

CS4670: Computer Vision

Noah Snavely

Lecture 4: Image Resampling

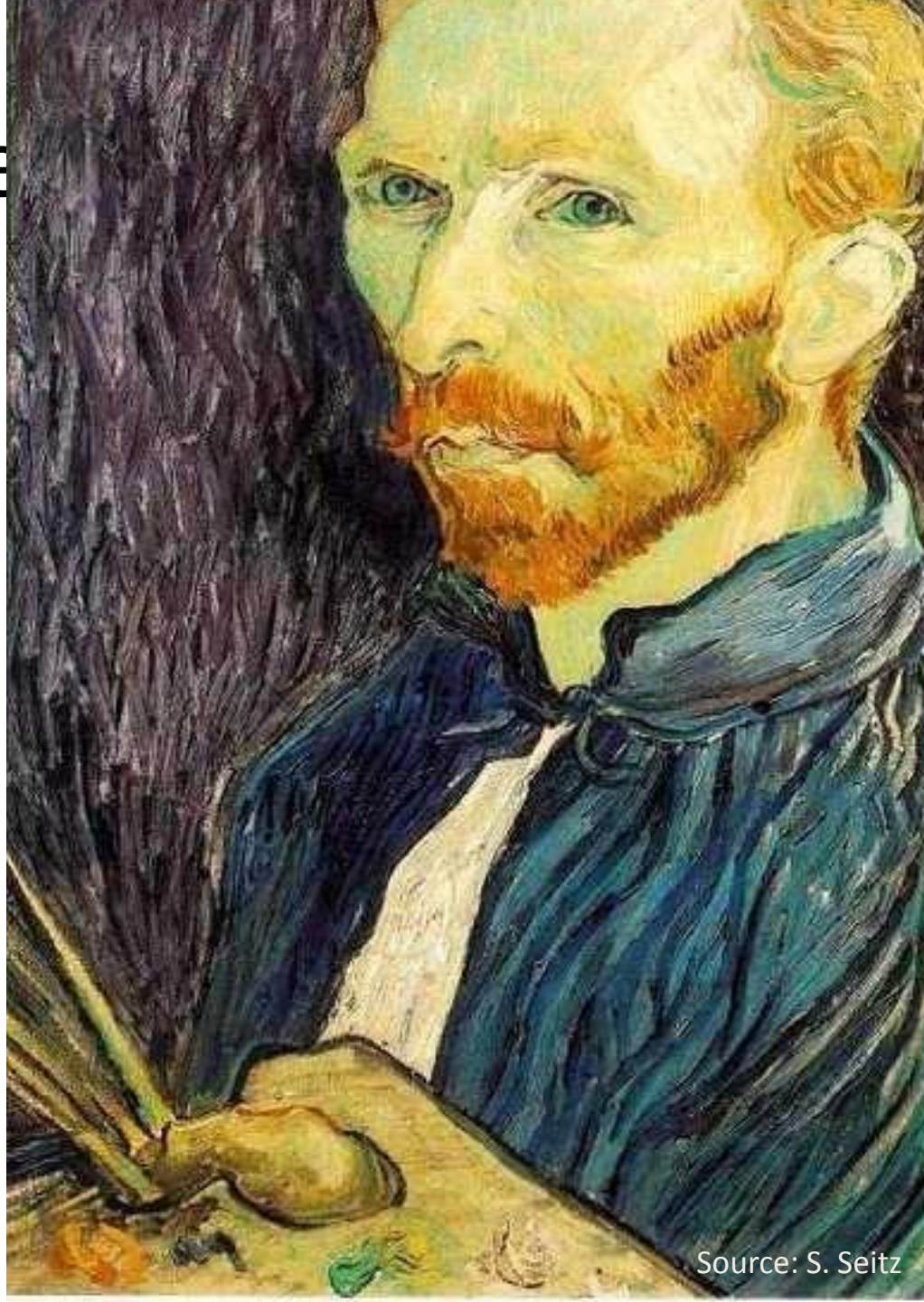


Announcements

- Project 1 assigned today 9/3
 - Part 1 due in one week (9/10), turnin via CMS
 - Kevin will demo at the end of class
 - Part 2 (using the cameraphone) due shortly after
- Guest lectures next week (and MW of week after)

