The University of Texas at Austin
Electrical and Computer Engineering
Cockrell School of Engineering

Spring 2022

INTRODUCTION TO COMPUTER VISION

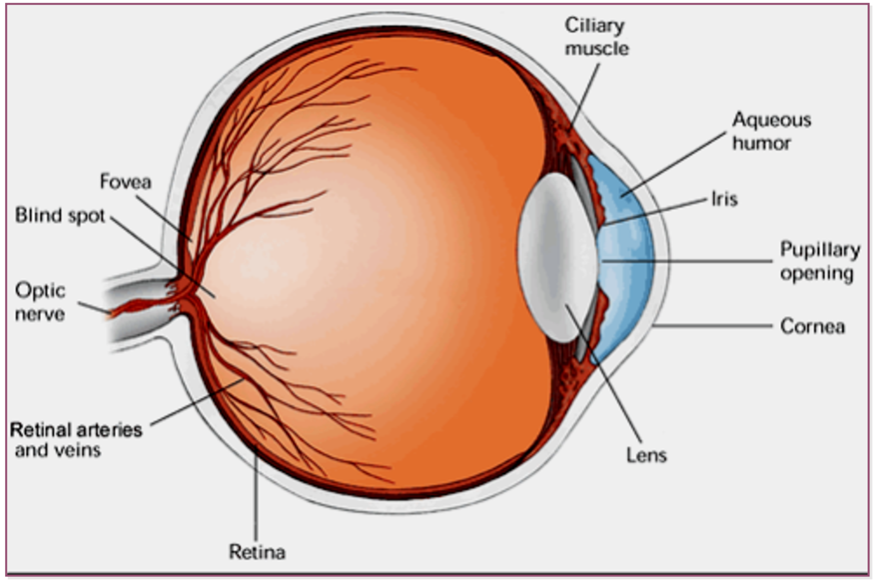
Atlas Wang
Assistant Professor, The University of Texas at Austin

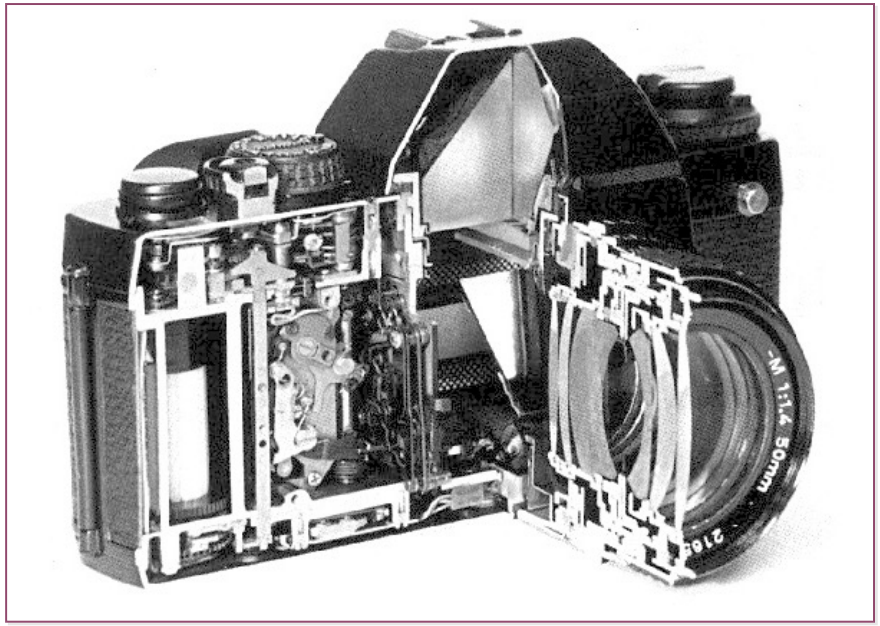
Visual Informatics Group@UT Austin
https://vita-group.github.io/

Many slides here were adapted from CMU 16-385 + Brown CSCI 1430

Image Formulation

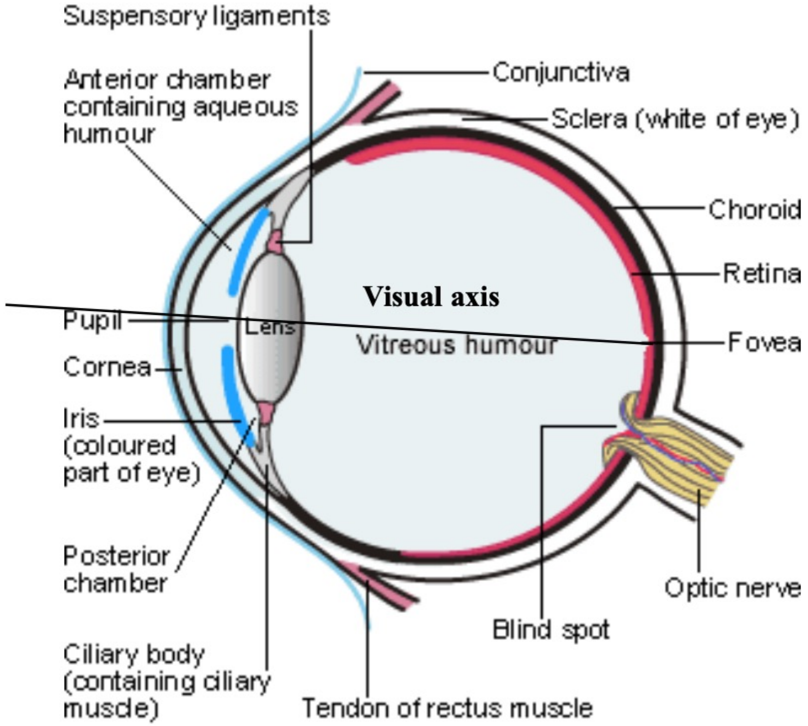
- Human: lens forms image on retina, sensors (rods and cones) respond to light
- Computer: lens system forms image, sensors (CCD, CMOS) respond to light





Overview of Human Vision: “Low-Level”

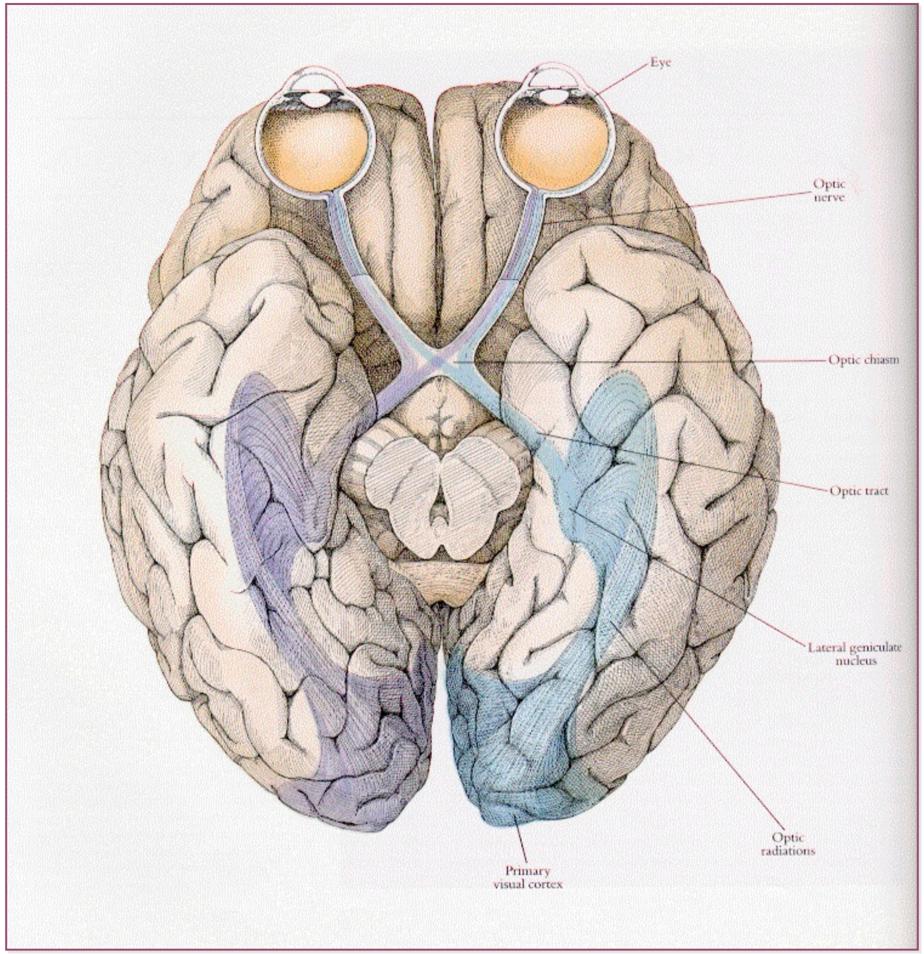
- Human visual perception plays a key role in composing our ”computer vision” techniques!
- **Lens** and **Cornea**: focusing on the objects
- Two receptors in the retina: **Cones** and **Rods**
  - Cones located in fovea and are sensitive to **color**
  - Rods give a general overall picture of view, are insensitive to color but sensitive to the **level of illumination**

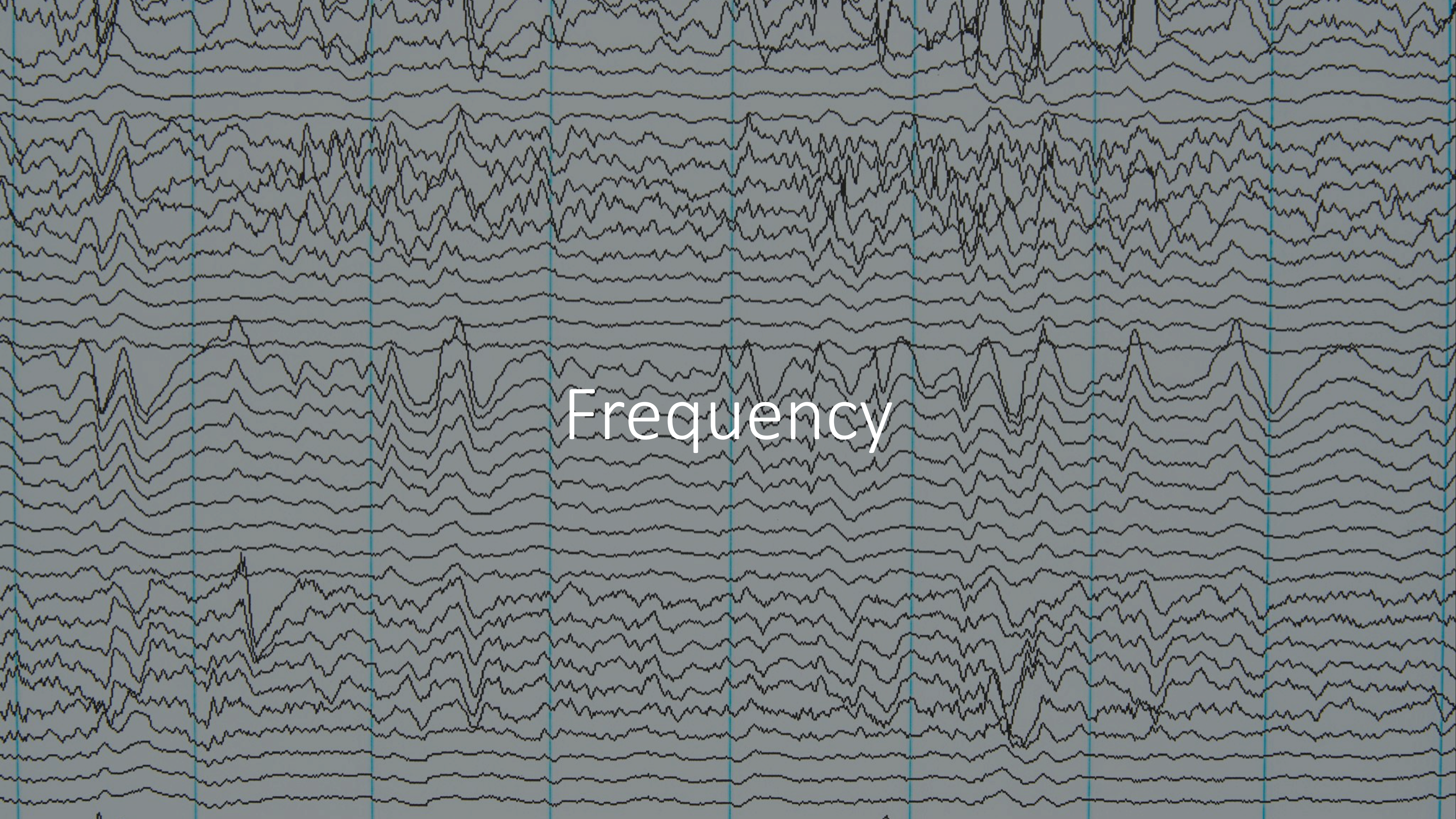


http://www.mydr.com.au/eye-health/eye-anatomy

Overview of Human Vision: "Mid & High-Level"

- Lateral Geniculate Nucleus – function unknown (visual adaptation?)
- Primary Visual Cortex ("**magnitude and phase**")
  - Simple cells: orientational sensitivity
  - Complex cells: directional sensitivity
- Further processing ("**what-where pathway**")
  - Temporal cortex: what is the object?
  - Parietal cortex: where is the object? How do I get it?
- Recognition-by-components (**RBC**) theory

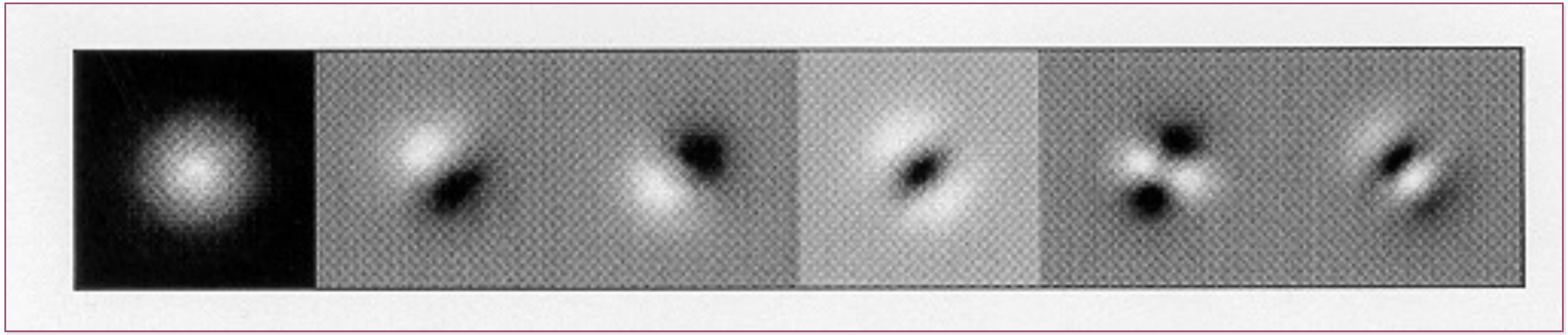




Frequency

Your Brain Secretly Thinks in the Frequency Domain

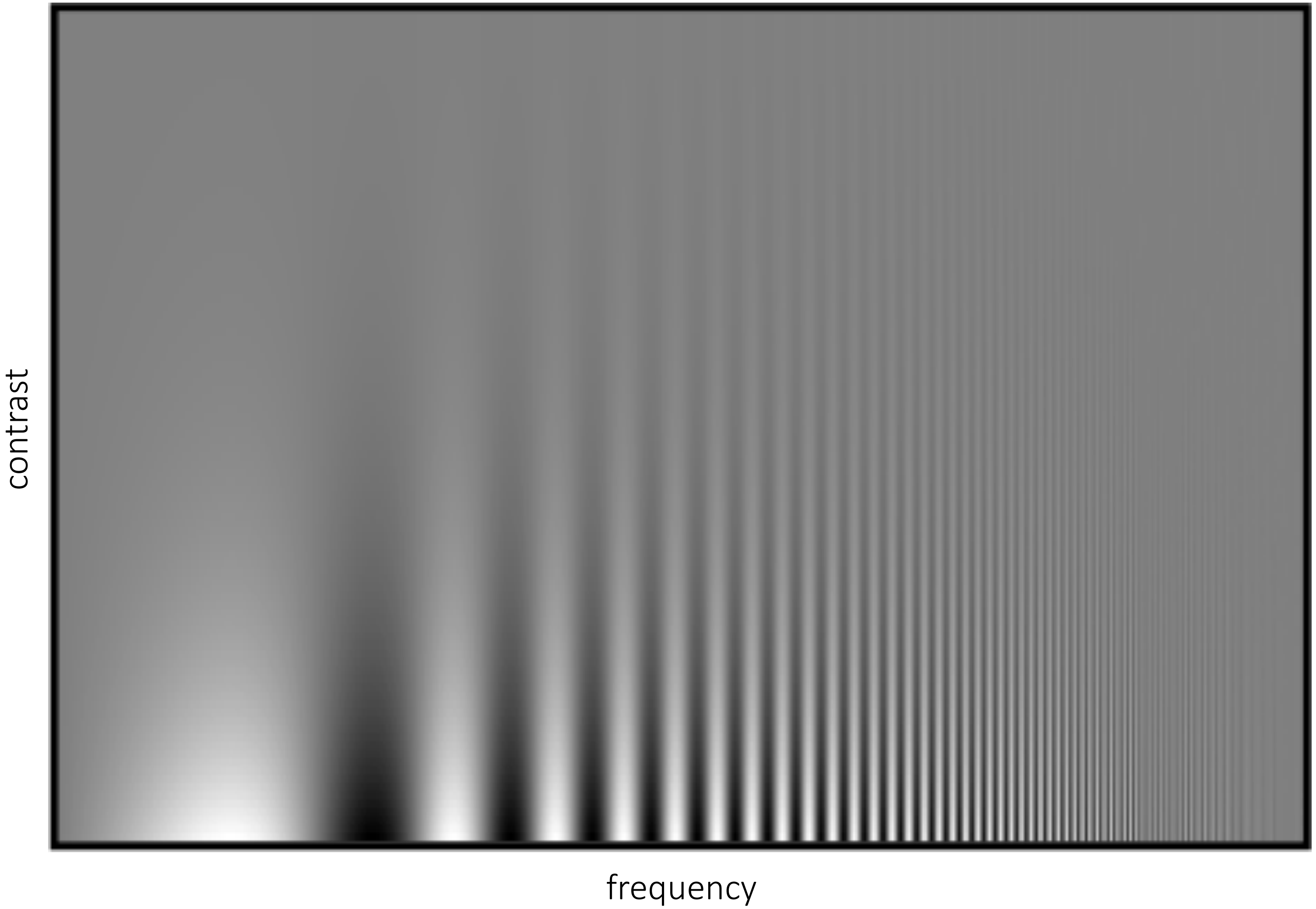
- Low-level human vision can be (partially) modeled as a set of *multi-resolution, and multi-orientation* filters



- Human perception cues are dominated by mid- to high-frequency bands
  - The spatial-frequency theory refers to the theory that the visual cortex operates on a code of spatial frequency, not on the code of straight edges and lines
  - When we see something from a distance, we are effectively subsampling it
  - "Depth Cues": Focus, Vergence, Stereo …

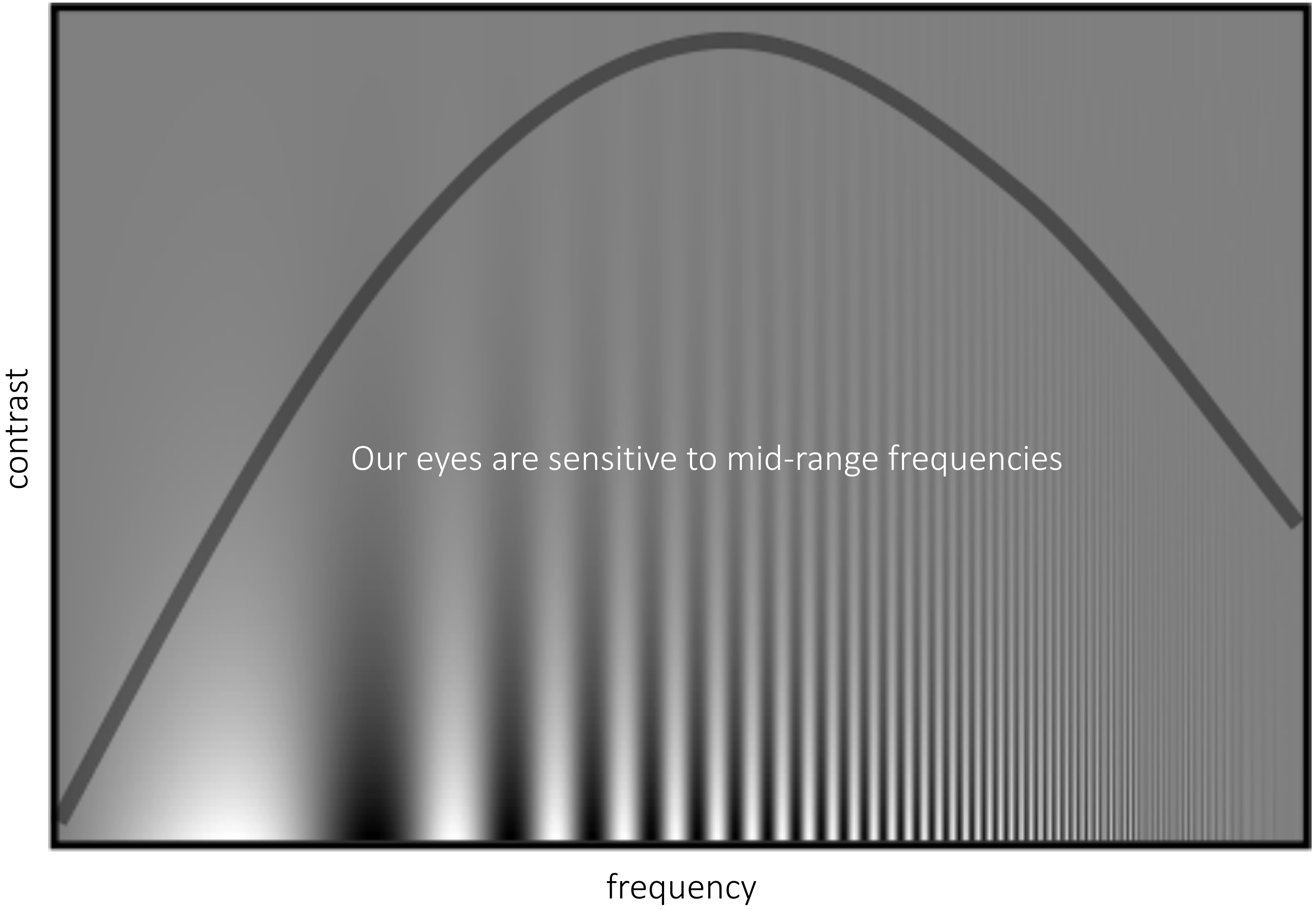
Variable frequency sensitivity

Experiment: Where do you see the stripes?



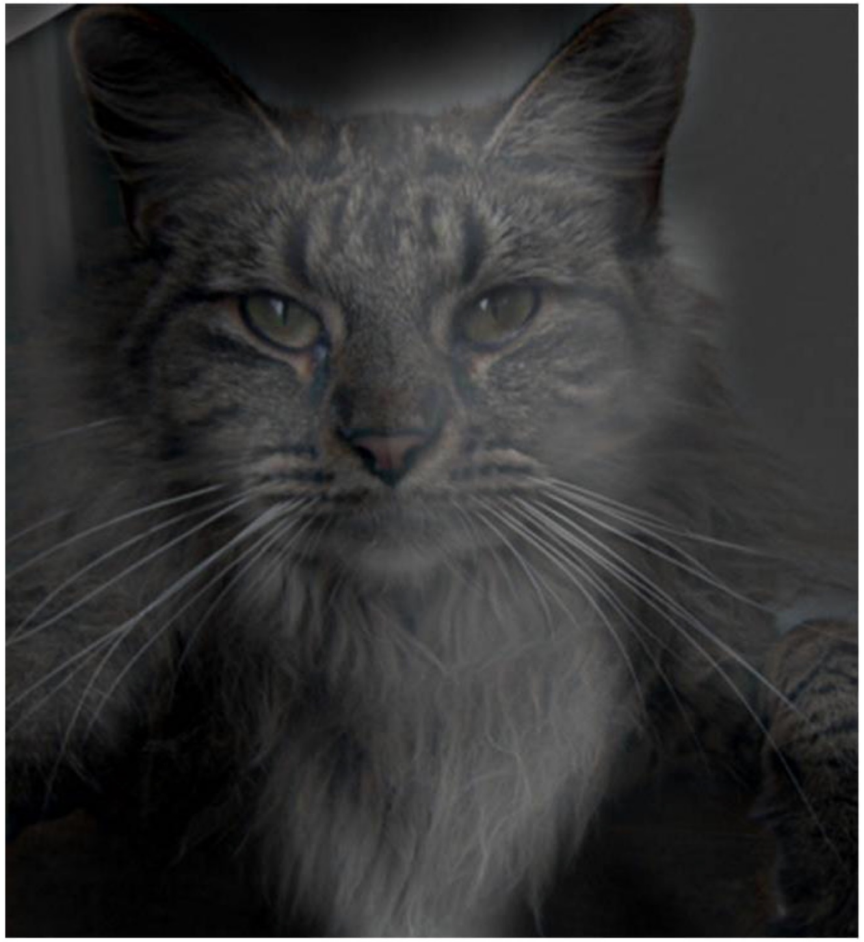
Variable frequency sensitivity

Campbell-Robson contrast sensitivity curve



- Early processing in humans filters for various orientations and scales of frequency
- Perceptual cues in the mid frequencies dominate perception

Example: Hybrid Images



?

Distance-dependent perception of hybrid images by human

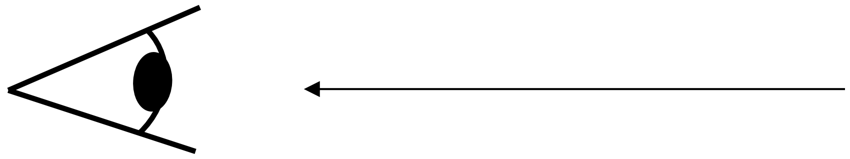
Are you still complaining deep networks are easily fooled? ☺

Light & Color

What is light?

