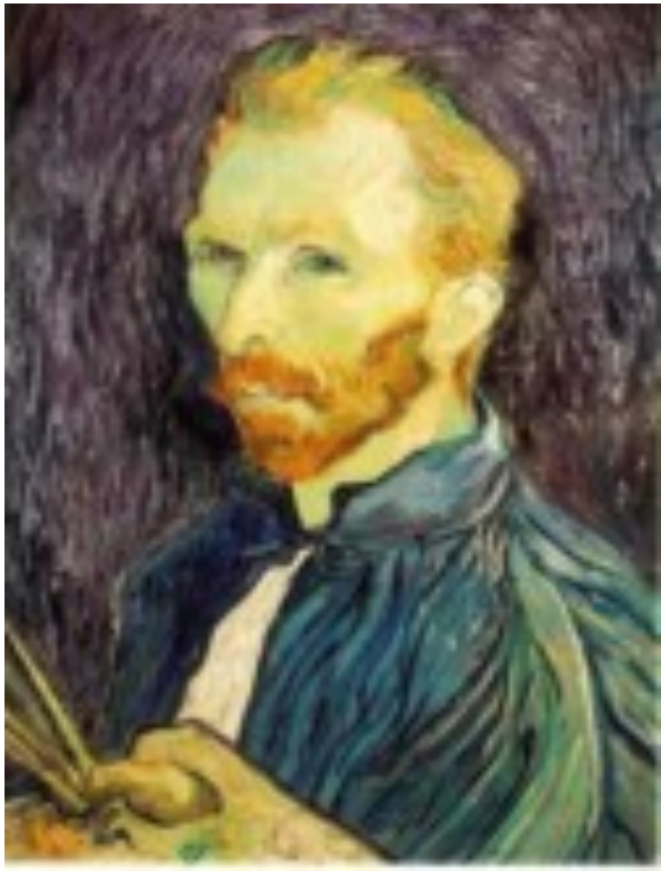

1/4 (2x zoom)

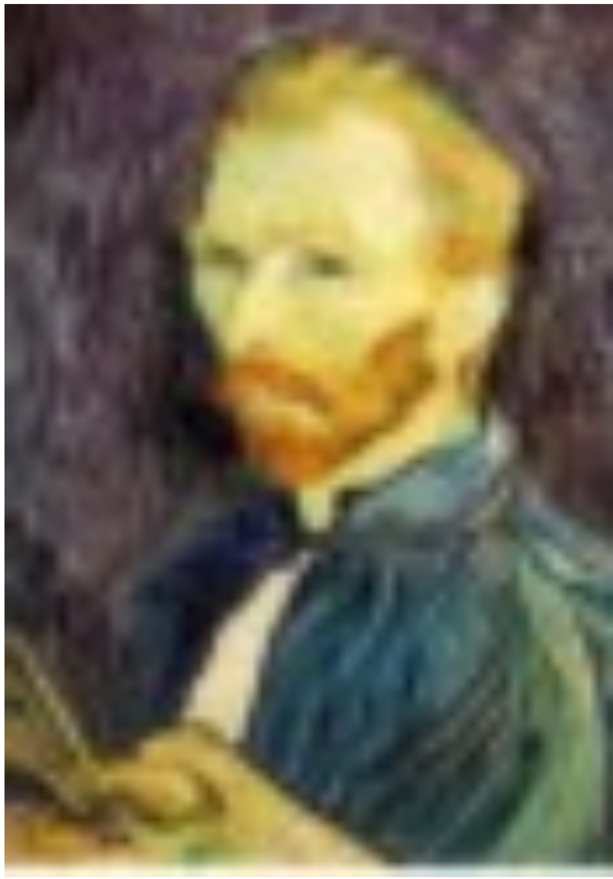

1/8 (4x zoom)

Slide by Steve Seitz

Subsampling with Gaussian pre-filtering

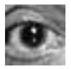

Gaussian 1/2

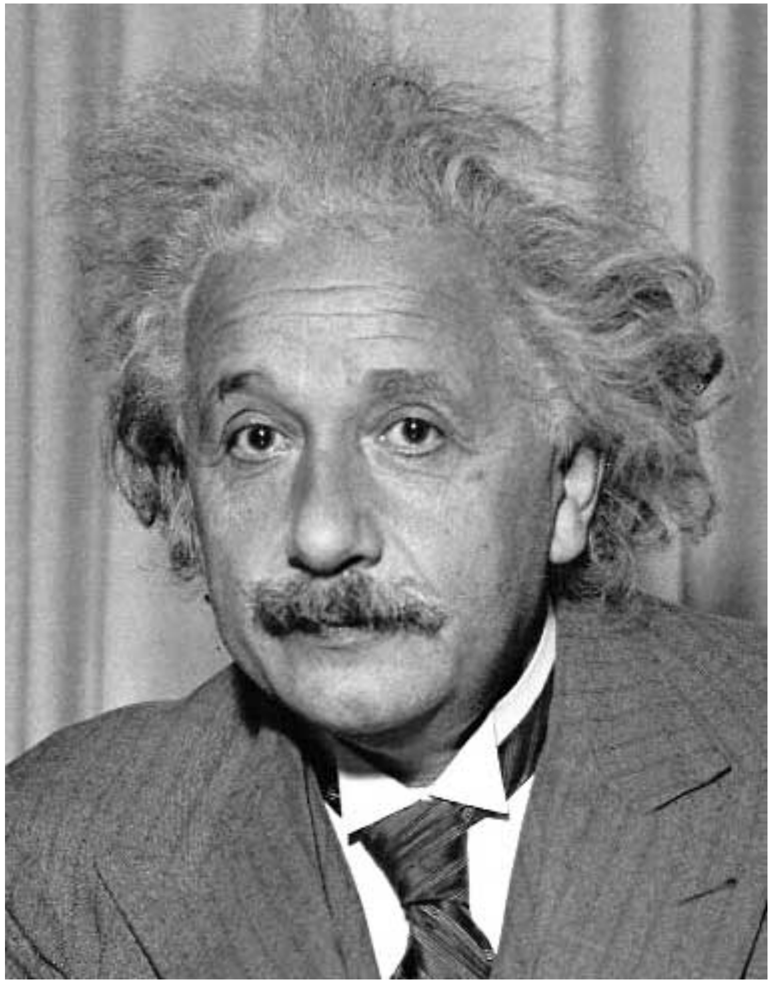

G 1/4


G 1/8

Slide by Steve Seitz

Using Filtering for Template matching

- Goal: find  in image



- Main challenge: What is a good similarity or distance measure between two patches?
  - Correlation
  - Zero-mean correlation
  - Sum Square Difference
  - Normalized Cross Correlation

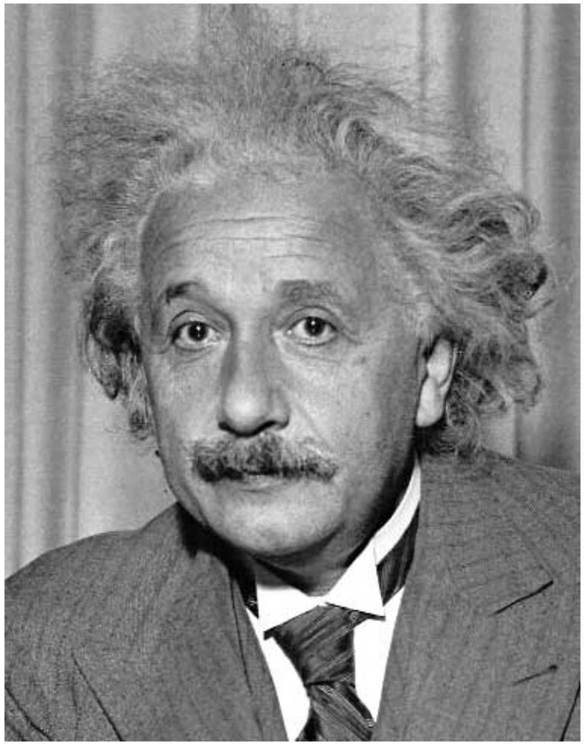
Slide from Derek Hoiem.

Matching with filters

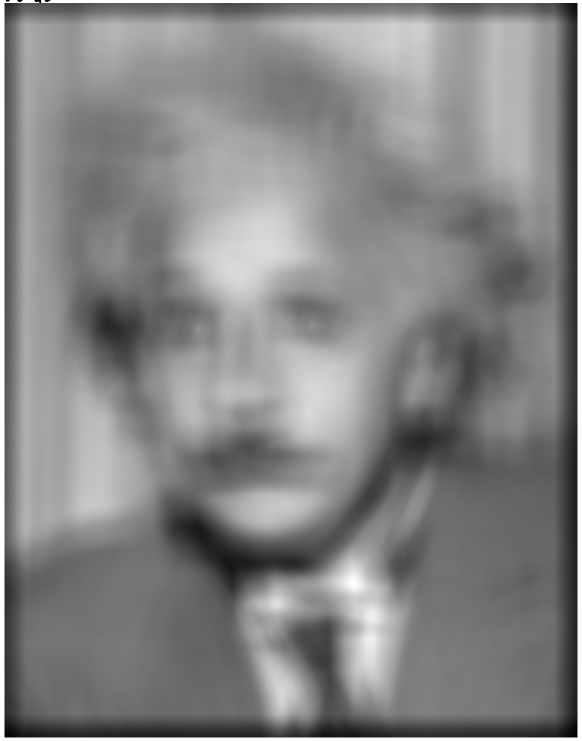
- Goal: find in image
- Method 0: filter the image with eye patch

$$h[m,n]=\sum_{k,l} g[k,l]\, f[m+k,n+l]$$

f = image
g = filter



Input



Filtered Image

What went wrong?

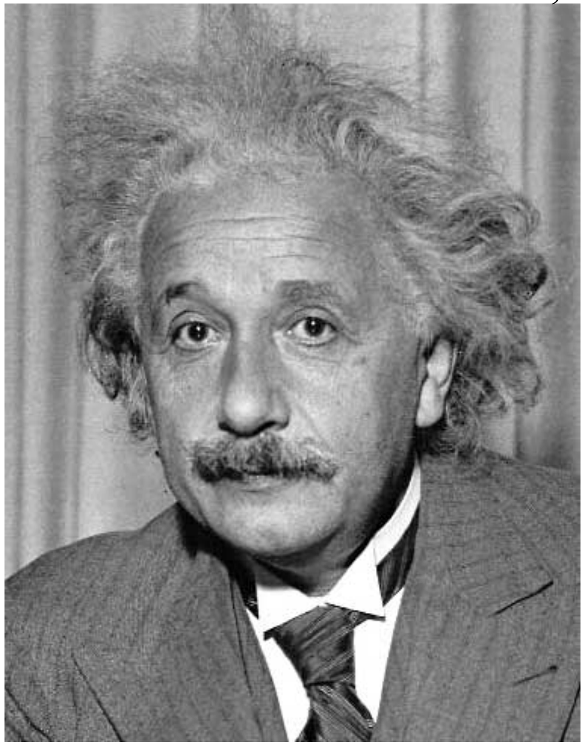
Slide from Derek Hoiem.