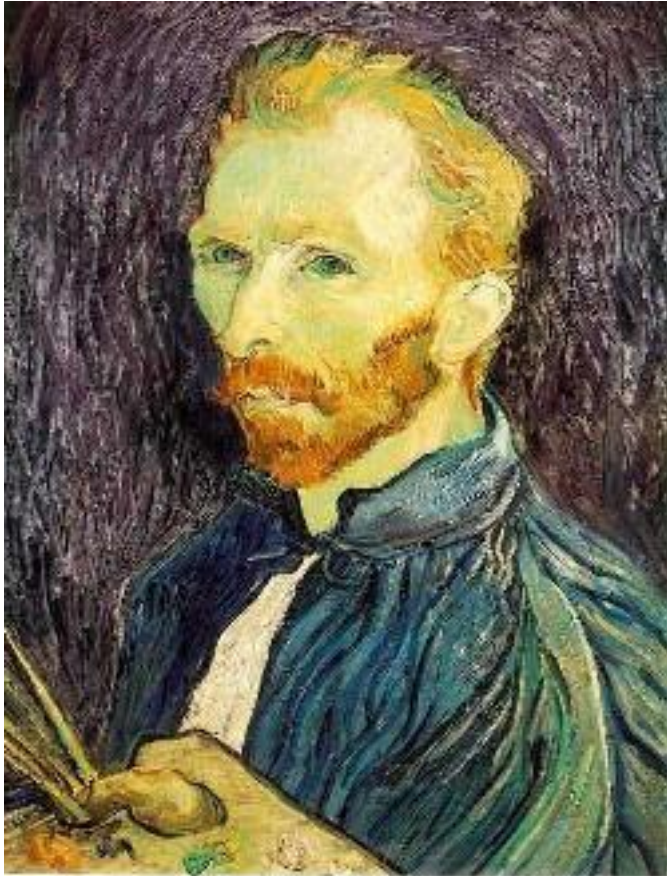
Image

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



Source: S. Seitz

Image sub-sampling



1/4



1/8

Throw away every other row and column to create a 1/2 size image
- called *image sub-sampling*

Image sub-sampling



1/2



1/4 (2x zoom)



1/8 (4x zoom)

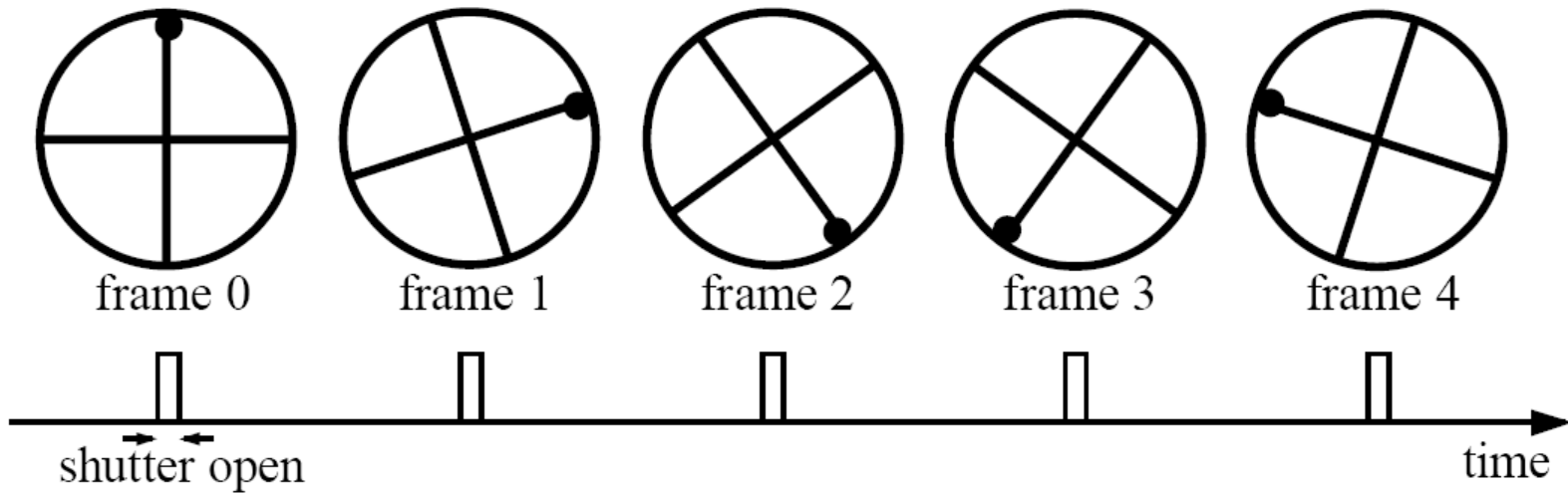
Why does this look so cruffy?