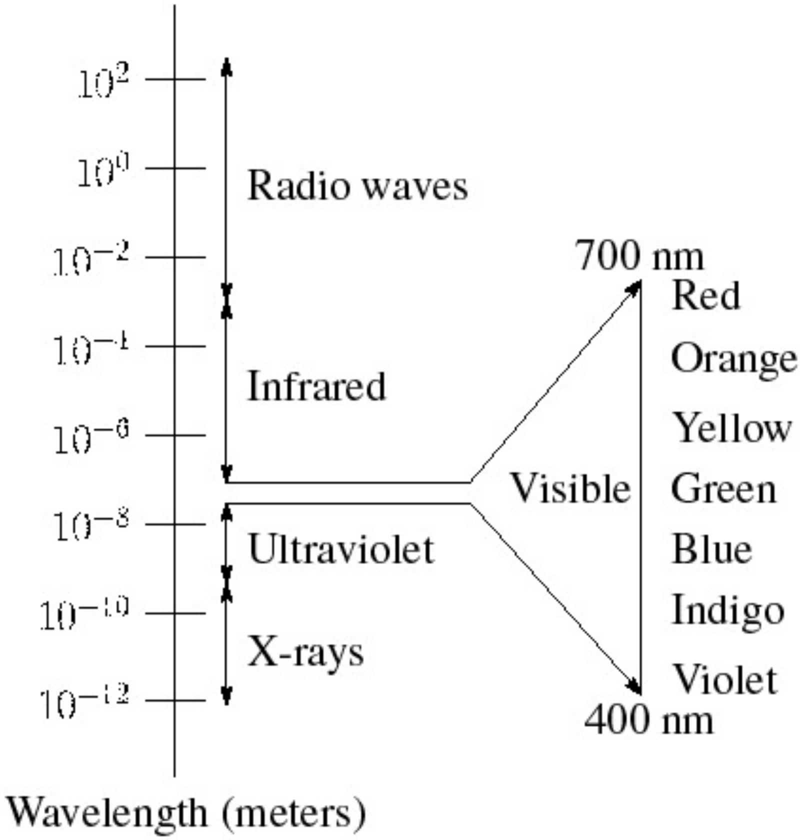
Electromagnetic radiation (EMR) moving along rays in space

- R(λ) is EMR, measured in units of power (watts)
  - λ is wavelength



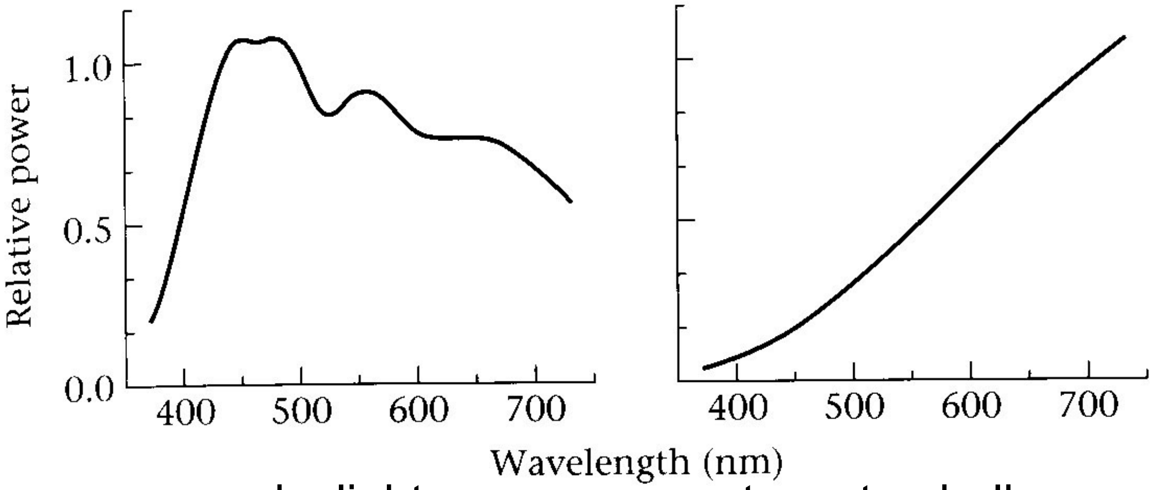
Perceiving light

- We “see” electromagnetic radiation in a range of wavelengths
- How do we convert radiation into “color”?
- What part of the spectrum do we see?



Light spectrum

- The appearance of light depends on its power **spectrum**
  - How much power (or energy) at each wavelength



daylight tungsten bulb

Our visual system converts a light spectrum into "color"

- This is a rather complex transformation

Our perceived brightness is often “relative”

- The apparent brightness depends on the surrounding region
  - **Brightness contrast**: a constant-colored region seem lighter or darker depending on the surround:

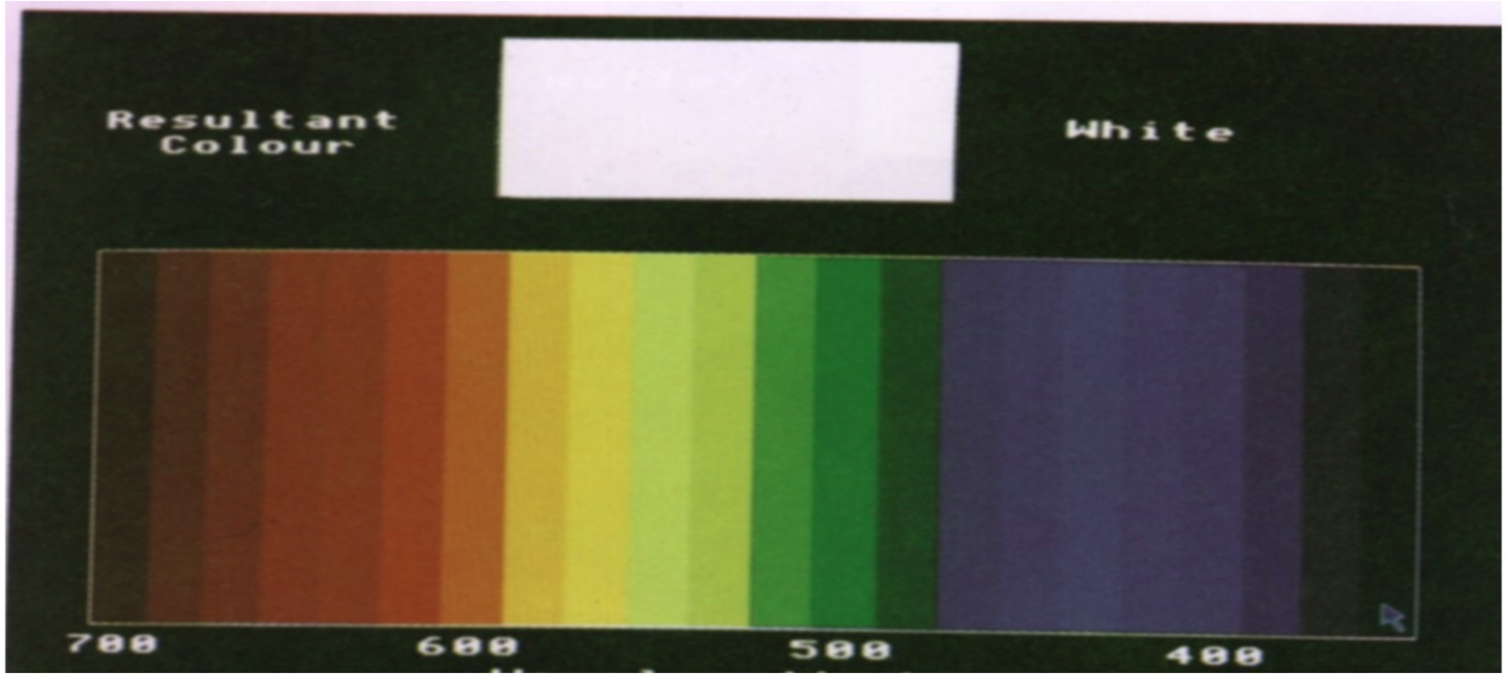
    - http://www.sandlotscience.com/Contrast/Checker_Board_2.htm

  - **Brightness constancy**: a surface looks the same under widely varying lighting conditions
    - For example, something white will appear to be the same shade of white no matter how much light it is being exposed to - noontime sunlight or a soft lamplight at night.
    - A type of psychological “perceptual constancy” (other constancy forms: color, shape …)

What is Color?

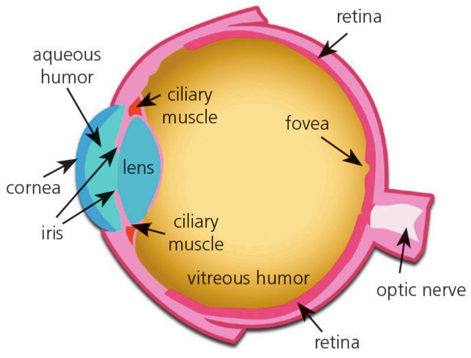
- We almost never see a “pure” wavelength of light; rather a mixture of wavelengths, each with a different “power”

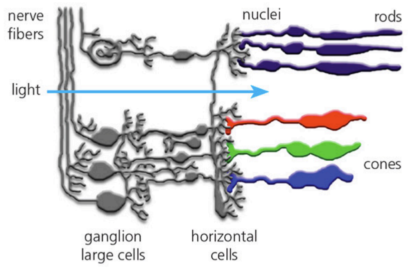
- Only some colors occur as pure wavelengths; mose are mixtures of pure colors (e.g. white)

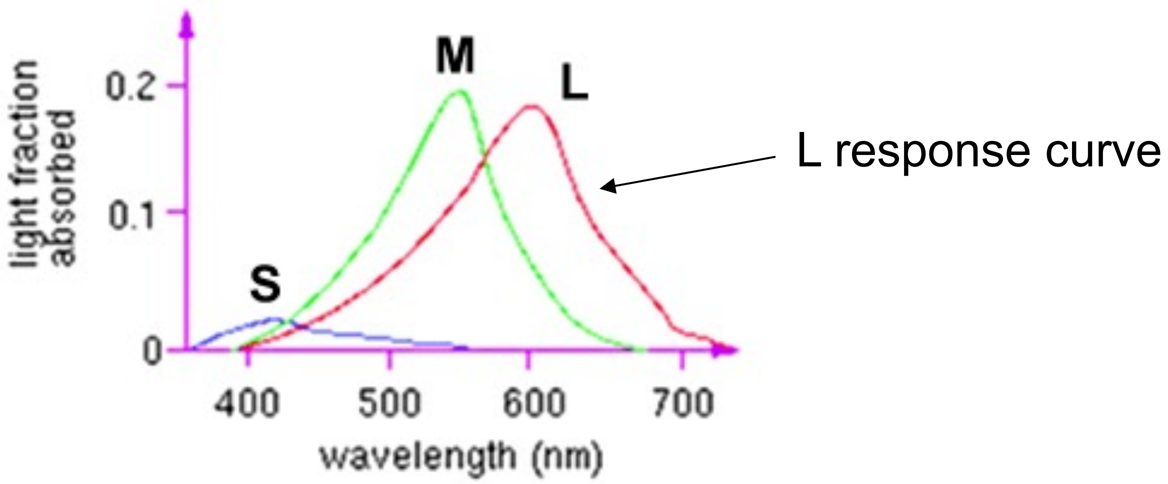


Source: G Hager Slides

Color perception

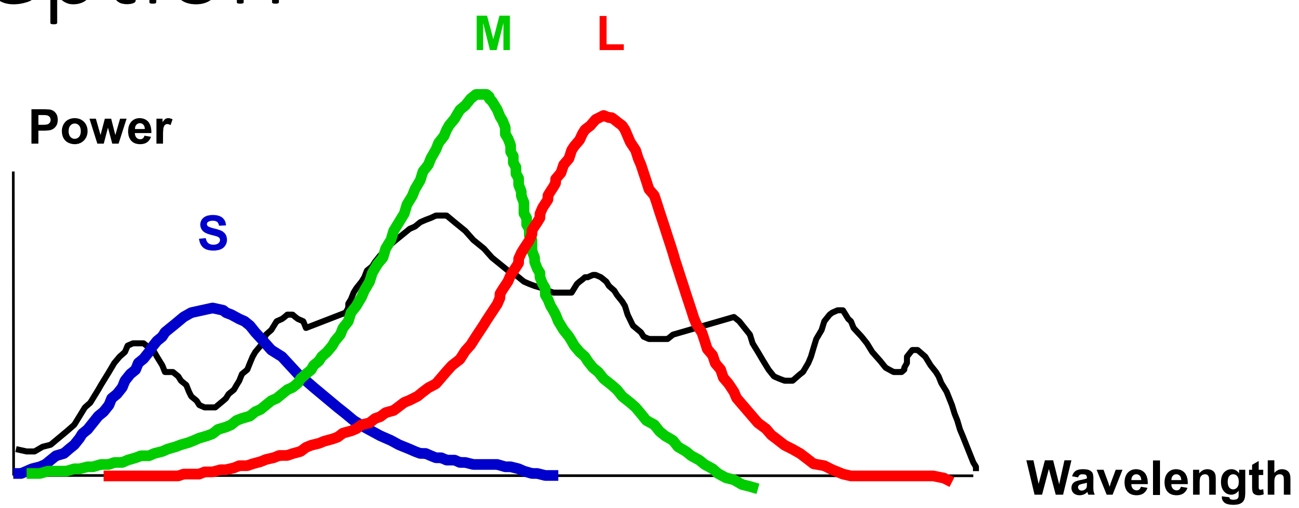






- Three types of cones
  - Each is sensitive in a different region of the spectrum, but regions overlap
    - Short (S) corresponds to blue
    - Medium (M) corresponds to green
    - Long (L) corresponds to red
  - Different sensitivities: we are more sensitive to green than red
    - varies from person to person (and with age)
  - Colorblindness—deficiency in at least one type of cone

Color perception



- Rods and cones act as filters on the spectrum
  - To get the output of a filter, multiply its response curve by the spectrum, integrate over all wavelengths
    - Each cone yields one number
  - Q: How can we represent an entire spectrum with 3 numbers?
  - A: We can't! Most of the information is lost.
    – As a result, two different spectra may appear indistinguishable by human eyes
    – Just like spatial "resolution", human eyes also have limited <u>"color resolution"</u>

Now, Digital Images!

Digital Image Formulation

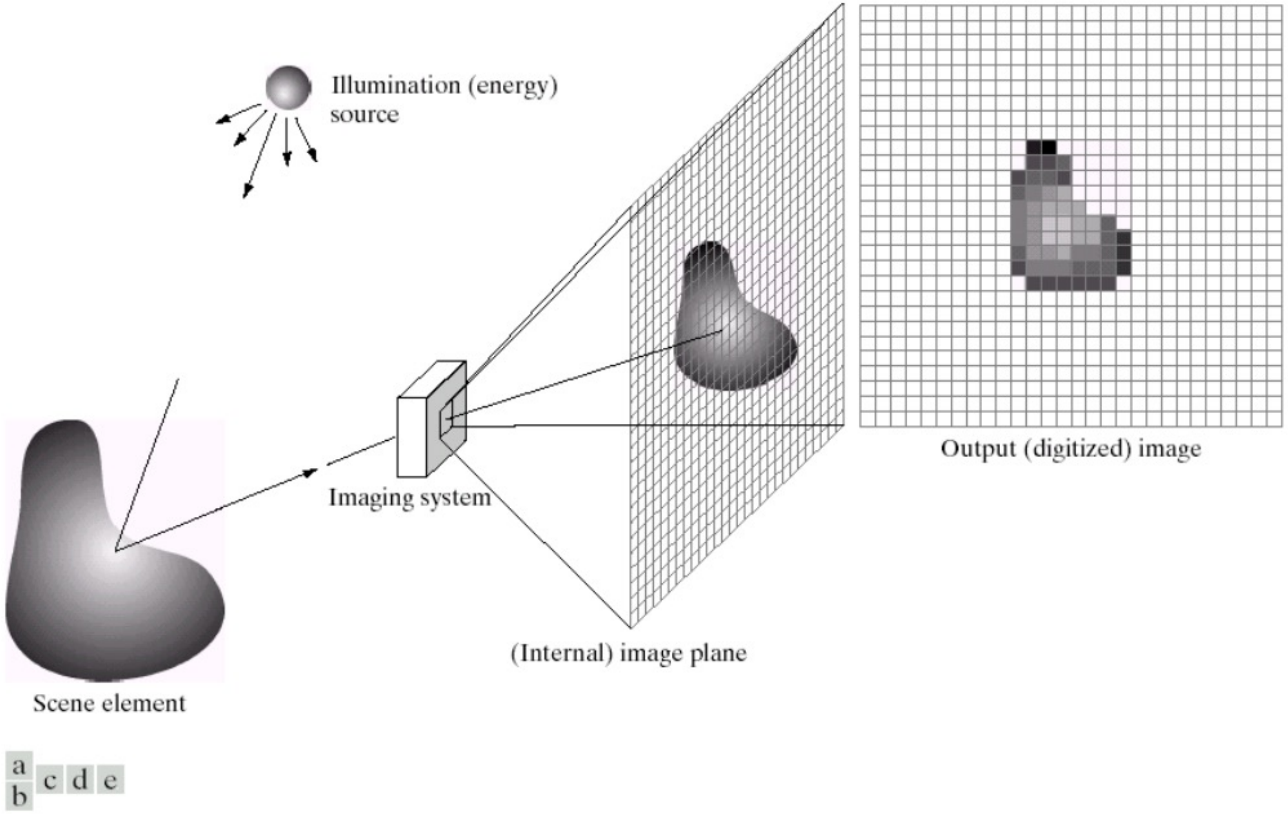
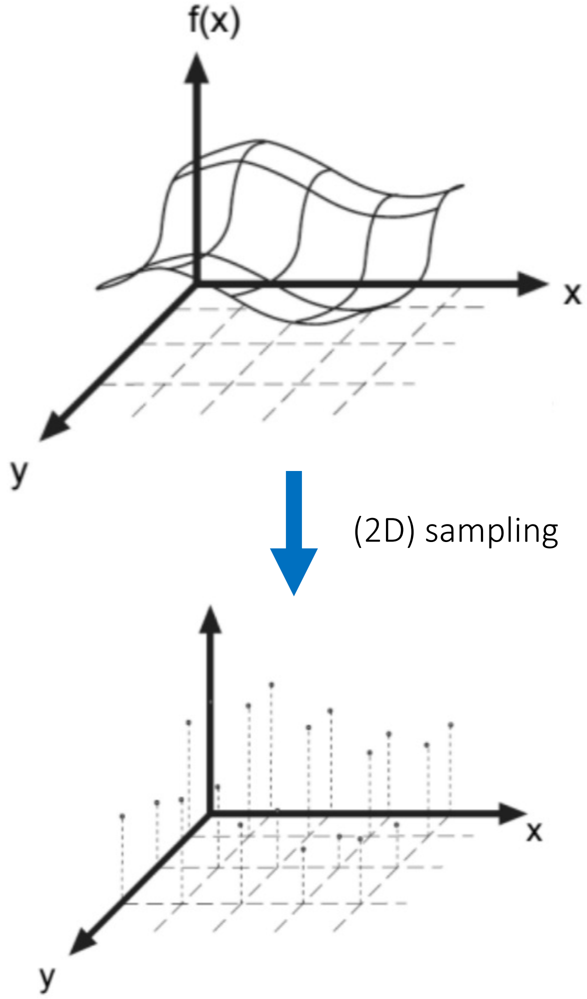


FIGURE 2.15 An example of the digital image acquisition process. (a) Energy ("illumination") source. (b) An element of a scene. (c) Imaging system. (d) Projection of the scene onto the image plane. (e) Digitized image.

Digital Image: Sampling of Continuous Visual World

- **Signal:** function depending on some variable with physical meaning
  - Our real visual world is always a continuous signal, do you agree?
- **Digital Image:** sampling of that function, dependent on **variables** of:
  - Two-axis: x-y coordinates
  - Three-axis: x-y-time (**video**)
- "Brightness/Color" is the **value** of the function for visible light, a.k.a. ***pixel***
  - Other possible function values in various "images": depth, heat…



Digital Image Representation

Binary

Gray scale

Color

Phil Noble / AP

Digital Images are Sampled and Quantized

- An image contains discrete number of pixels

  Pixel value:

  - "binary": 0 or 1
  - "grayscale" (or "intensity"): [0,255]
  - "color": RGB: [R, G, B]

[90, 0, 53]

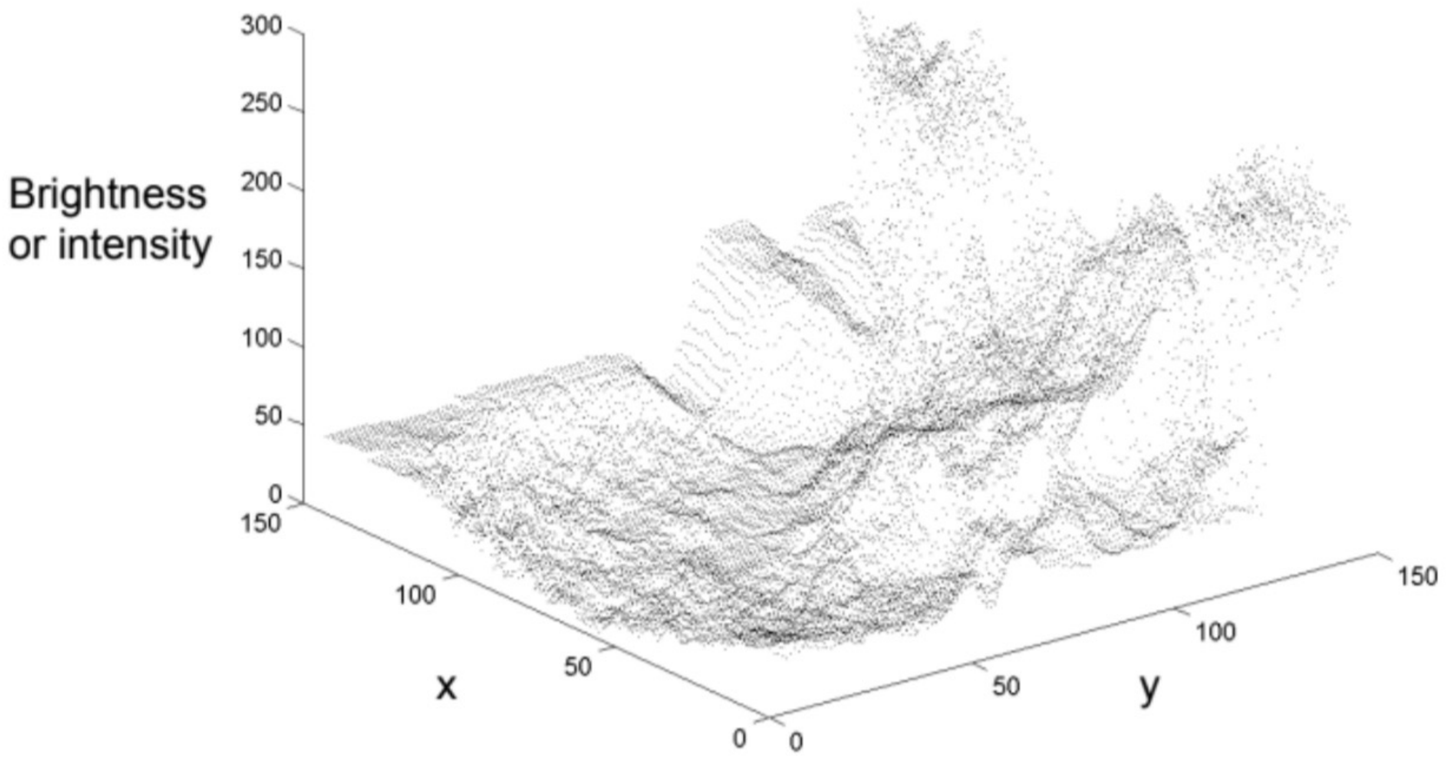
Phil Noble / AP

[249, 215, 203]

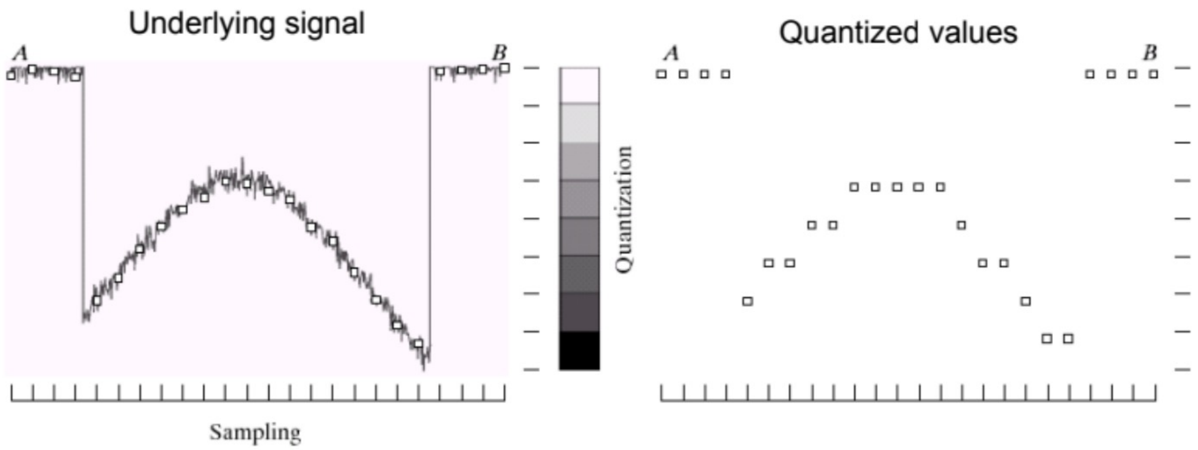
[213, 60, 67]

Samples = pixels
Quantization = number of bits per pixel

Grayscale Digital Image as “Height Map”



Digital Values can be Quantized Further



8 bit – 256 levels   4 bit – 16 levels   2 bit – 4 levels   1 bit – 2 levels

- We often call this *bit depth*
- For photography, this is also related to *dynamic range*

Is An Image Just A Matrix?

```
>>> from matplotlib import pyplot as p
>>> I = r.rand(256,256);
>>> p.imshow(I);
>>> p.show();
```

Is it an image?

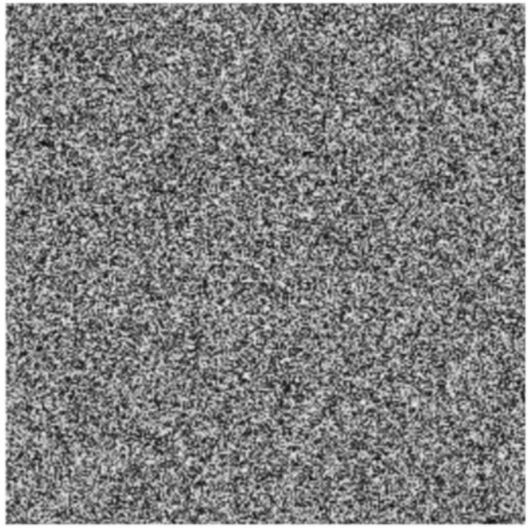


Image is a high-structured 2D signal!!

- *(piece-wise) smoothness, self-similar patterns (fractal), "reducible" to the composition of basic units (subspace)…*
  - *A wealth of "image priors", although not always explicit*
- It takes great luck for a 2D matrix to be an image!

8bit = 256 values ^ 65,536

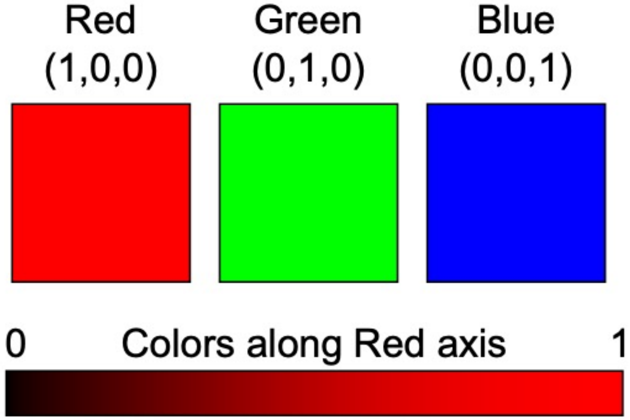
Computer vision makes sense of an extremely high-dimensional space

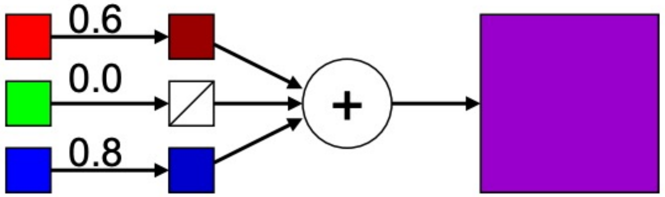
- *Using **low-dimensional**, explainable models*

"Natural Image Manifold"

- The distribution of natural images (or patches) is similar to the mass distribution in the universe, where there are high-density and low-density areas
- This "manifold" has to be highly nonlinear, inherently **low-dimensional**, and **locally smooth** … (do you understand why?)