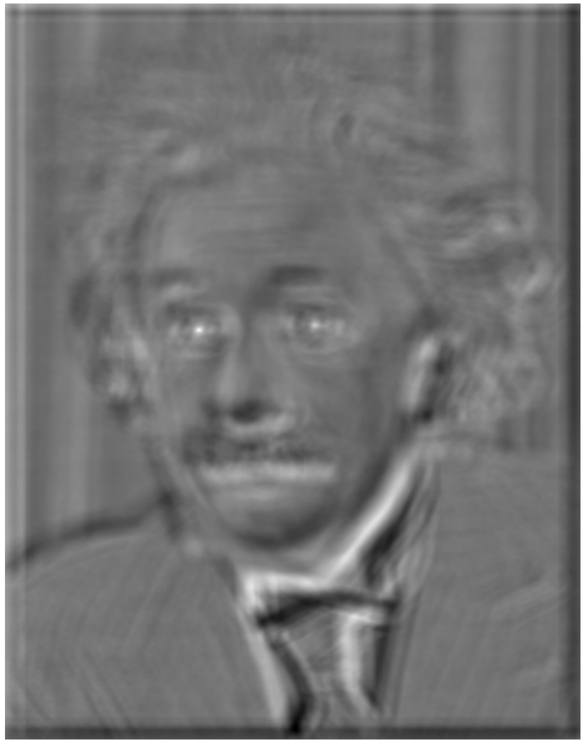
Matching with filters

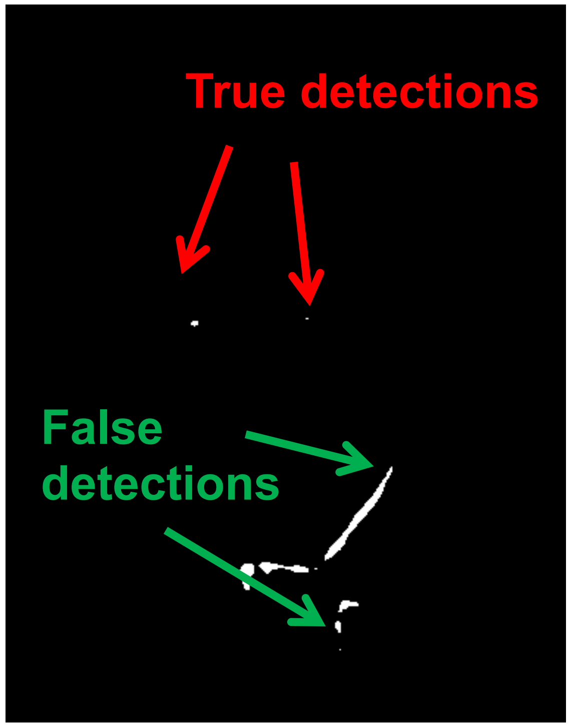
- Goal: find in image
- Method 1: filter the image with zero-mean eye

$$h[m,n]=\sum_{k,l}(g[k,l]-\bar{g})\,(f[m+k,n+l])$$

mean of template g


Input


Filtered Image (scaled)



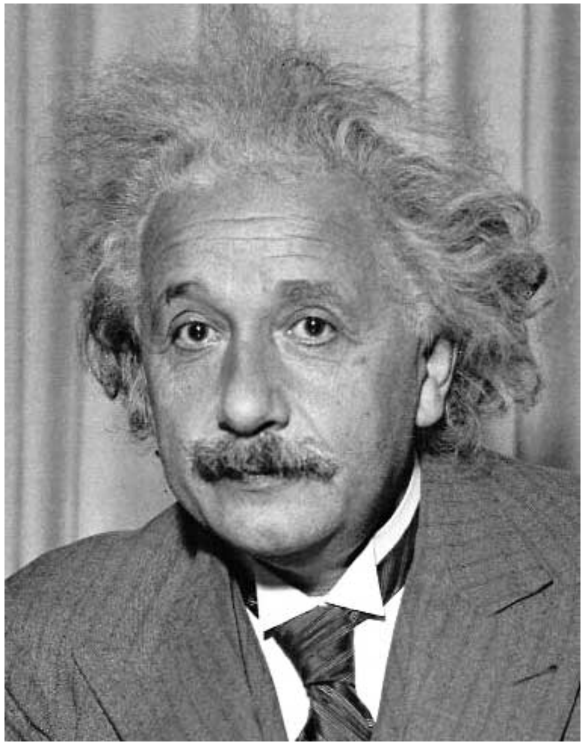
Thresholded Image

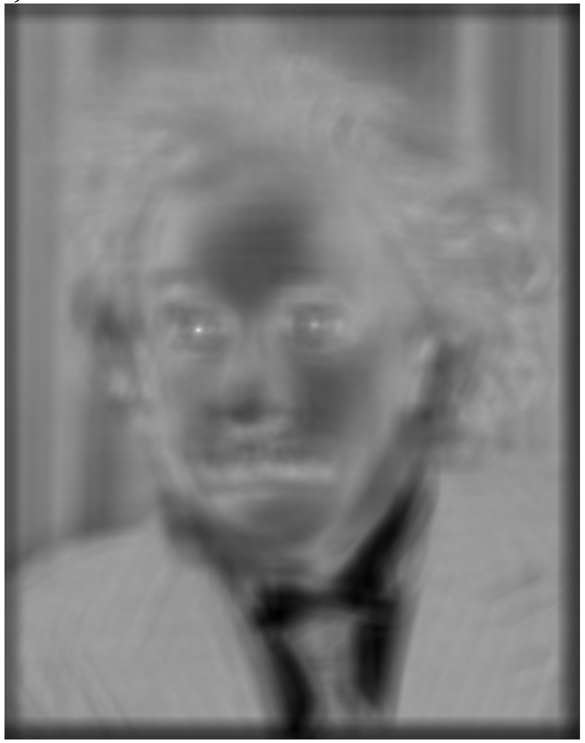
Slide from Derek Hoiem.

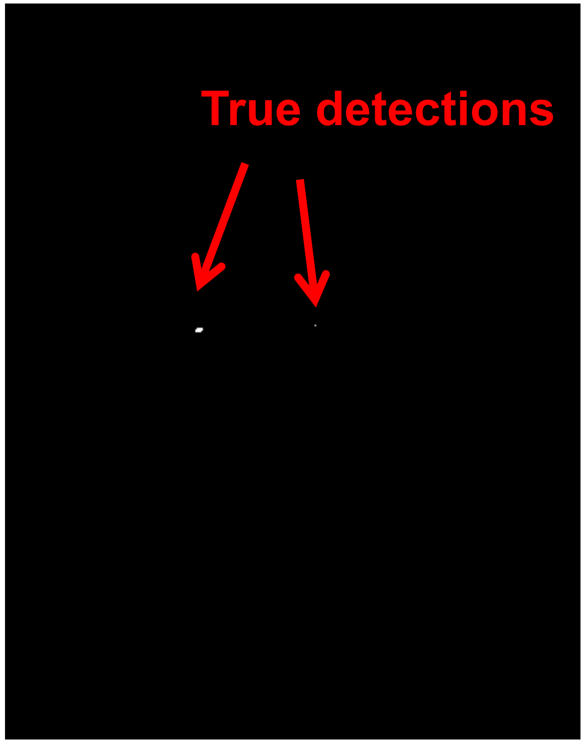
Matching with filters

- Goal: find in image
- Method 2: SSD

$$h[m,n] = \sum_{k,l} (g[k,l] - f[m+k,n+l])^2$$


Input


1- sqrt(SSD)



Thresholded Image

Slide from Derek Hoiem.

Matching with filters

Can SSD be implemented with linear filters?

$$h[m,n] = \sum_{k,l} (g[k,l] - f[m+k, n+l])^2$$

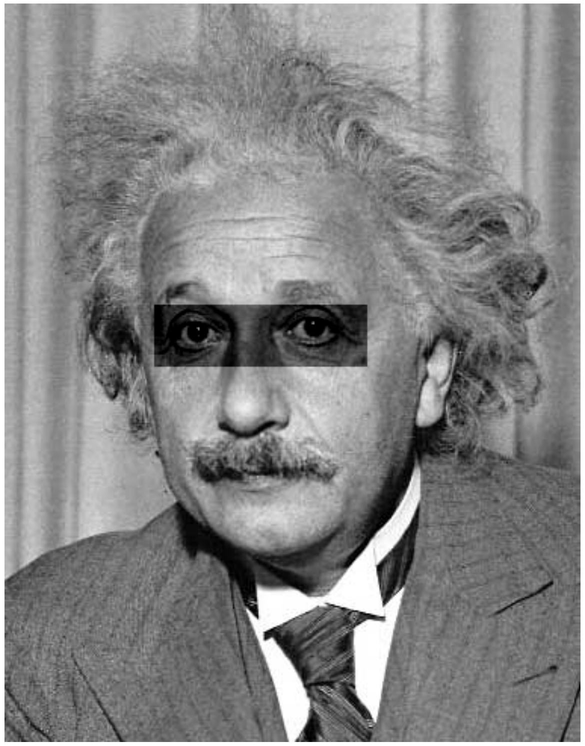
Slide from Derek Hoiem.

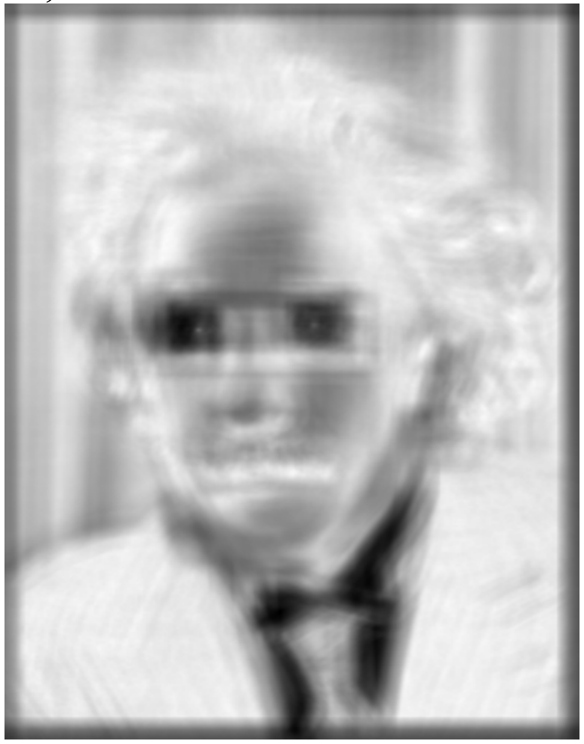
Matching with filters

- Goal: find in image
- Method 2: SSD

$$h[m,n]=\sum_{k,l}(g[k,l]-f[m+k,n+l])^2$$

What's the potential downside of SSD?


