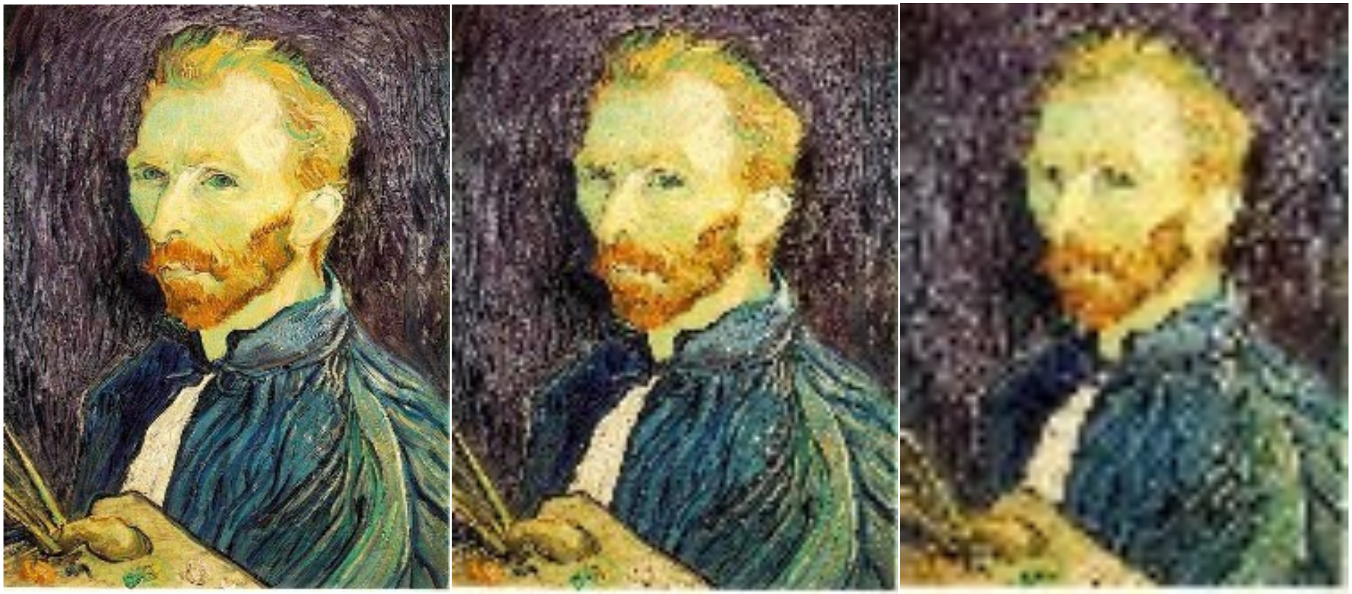
CSCI 497P/597P: Computer Vision



Lecture 5:

Sobel Filter

Image Frequency Content

Downsampling and Gaussian Pyramids

Announcements

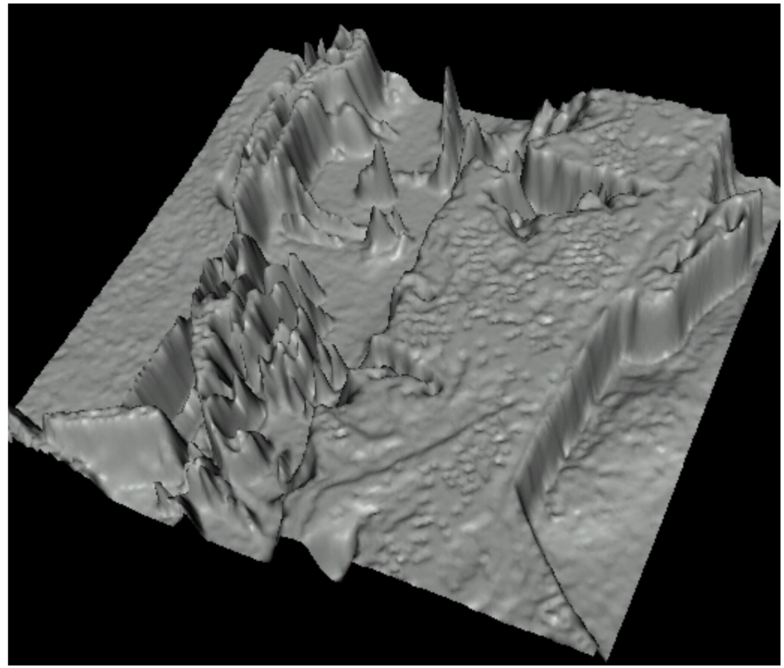
- Project 1 is not out yet.

Goals

- Understand how the Sobel filter works to detect edges in noisy images.
- Have an intuitive understanding of what constitutes high frequency and low frequency image content.
- Know how to make images smaller:
  - The naive way via subsampling (and why this is bad)
  - The better way by prefiltering (and why this is better)
- Understand how and why to construct a Gaussian Pyramid

An edge is an "intensity cliff" with rapid rate of change (derivative)

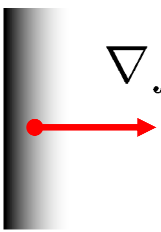
f(x,y) as brightness



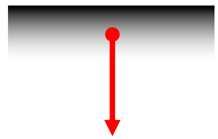
f(x,y) as height

Image Gradient as Edge Detector

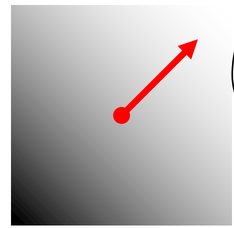
$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$$



$$\nabla f = \left[\frac{\partial f}{\partial x}, 0 \right]$$



$$\nabla f = \left[0, \frac{\partial f}{\partial y} \right]$$



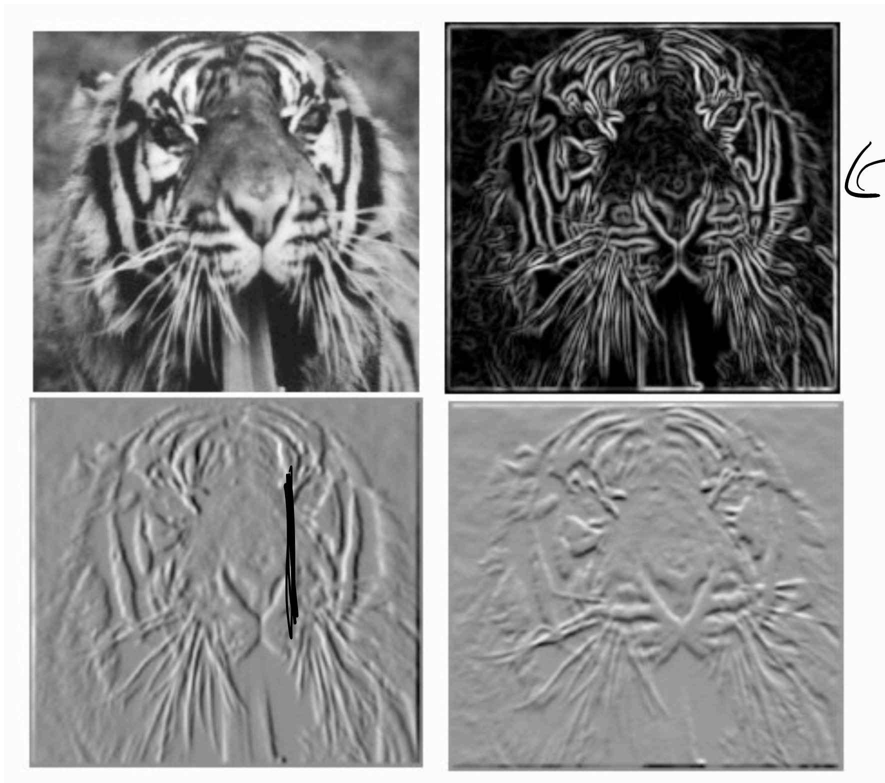
$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$$

What is the edge **strength**?

What is the edge **direction**?