Wagon-wheel effect

Imagine a spoked wheel moving to the right (rotating clockwise).

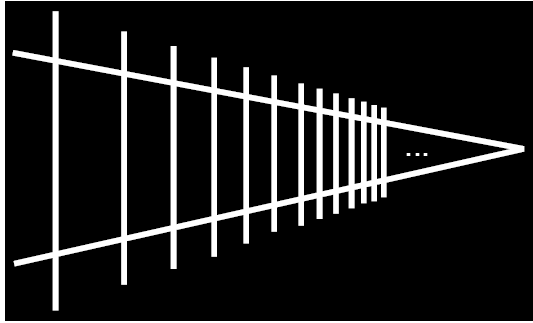
Mark wheel with dot so we can see what's happening.

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)

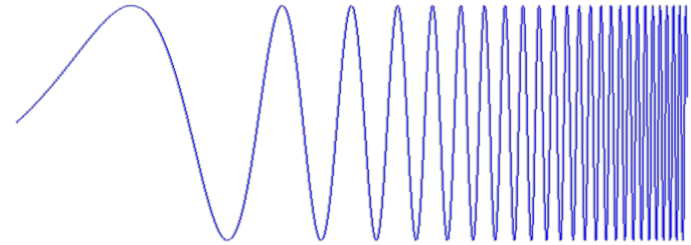
Alias: n., an assumed name



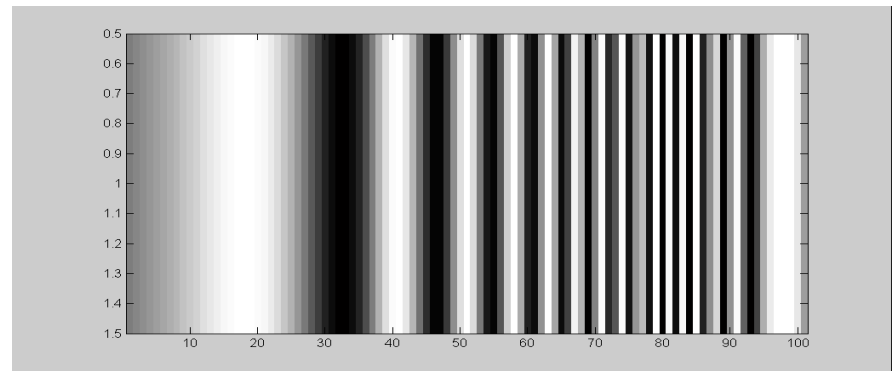
Picket fence receding
Into the distance will
produce aliasing...

WHY?

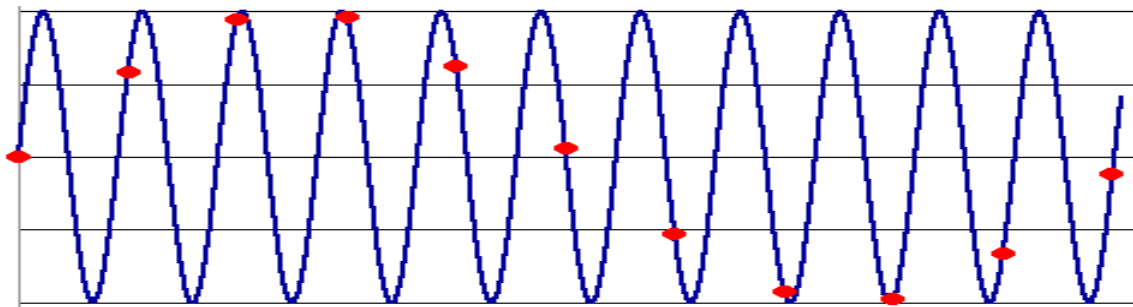
Input signal:



Matlab output:



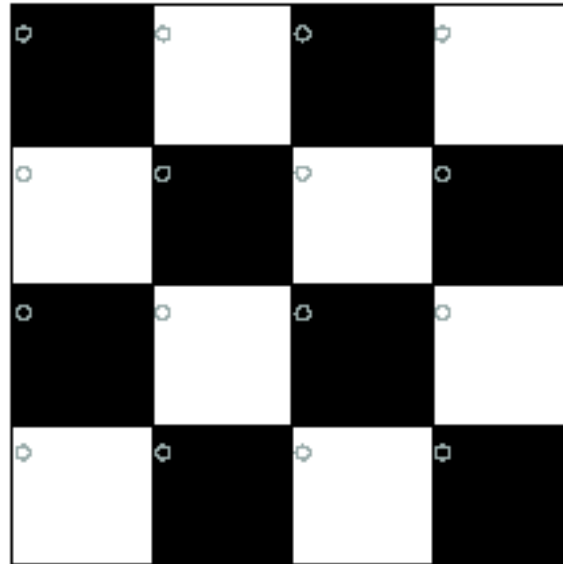
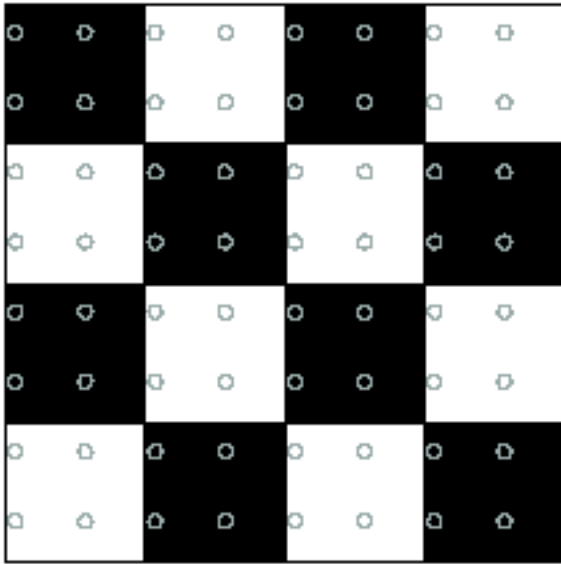
`x = 0:.05:5; imagesc(sin((2.^x).*x))`



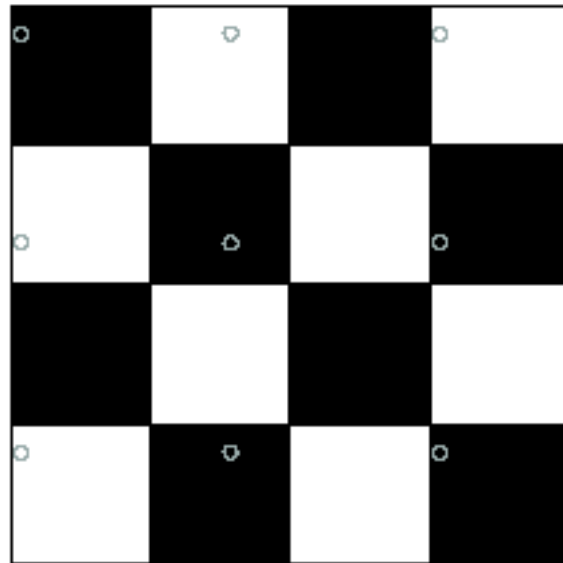
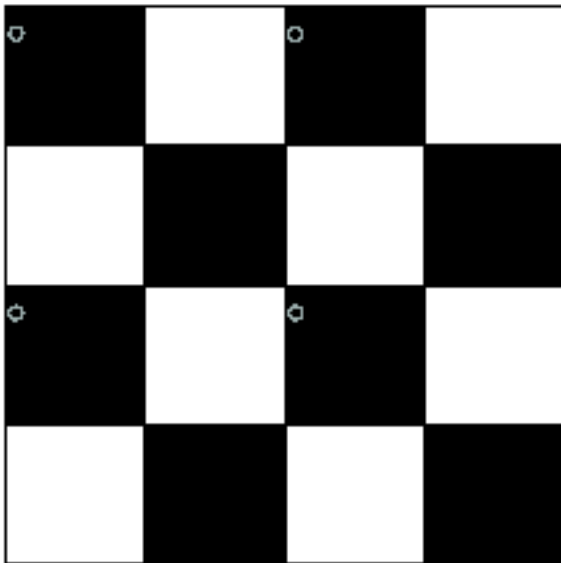
Not enough samples