Color Image: Three-Channel RGB Model



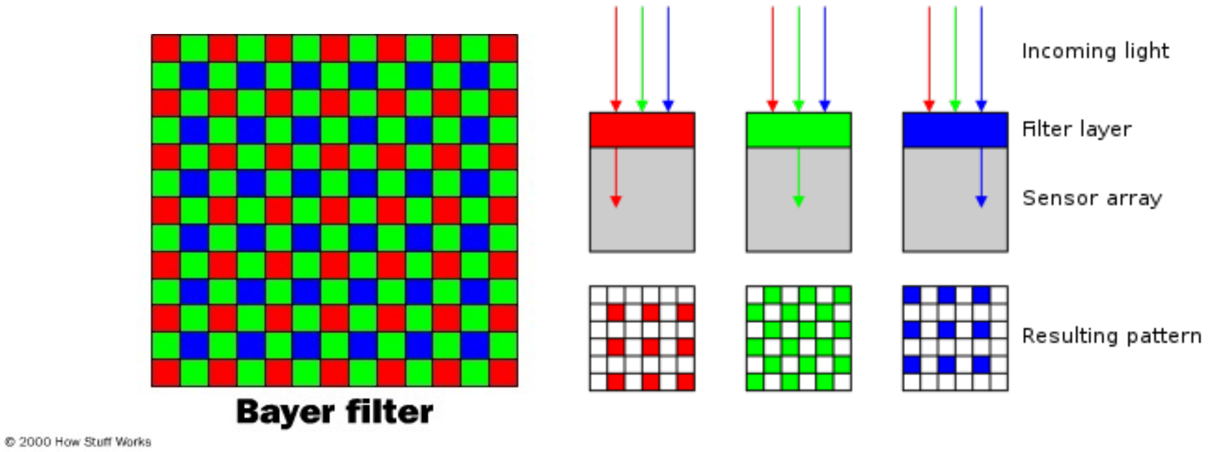


Universal, yet non-perceptual…

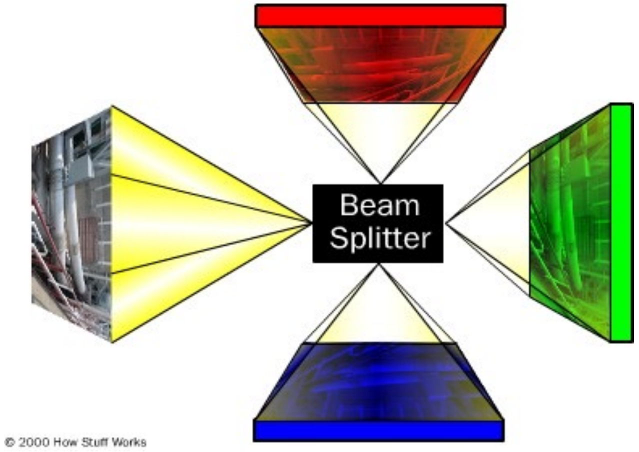
- *The three channels are strongly correlated!*

How Color Cameras Work?

- 1 CCD cameras
  - A **Bayer** pattern is placed in front of the CCD
  - A **Demosaicing** process reads the pixels in a region and computes color and intensity



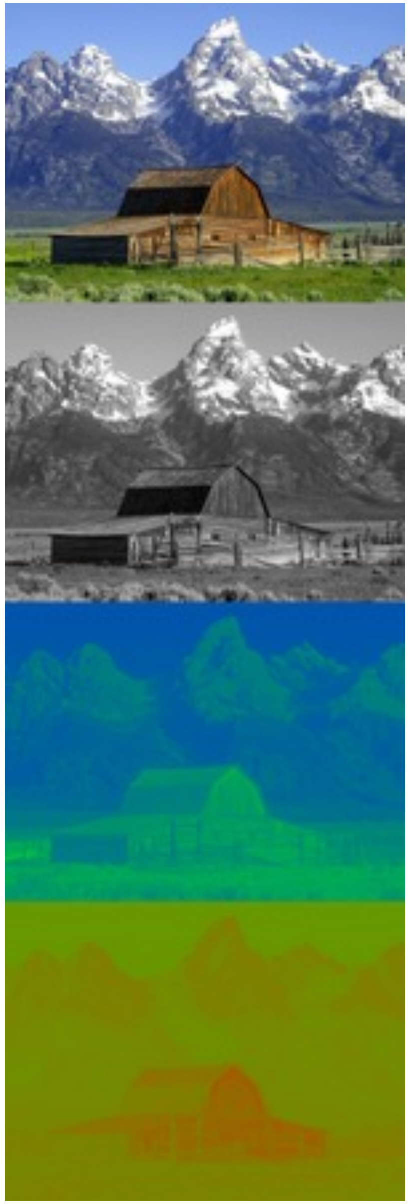
- 3 CCD camera use a beam splitter and 3 separate CCDs
  - higher color fidelity
  - needs lots of light
  - requires careful alignment of ccds



Color Space Representations

$$\begin{bmatrix} Y' \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.14713 & -0.28886 & 0.436 \\ 0.615 & -0.51499 & -0.10001 \end{bmatrix} \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix},$$

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.13983 \\ 1 & -0.39465 & -0.58060 \\ 1 & 2.03211 & 0 \end{bmatrix} \begin{bmatrix} Y' \\ U \\ V \end{bmatrix}.$$



RGB system (most common):
linear additive color mixing

YUV system (popular in color TV):

- Y stands for the luma component (brightness)
- U and V are the chrominance (color) components

Color Space Representations

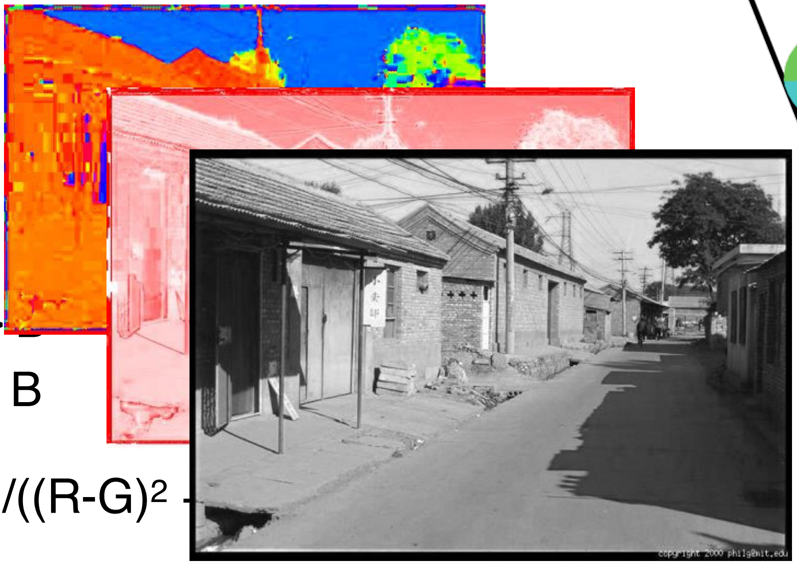
- **HSI** (hue, saturation, intensity) is a **nonlinear** representation of color space. Note the non-uniform treatment of color
- Popular in computer graphics researchers to more closely align with the way human vision perceives color-making attributes

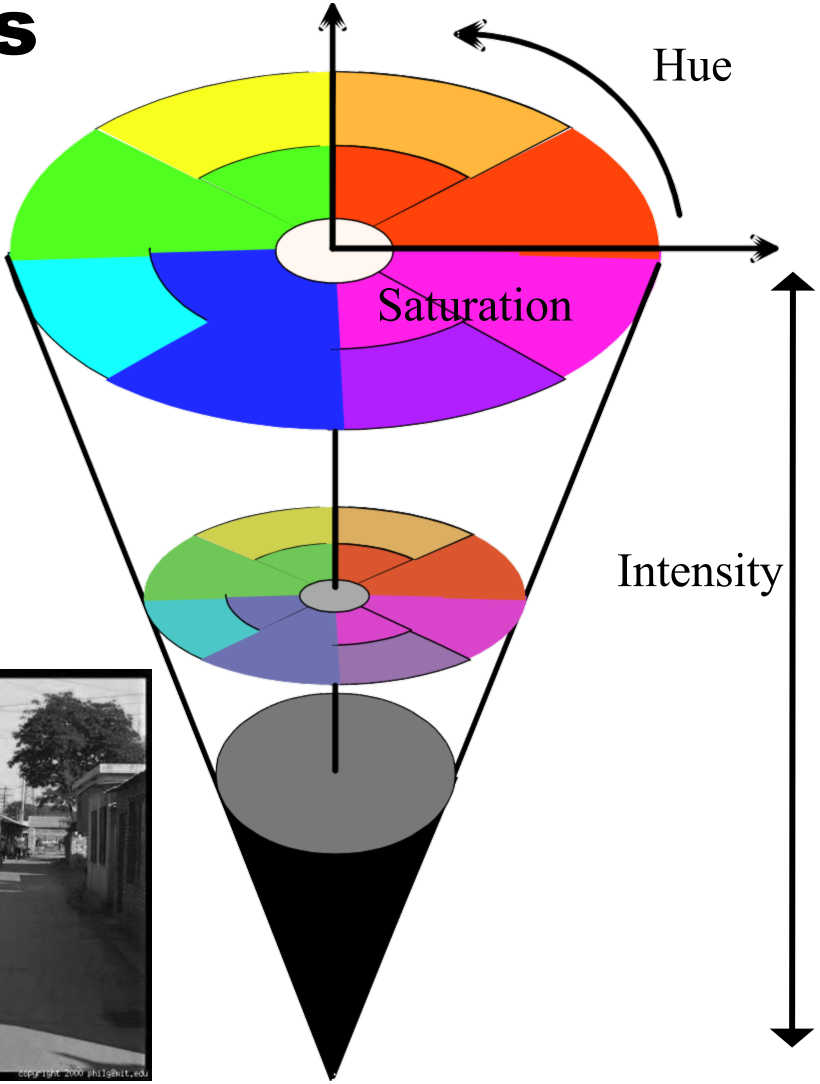
$I = (R+G+B)/3$

G > B

G < B

-B)/((R-G)² -





Bit Color Depth

1 bit

2 bits

4 bits

8 bits

24 bits

1= ON 0 =OFF

00 01 10 11
Different shades of gray

24-bit ***TrueColor*** can represent more than 16.7 million unique colors

- More colors than the human eye can distinguish!

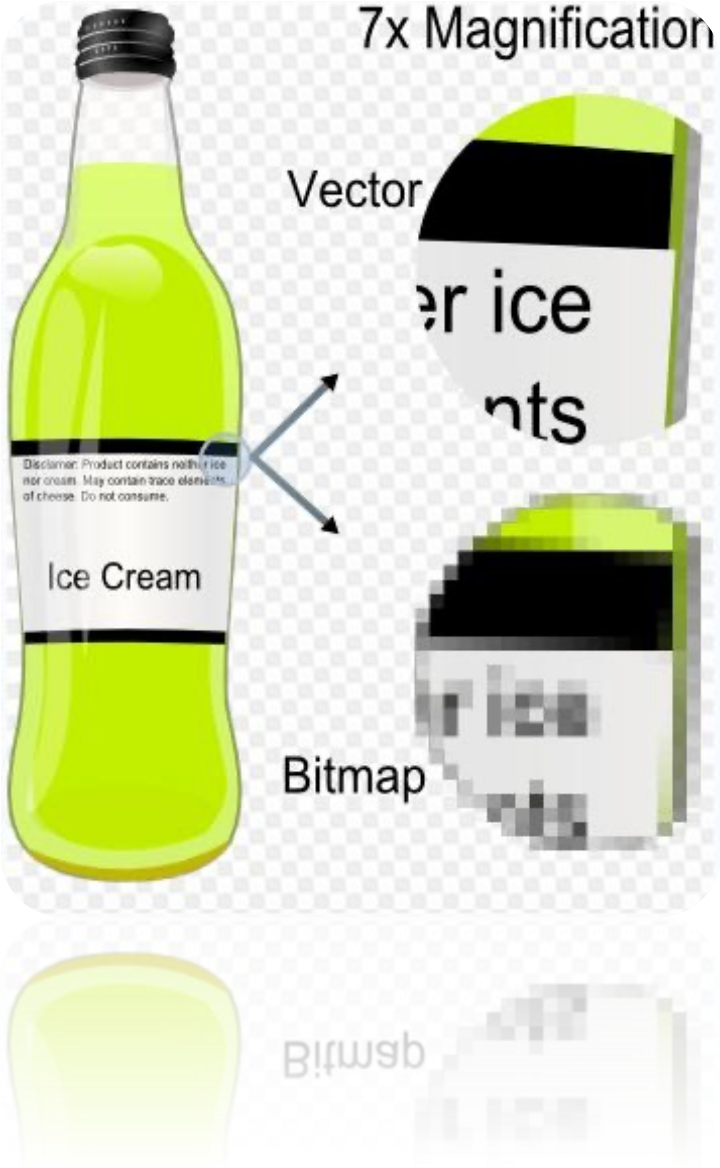
Video: Frame-by-Frame Image Sequence

30 frames/second

Raster vs Vector Graphics

- Raster graphics: made up of pixels
  - Resolution dependent
  - Cannot be scaled without losing quality
  - Can represent photo realistic elements better than vector graphics

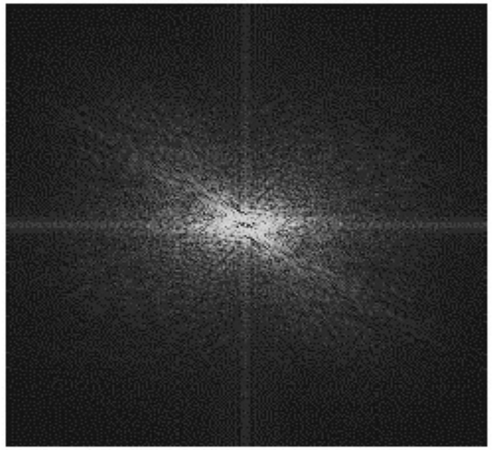
- Vector graphics: geometric primitives, composed of paths
  - Mathematical equations
  - Resolution independent
  - Can be scaled to any size without losing quality
  - Best for cartoon-like images
  - 3D modeling



Spatial and Frequency Domains

- Spatial domain
  - refers to planar region of intensity values at time *t*

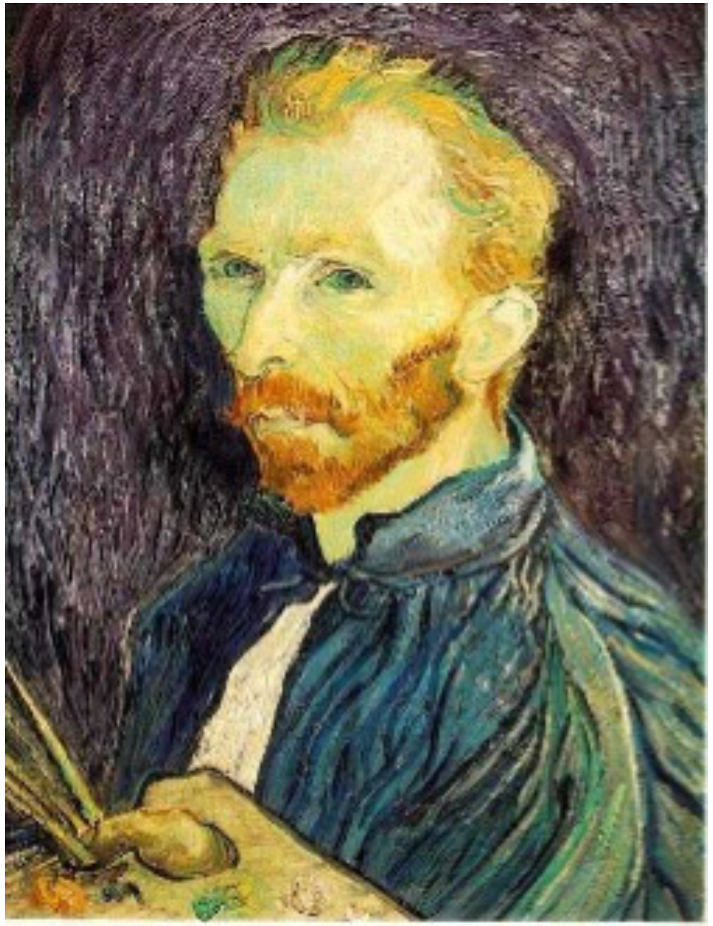
- **Frequency domain**
  - think of each color plane as a sinusoidal function of changing intensity values
  - refers to organizing pixels according to their changing intensity (frequency)




This image is too big to fit on the screen.
How would you reduce it to half its size?

Naïve image downsampling

Throw away half the rows and columns



1/2

delete even rows
delete even columns

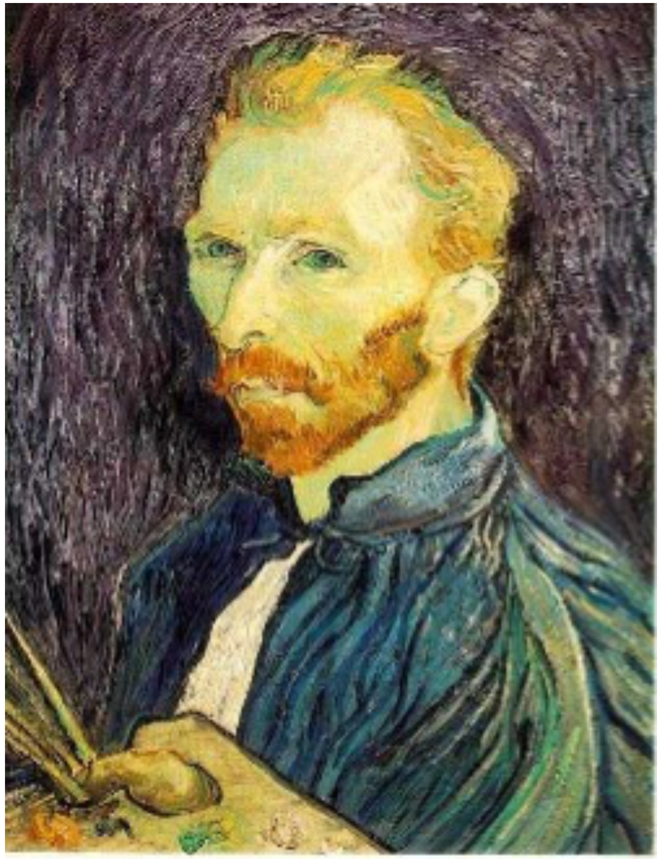
1/4

delete even rows
delete even columns

1/8

What is the problem with this approach?

Naïve image downsampling



1/2

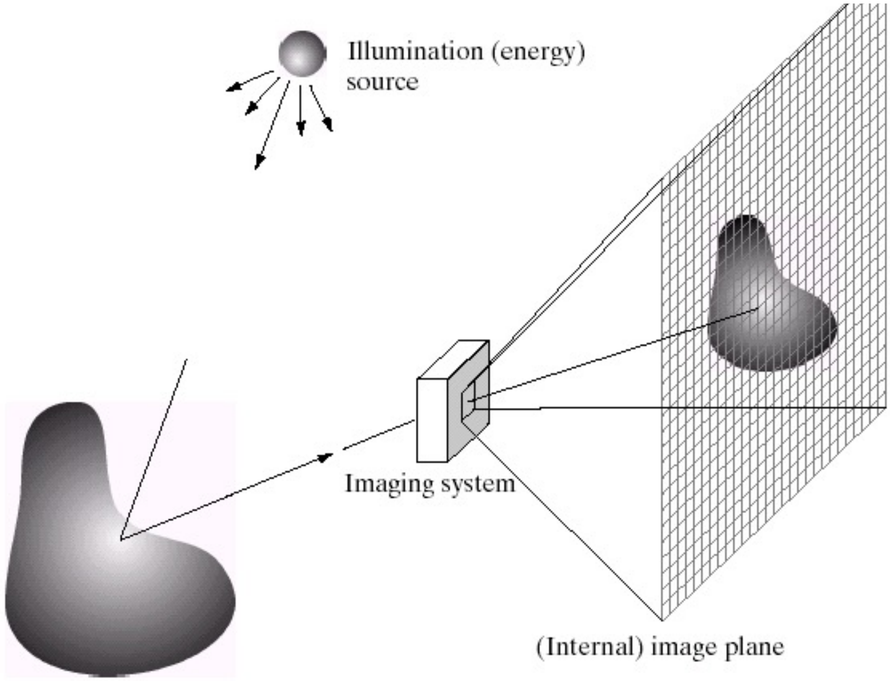
1/4 (2x zoom)

1/8 (4x zoom)

What is the 1/8 image so pixelated (and do you know what this effect is called)?

The Devil of Digital Sampling:
Aliasing

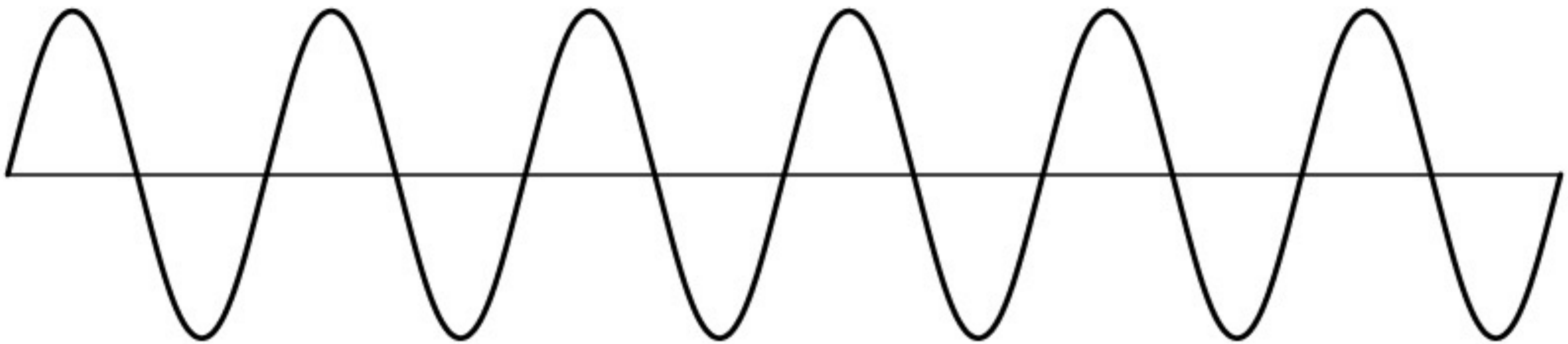
Reminder



Images are a *discrete*, or *sampled*, representation of a *continuous* world

Sampling

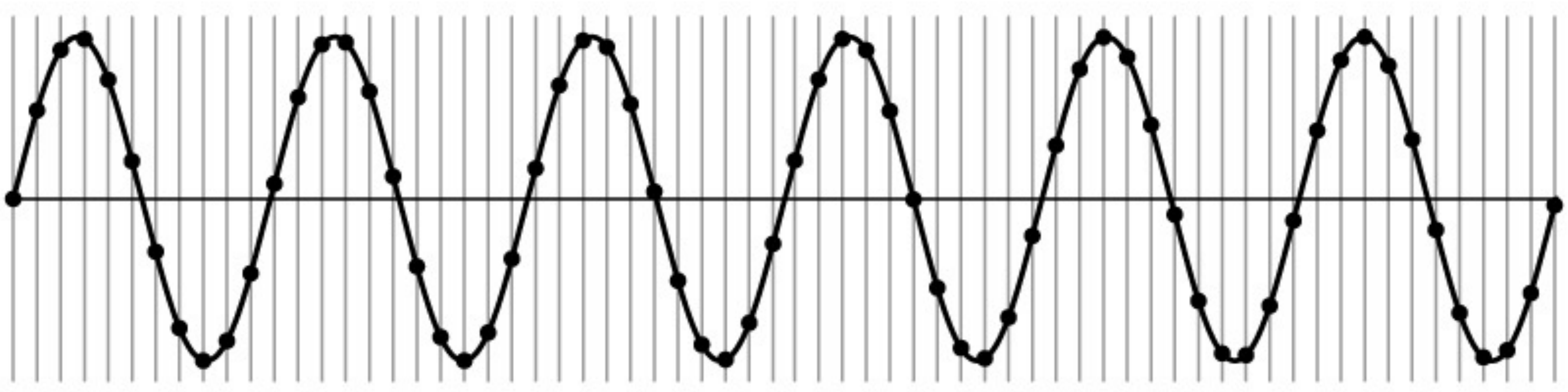
Very simple example: a sine wave



How would you discretize this signal?

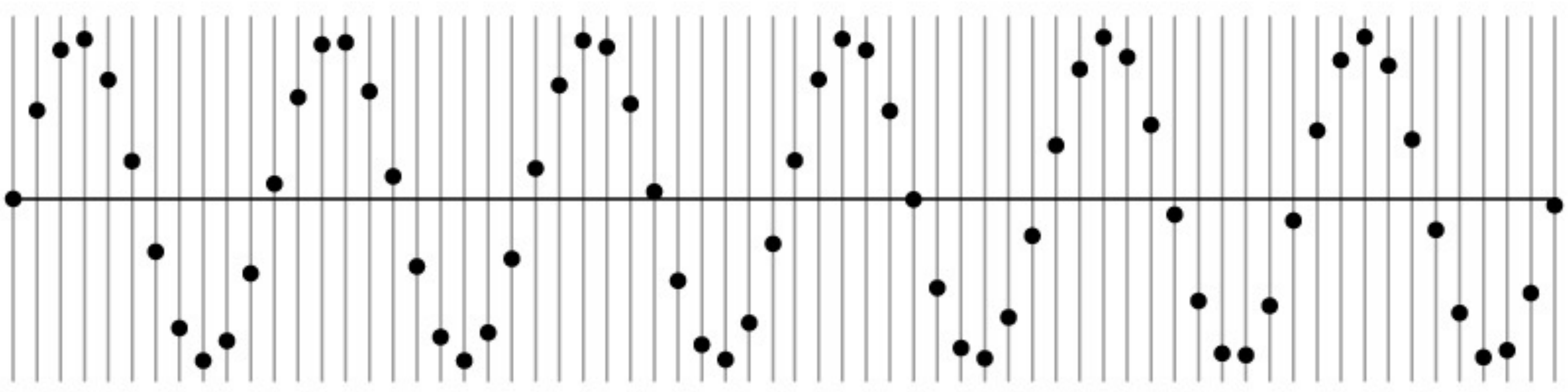
Sampling

Very simple example: a sine wave



Sampling

Very simple example: a sine wave

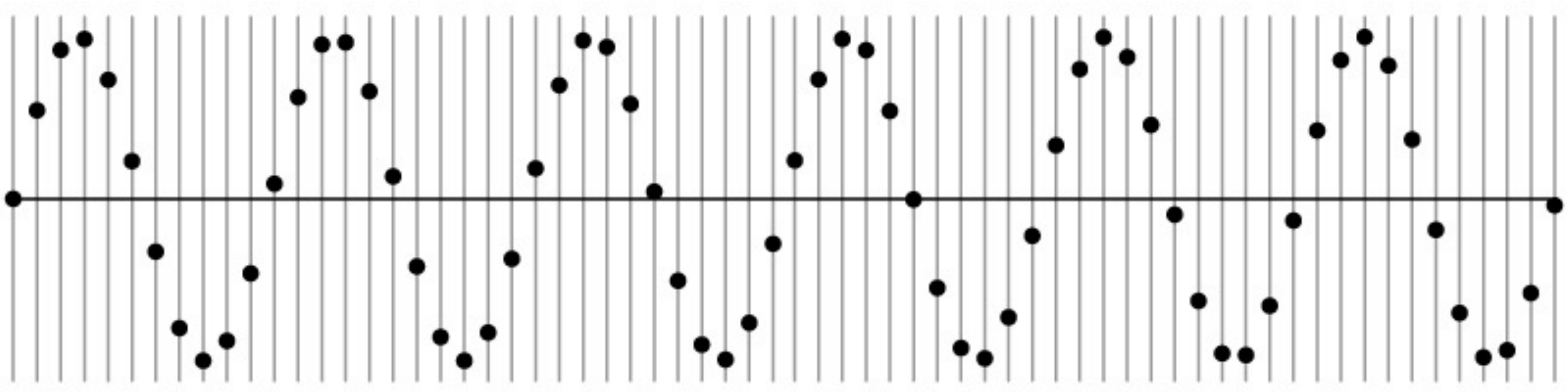


How many samples should I take?