Image Gradient: Visually

f



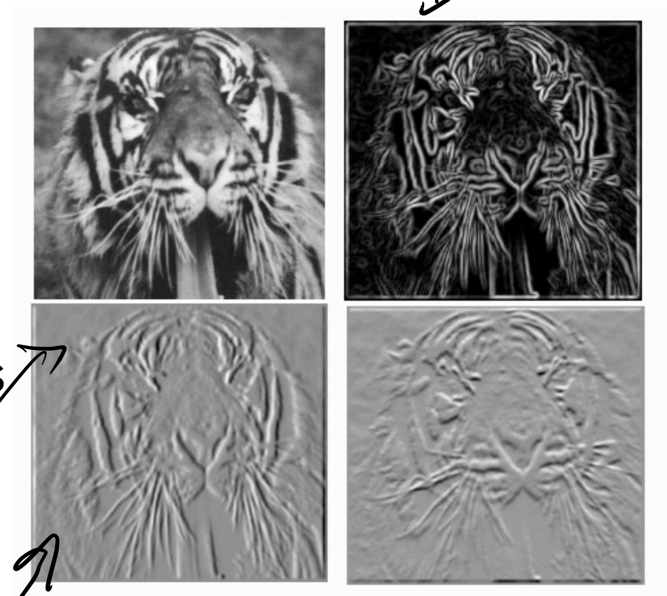
← edge strength

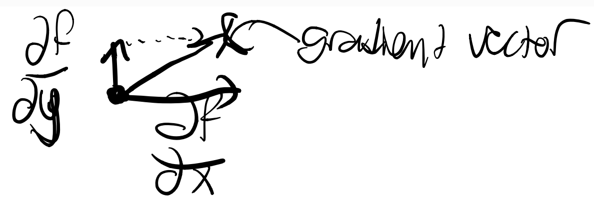
$\frac{\partial f}{\partial x}$

$\frac{\partial f}{\partial y}$

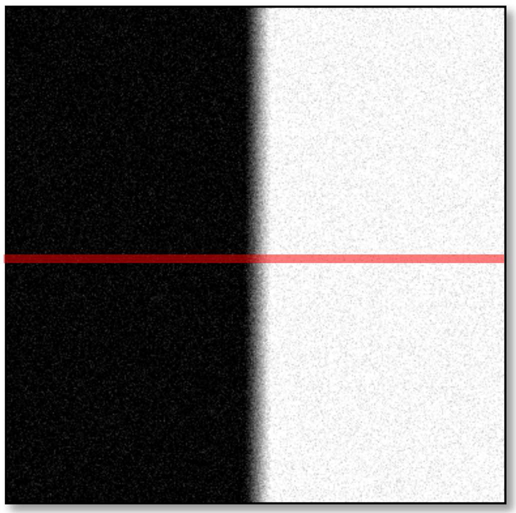
Aside: why are the derivatives grayish, not blackish?

- Images are nonnegative
- Derivatives can be negative!
- Scale and shift derivative values to display in the range 0-255
    - Gray is zero,
      darker is negative,
      lighter is positive



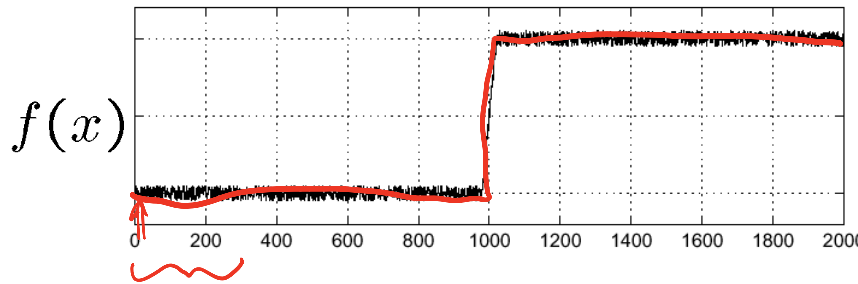


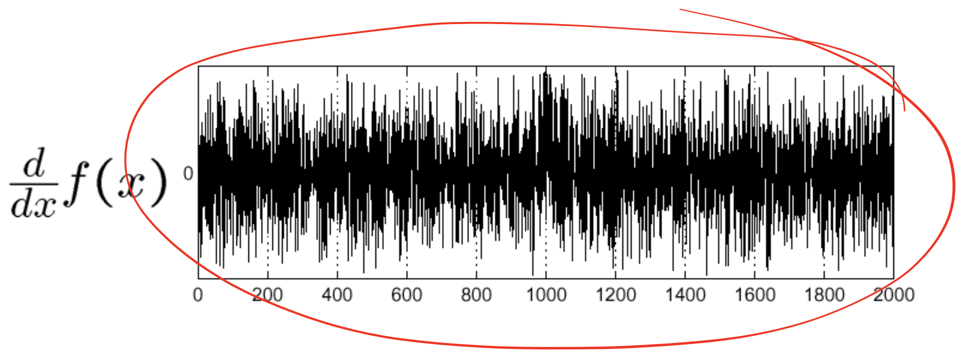
Images (still) aren't perfect



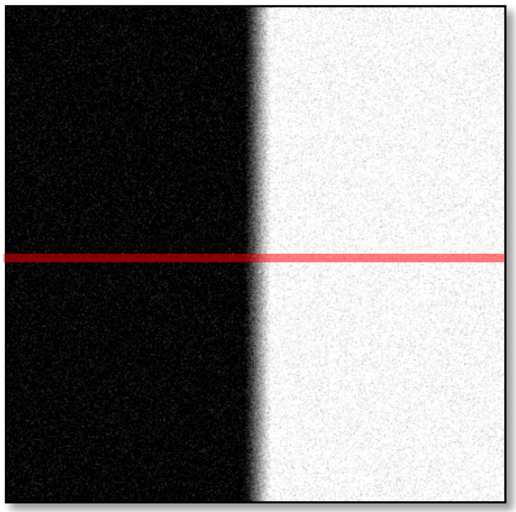
Noisy image

scanline



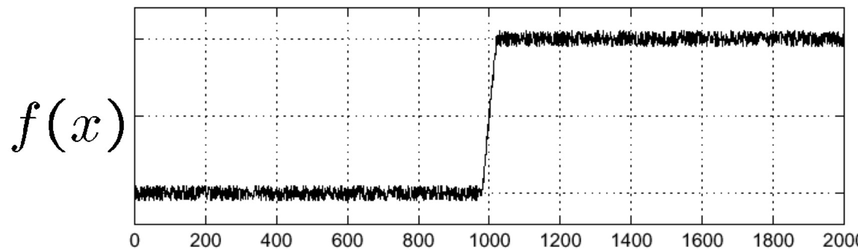


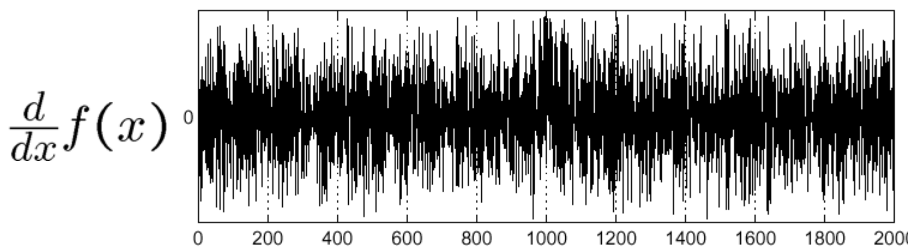
Images (still) aren't perfect



Noisy image

scanline





Can you find the edge?