Aj-aj-aj:
Alias!

Nyquist limit – 2D example



Good sampling



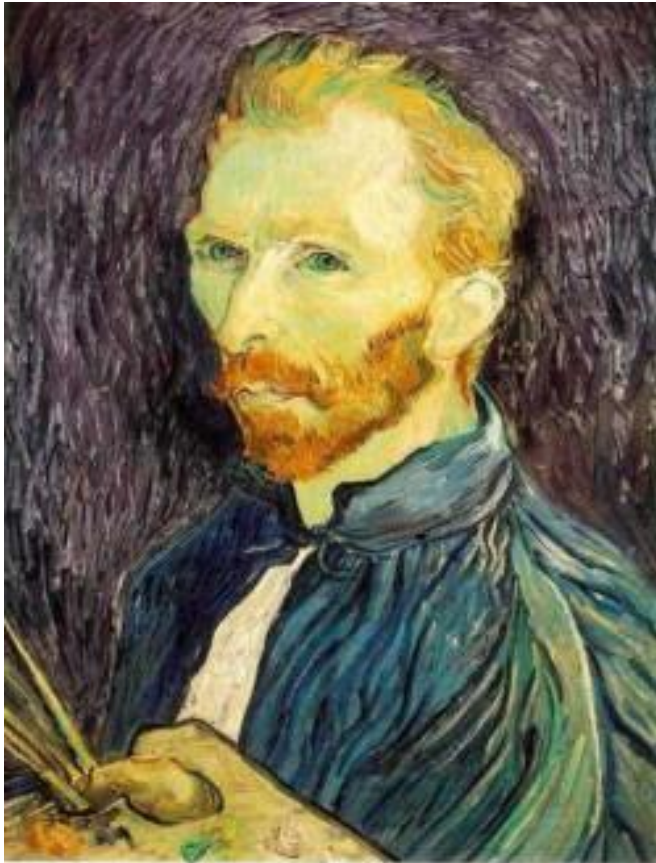
Bad sampling

Aliasing

- When downsampling by a factor of two
 - Original image has frequencies that are too high

- How can we fix this?

Gaussian pre-filtering



Gaussian 1/2



G 1/4



G 1/8

- Solution: filter the image, *then* subsample

Subsampling with Gaussian pre-filtering



Gaussian 1/2



G 1/4



G 1/8

- Solution: filter the image, *then* subsample

Compare with...



1/2



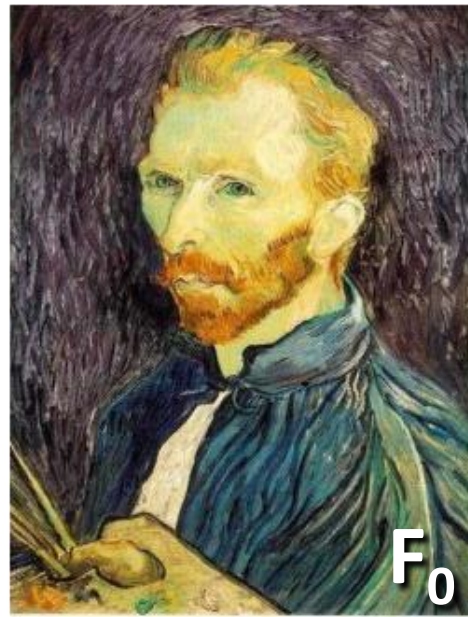
1/4 (2x zoom)