Can I take as *many* samples as I want?

Sampling

Very simple example: a sine wave

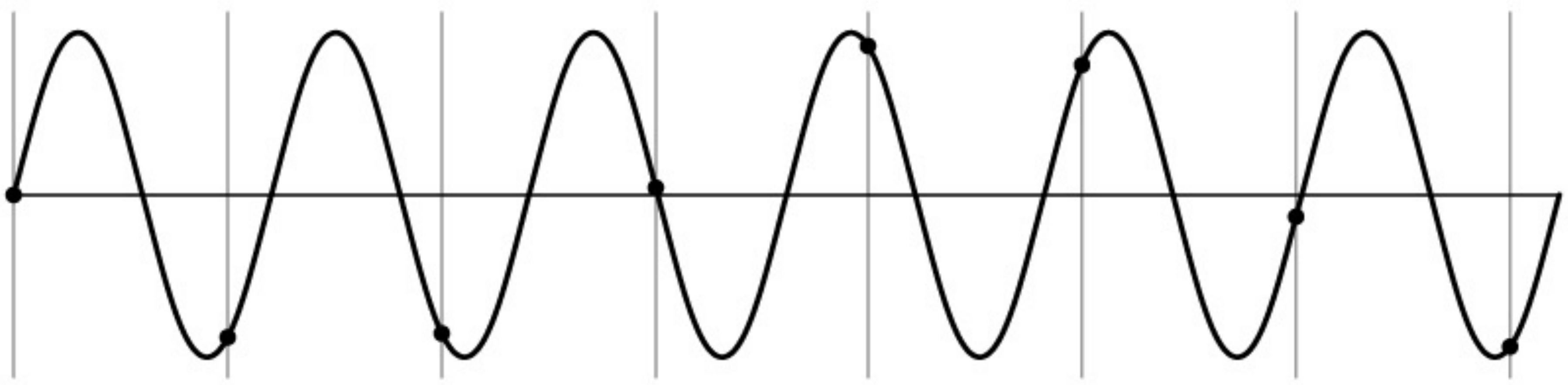


How many samples should I take?

Can I take as *few* samples as I want?

Undersampling

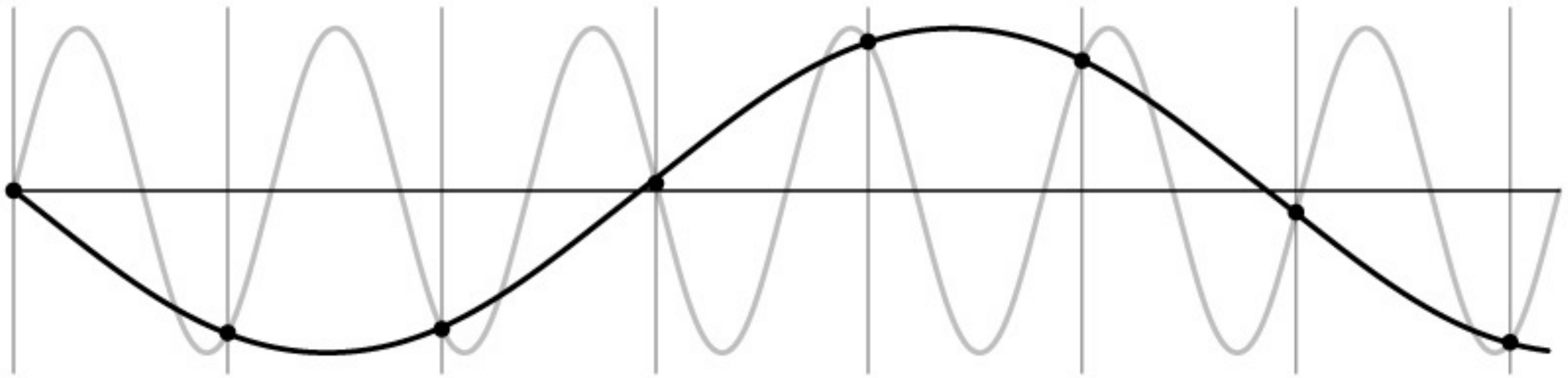
Very simple example: a sine wave



Unsurprising effect: information is lost.

Undersampling

Very simple example: a sine wave

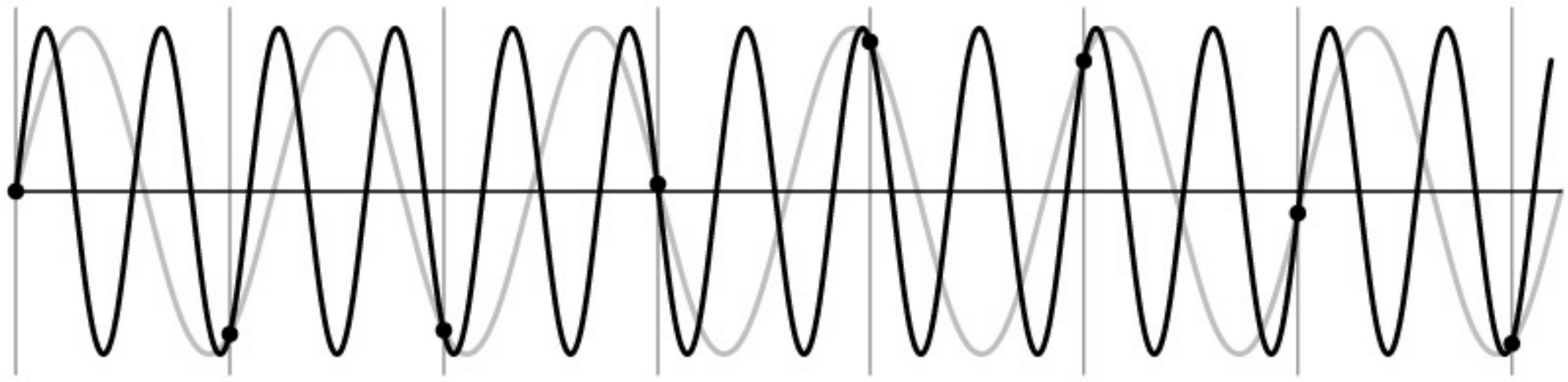


Unsurprising effect: information is lost.

Surprising effect: can confuse the signal with one of *lower* frequency.

Undersampling

Very simple example: a sine wave

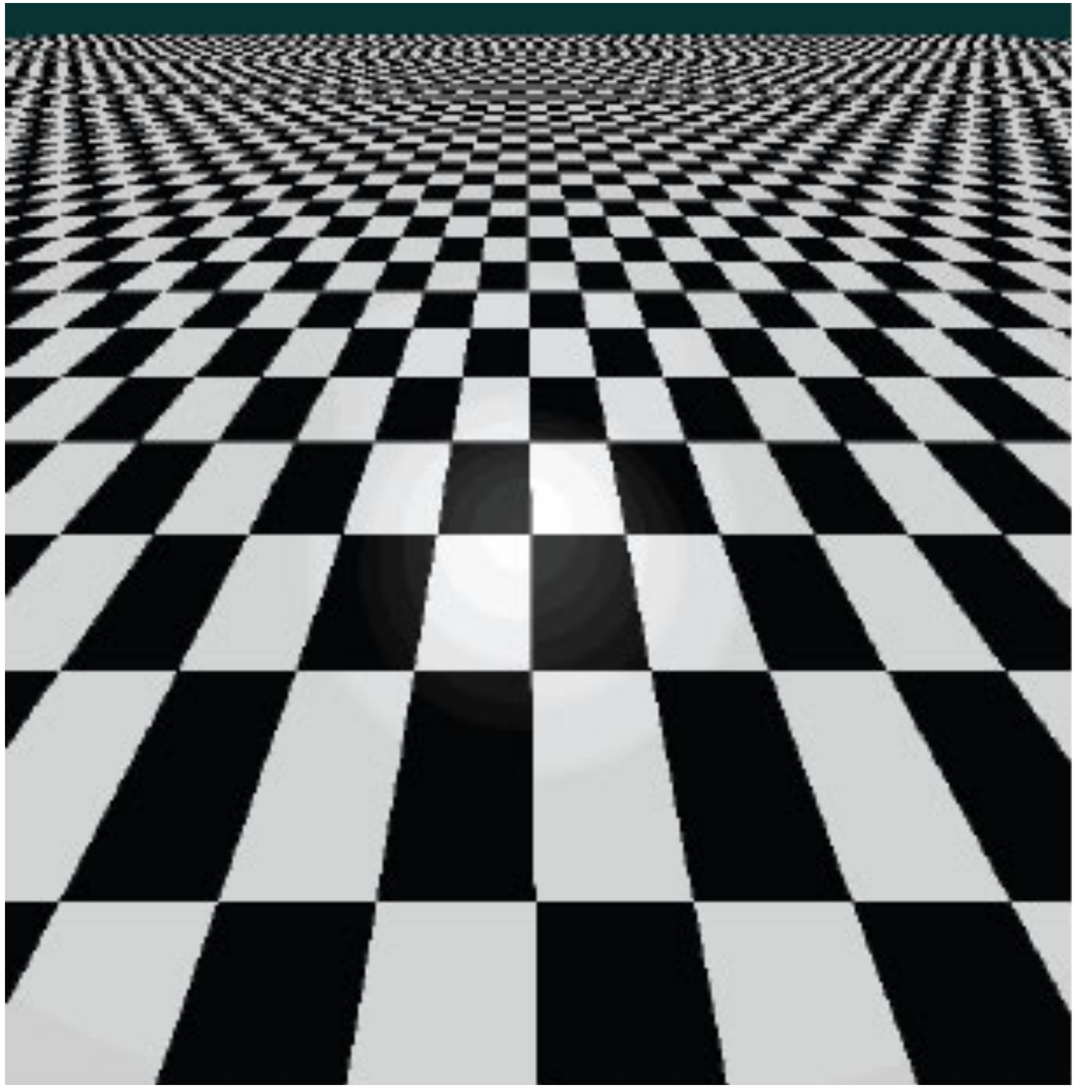


Unsurprising effect: information is lost.

Surprising effect: can confuse the signal with one of *lower* frequency.

Note: we could always confuse the signal with one of *higher* frequency.

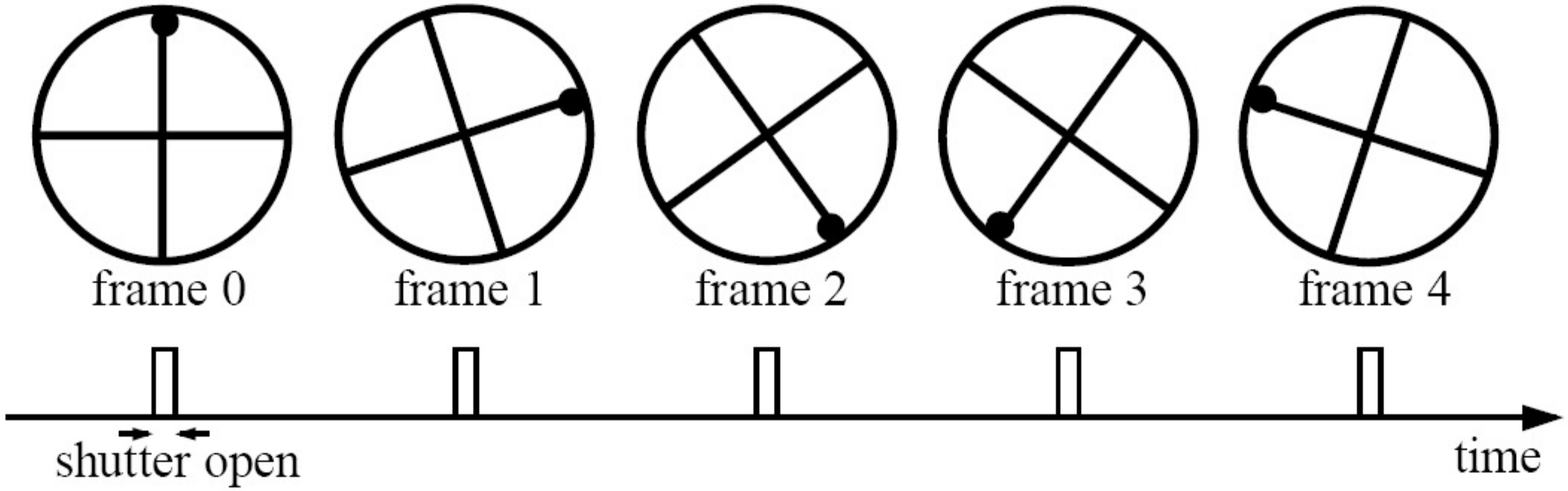
Aliasing in textures



Temporal aliasing

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)

Wagon wheel effect

Anti-aliasing

How would you deal with aliasing?

Anti-aliasing

How would you deal with aliasing?

Approach 1: Oversample the signal

Anti-aliasing

How would you deal with aliasing?

Approach 1: Oversample the signal

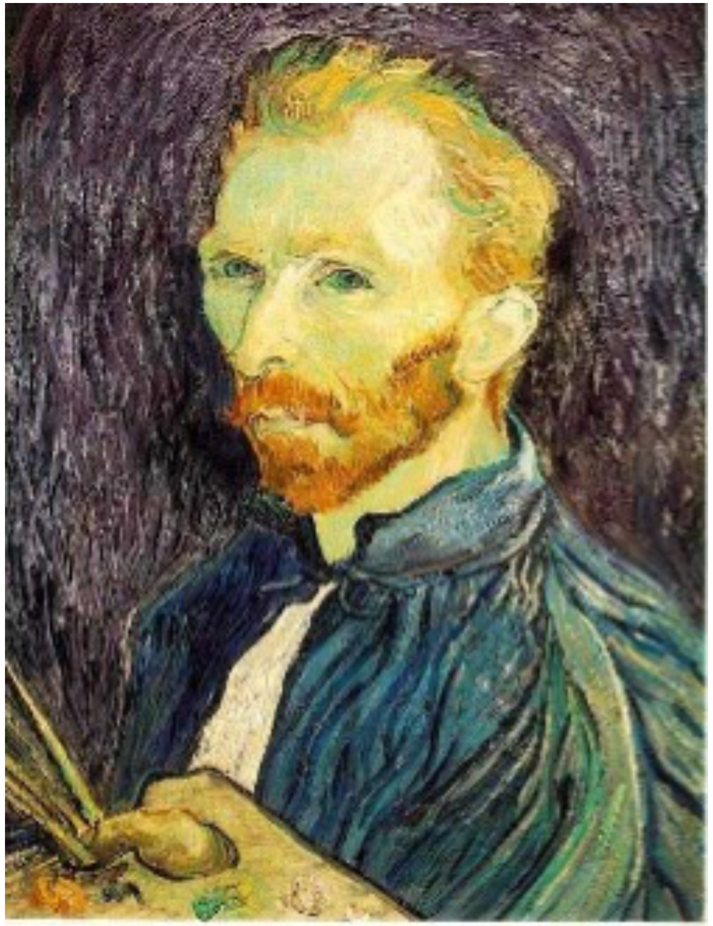
Approach 2: Smooth the signal

- Remove some of the detail effects that cause aliasing.
- Lose information, but better than aliasing artifacts.

How would you smooth a signal?

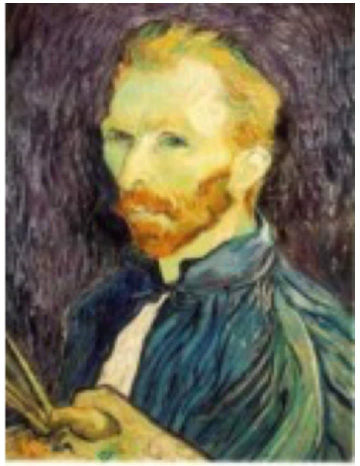
Better image downsampling

Apply a smoothing filter first, then throw away half the rows and columns



1/2

Gaussian filter
delete even rows
delete even columns

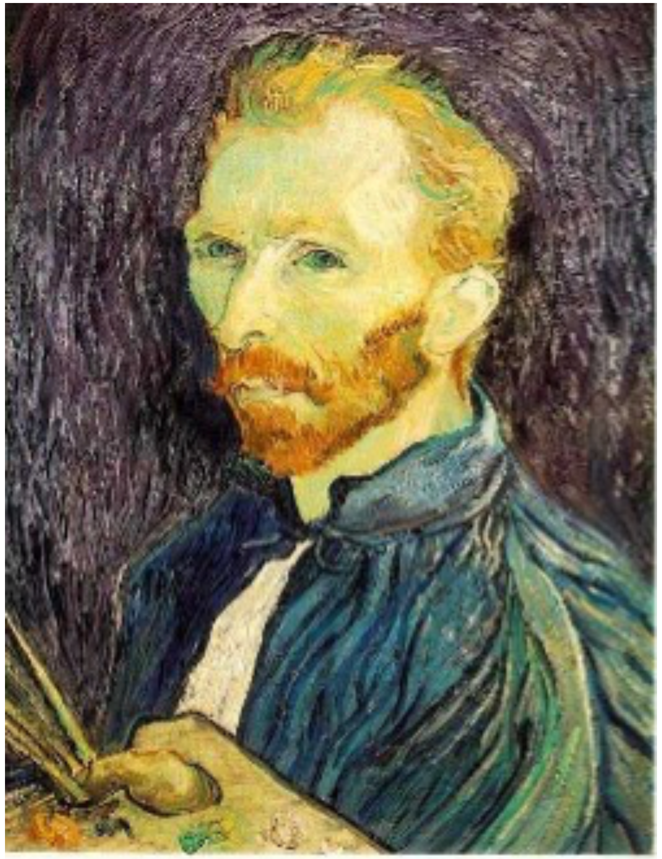


1/4

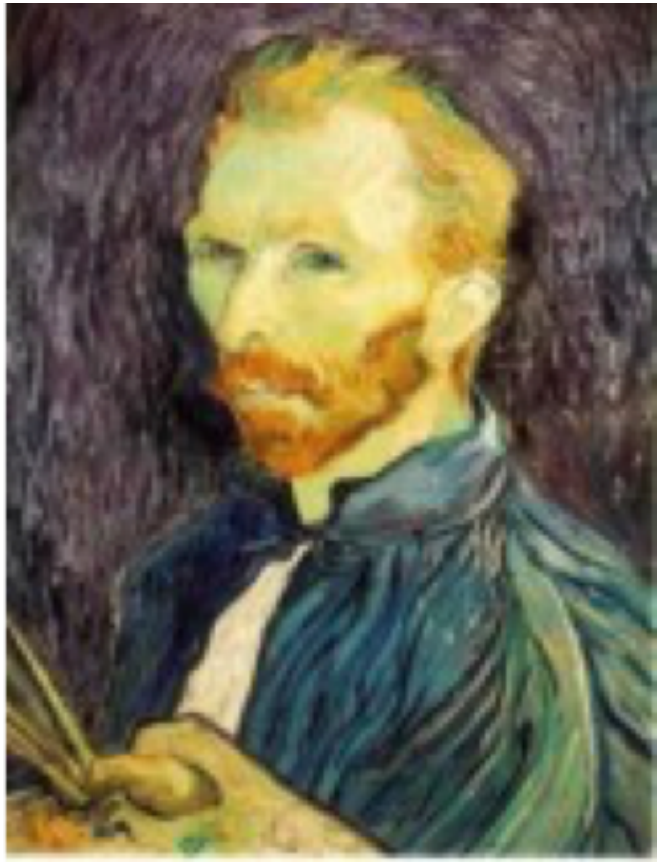
Gaussian filter
delete even rows
delete even columns

1/8

Better image downsampling



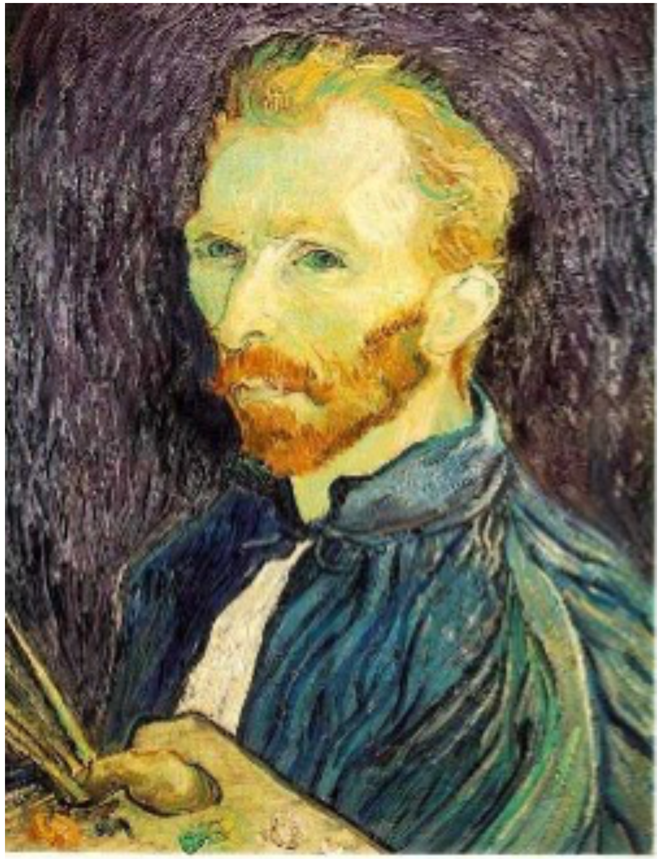
1/2



1/4 (2x zoom)

1/8 (4x zoom)

Naïve image downsampling



1/2

1/4 (2x zoom)

1/8 (4x zoom)

Anti-aliasing

Question 1: How much smoothing do I need to do to avoid aliasing?

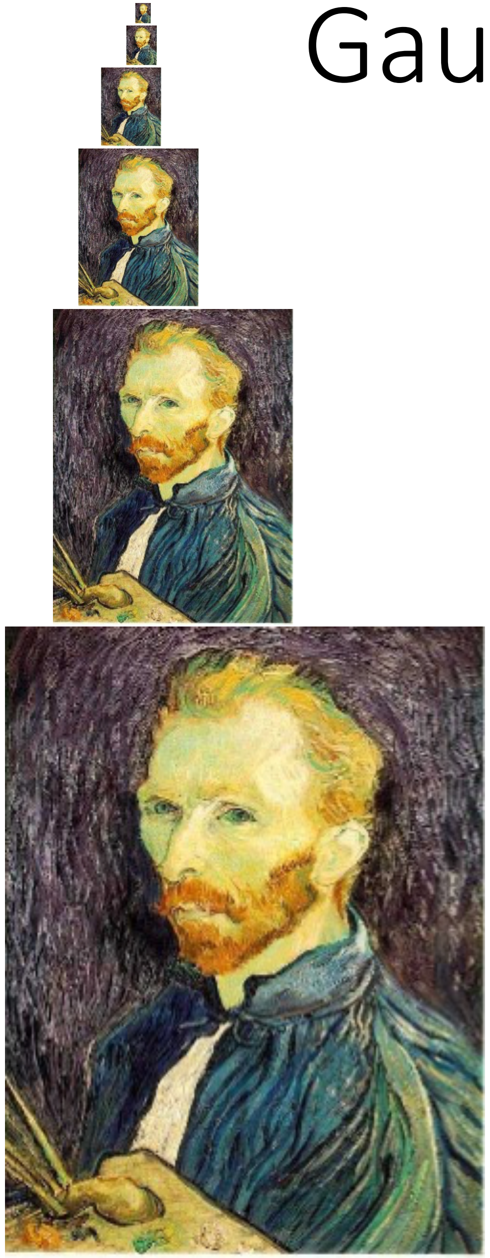
Question 2: How many samples do I need to take to avoid aliasing?

Answer to both: Enough to reach the Nyquist limit.

We'll see what this means soon.

Image pyramid: Gaussian and Laplacian

Gaussian image pyramid



The name of this sequence of subsampled images

Constructing a Gaussian pyramid

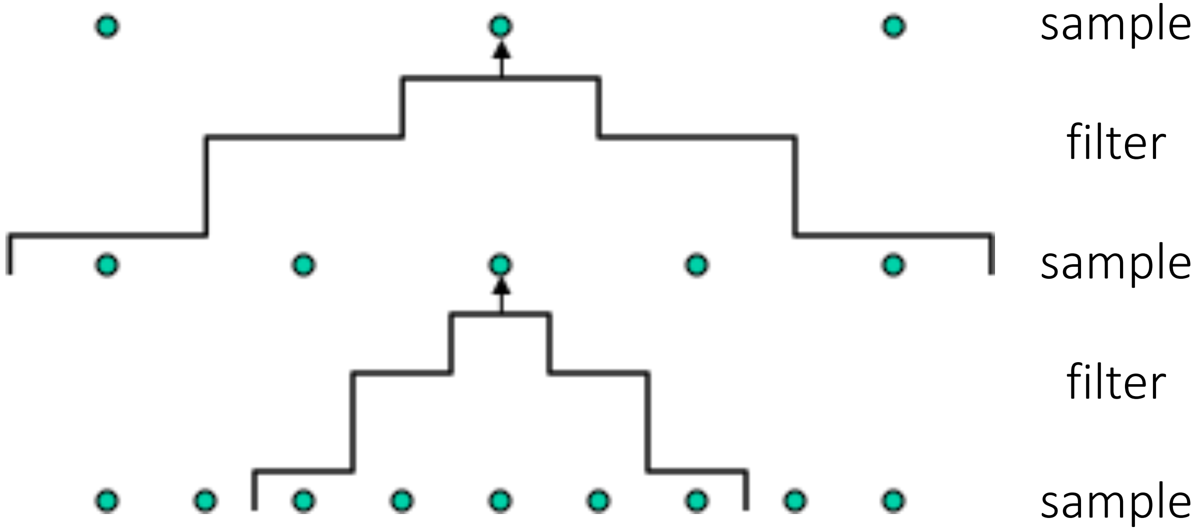
Algorithm

repeat:

filter

subsample

until min resolution reached



Question: How much bigger than the original image is the whole pyramid?