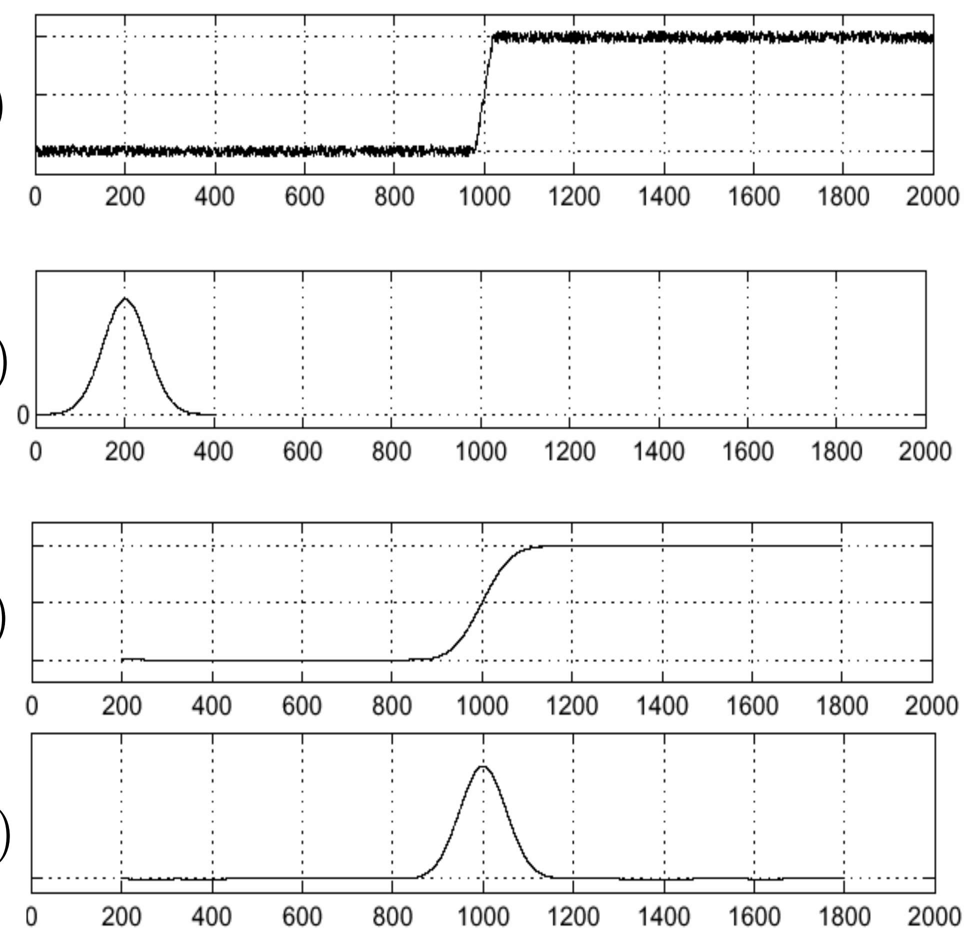
A solution: smooth (blur!) it first

$f(x)$

$w(x)$

$(f * w)(x)$

$$\frac{d}{dx}(f * w)(x)$$



An Edge Detection Filter

- Blur, then take the derivative:

$$f * \left(\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \right)$$

Blur

Gradient

derivative of Gaussian

≈ sobel filter

An Edge Detection Filter

- Blur, then take the derivative:

f *

1	2	1
2	4	2
1	2	1

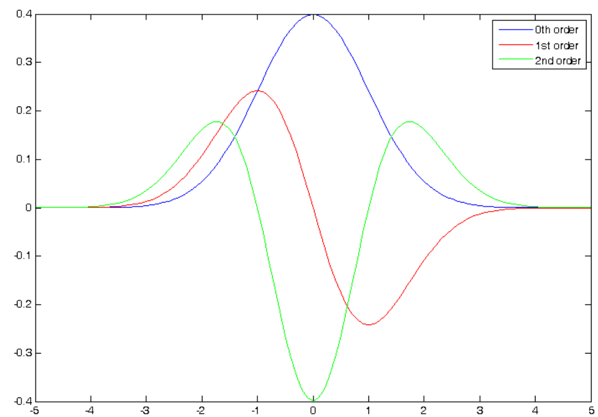
*

0	0	0
-1	0	1
0	0	0

Or, do the composition in the continuous domain then build a discrete approximation:

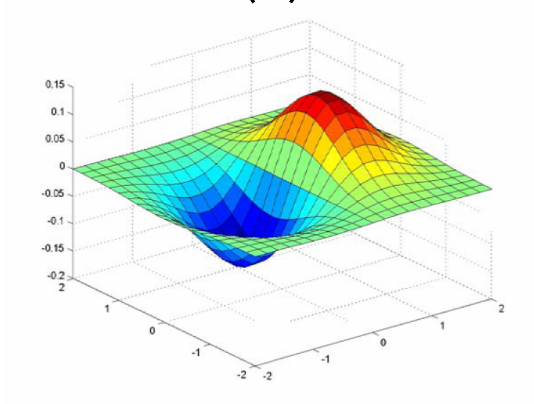
$$G_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$$

$$G'_\sigma(x) = \frac{d}{dx} G_\sigma(x) = -\frac{1}{\sigma}\left(\frac{x}{\sigma}\right) G_\sigma(x)$$

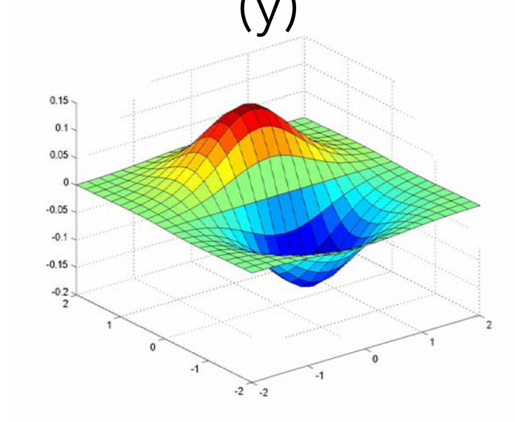


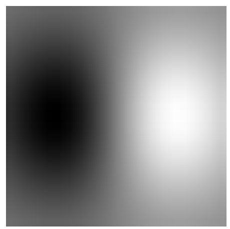
Derivative-of-Gaussian Filter

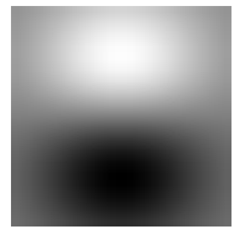
(x)



(y)







Sobel filter: a 3x3 approximation of the DoG:

$\frac{1}{8}$

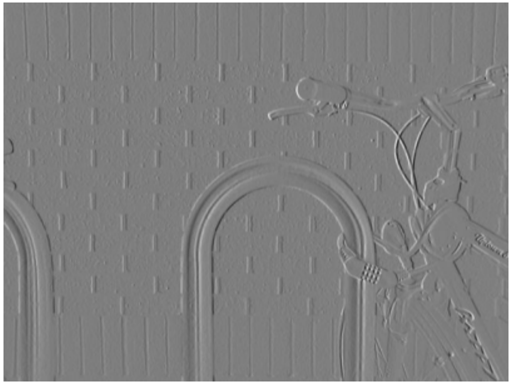
-1	0	1
-2	0	2
-1	0	1

$\frac{1}{8}$

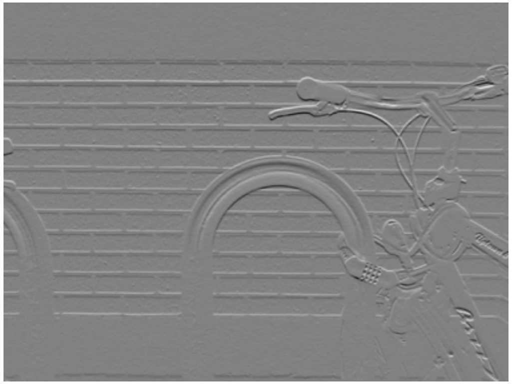
1	2	1
0	0	0
-1	-2	-1

Sobel filter: example

input



x sobel filtered



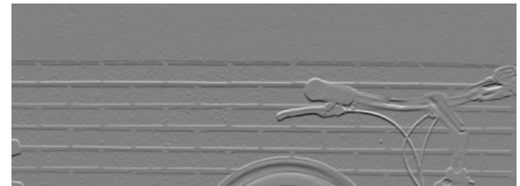
y sobel filtered

$\begin{bmatrix} a \\ b \end{bmatrix}$

$\sqrt{a^2 + b^2}$

Sobel filter: example





(as estimated by Sobel filters)

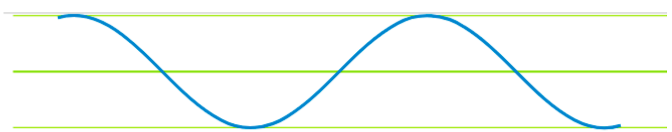
approximate gradient magnitude

Questions?