Input


1- sqrt(SSD)

Slide from Derek Hoiem.

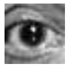
Matching with filters

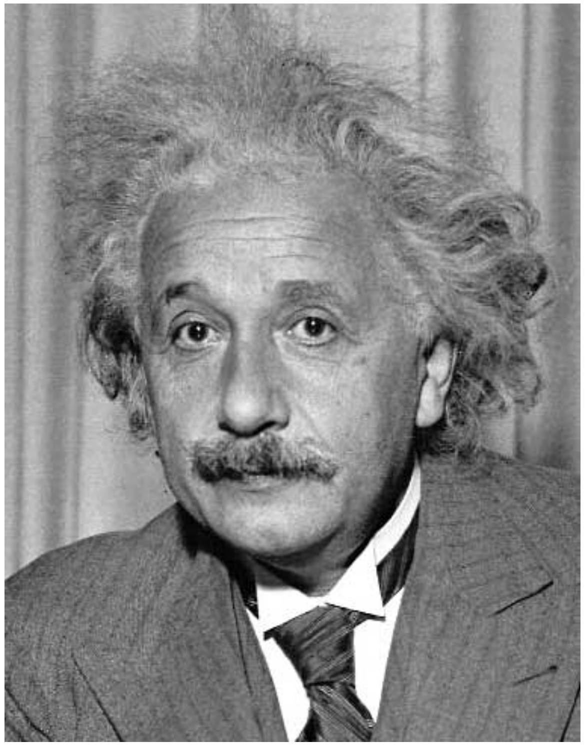
- Goal: find in image
- Method 3: Normalized cross-correlation

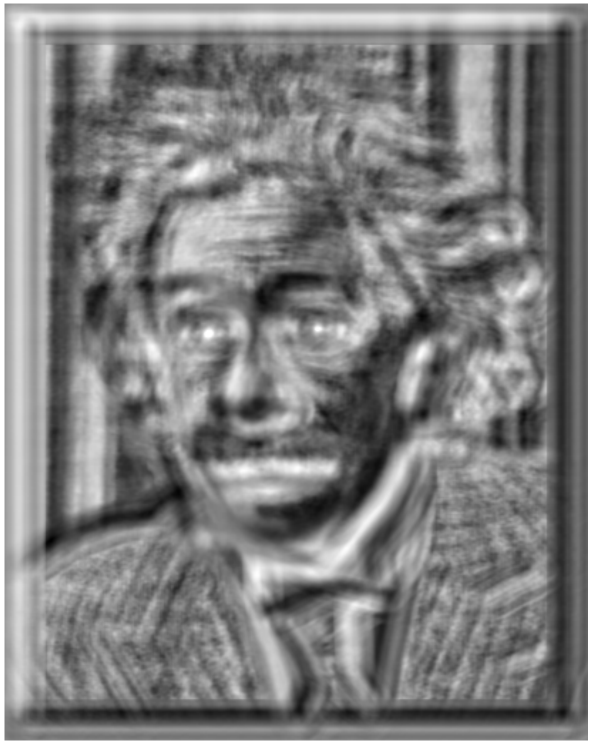
$$h[m,n]=\frac{\sum_{k,l}(g[k,l]-\overset{\substack{\text{mean template}\\ \downarrow}}{\overline{g}})(f[m+k,n+l]-\overset{\substack{\text{mean image patch}\\ \downarrow}}{\bar{f}_{m,n}})}{\left(\sum_{k,l}(g[k,l]-\overline{g})^2\sum_{k,l}(f[m+k,n+l]-\bar{f}_{m,n})^2\right)^{0.5}}$$

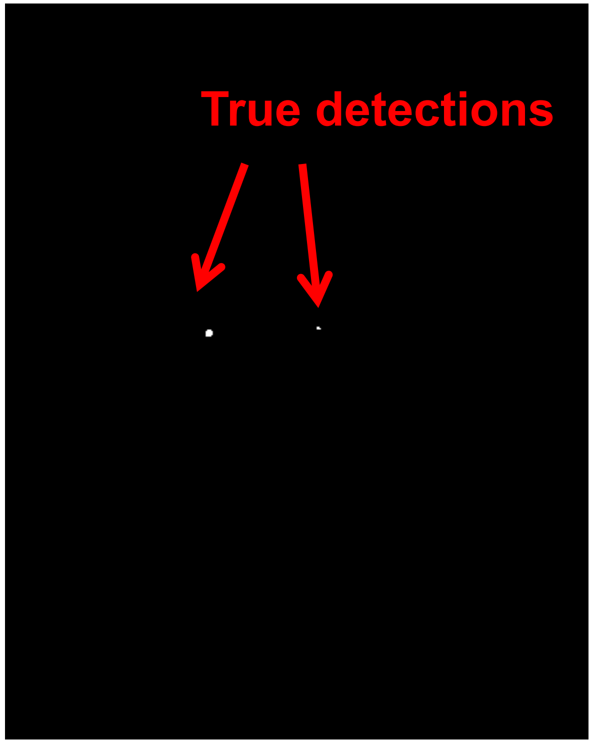
Slide from Derek Hoiem.

Matching with filters

- Goal: find  in image
- Method 3: Normalized cross-correlation


Input


Normalized X-Correlation

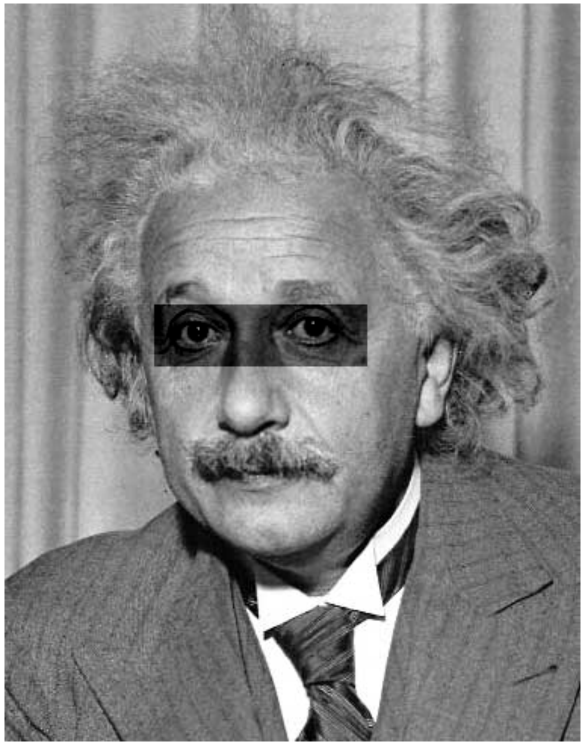


Thresholded Image

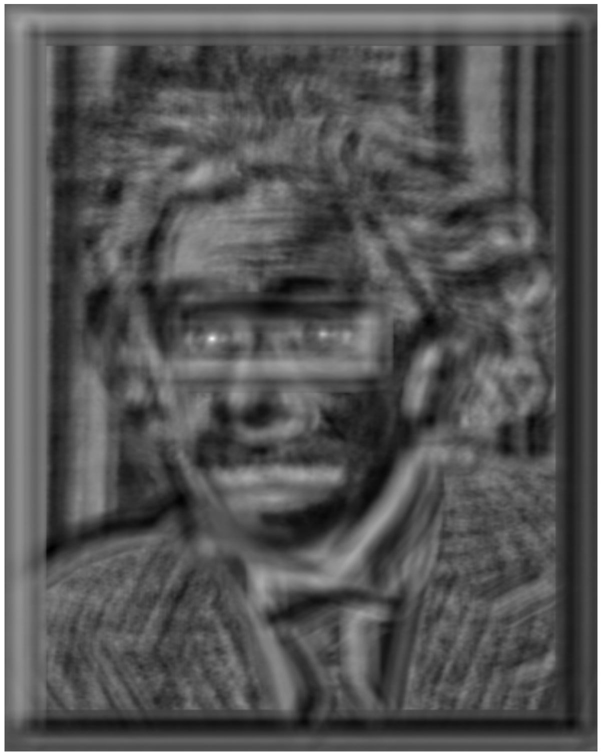
Slide from Derek Hoiem.

Matching with filters

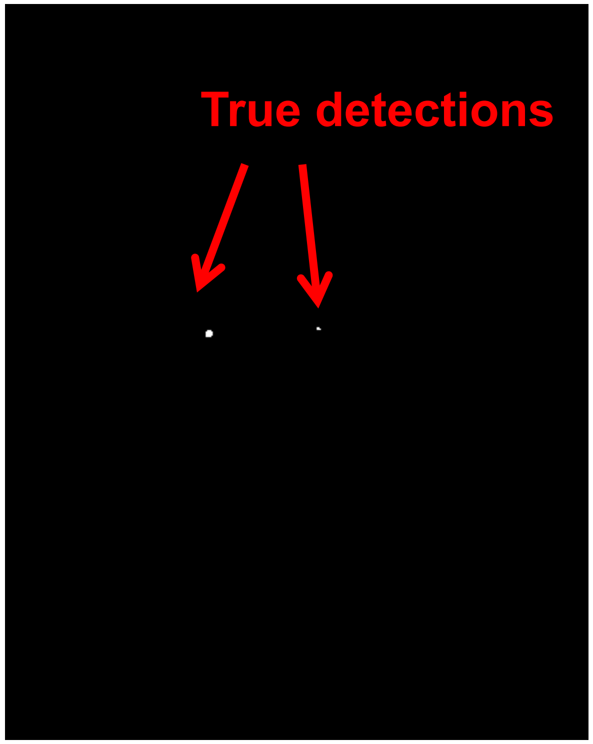
- Goal: find in image
- Method 3: Normalized cross-correlation



Input



Normalized X-Correlation



Thresholded Image

Slide from Derek Hoiem.

Q: What is the best method to use?

A: Depends

- Zero-mean filter: fastest but not a great matcher
- SSD: next fastest, sensitive to overall intensity
- Normalized cross-correlation: slowest, invariant to local average intensity and contrast

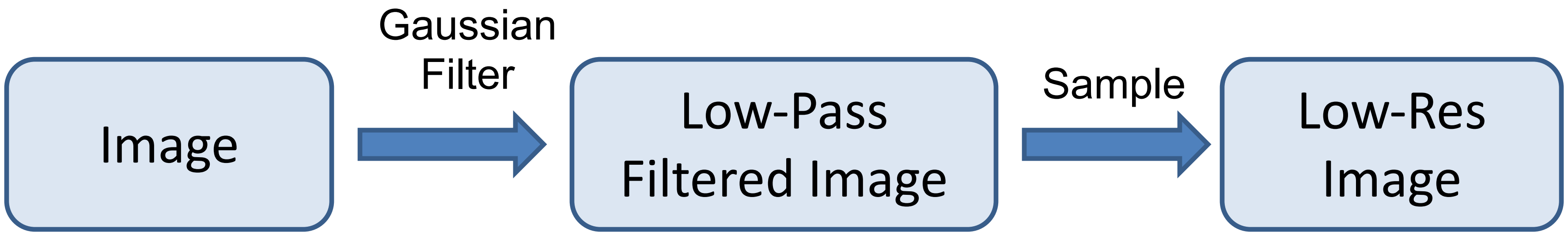
Slide from Derek Hoiem.