Constructing a Gaussian pyramid

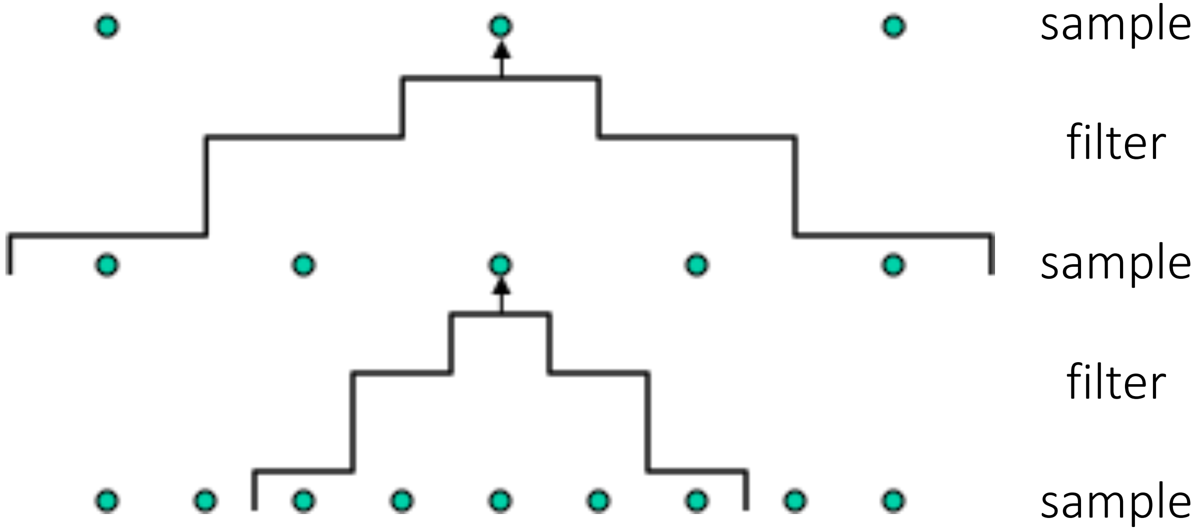
Algorithm

repeat:

filter

subsample

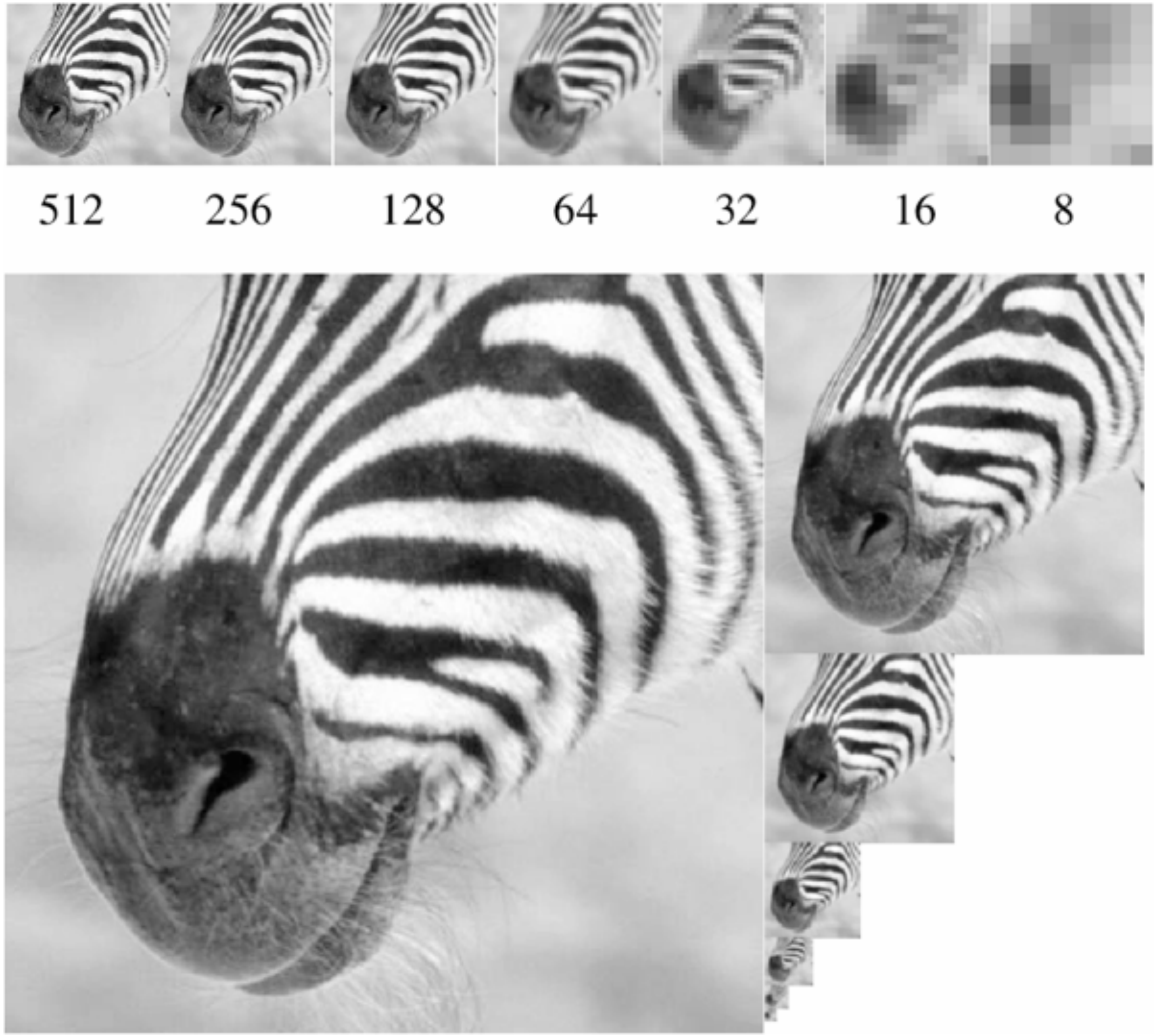
until min resolution reached



Question: How much bigger than the original image is the whole pyramid?

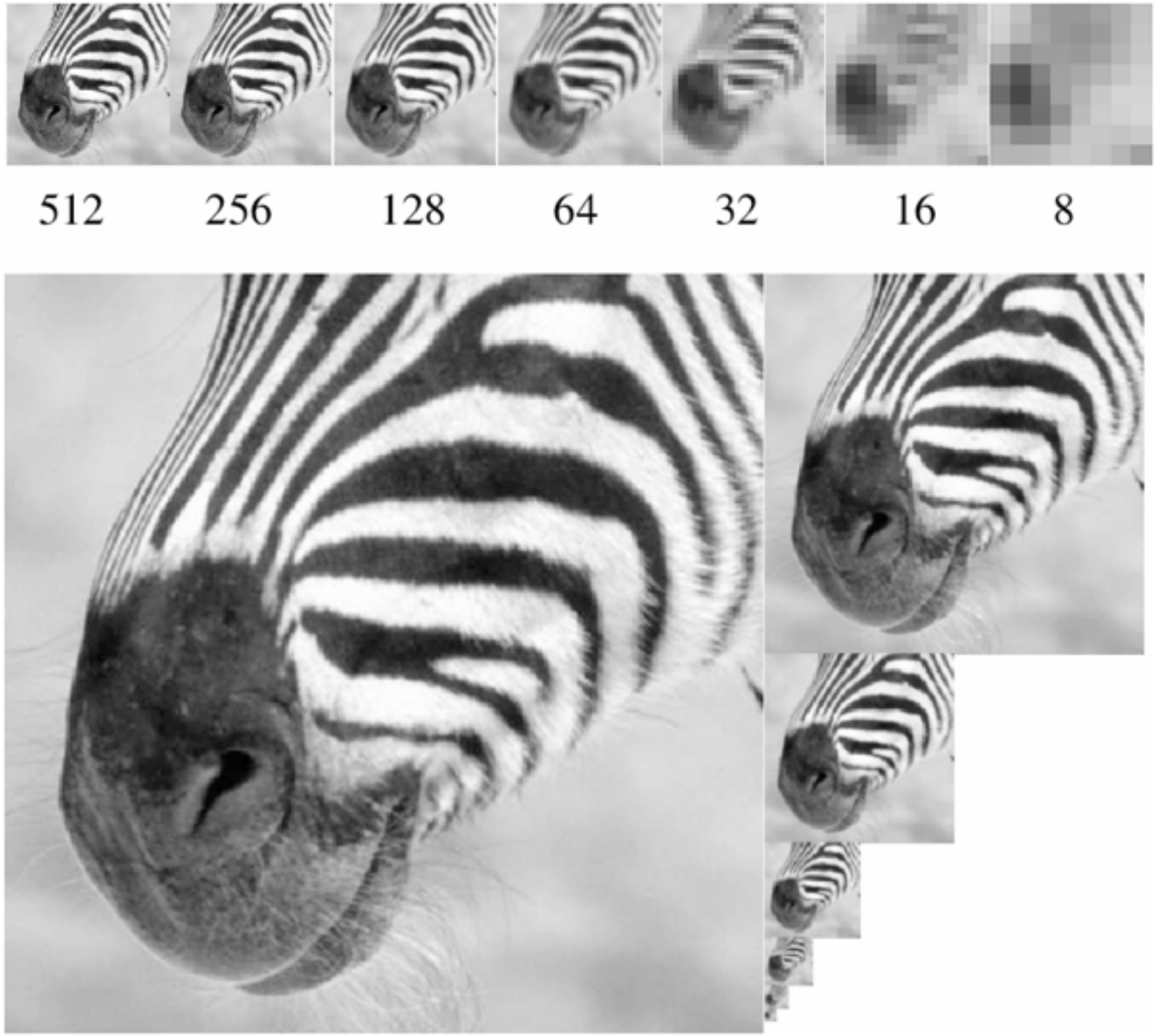
Answer: Just 4/3 times the size of the original image! (How did I come up with this number?)

Some properties of the Gaussian pyramid



What happens to the details of the image?

Some properties of the Gaussian pyramid

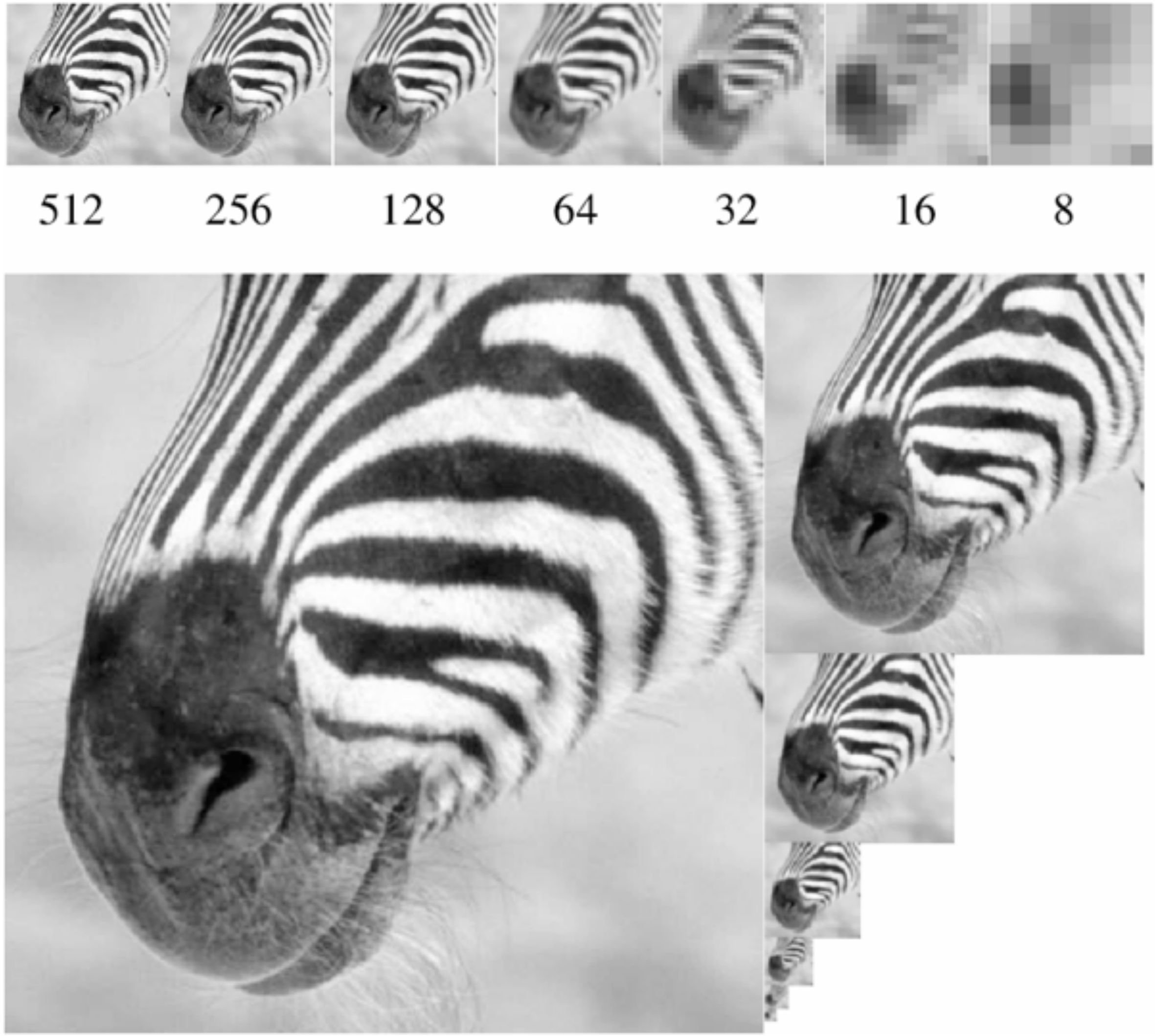


What happens to the details of the image?

- They get smoothed out as we move to higher levels.

What is preserved at the higher levels?

Some properties of the Gaussian pyramid



What happens to the details of the image?

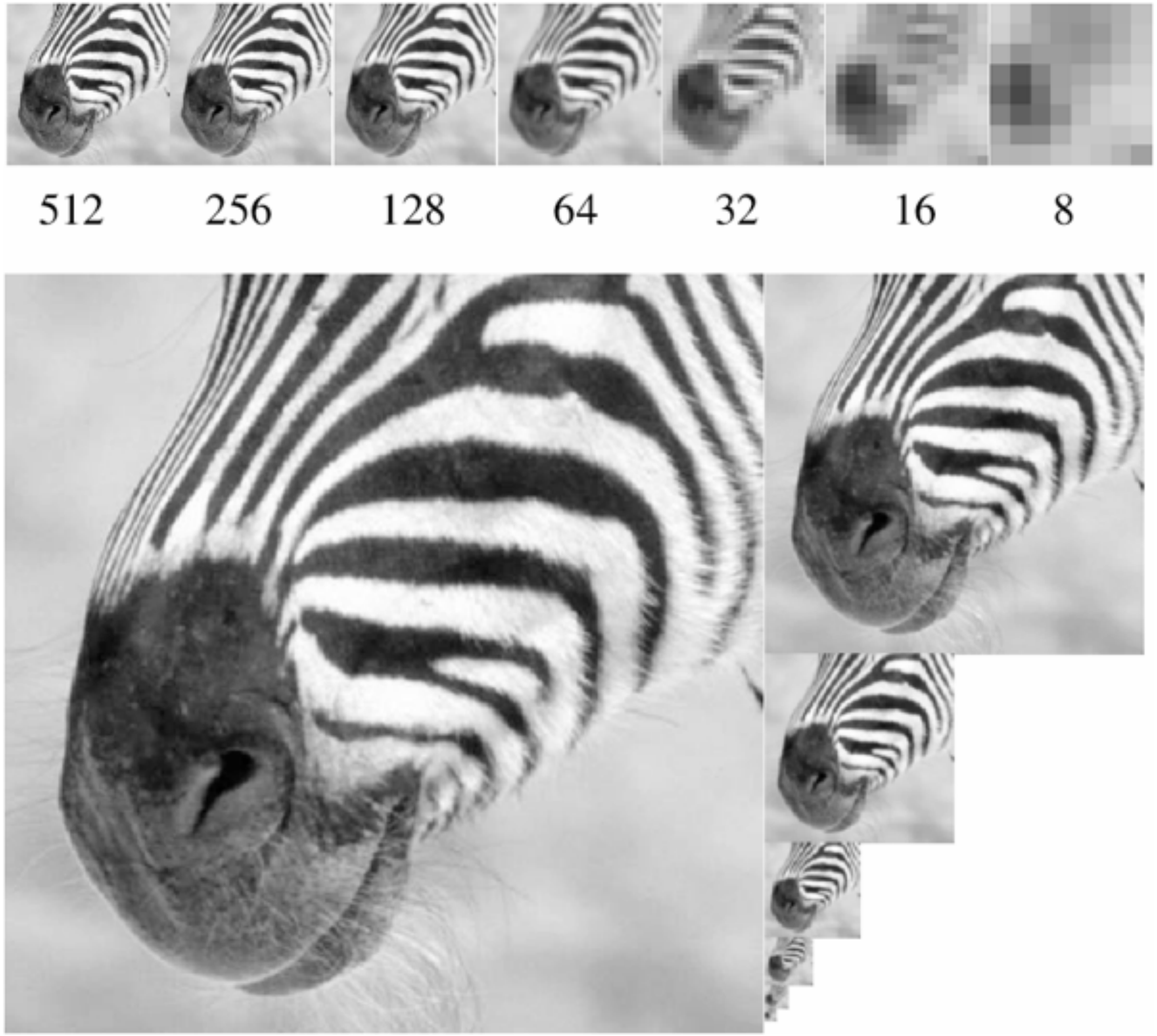
- They get smoothed out as we move to higher levels.

What is preserved at the higher levels?

- Mostly large uniform regions in the original image.

How would you reconstruct the original image from the image at the upper level?

Some properties of the Gaussian pyramid



What happens to the details of the image?

- They get smoothed out as we move to higher levels.

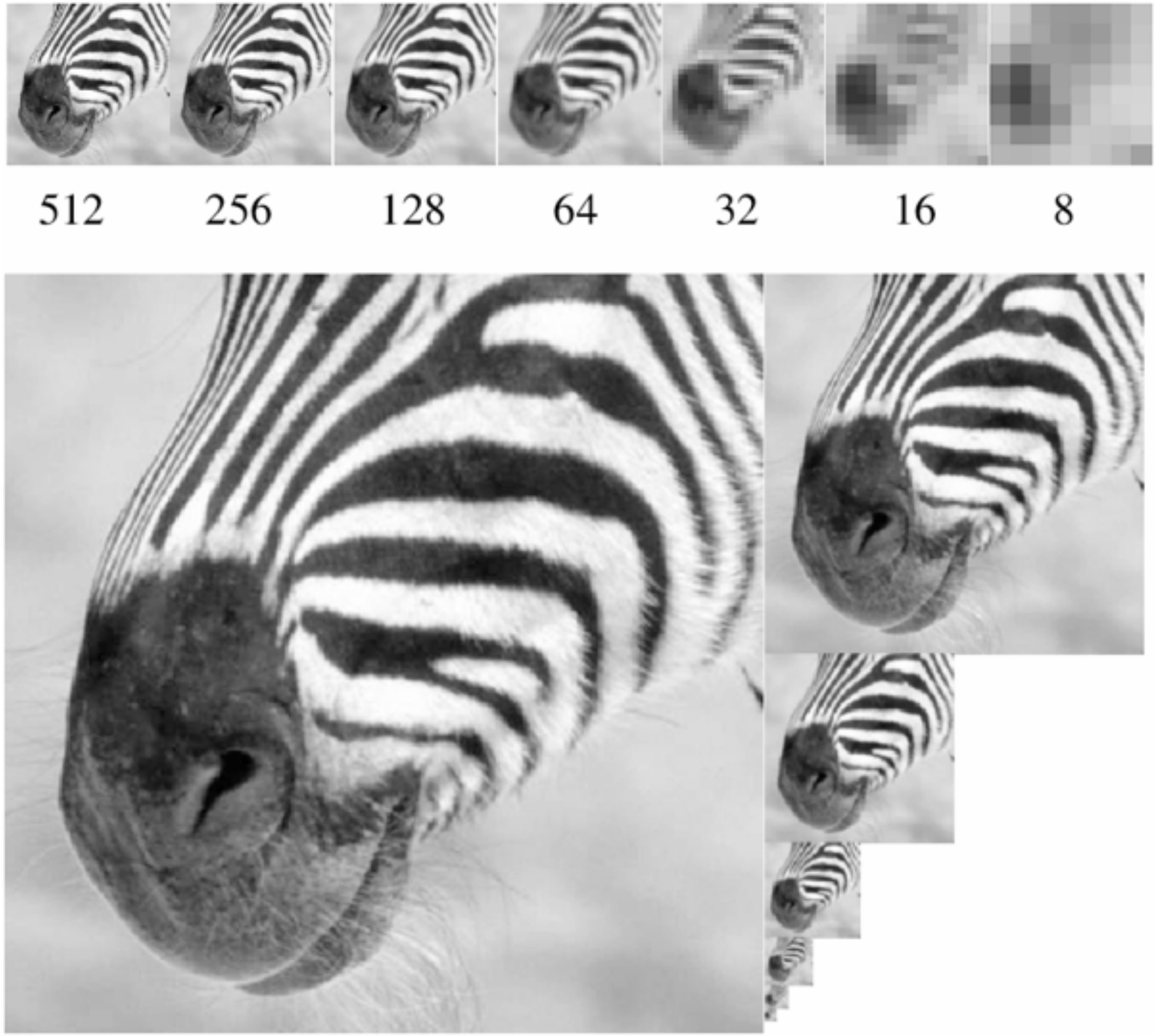
What is preserved at the higher levels?

- Mostly large uniform regions in the original image.

How would you reconstruct the original image from the image at the upper level?

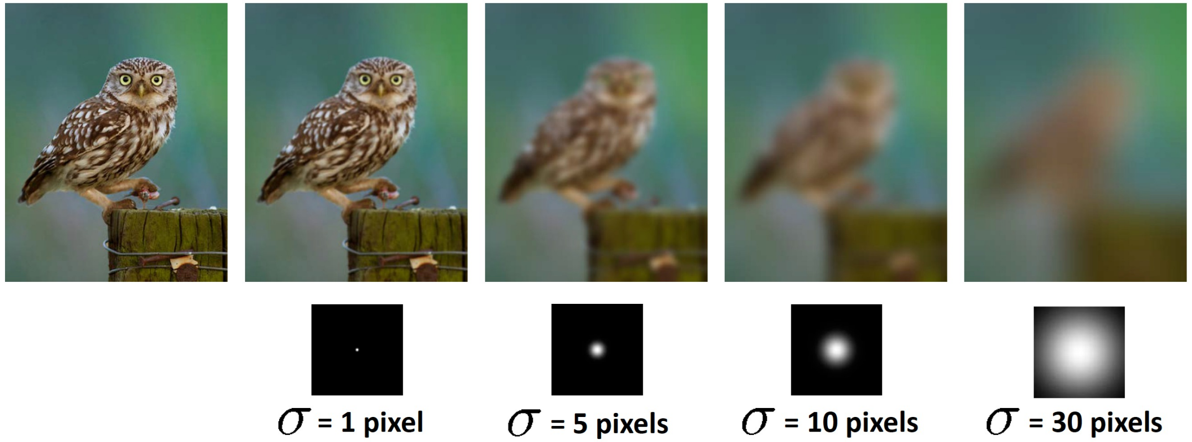
- That’s not possible.

Relating Nyquist-Shannon theorem to Gaussian pyramid



- Gaussian blurring is low-pass filtering.
- By blurring we try to sufficiently decrease the Nyquist frequency to avoid aliasing.

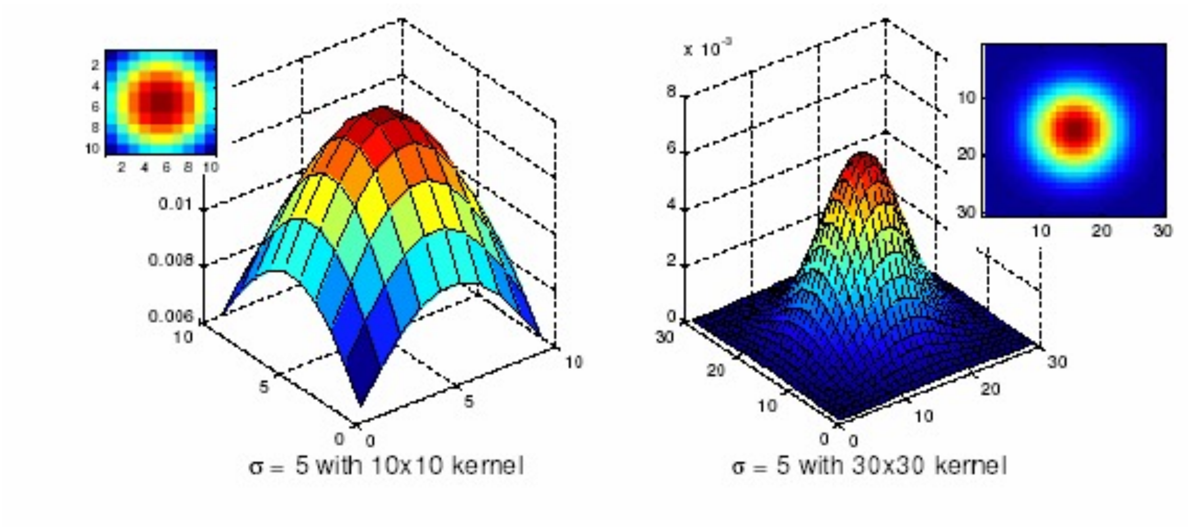
How large should the Gauss blur we use be?



Choosing blur level & kernel width

Practically, you have two parameters to choose: blur level σ, and the (discrete) filter size

- Q1: How to choose appropriate σ, knowing the down-sampling rate s?
- One possible empirical rule: $\sigma = \sqrt{s/2}$ **why?**



- Q2: The Gaussian function has infinite support, but discrete filters use finite kernels!
- Values at edges should be near zero. Practically, we set filter half-width to about 3σ

Blurring is lossy

level 0

-

level 1 (before downsampling)

=

residual

What does the residual look like?