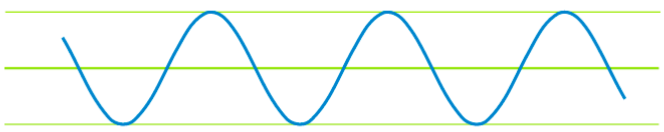
Let's talk about waves

- Frequency of a wave (e.g., a sine wave): how quickly does it cycle?

- A lower frequency wave:

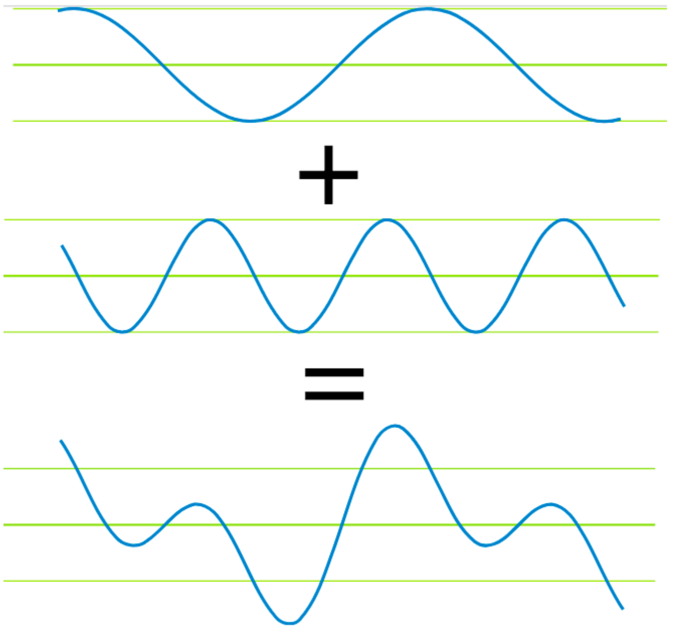


- A higher frequency wave:



Let's talk about waves

- We can add waves of different frequencies to get more complicated functions:



Let's talk about waves

(or: the least formal treatment of Fourier analysis you'll ever see)

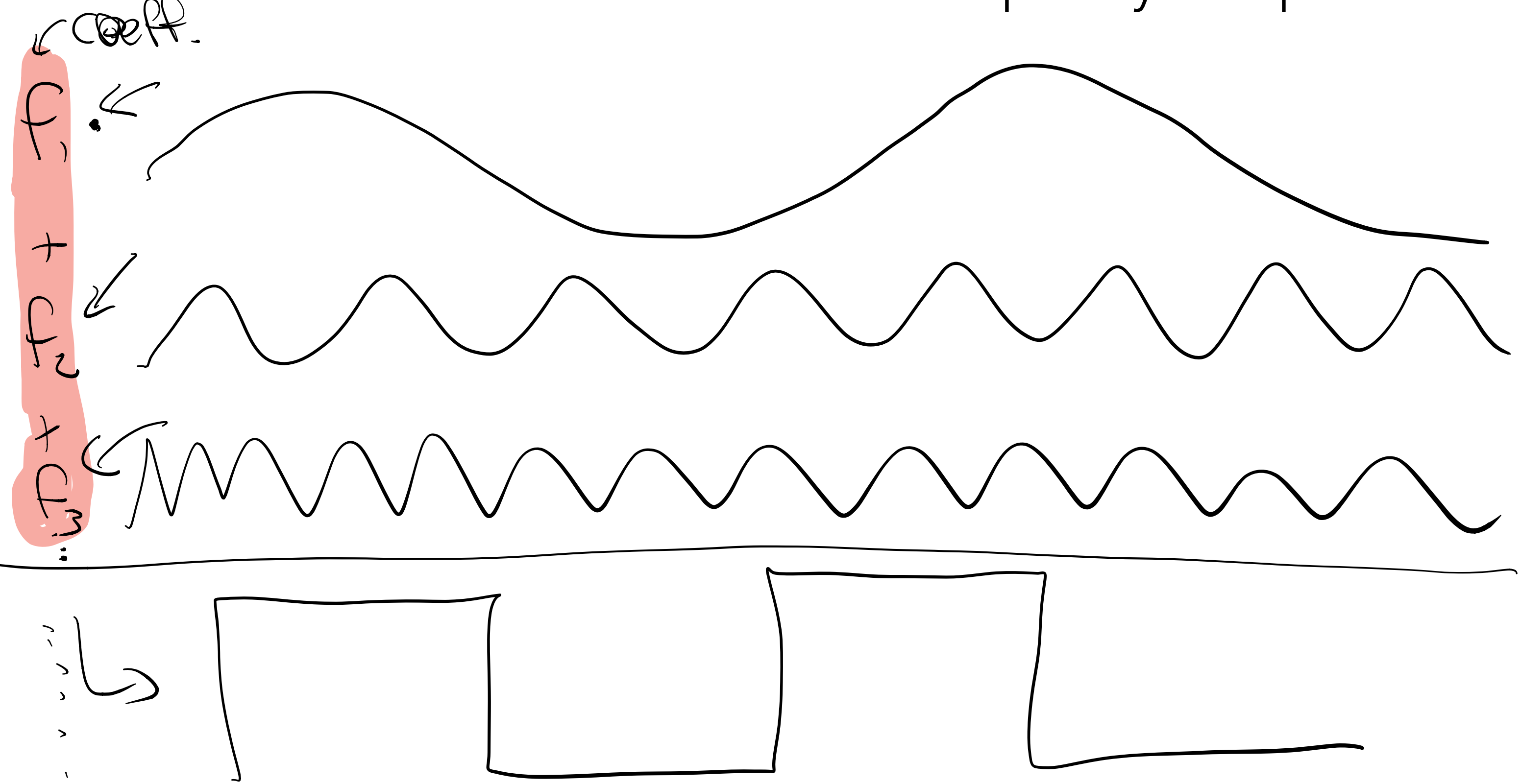
- It turns out: you can take **any** function and build it out of waves!

For the pictures (but you can read the math if you'd like!)
https://www.mathsisfun.com/calculus/fourier-series.html

This is the Fourier decomposition

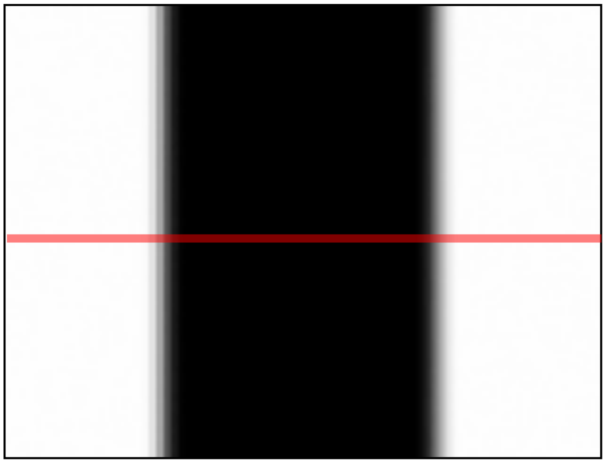
"Frequency content" of a signal

Refers to the coefficients of each frequency component.

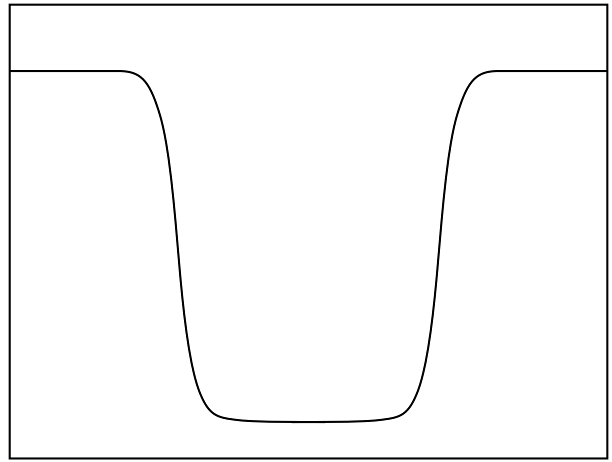


Let's talk about images

A scanline is a 1D function - we can build it out of waves!