Q: What if we want to find larger or smaller eyes?

A: Image Pyramid

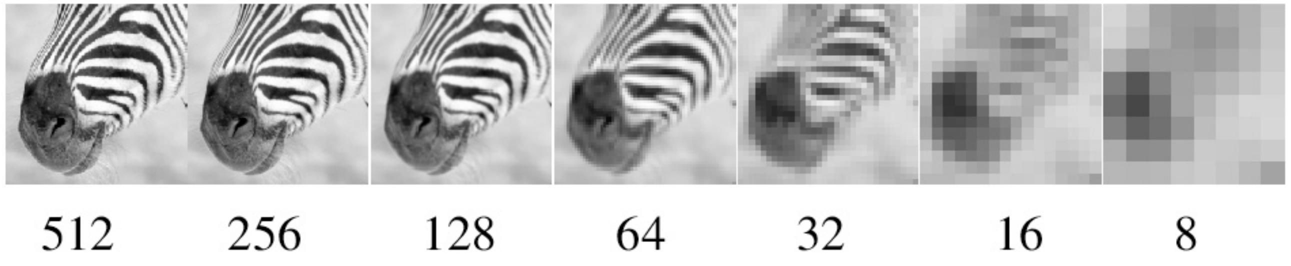
Slide from Derek Hoiem.

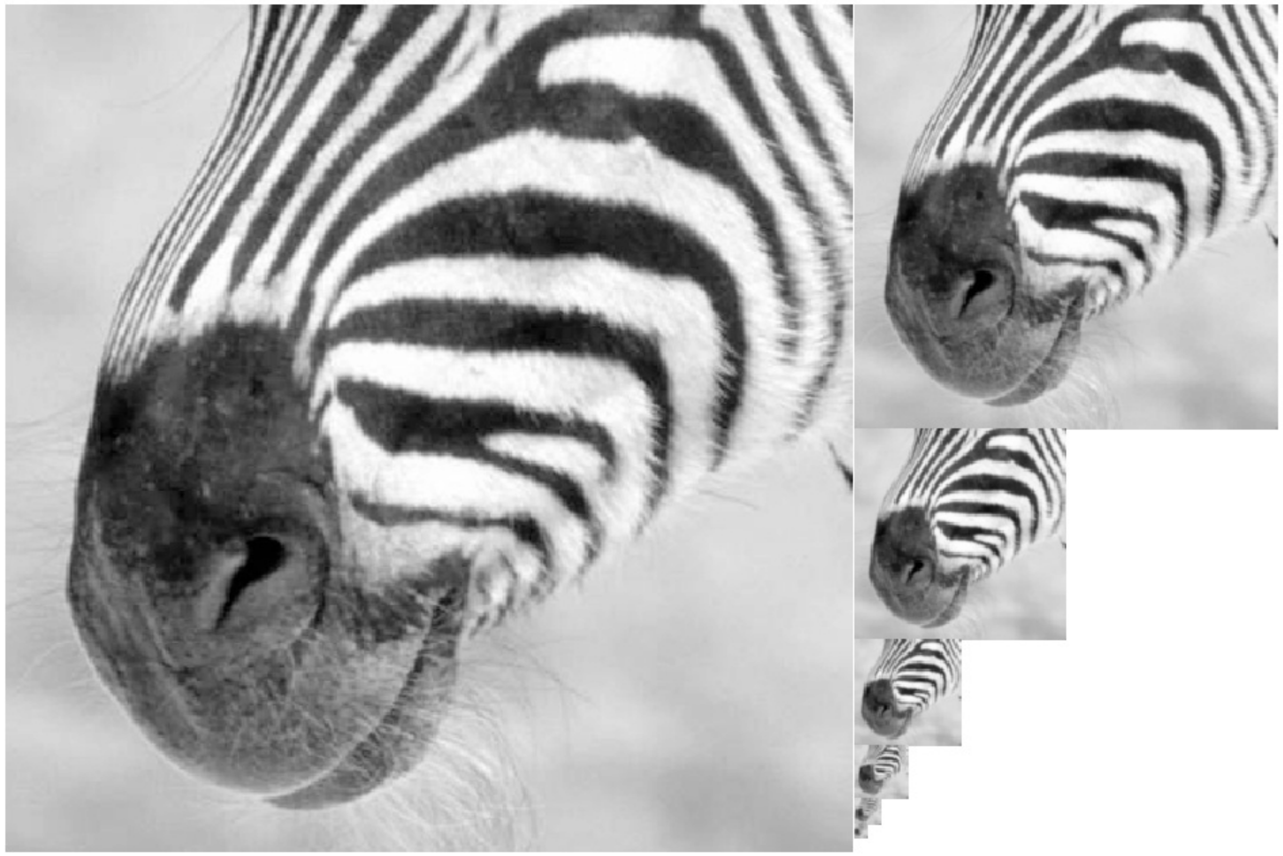
Review of Sampling



Slide from Derek Hoiem.

Gaussian pyramid





Source: Forsyth

Template Matching with Image Pyramids

Input: Image, Template

1. Match template at current scale

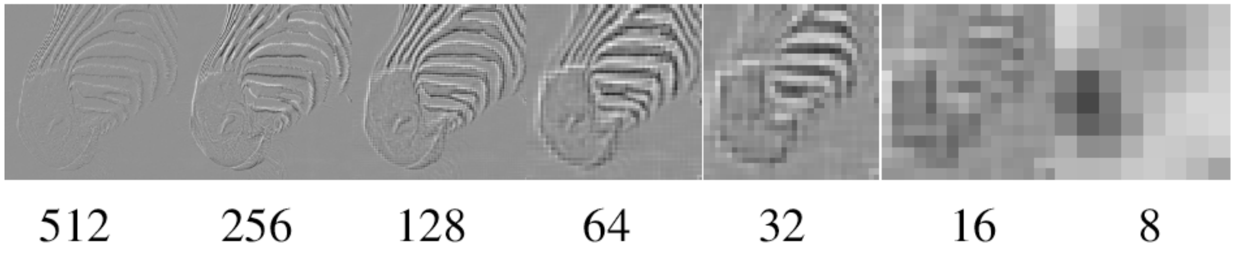
2. Downsample image
   - In practice, scale step of 1.1 to 1.2

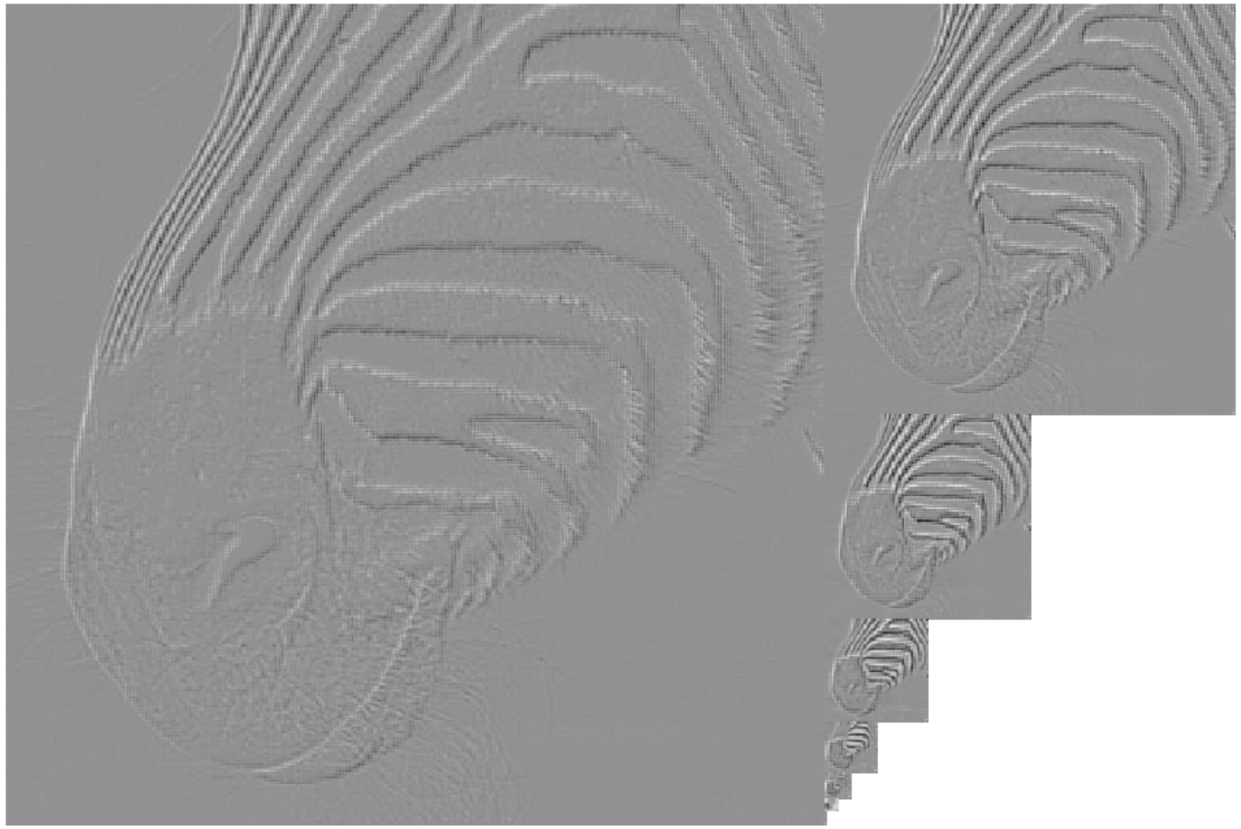
3. Repeat 1-2 until image is very small

4. Take responses above some threshold, perhaps with non-maxima suppression

Slide from Derek Hoiem.