Blurring is lossy

level 0

-

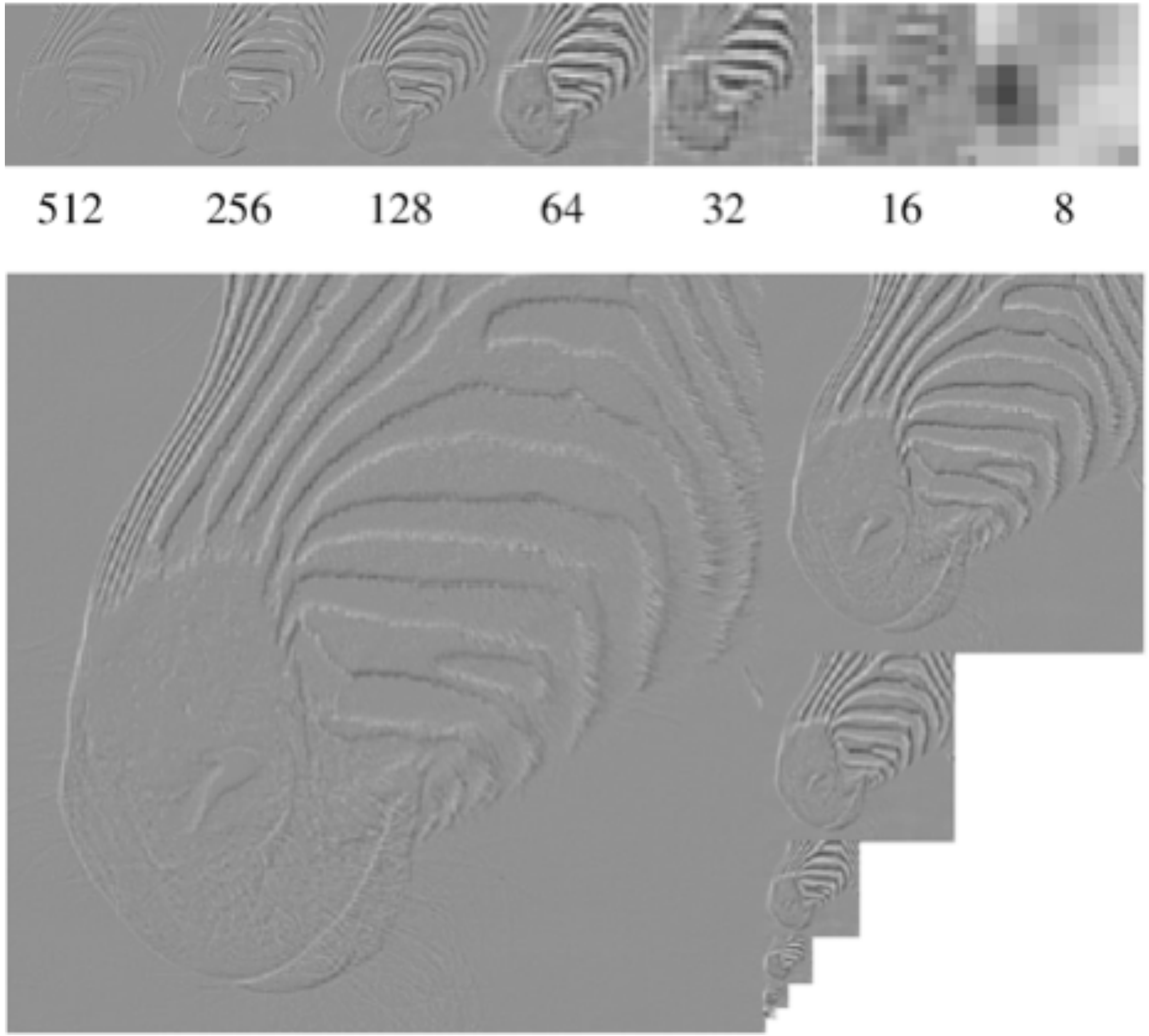
level 1 (before downsampling)

=

residual

Can we make a pyramid that is lossless?

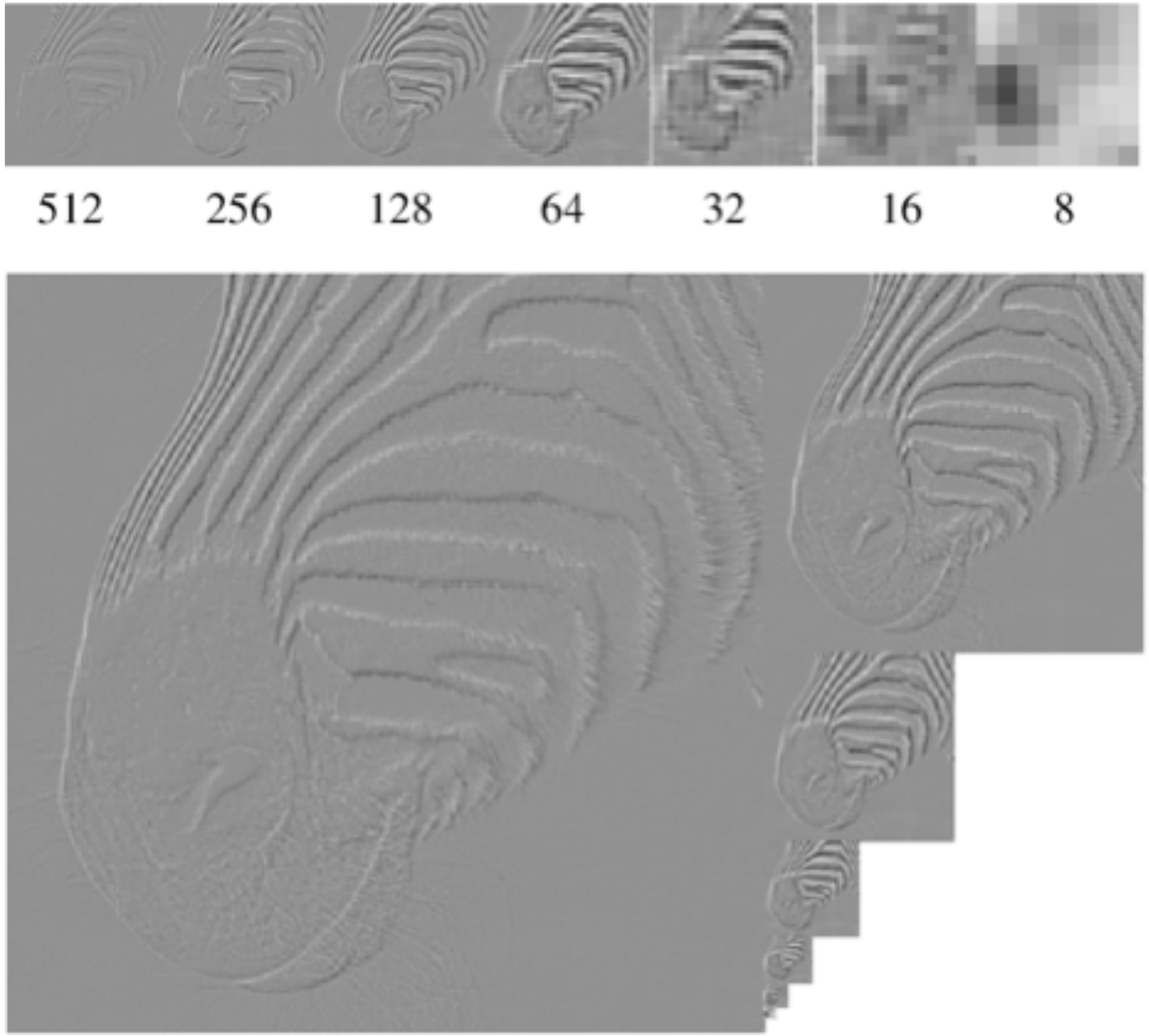
Laplacian image pyramid



At each level, retain the residuals instead of the blurred images themselves.

Can we reconstruct the original image using the pyramid?

Laplacian image pyramid



At each level, retain the residuals instead of the blurred images themselves.

Can we reconstruct the original image using the pyramid?

- Yes we can!

What do we need to store to be able to reconstruct the original image?

Let’s start by looking at just one level

level 0

=

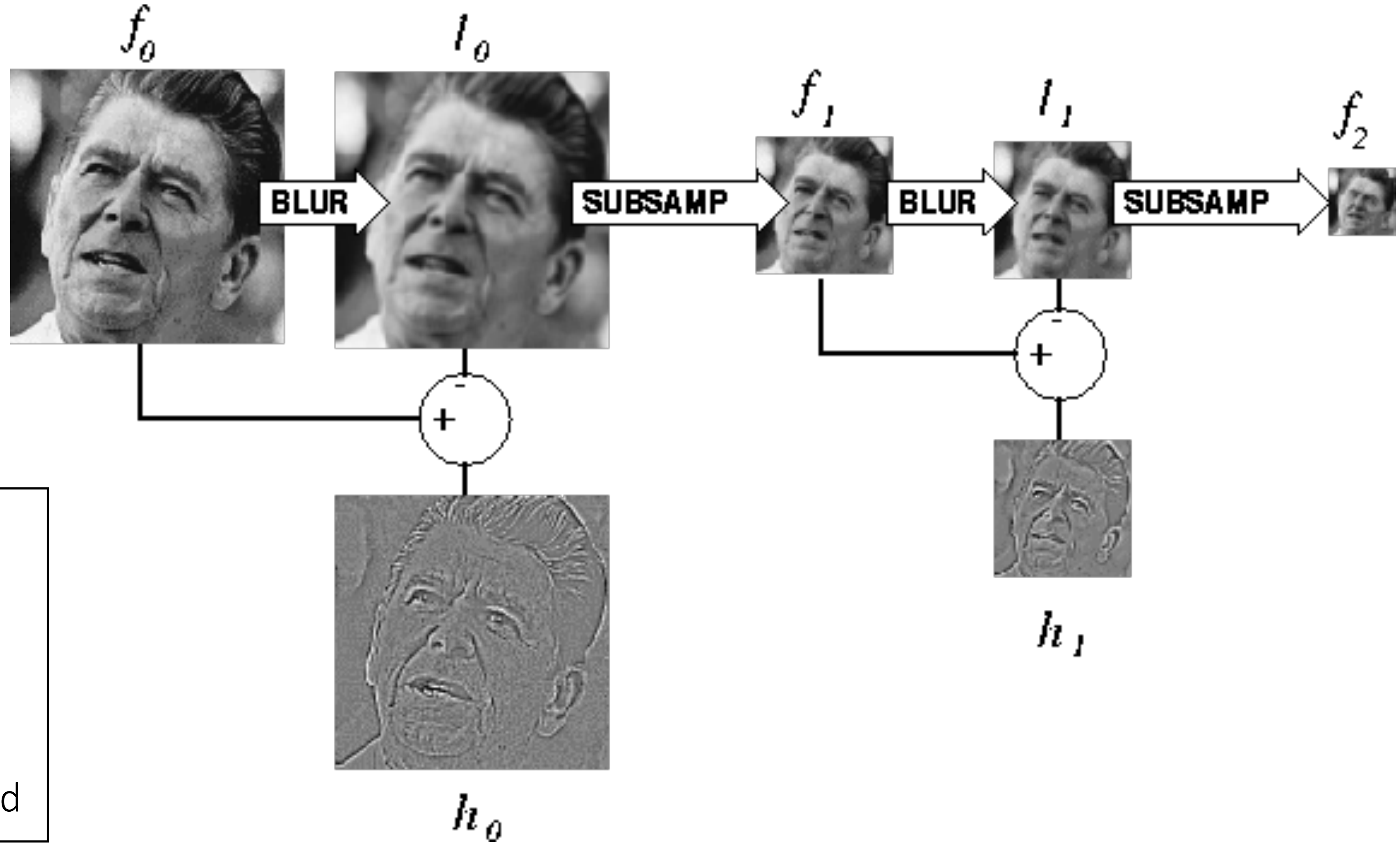
level 1 (upsampled)

+

residual

Does this mean we need to store both residuals and the blurred copies of the original?

Constructing a Laplacian pyramid



Algorithm

repeat:

filter

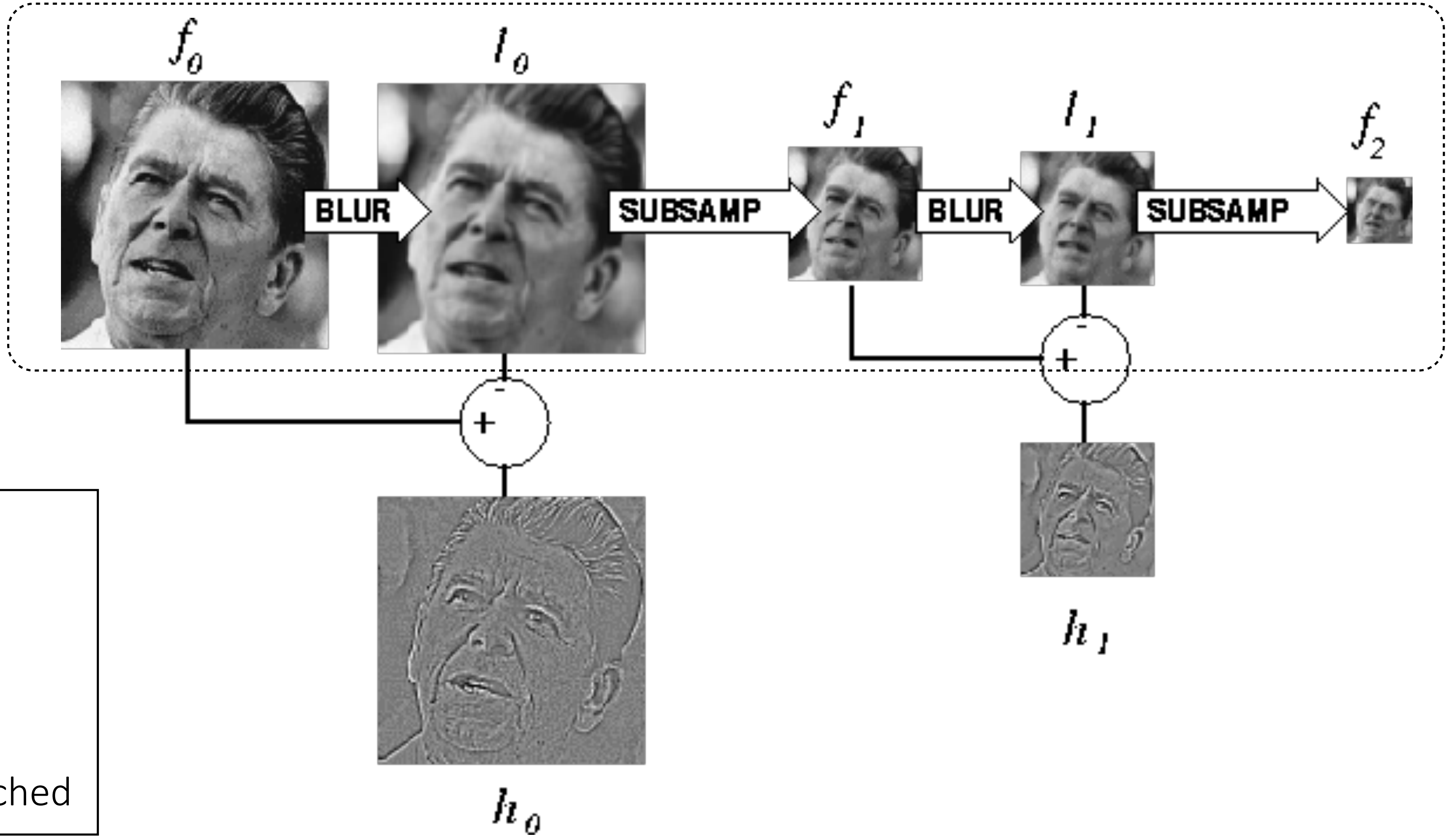
compute residual

subsample

until min resolution reached

Constructing a Laplacian pyramid

What is this part?



Algorithm

repeat:

filter

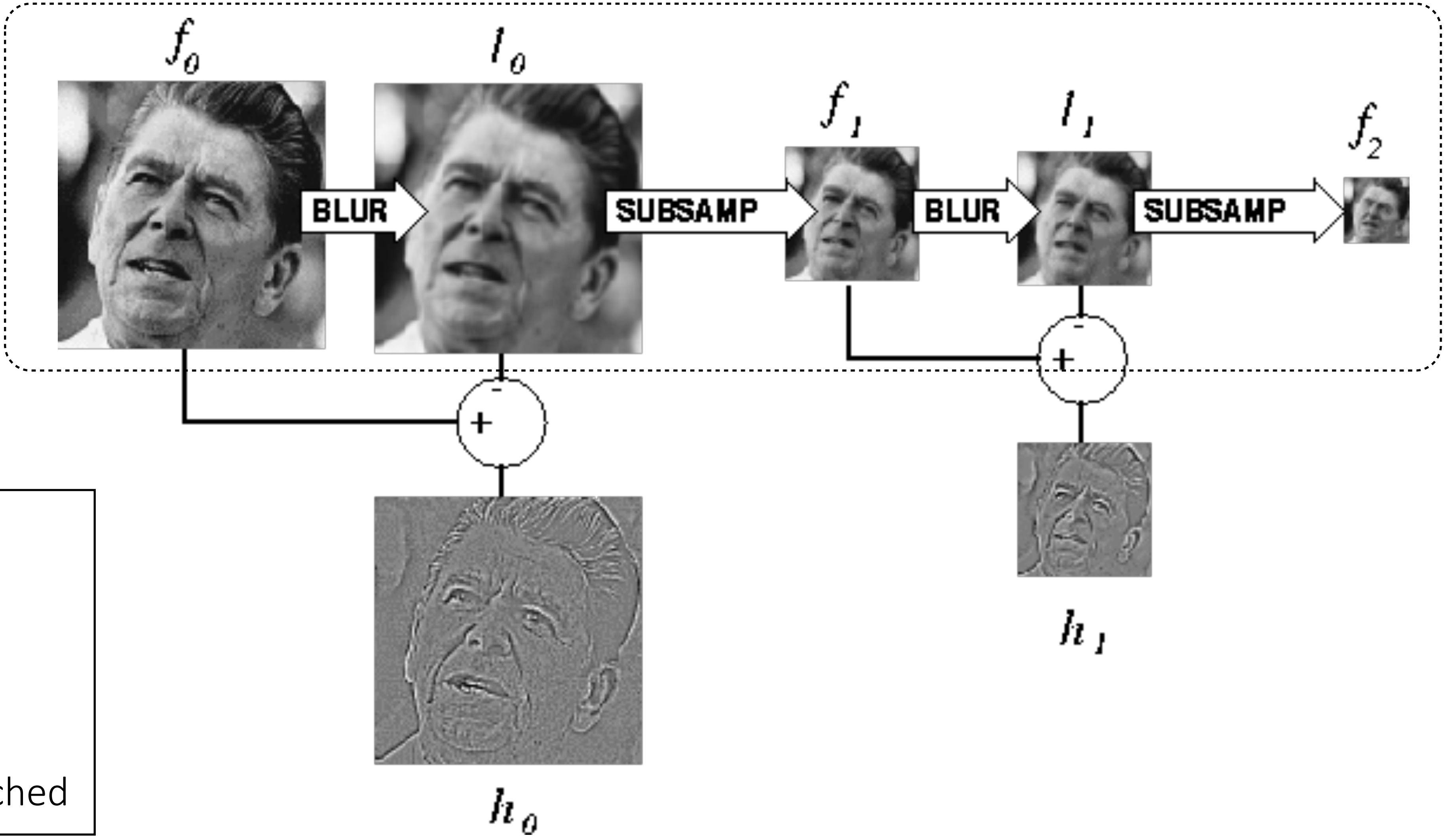
compute residual

subsample

until min resolution reached

Constructing a Laplacian pyramid

It's a Gaussian pyramid.



Algorithm

repeat:

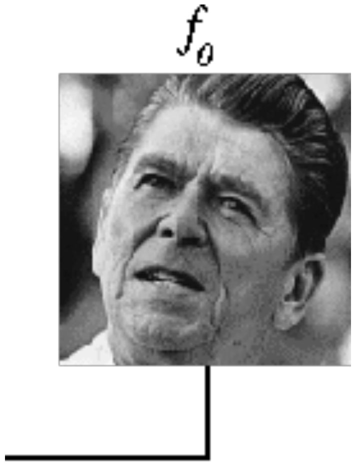
filter

compute residual

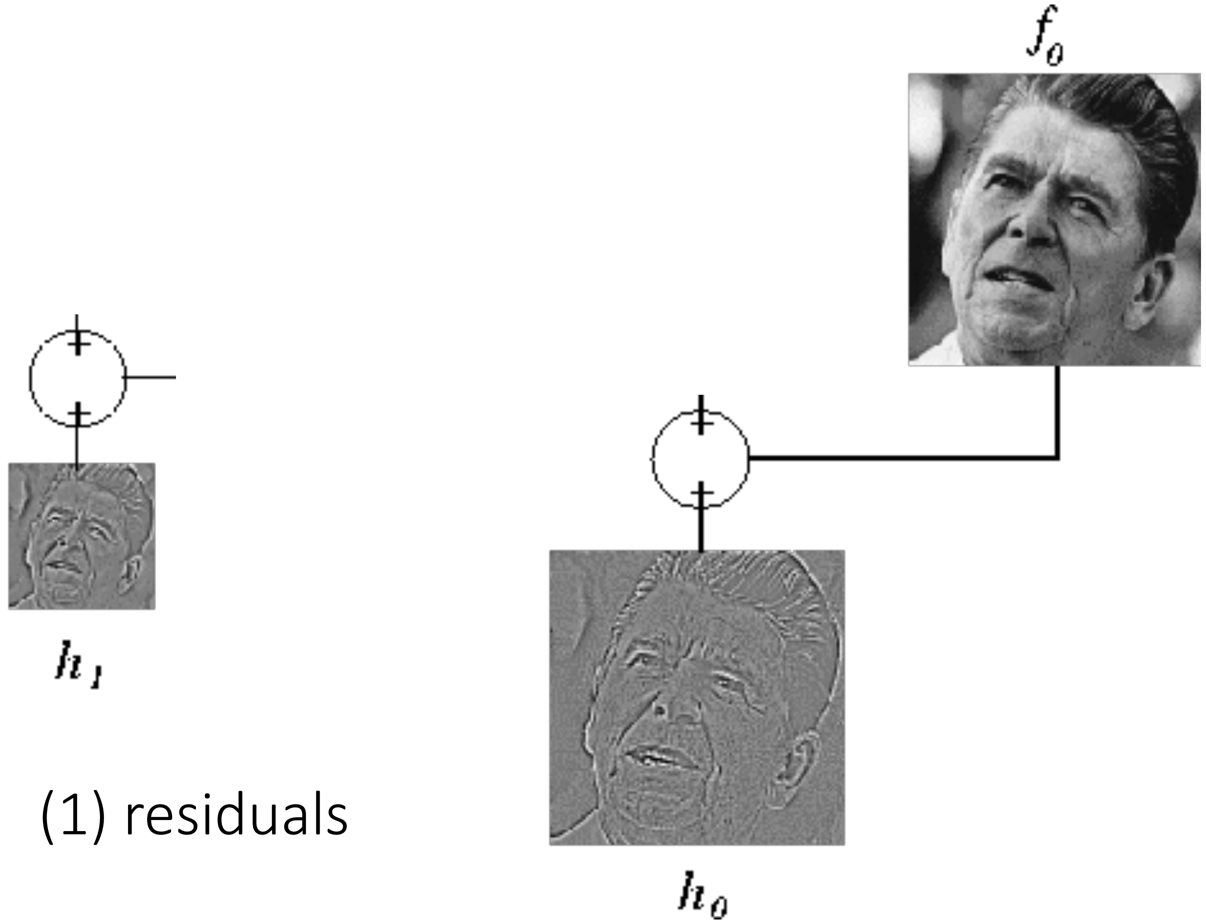
subsample

until min resolution reached

What do we need to construct the original image?



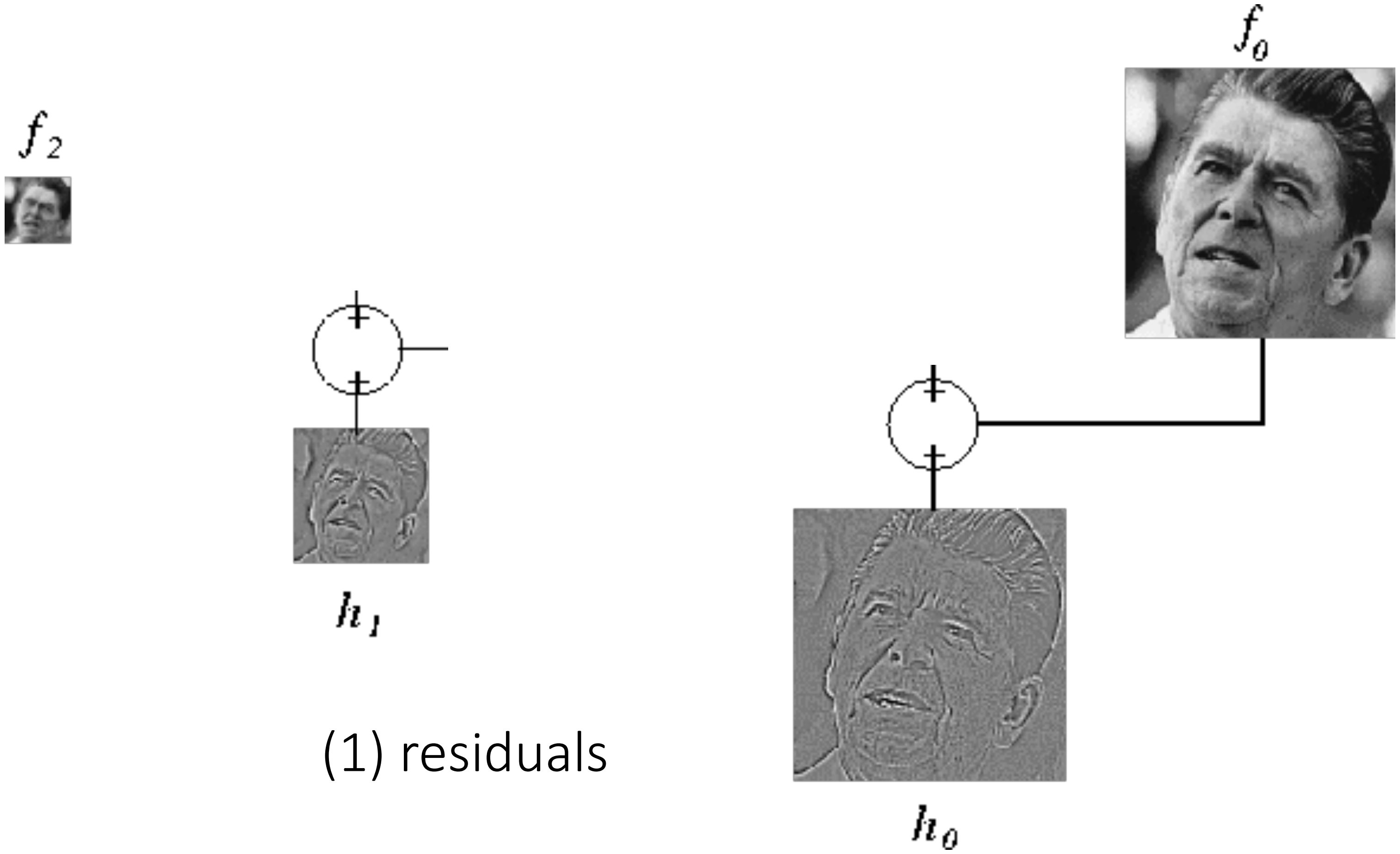
What do we need to construct the original image?