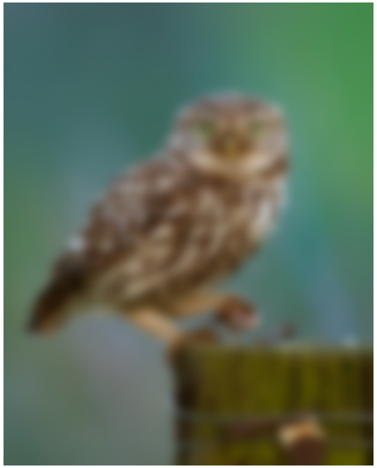
image

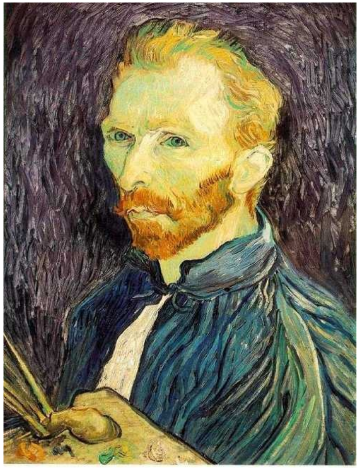


1D scanline

Let's talk about images

A scanline is a 1D function - we can build it out of waves!
Also (presented without proof): All of this generalizes to 2D.

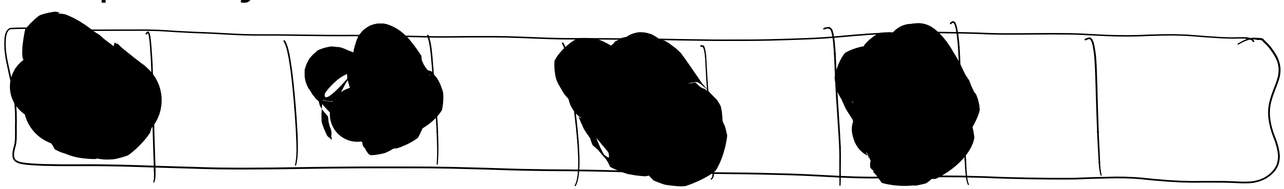




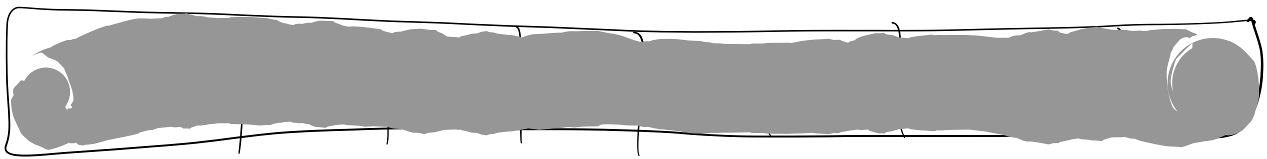
Exercise

- Here's a 1D image (or scanline):

- Design a scanline with the **most** possible high frequency content:



- Design a scanline with the **least** possible high frequency content:



Back to our definitions of image...

- Our digital image is a discrete, sampled version of some ideal continuous function.

- There's a limit to the frequencies we can represent

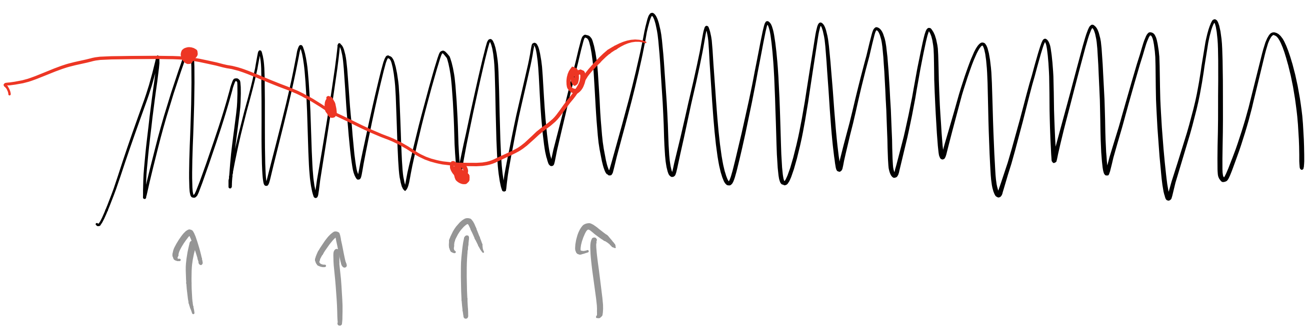
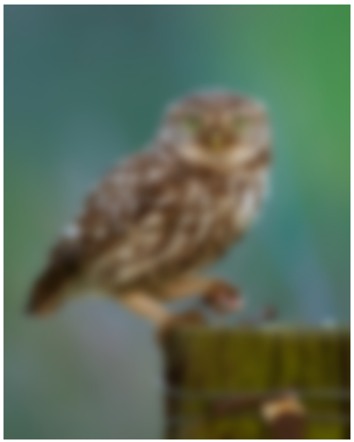
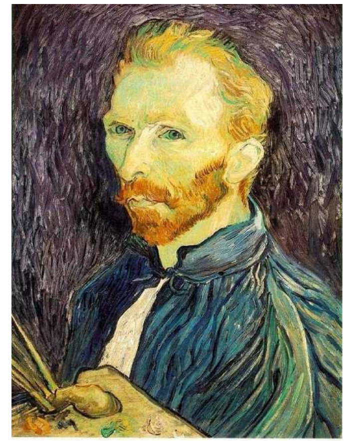


Image Frequency Content

Think: "how quickly pixels tend to change"