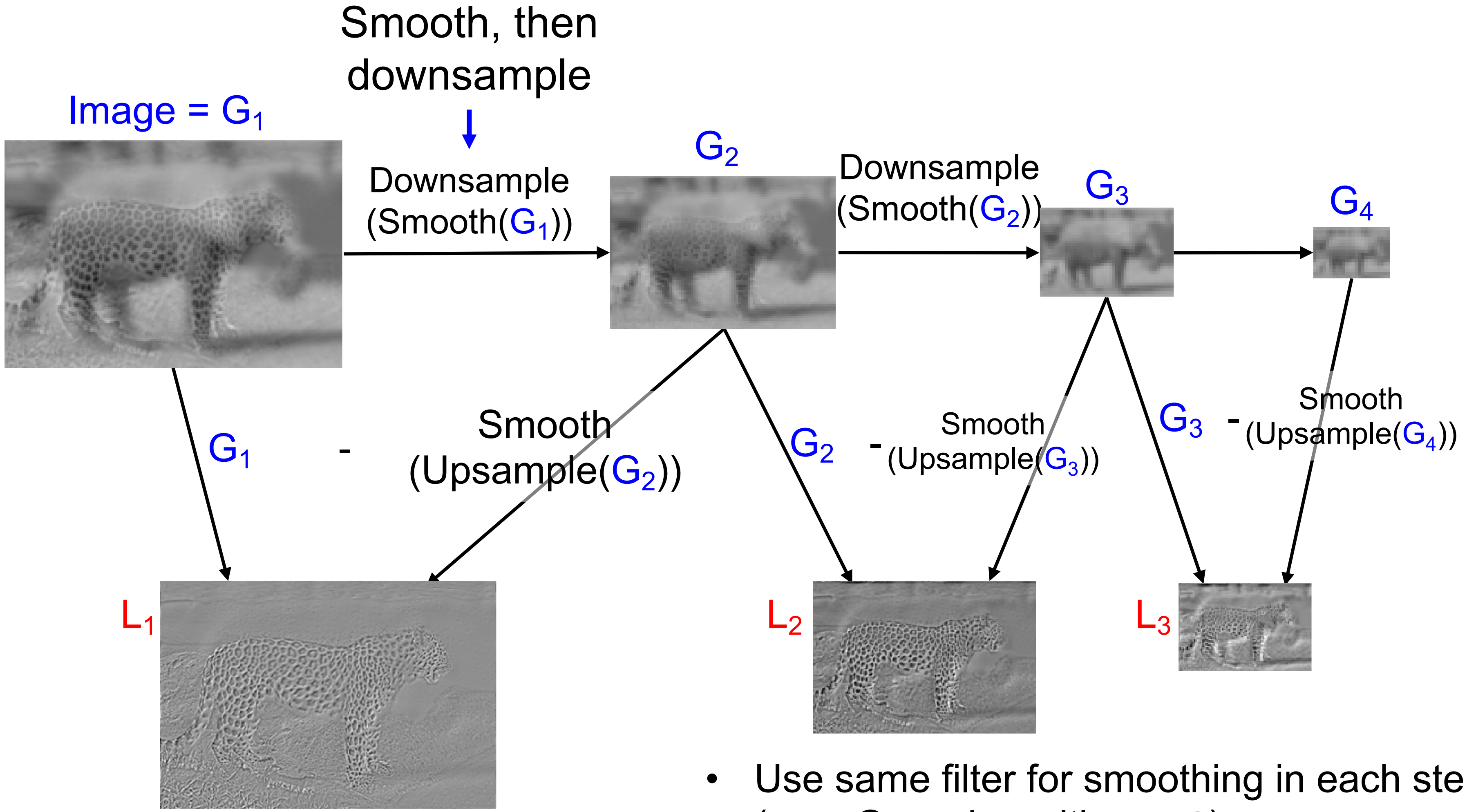
Laplacian pyramid





Source: Forsyth

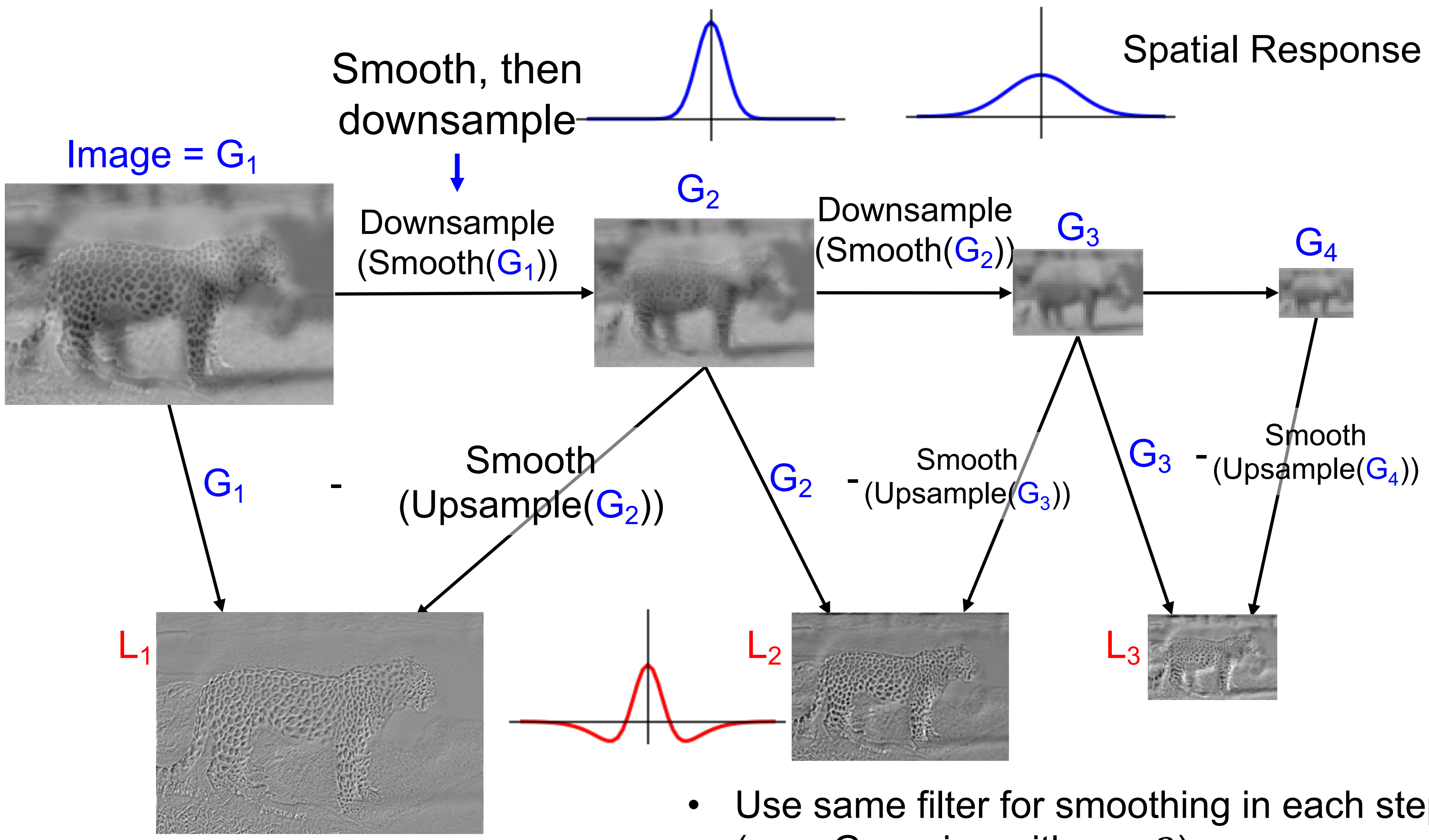
Creating the Difference of Gaussian Pyramid


Smooth, then downsample
Image = G_1
G_2
G_3
G_4
Downsample (Smooth(G_1))
Downsample (Smooth(G_2))
G_1 - Smooth (Upsample(G_2))
G_2 - Smooth (Upsample(G_3))
G_3 - Smooth (Upsample(G_4))
L_1
L_2
L_3

- Use same filter for smoothing in each step (e.g., Gaussian with $\sigma = 2$)
- Downsample/upsample with “nearest” interpolation

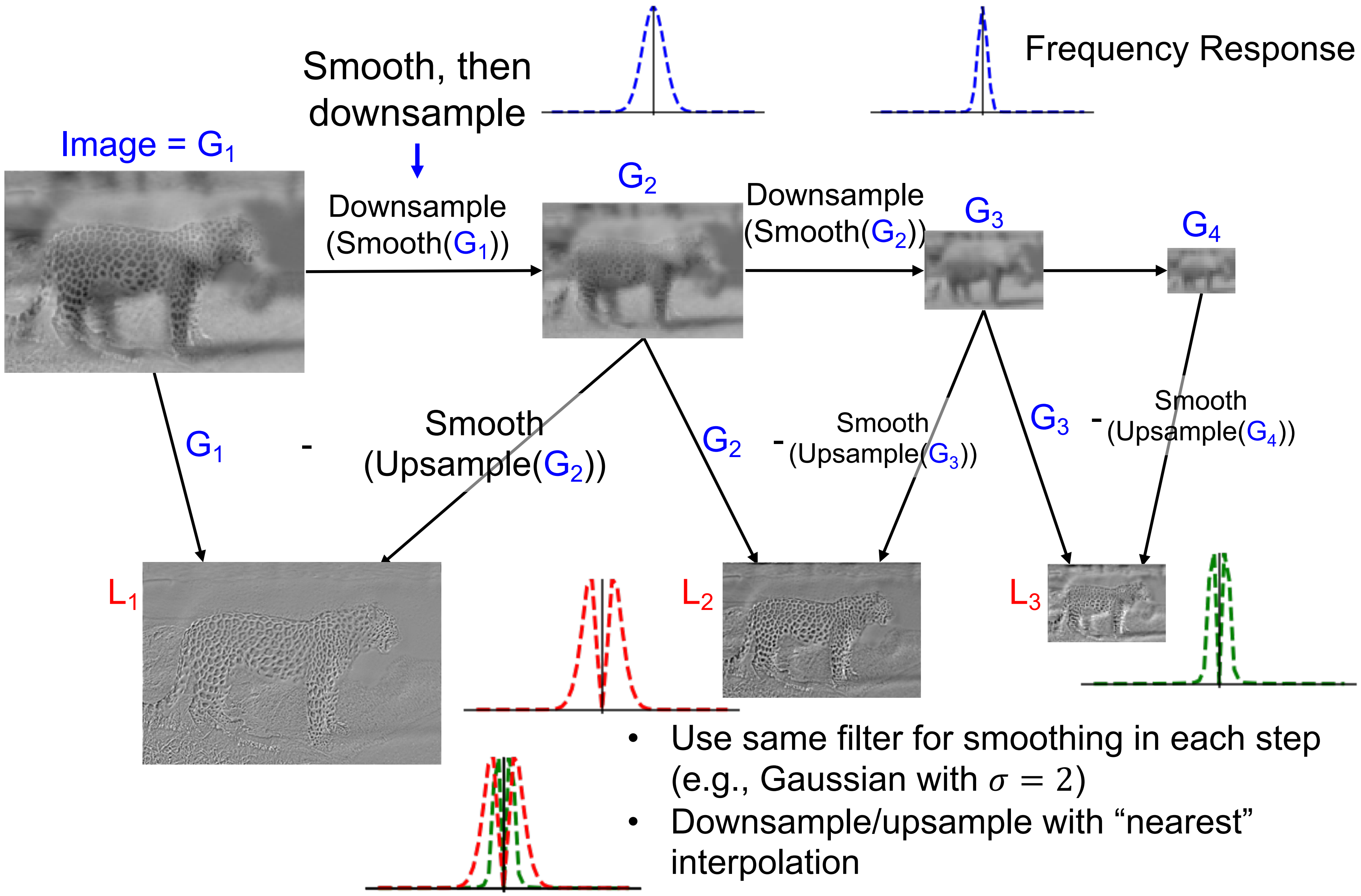
Leopard, Elephant image from Olivia and Torralba