(1) residuals

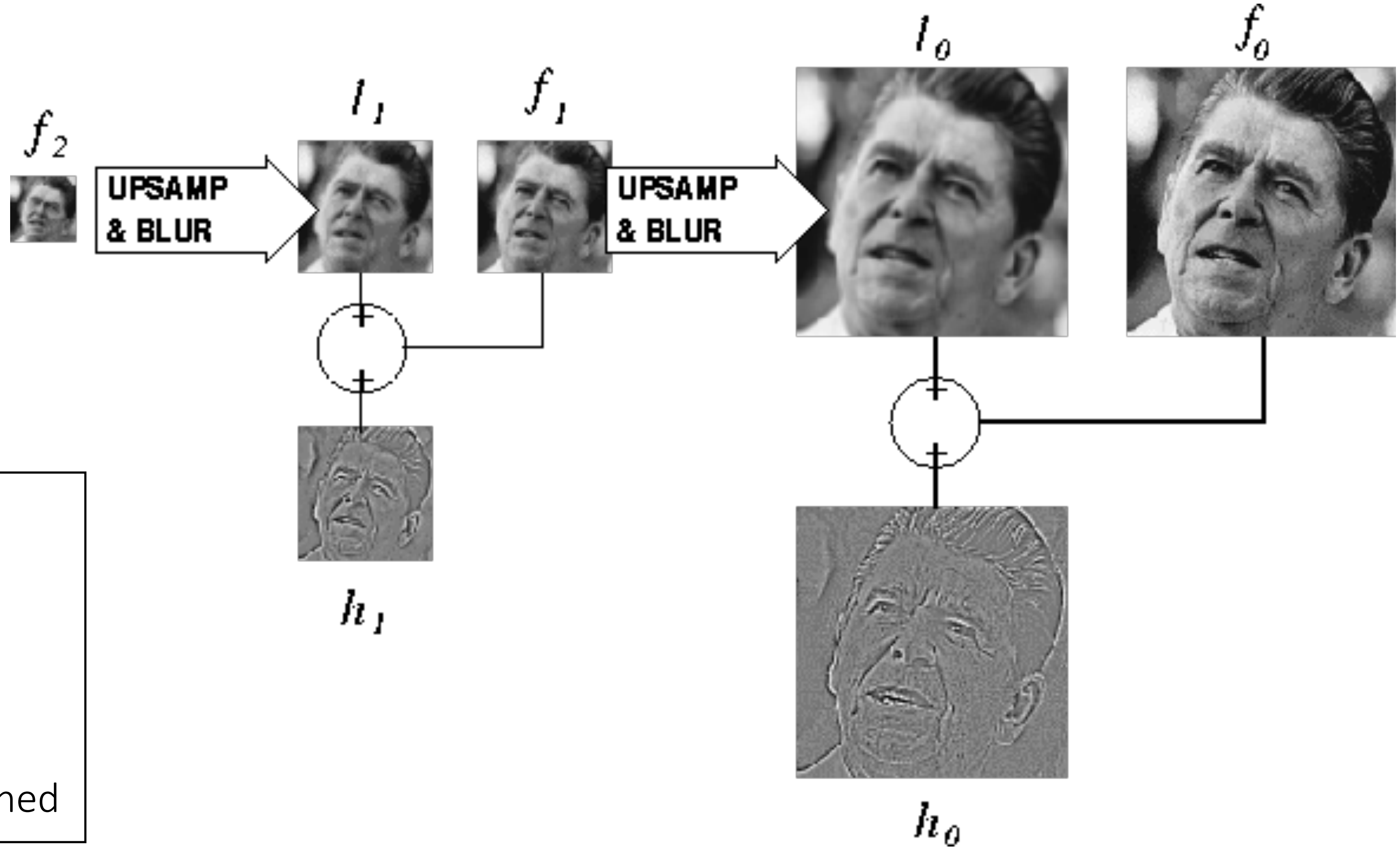
What do we need to construct the original image?

(2) smallest image



(1) residuals

Reconstructing the original image



Algorithm

repeat:

upsample

sum with residual

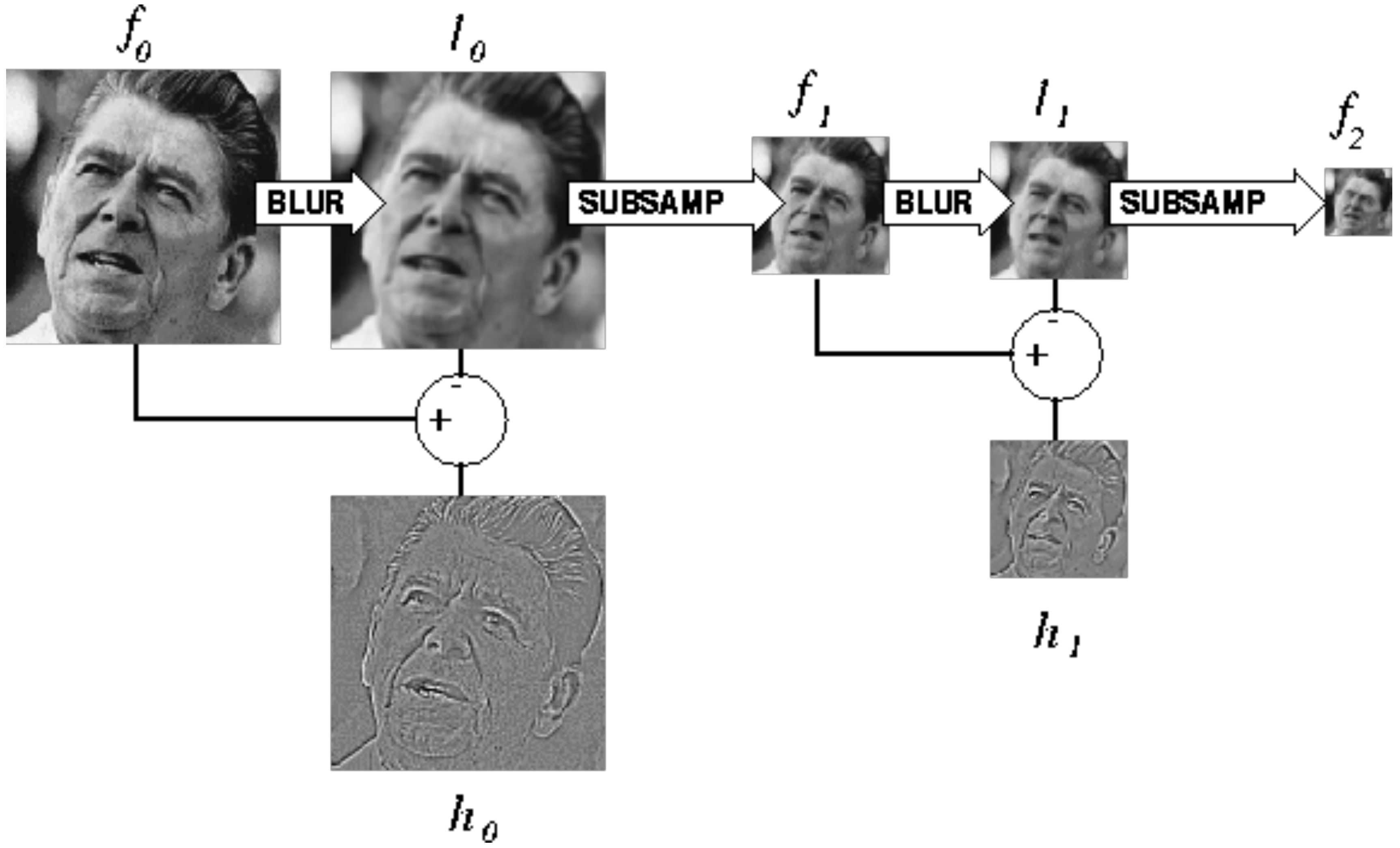
until orig resolution reached

Gaussian vs Laplacian Pyramid

Shown in opposite order for space.

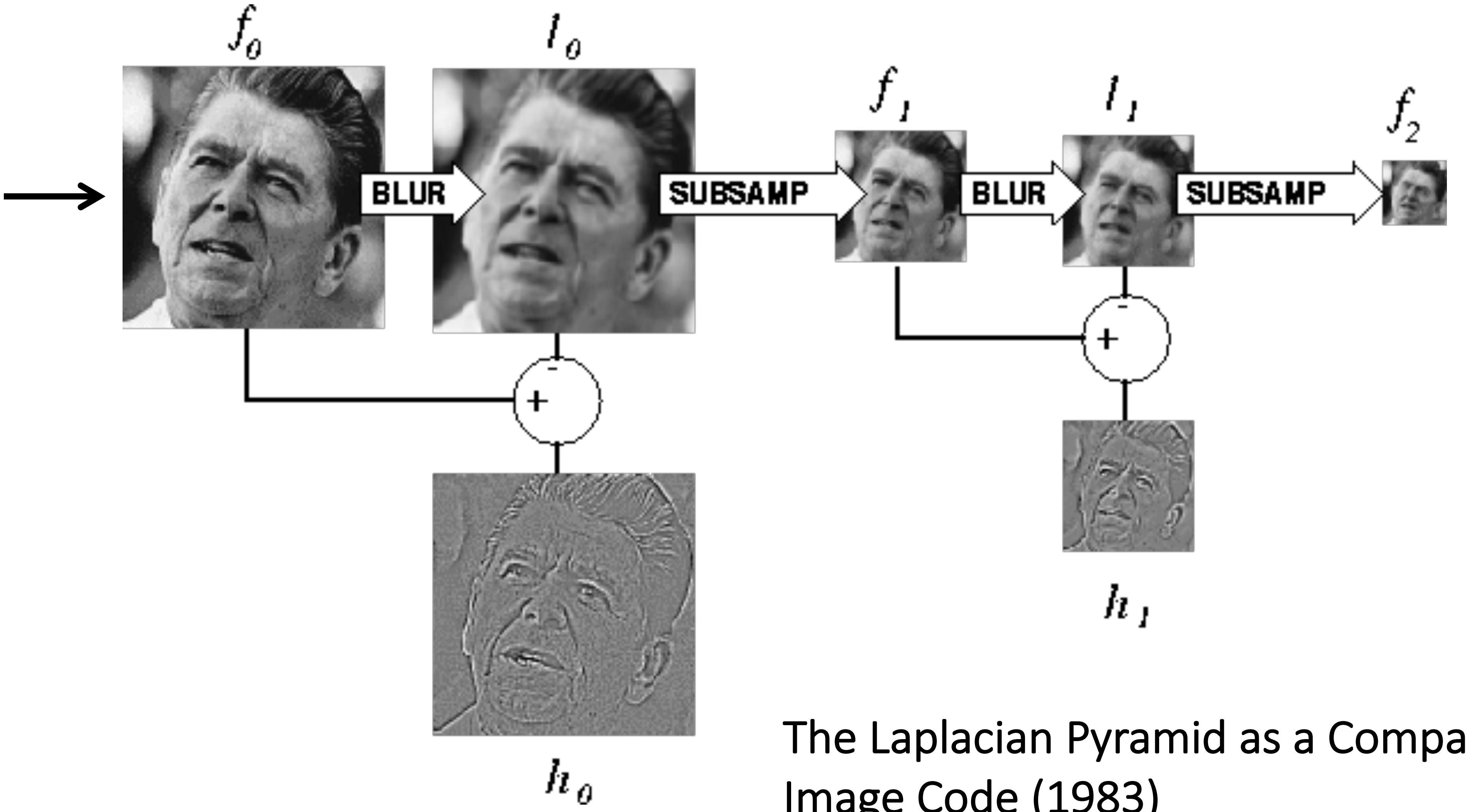
Which one takes more space to store?

Why Reagan?



Why Reagan?

Ronald Reagan was President when the Laplacian pyramid was invented



The Laplacian Pyramid as a Compact Image Code (1983)

Peter J. Burt , Edward H. Adelson

Still used extensively

Still used extensively

foreground details enhanced, background details reduced

input image

user-provided mask

Other types of pyramids

Steerable pyramid: At each level keep multiple versions, one for each direction.

Wavelets: Huge area in image processing (see 18-793).

What are image pyramids used for?

image compression

multi-scale texture mapping

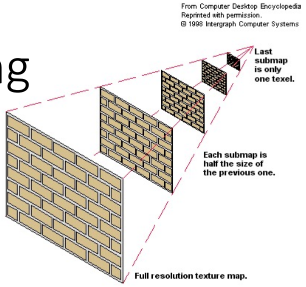
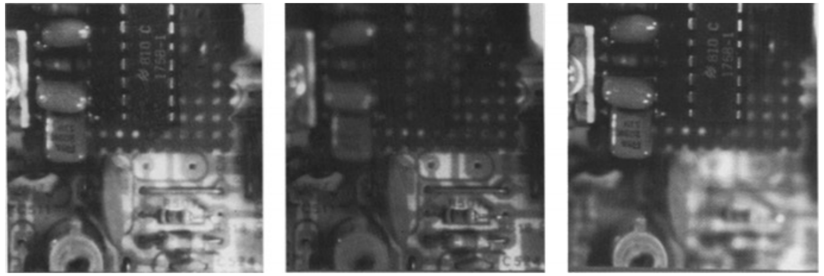
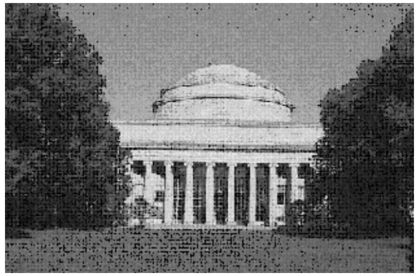


image blending

focal stack compositing

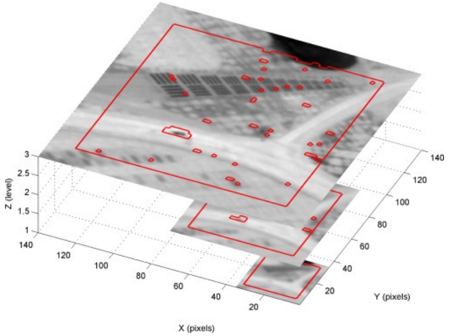


denoising



multi-scale detection

multi-scale registration



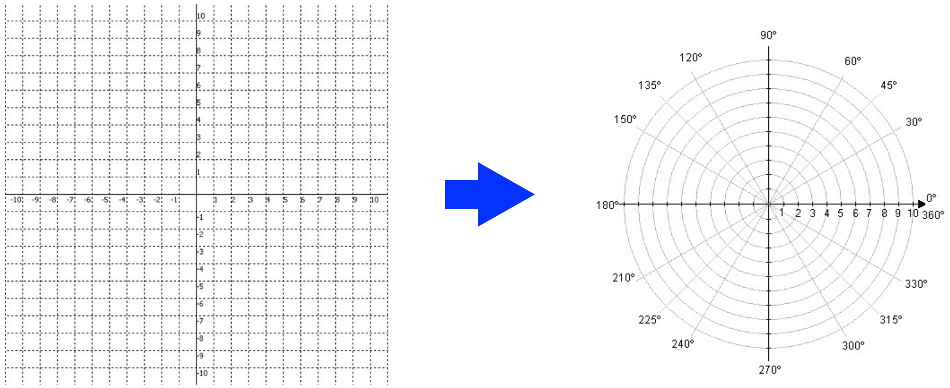
Fourier transform

Recalling some basics

Complex numbers have two parts:

rectangular coordinates

$$\underbrace{R}_{\text{real}} + \underbrace{jI}_{\text{imaginary}}$$



Recalling some basics

Complex numbers have two parts:

rectangular coordinates

$$R + jI$$

real imaginary

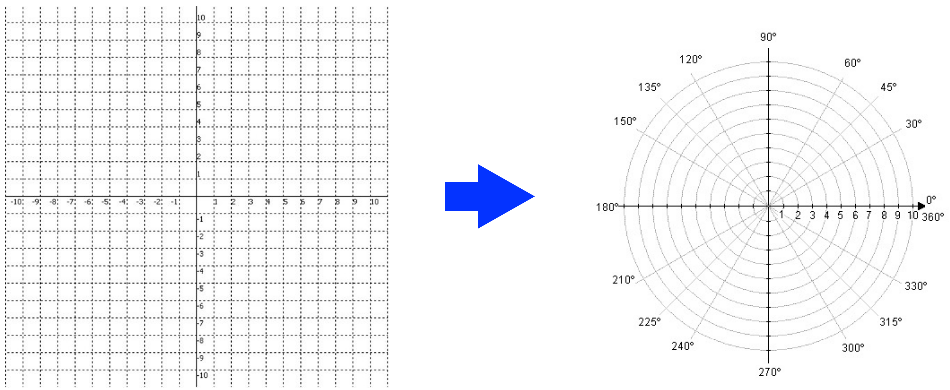
Alternative reparameterization:

polar coordinates

$$r(\cos\theta + j\sin\theta)$$

polar transform

$$\theta = \tan^{-1}(\frac{I}{R}) \quad r = \sqrt{R^2 + I^2}$$



polar transform

Recalling some basics

Complex numbers have two parts:

rectangular coordinates

$$R + jI$$

real imaginary

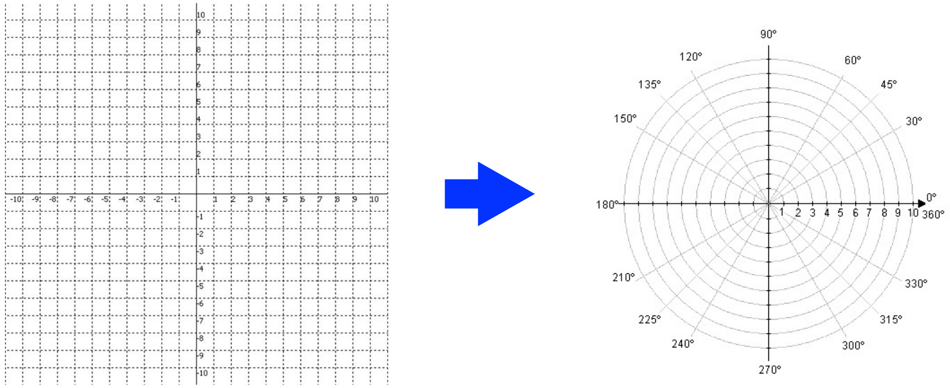
Alternative reparameterization:

polar coordinates

$$r(\cos\theta + j\sin\theta)$$

polar transform

$$\theta = \tan^{-1}\left(\frac{I}{R}\right) \quad r = \sqrt{R^2 + I^2}$$

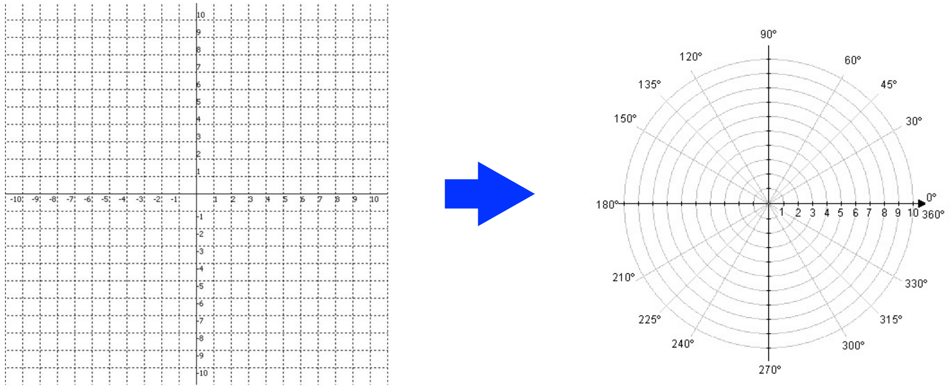


polar transform

How do you write these in exponential form?

Recalling some basics

Complex numbers have two parts:



rectangular coordinates

$$R + jI$$

real imaginary

Alternative reparameterization:

polar coordinates

$$r(\cos\theta + j\sin\theta)$$

polar transform

$$\theta = \tan^{-1}\left(\frac{I}{R}\right) \quad r = \sqrt{R^2 + I^2}$$

or equivalently

$$re^{j\theta}$$

Euler's formula

$$e^{j\theta} = \cos\theta + j\sin\theta$$

exponential form

This will help us understand the Fourier transform equations

Fourier transform

	Fourier transform	inverse Fourier transform
continuous	$F(k) = \int_{\infty}^{-\infty} f(x) e^{-j2\pi kx} dx$	$f(x) = \int_{\infty}^{-\infty} F(k) e^{j2\pi kx} dk$
discrete	$F(k) = \frac{1}{N} \sum_{x=0}^{N-1} f(x) e^{-j2\pi kx/N}$ $k = 0, 1, 2, \ldots, N-1$	$f(x) = \sum_{k=0}^{N-1} F(k) e^{j2\pi kx/N}$ $x = 0, 1, 2, \ldots, N-1$

'summation of sine waves'

Computing the discrete Fourier transform (DFT)

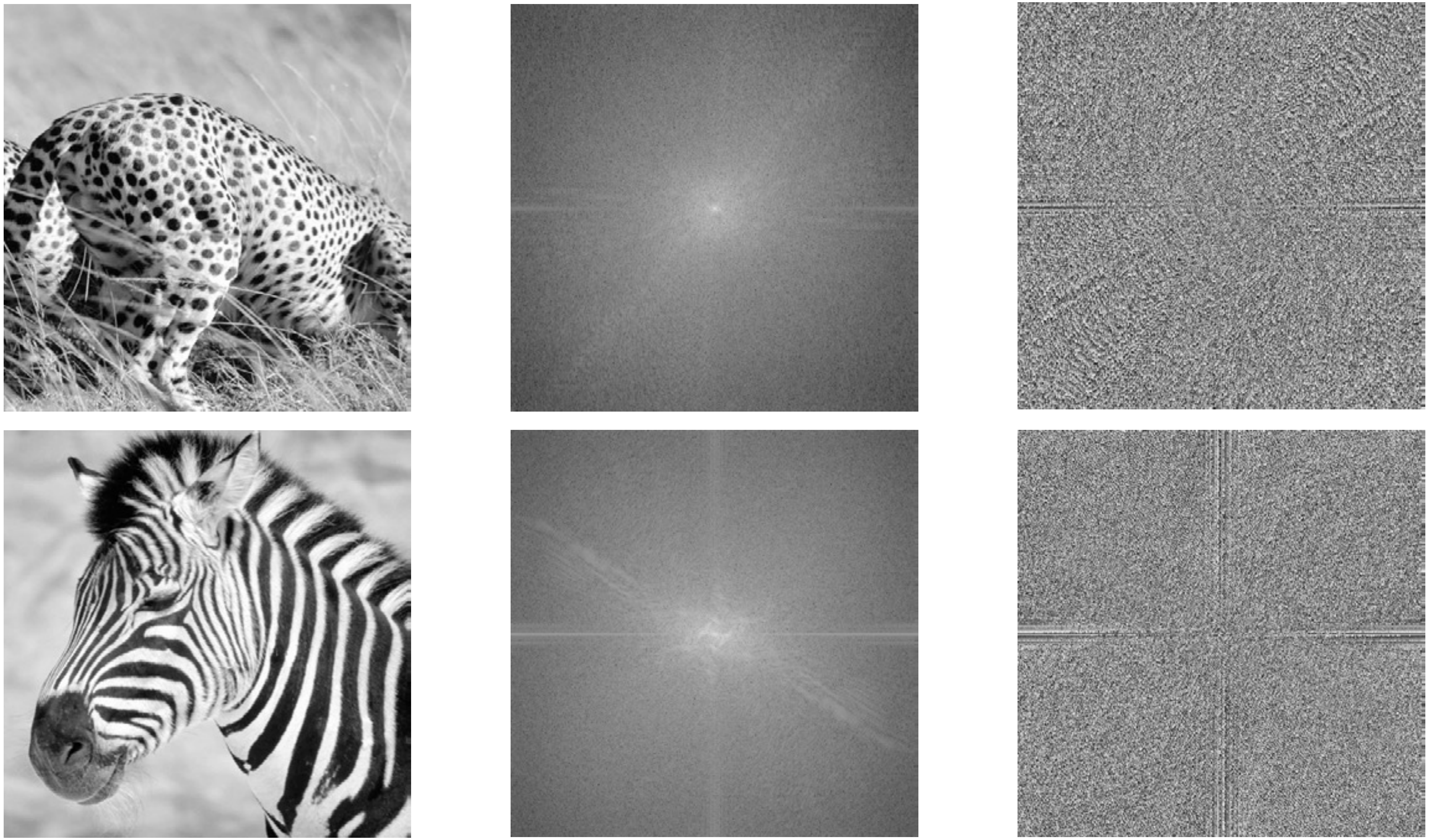
$$F(k) = \frac{1}{N} \sum_{x=0}^{N-1} f(x) e^{-j2\pi kx/N}$$ is just a matrix multiplication:

$$\boldsymbol{F} = \boldsymbol{W} \boldsymbol{f}$$

$$\begin{bmatrix} F(0) \\ F(1) \\ F(2) \\ F(3) \\ \vdots \\ F(N-1) \end{bmatrix} = \begin{bmatrix} W^0 & W^0 & W^0 & W^0 & \cdots & W^0 \\ W^0 & W^1 & W^2 & W^3 & \cdots & W^{N-1} \\ W^0 & W^2 & W^4 & W^6 & \cdots & W^{N-2} \\ W^0 & W^3 & W^6 & W^9 & \cdots & W^{N-3} \\ \vdots & & & & \ddots & \vdots \\ W^0 & W^{N-1} & W^{N-2} & W^{N-3} & \cdots & W^1 \end{bmatrix} \begin{bmatrix} f(0) \\ f(1) \\ f(2) \\ f(3) \\ \vdots \\ f(N-1) \end{bmatrix} \qquad \boldsymbol{W = e^{-j2\pi/N}}$$

In practice this is implemented using the *fast Fourier transform* (FFT) algorithm.