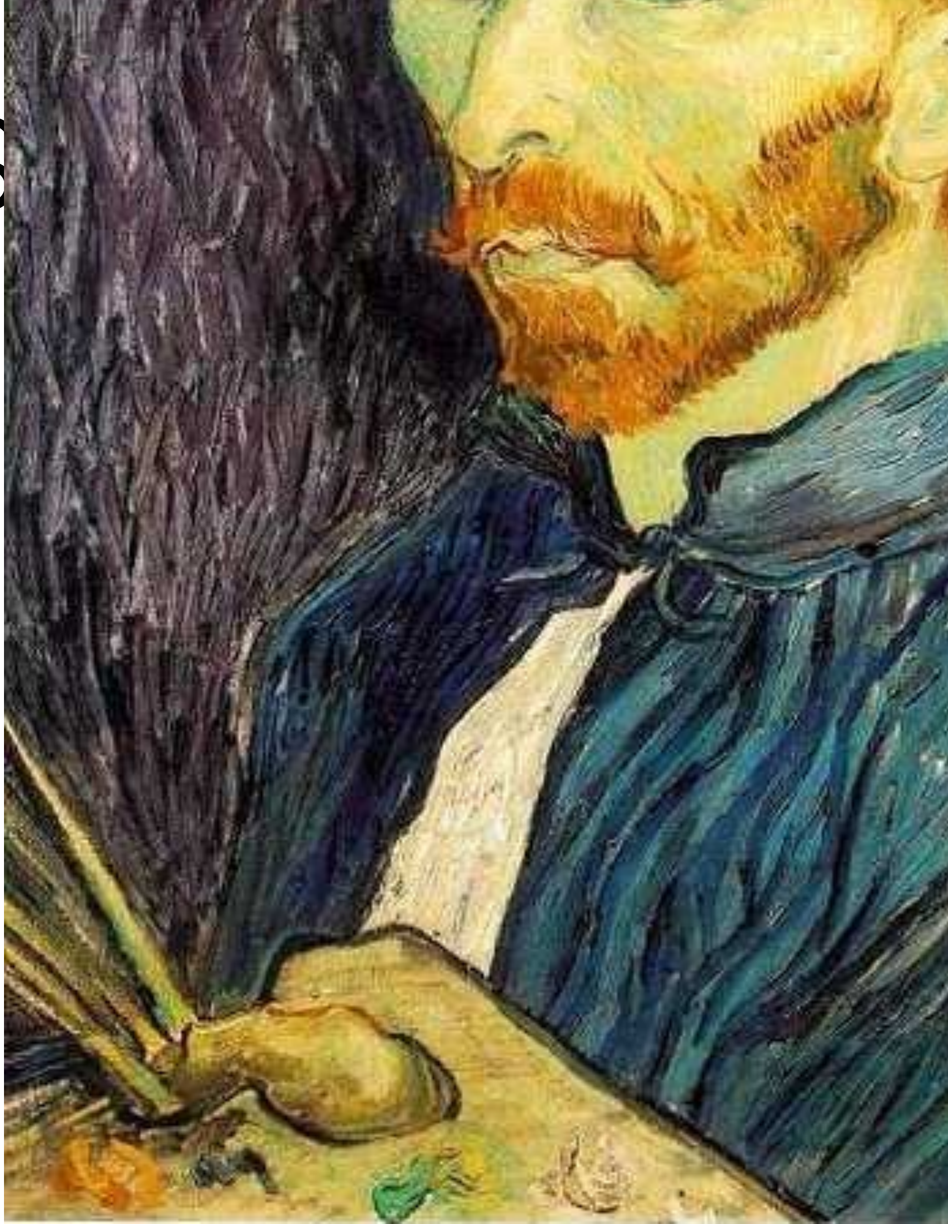


How does this relate to things we've seen so far?

- Blurring *removes* high frequencies
- Sharpening *amplifies* high frequencies
- Hard *edges* characterize high frequency content

Questions?

Image S

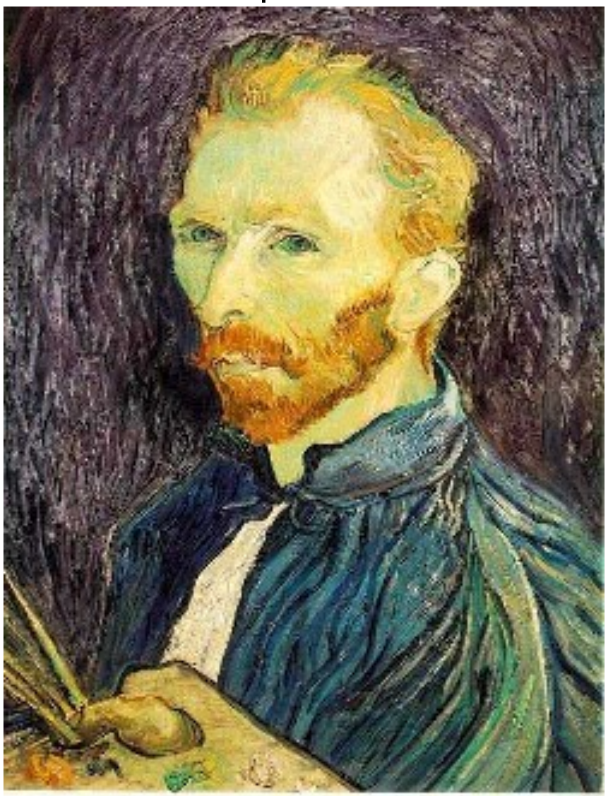


This image is too big to fit on the screen. How can we generate a half-sized version?

Image Subsampling

Only keep every...