Creating the Difference of Gaussian Pyramid


Smooth, then downsample
Spatial Response
Image = G_1
Downsample $(\text{Smooth}(G_1))$
G_2
Downsample $(\text{Smooth}(G_2))$
G_3
G_4
G_1 - Smooth $(\text{Upsample}(G_2))$
G_2 - Smooth $(\text{Upsample}(G_3))$
G_3 - Smooth $(\text{Upsample}(G_4))$
L_1
L_2
L_3

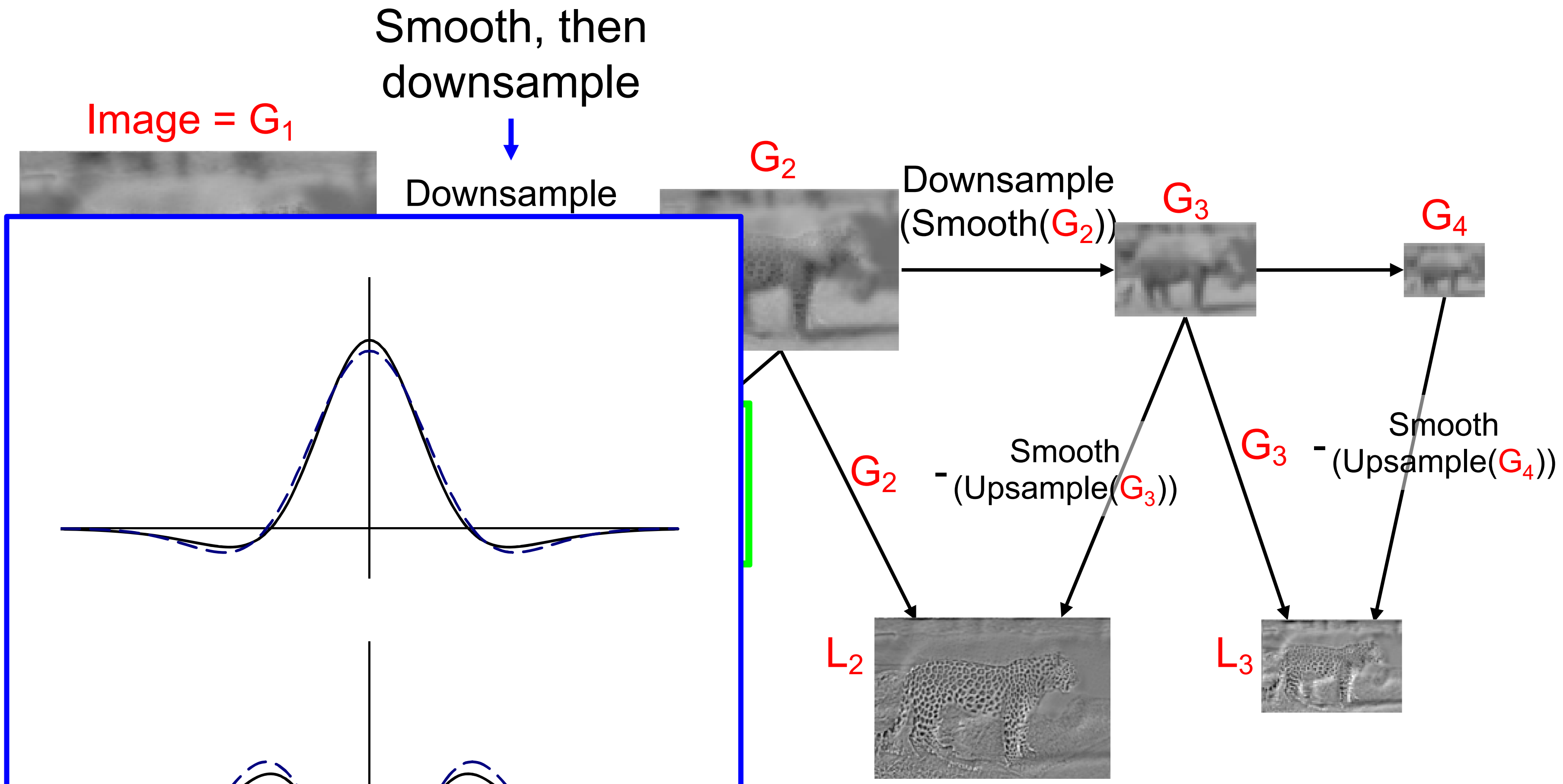
- Use same filter for smoothing in each step (e.g., Gaussian with $\sigma = 2$)
- Downsample/upsample with “nearest” interpolation

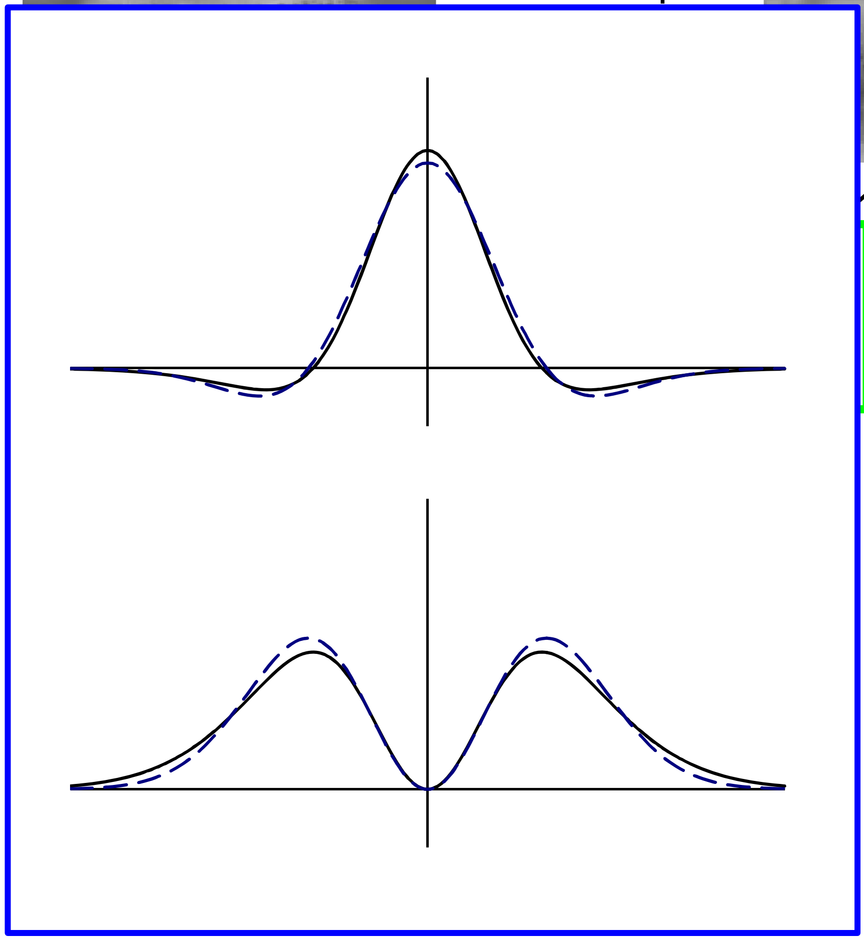
Creating the Difference of Gaussian Pyramid


Smooth, then downsample
Frequency Response
Image = G_1
Downsample (Smooth(G_1))
G_2
Downsample (Smooth(G_2))
G_3
G_4
G_1 - Smooth (Upsample(G_2))
G_2 - Smooth (Upsample(G_3))
G_3 - Smooth (Upsample(G_4))
L_1
L_2
L_3

- Use same filter for smoothing in each step (e.g., Gaussian with $\sigma = 2$)
- Downsample/upsample with “nearest” interpolation

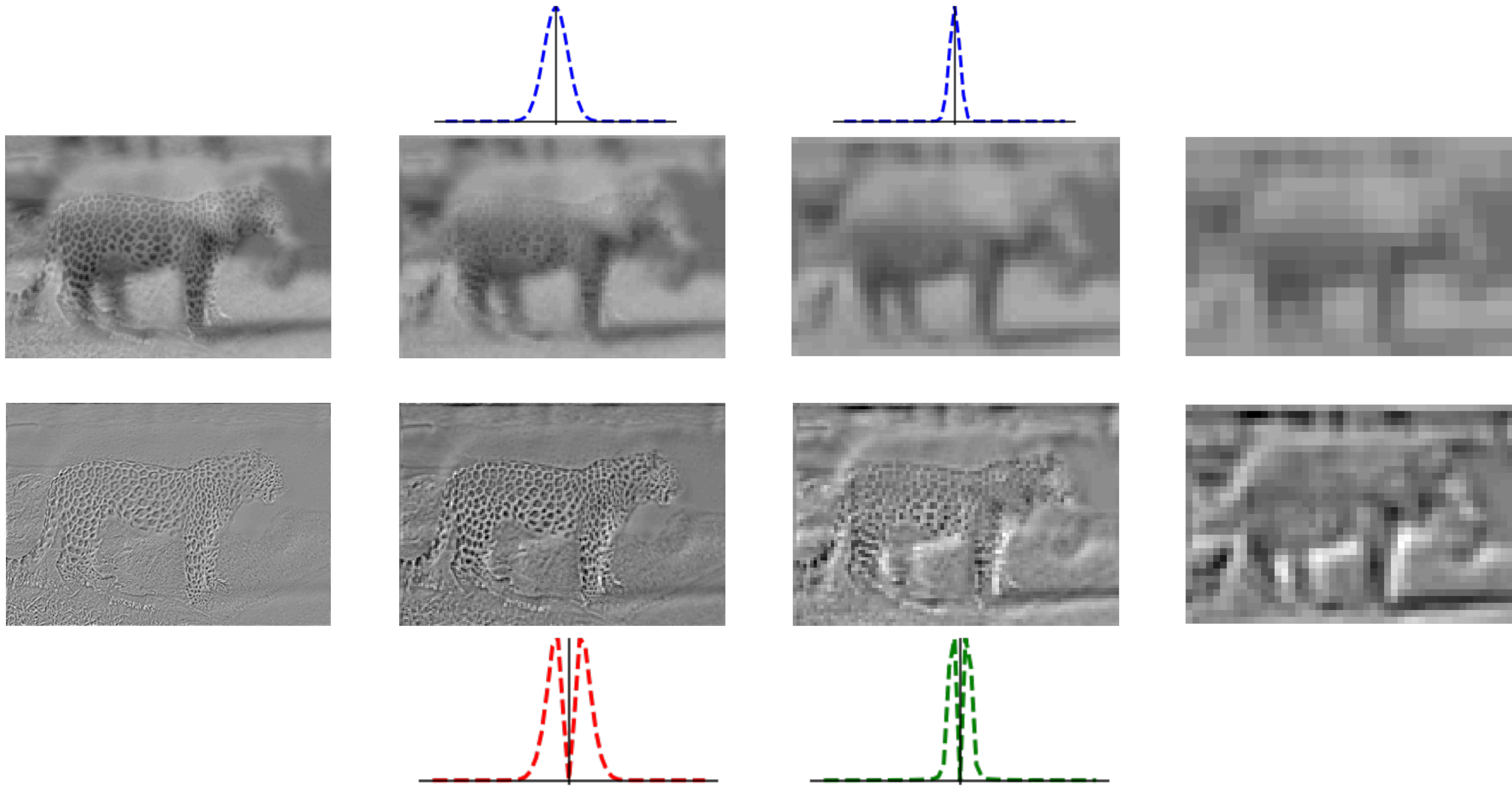
Creating the Difference of Gaussian Pyramid