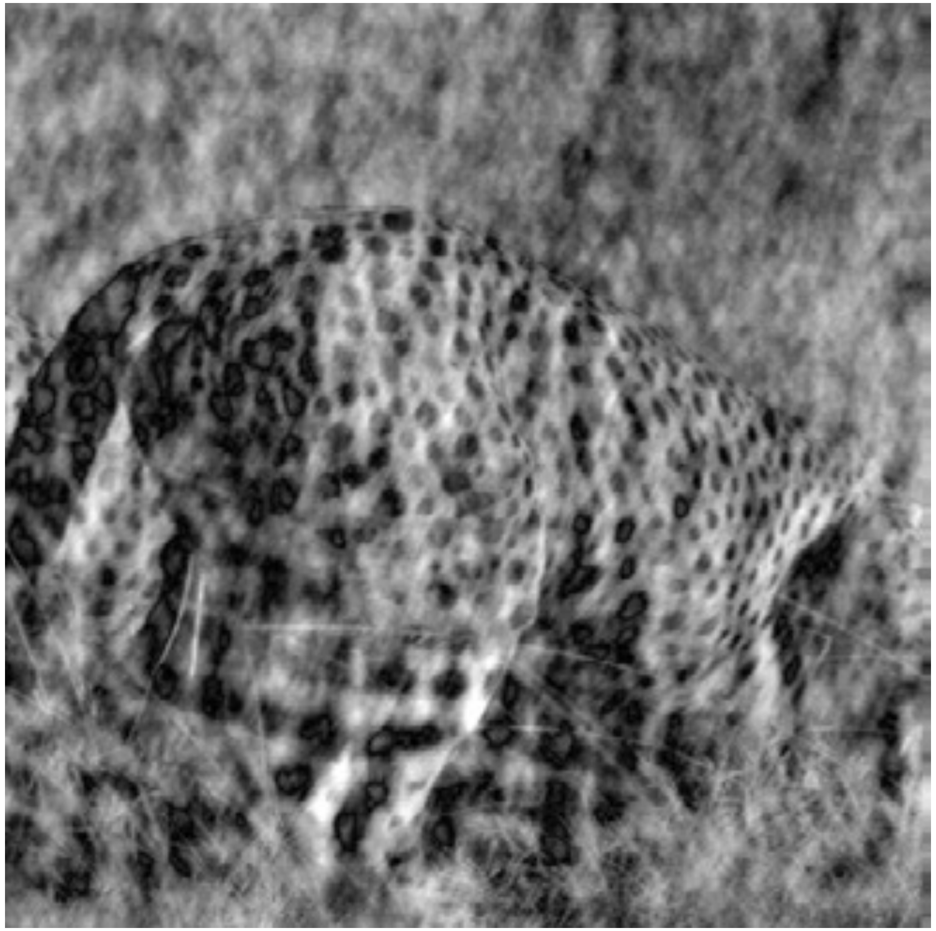
Fourier transforms of natural images



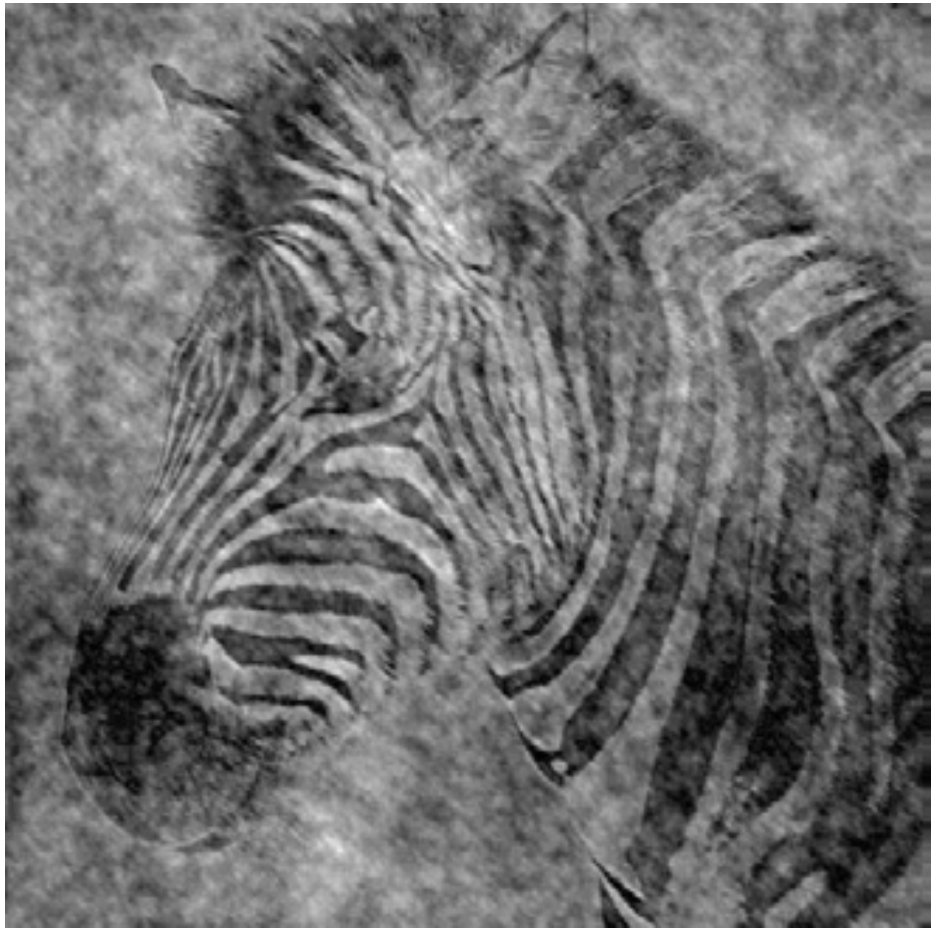
original amplitude phase

Fourier transforms of natural images

Image phase matters!



cheetah phase with zebra amplitude



zebra phase with cheetah amplitude

Frequency-Domain View of Filtering

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!

Convolution for 1D continuous signals

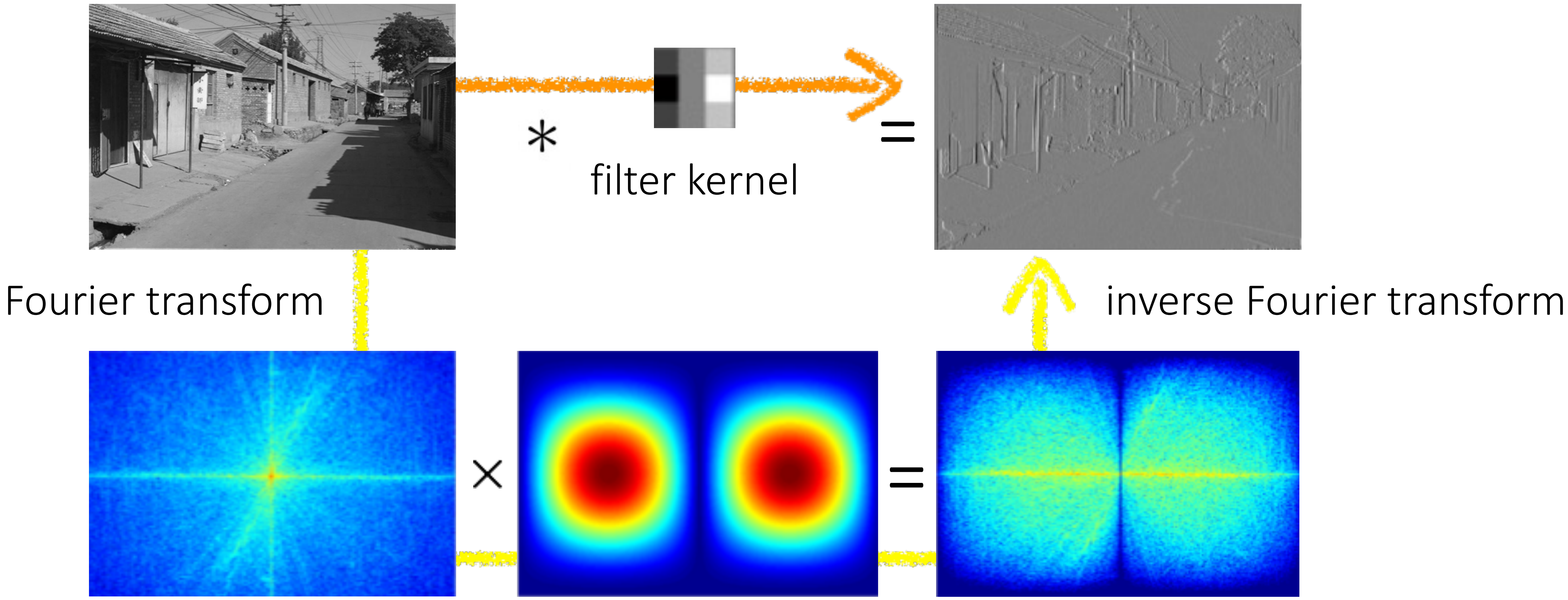
Definition of linear shift-invariant filtering as convolution:

$$\underbrace{(f * g)(x)}_{\text{filtered signal}} = \int_{-\infty}^{\infty} \underbrace{f(y)}_{\text{filter}}\,\underbrace{g(x-y)}_{\text{input signal}}\,dy$$

Using the convolution theorem, we can interpret and implement all types of linear shift-invariant filtering as multiplication in frequency domain.

Why implement convolution in frequency domain?

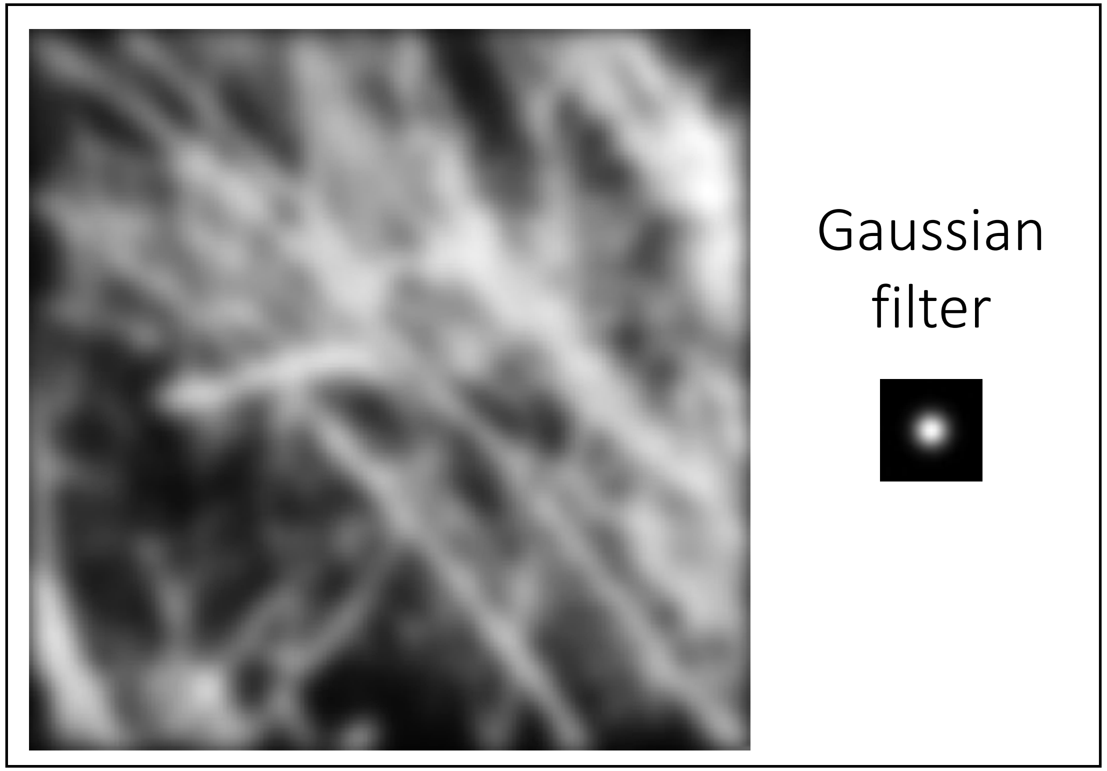
Spatial domain filtering



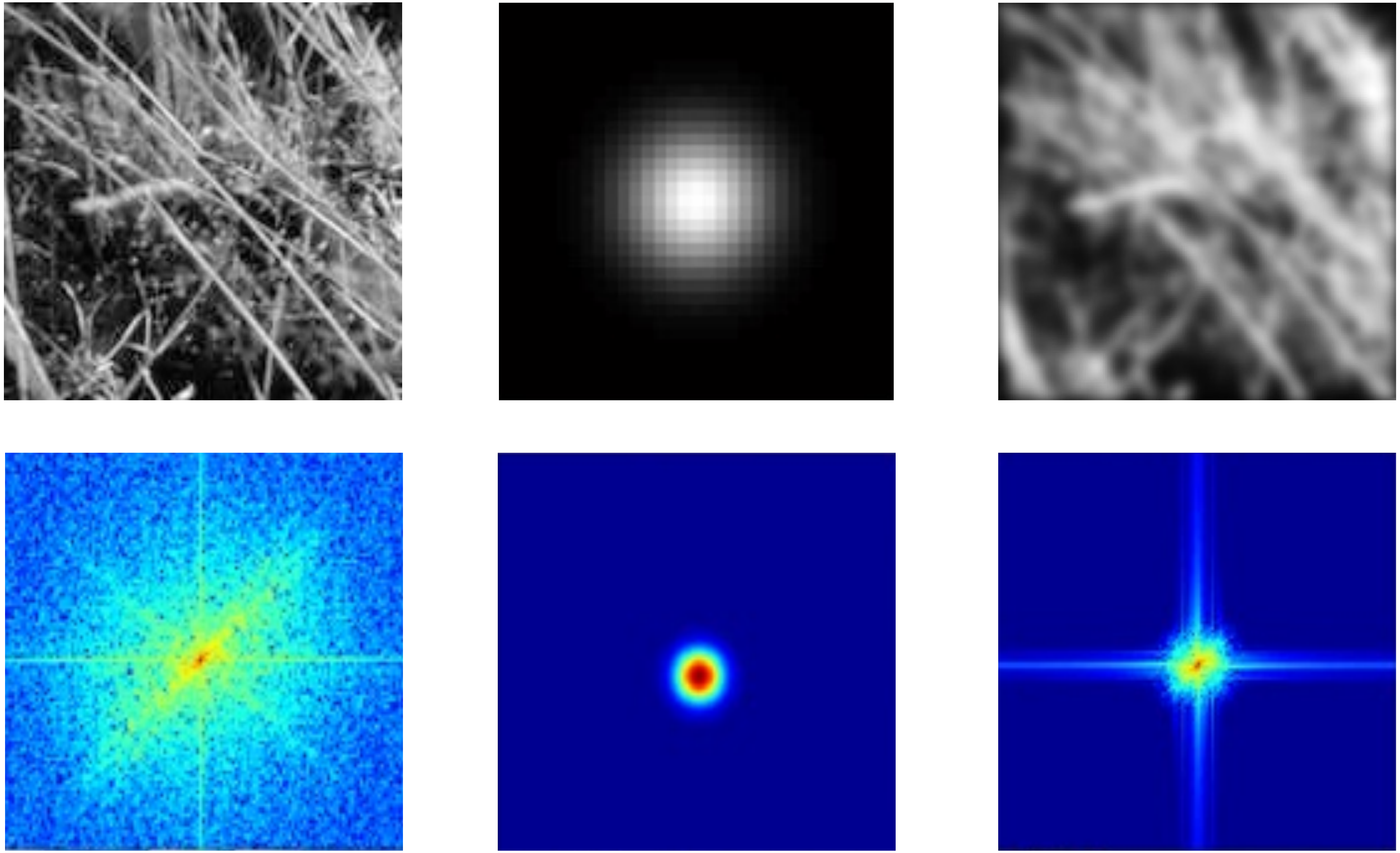
Frequency domain filtering

Revisiting blurring

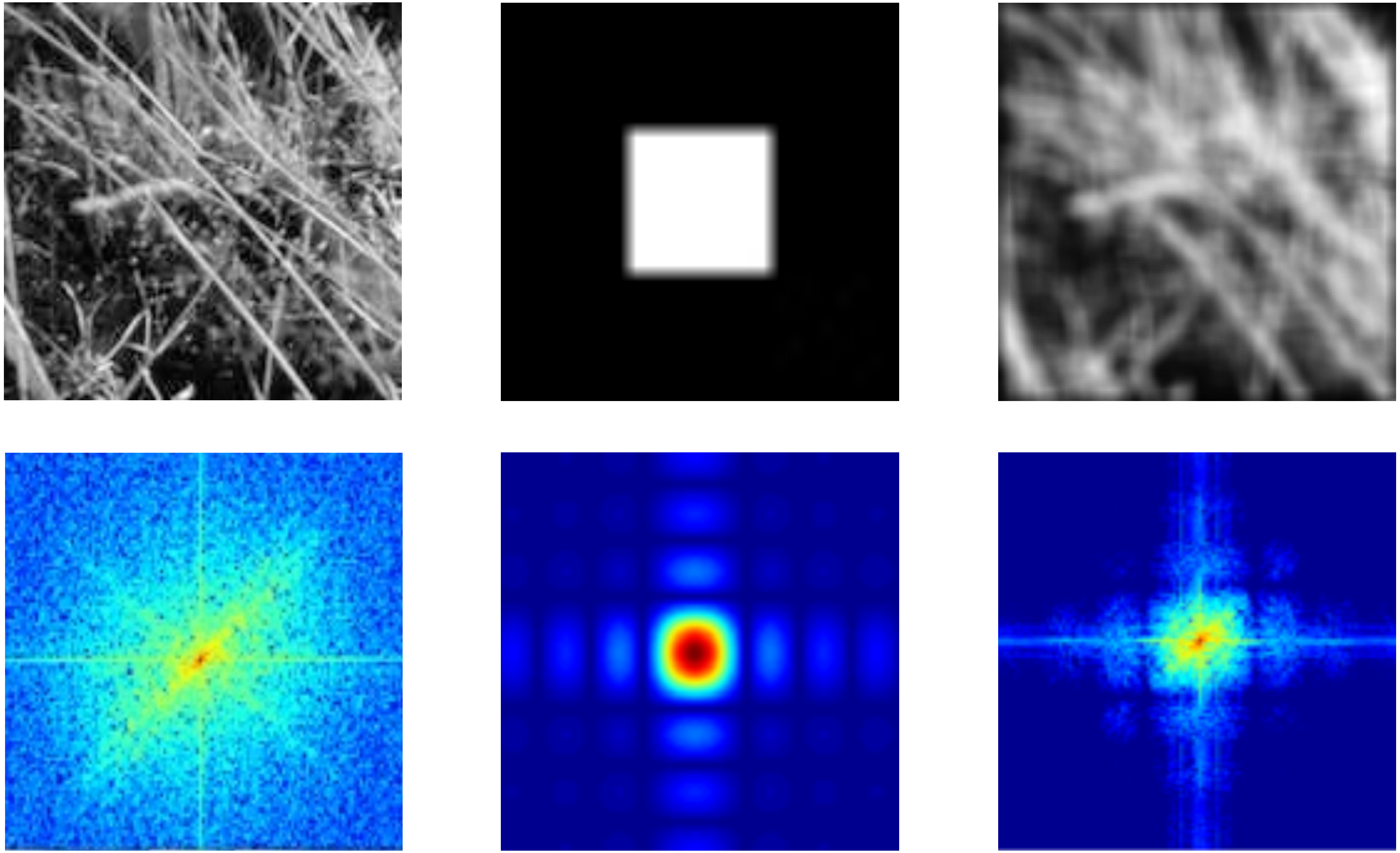
Why does the Gaussian give a nice smooth image, but the square filter give edgy artifacts?



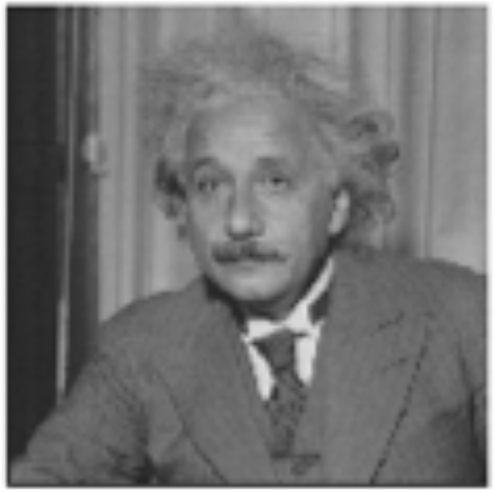
Gaussian blur



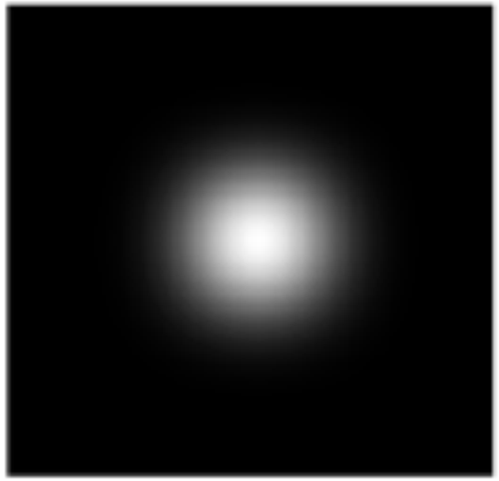
Box blur

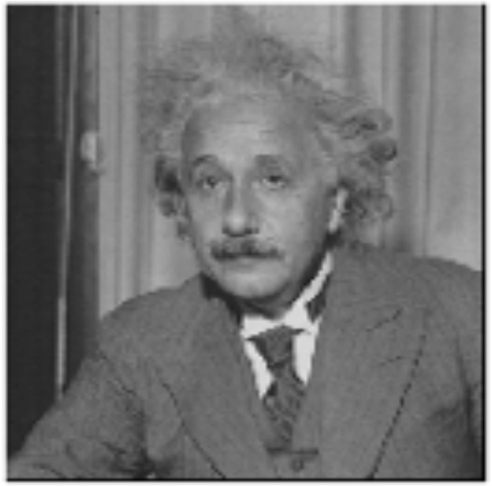


More filtering examples



?

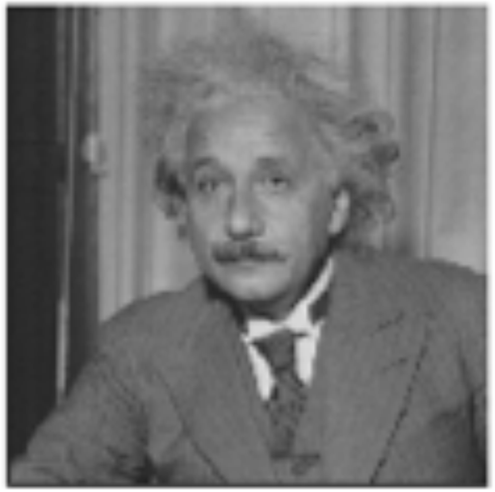


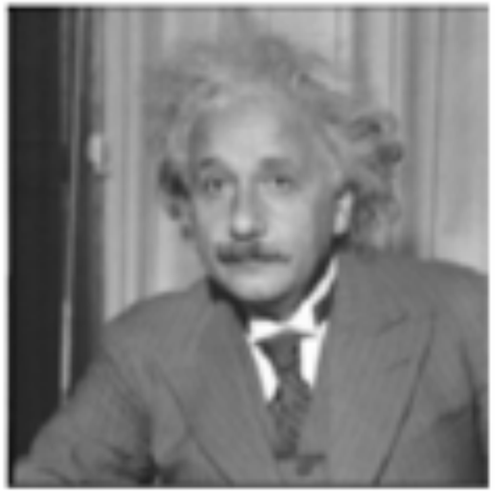


?

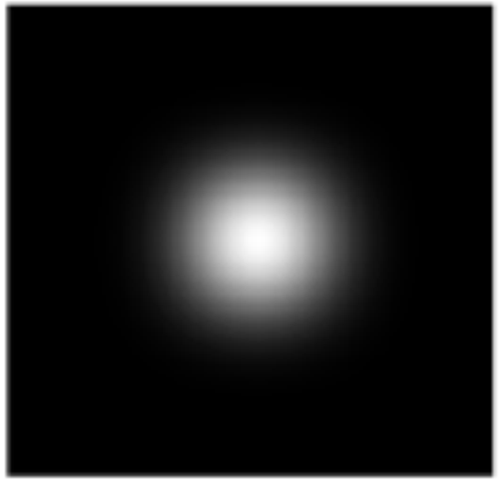
filters shown in frequency-domain

More filtering examples

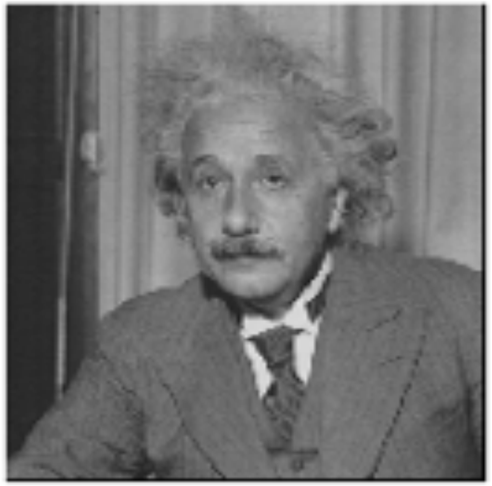




low-pass



filters shown in frequency-domain

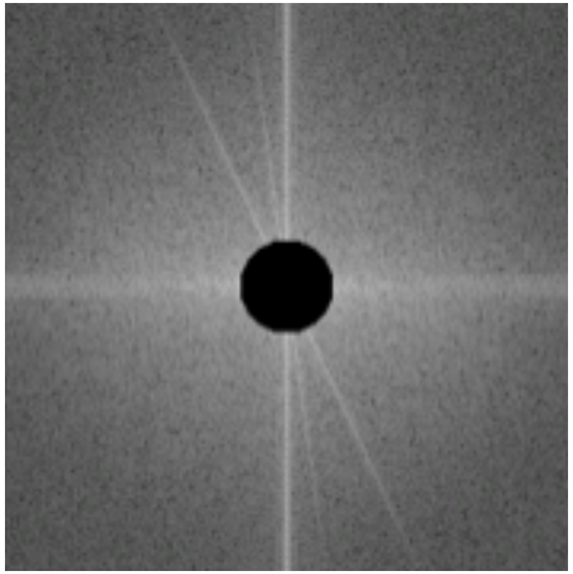


band-pass

More filtering examples

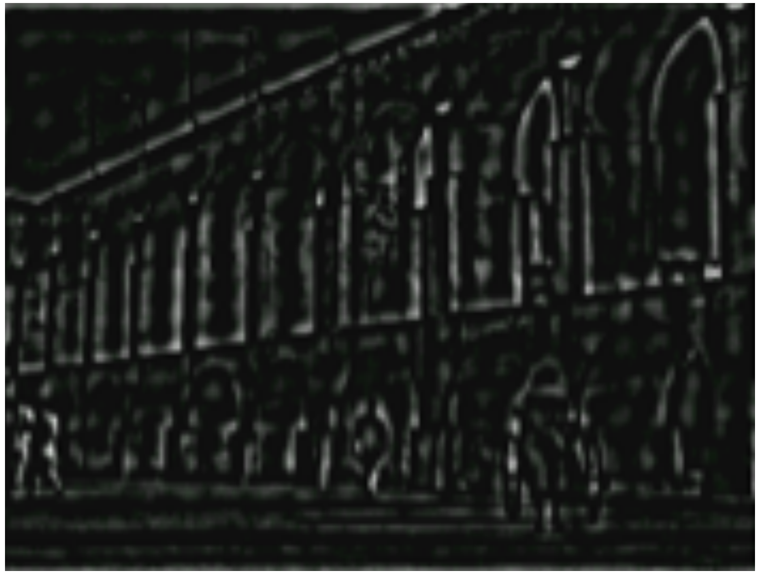
?

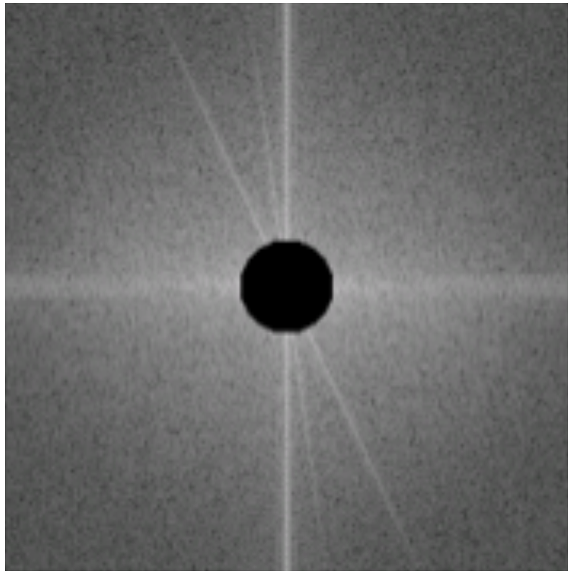
high-pass



More filtering examples

high-pass





The University of Texas at Austin
Electrical and Computer Engineering
Cockrell School of Engineering