2nd pixel



1/2

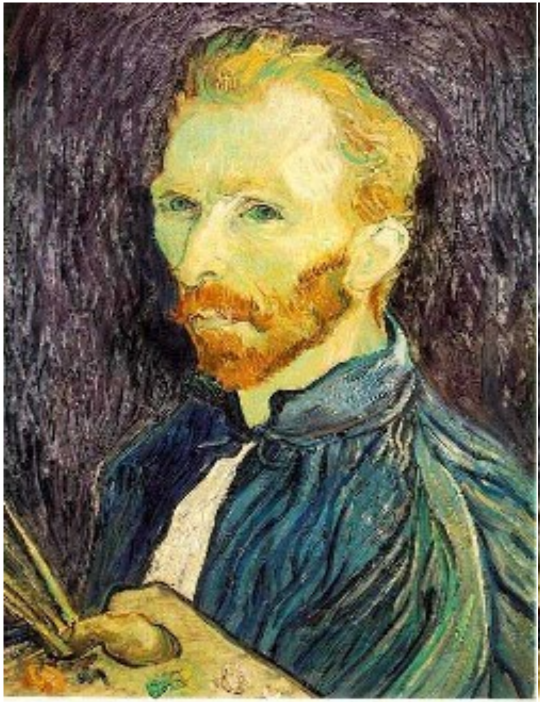
4th pixel

1/4

8th pixel

1/8

Image Subsampling

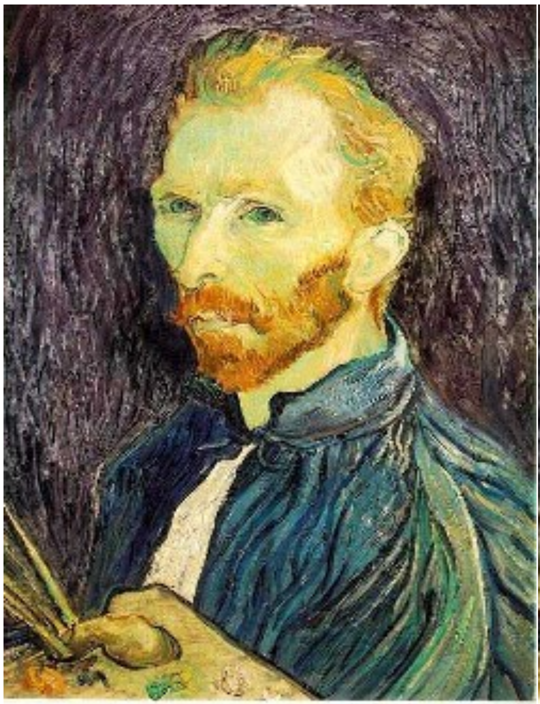


1/2

1/4

1/8

Image Subsampling



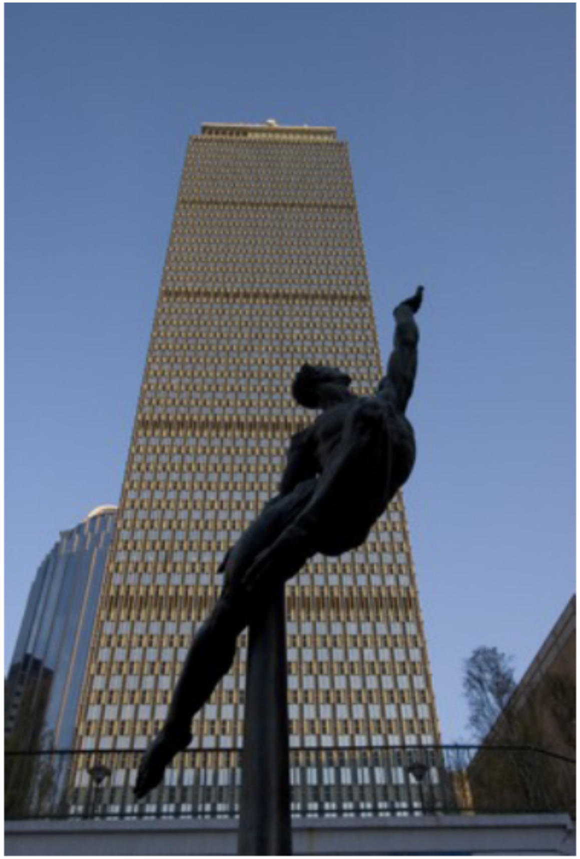
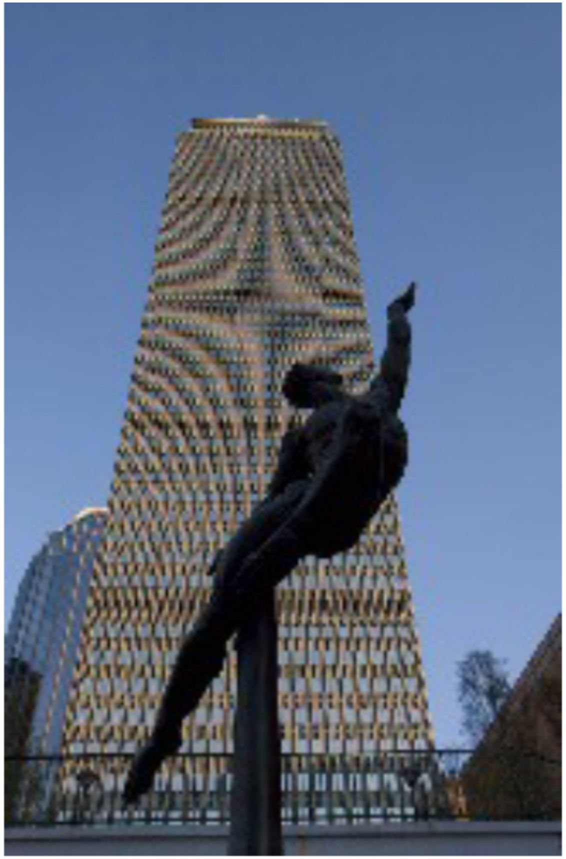
1/2

1/4

1/8

Why does this look so crufty?

Subsampling: Another example

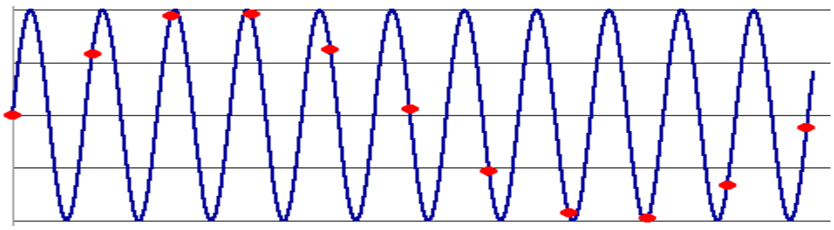
 

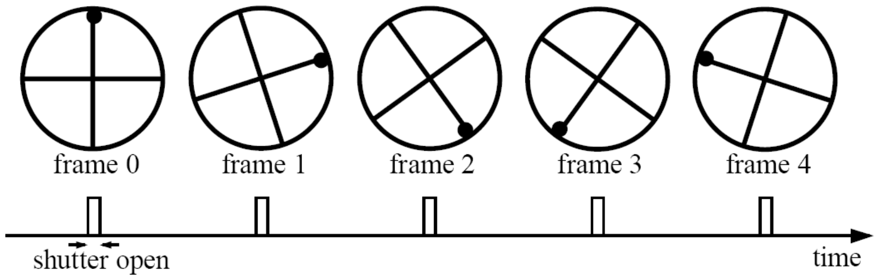
Source: Fredo Durand

Aliasing

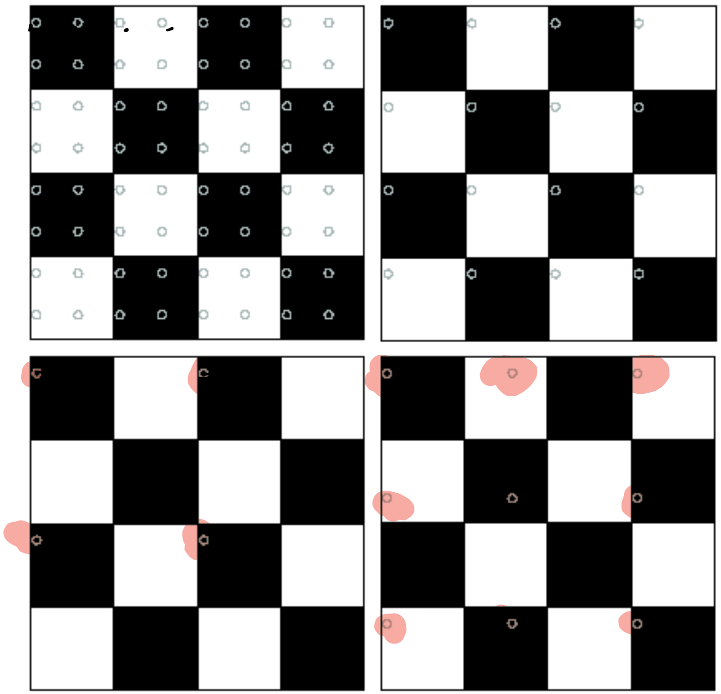
- Let's look back at our highest-frequency scanline:

Aliasing





Aliasing in 2D



good

Aliasing

- Let's look back at our highest-frequency scanline:

- What's the "right" (i.e., best we can do) answer?

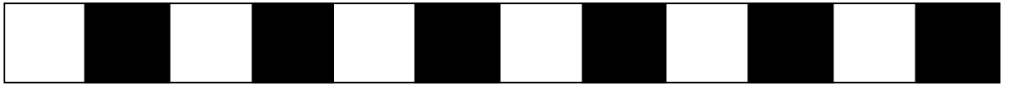
Aliasing

- Let's look back at our highest-frequency scanline:

- What's the "right" (i.e., best we can do) answer?
- If we walked far away, what we'd see is:

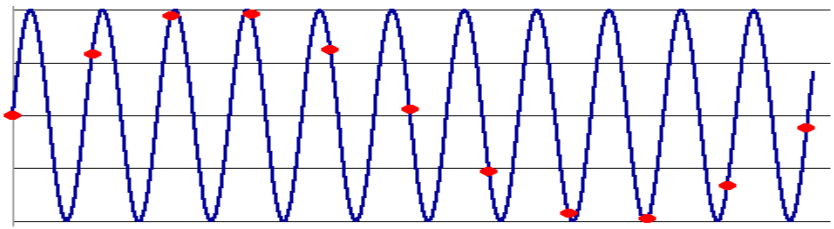
Aliasing

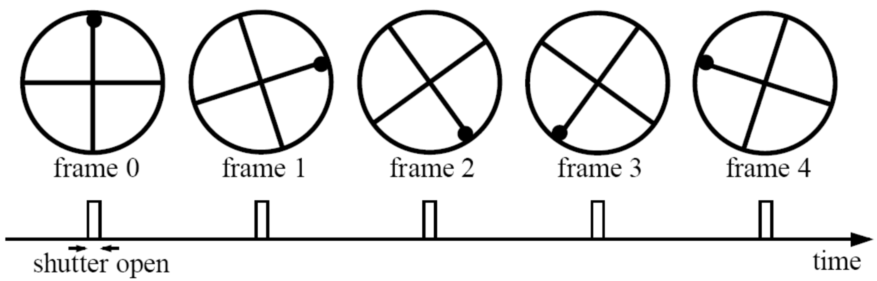
- Let's look back at our highest-frequency scanline:



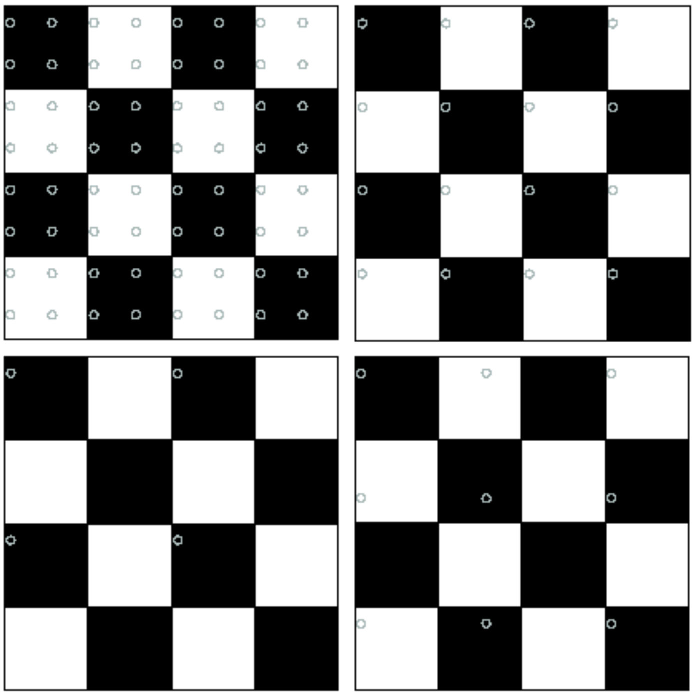
- What's the "right" (i.e., best we can do) answer?
- If we walked far away, what we'd see is:

Aliasing





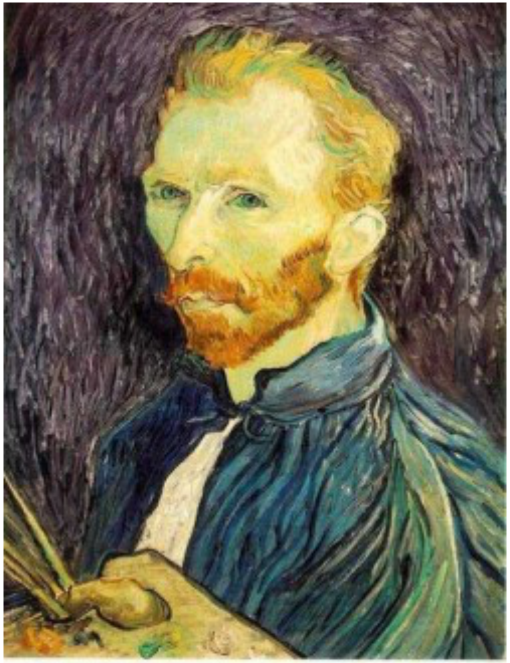
Aliasing in 2D



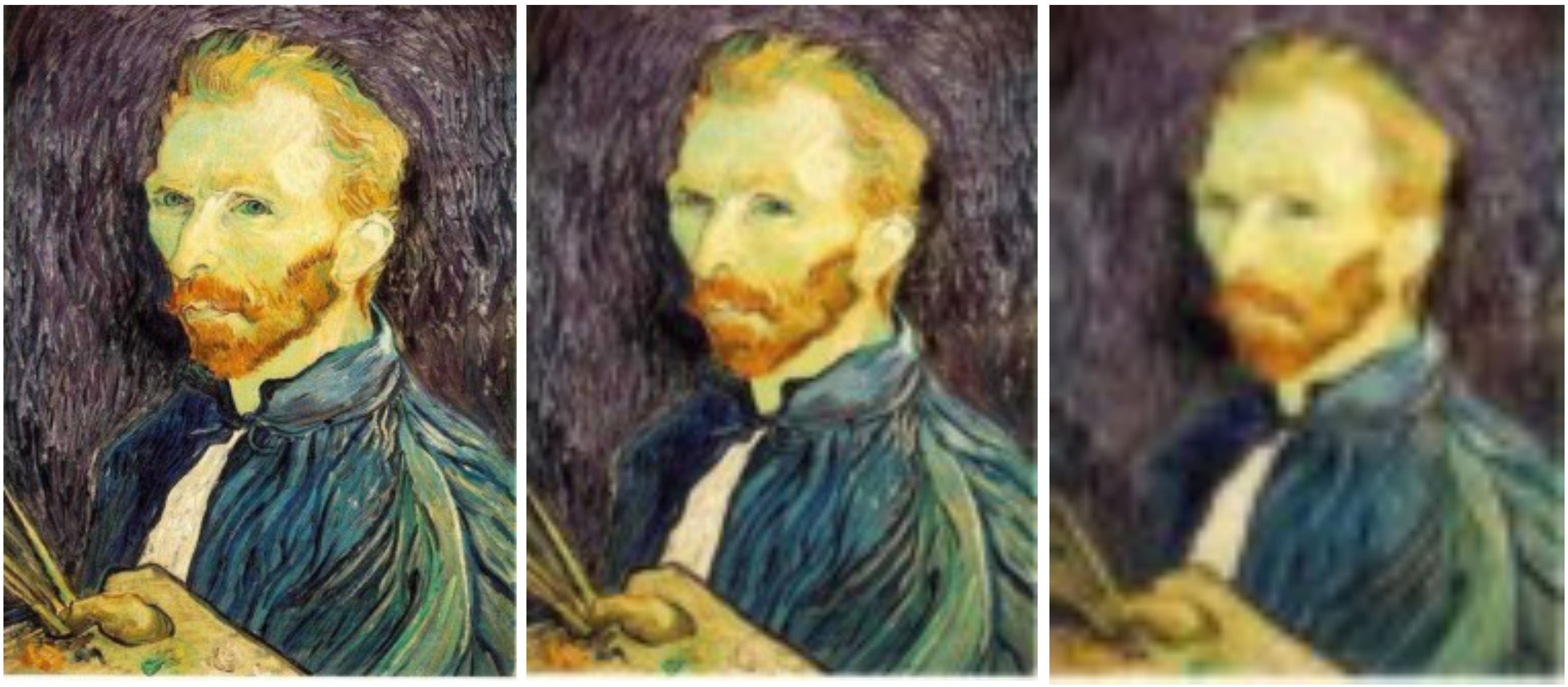
Aliasing - the best we can do

Blurring *removes* high frequencies.

So: blur (pre-filter) the image, *then* subsample it.



Downsampling with Gaussian Pre-filtering

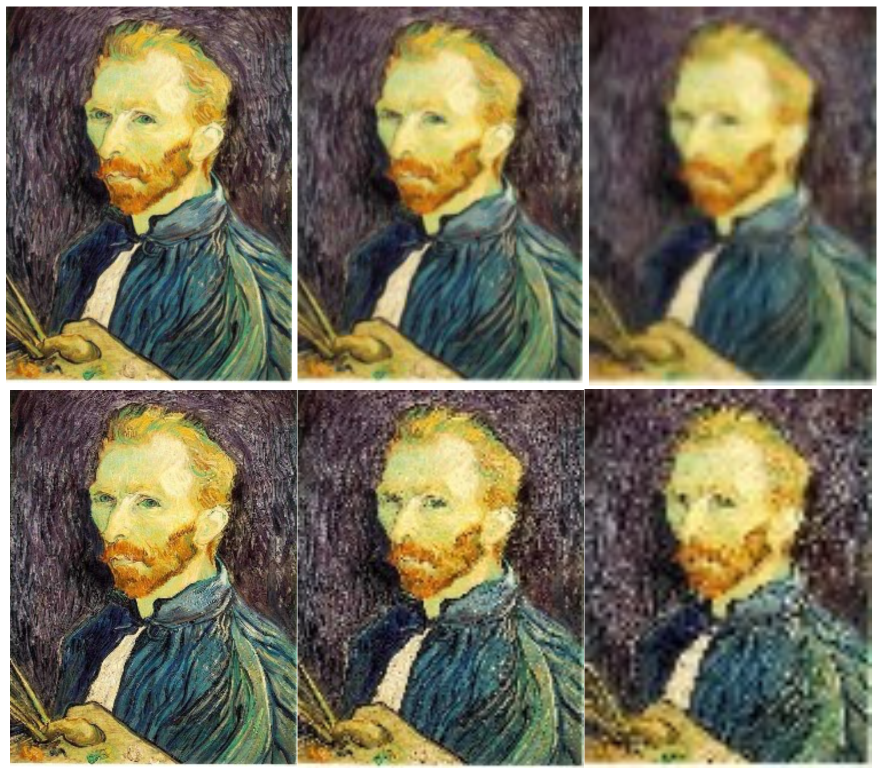


1/2 1/4 1/8

Downsampling with Gaussian Pre-filtering

with

without



1/2 1/4 1/8