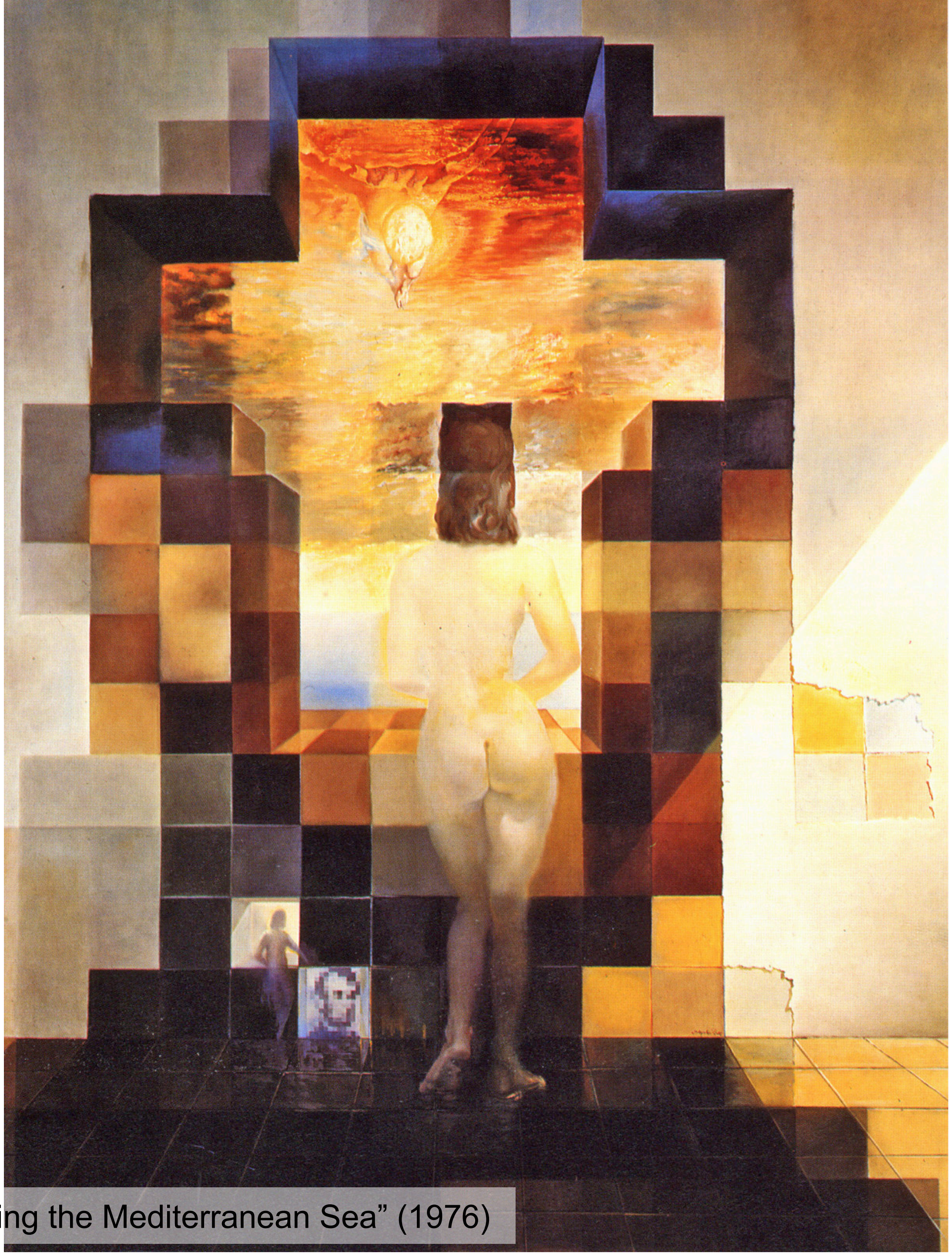

Smooth, then downsample
Image = G_1
Downsample
G_2
Downsample (Smooth(G_2))
G_3
G_4
G_2 - Smooth (Upsample(G_3))
G_3 - Smooth (Upsample(G_4))
L_2
L_3



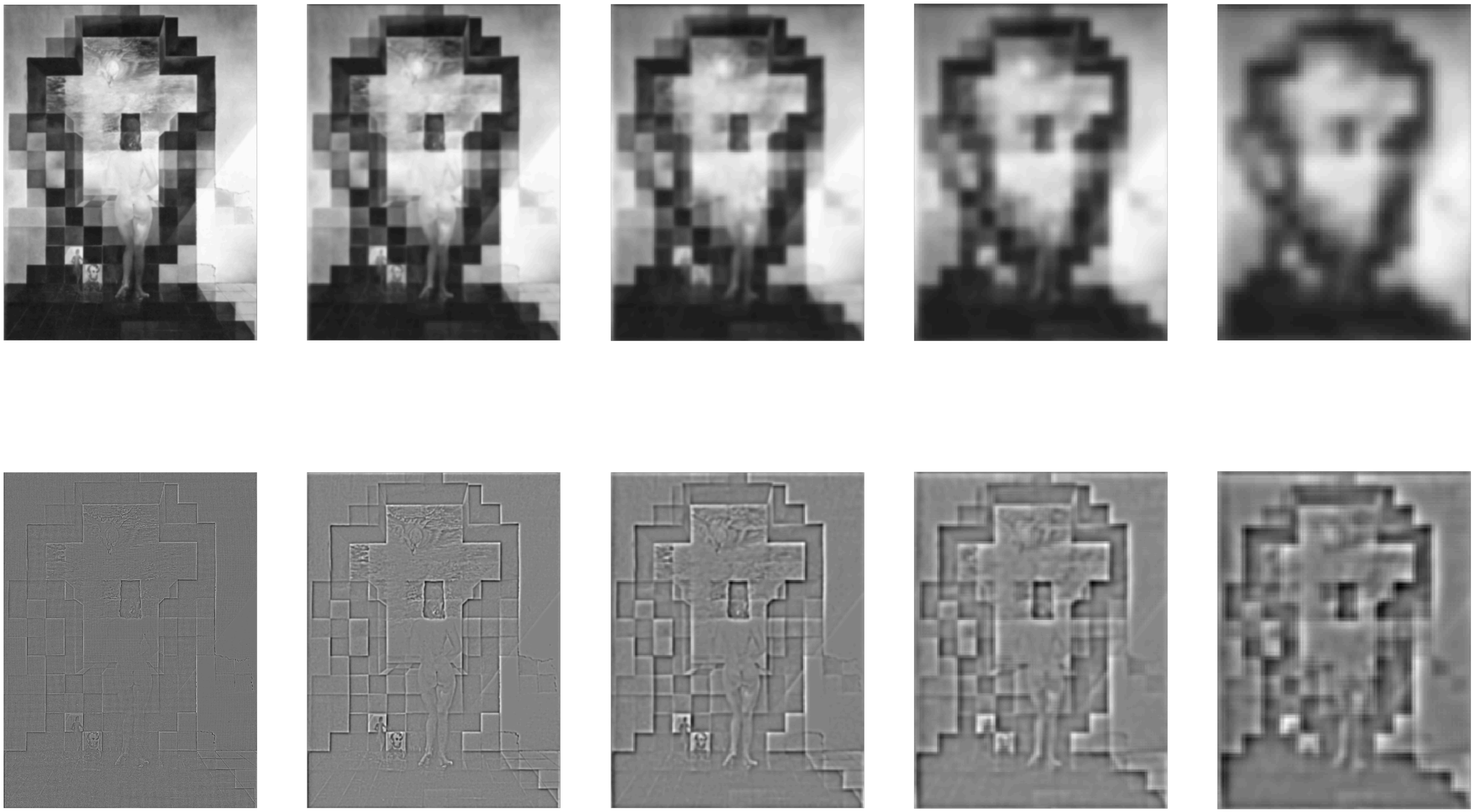
- Can also use Smooth(G_1), but then reverse isn't the exact same.
- Technically, this is a Difference of Gaussian pyramid and not a Laplacian pyramid.

Images in a Difference of Gaussian Pyramid




Dali: “Gala Contemplating the Mediterranean Sea” (1976)

Images in a Difference of Gaussian Pyramid



Dali: “Gala Contemplating the Mediterranean Sea” (1976)

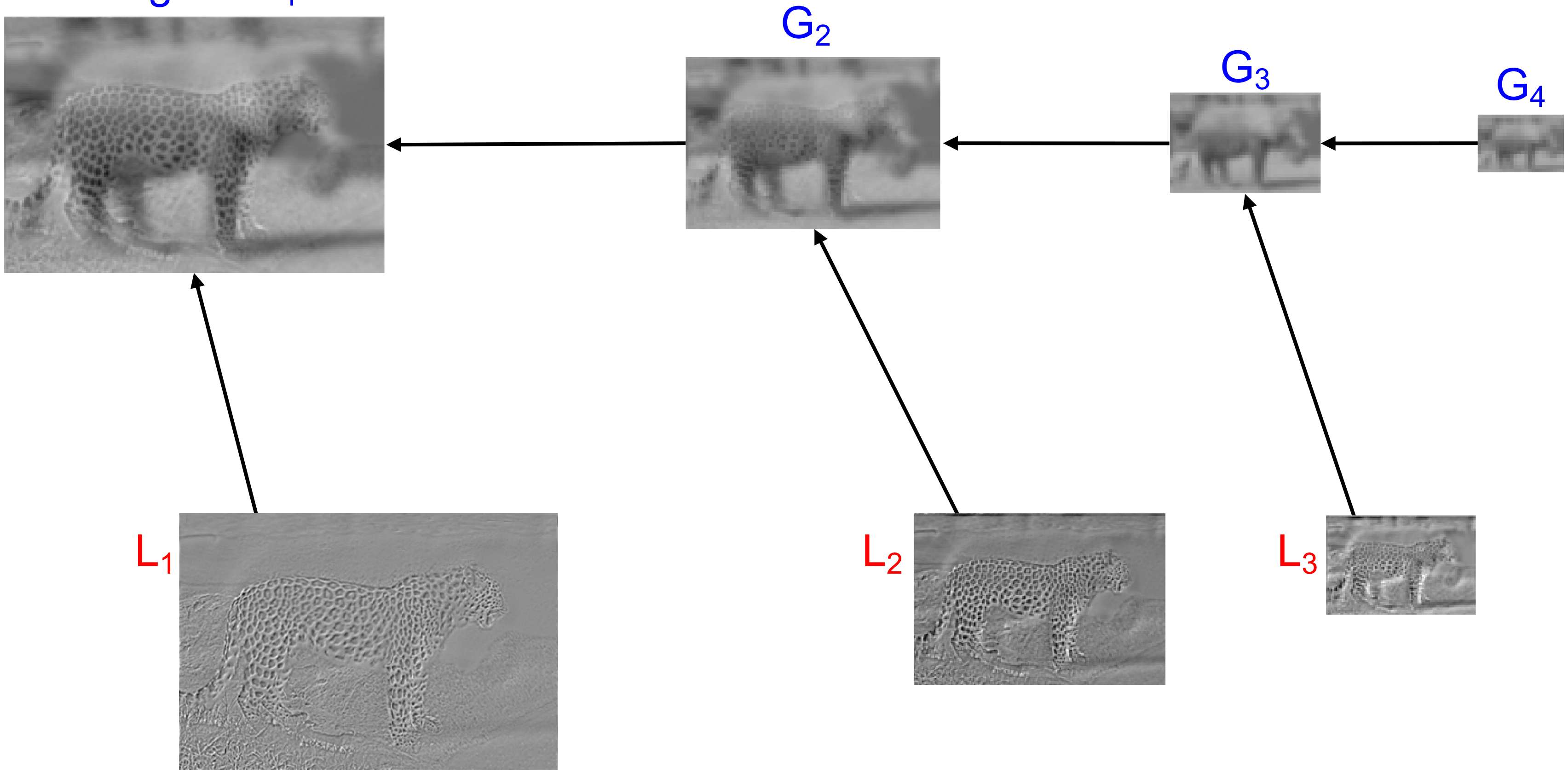
Reconstructing from Diff of Gauss Pyramid

$G_1 = L_1 +$ Smooth(Upsample(G_2))

$G_2 = L_2 +$ Smooth(Upsample(G_3))

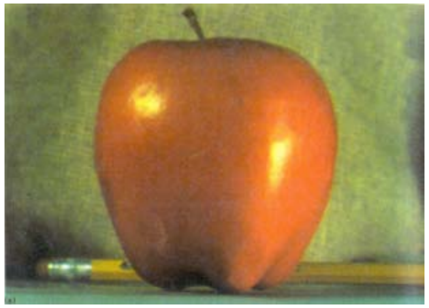
$G_3 = L_3 +$ Smooth(Upsample(G_4))

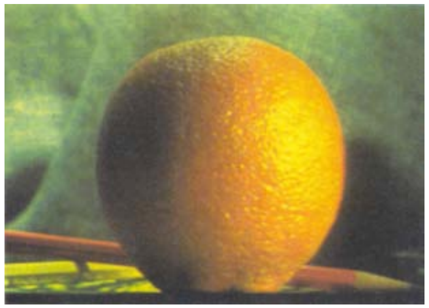
Image = G_1

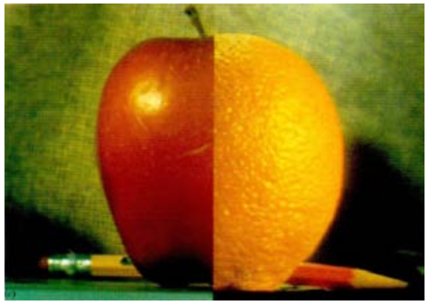


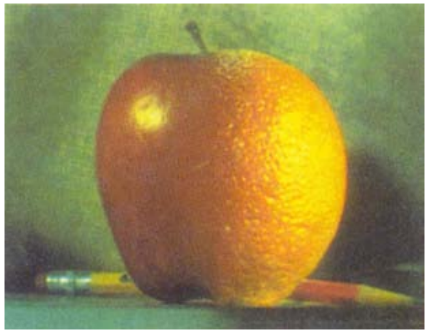
- Use same filter for smoothing as in deconstruction
- Upsample with “nearest” interpolation
- Reconstruction will be nearly lossless

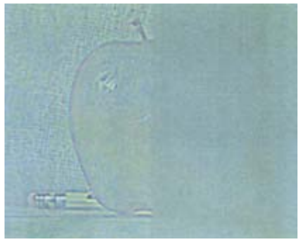
Application: Image Blending

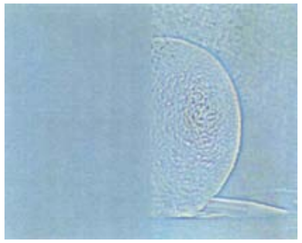

(a)

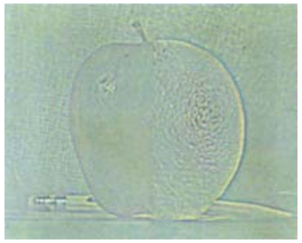

(b)

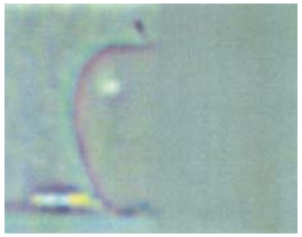

(c)

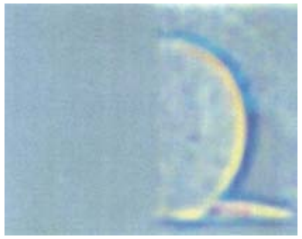

(d)

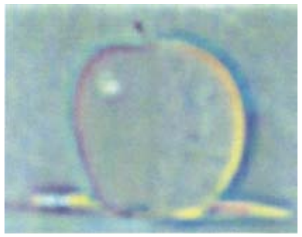

(a)

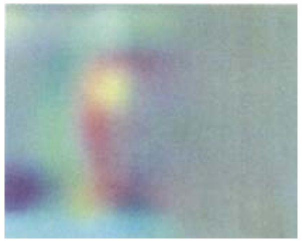

(b)

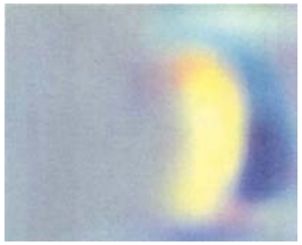

(c)

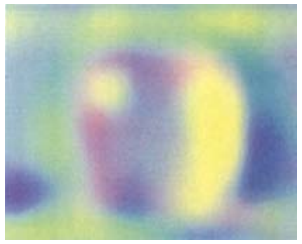

(d)

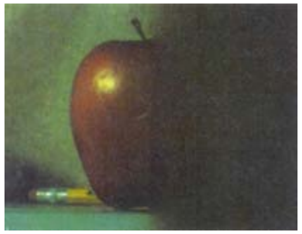

(e)

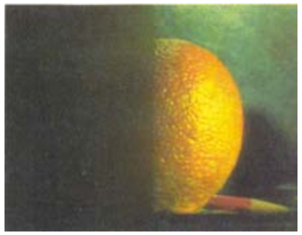

(f)

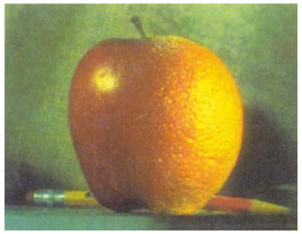

(g)


(h)


(i)







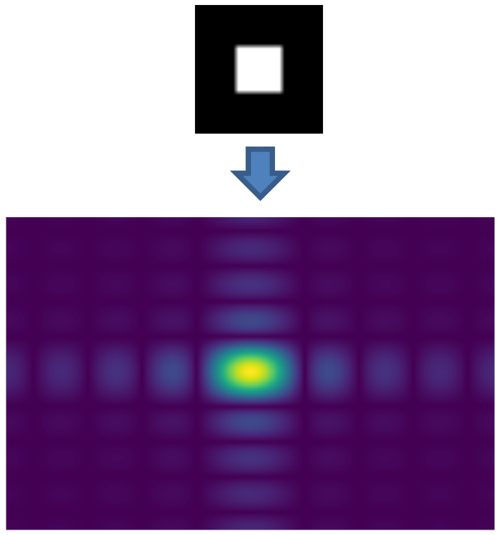
Laplacian pyramid blending (Burt and Adelson 1983b)

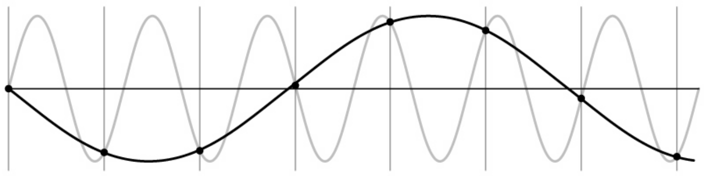
Major uses of image pyramids

- Compression
- Object detection
  - Scale search
  - Features
- Detecting stable interest points
- Registration
  - Course-to-fine

Recap

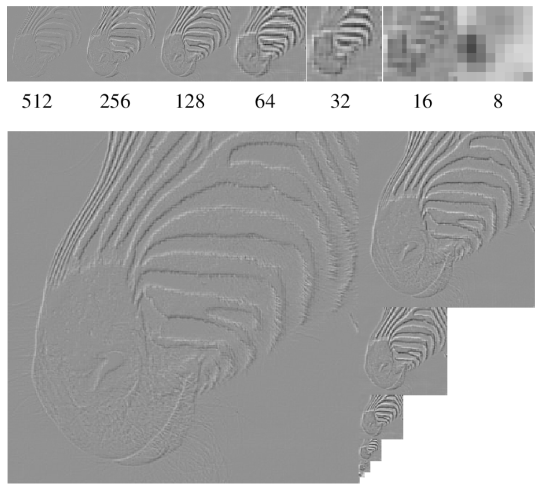
- Sometimes it makes sense to think of filtering in the frequency domain
  - Fourier analysis





- Sampling and Aliasing

- Image Pyramids


512
256
128
64
32
16
8