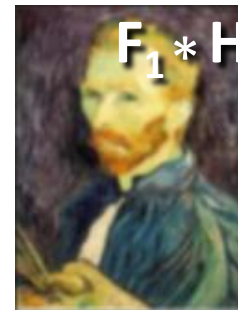
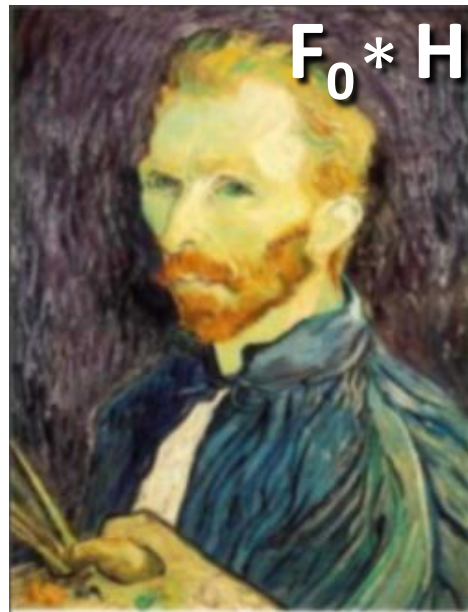
1/8 (4x zoom)

Gaussian pre-filtering

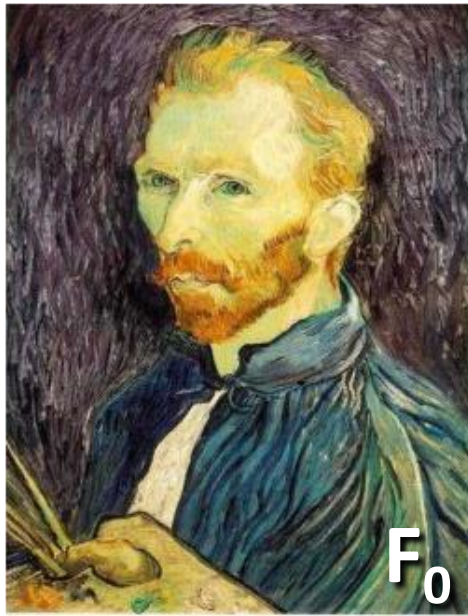
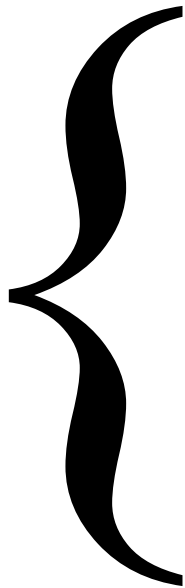
- Solution: filter the image, *then* subsample



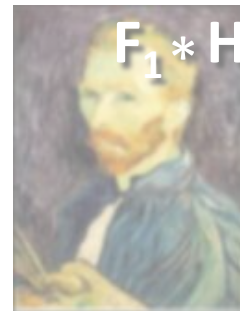
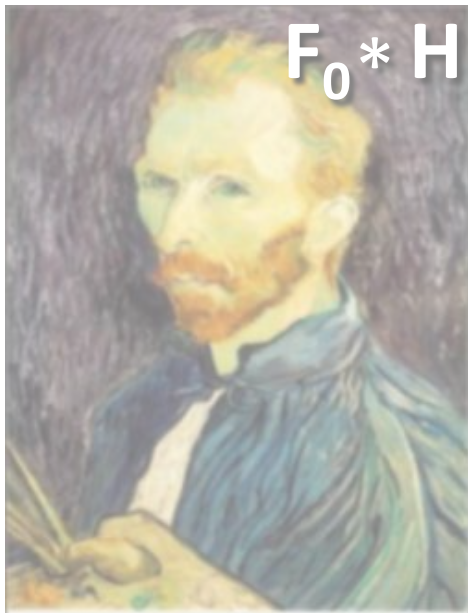
...



Gaussian pyramid



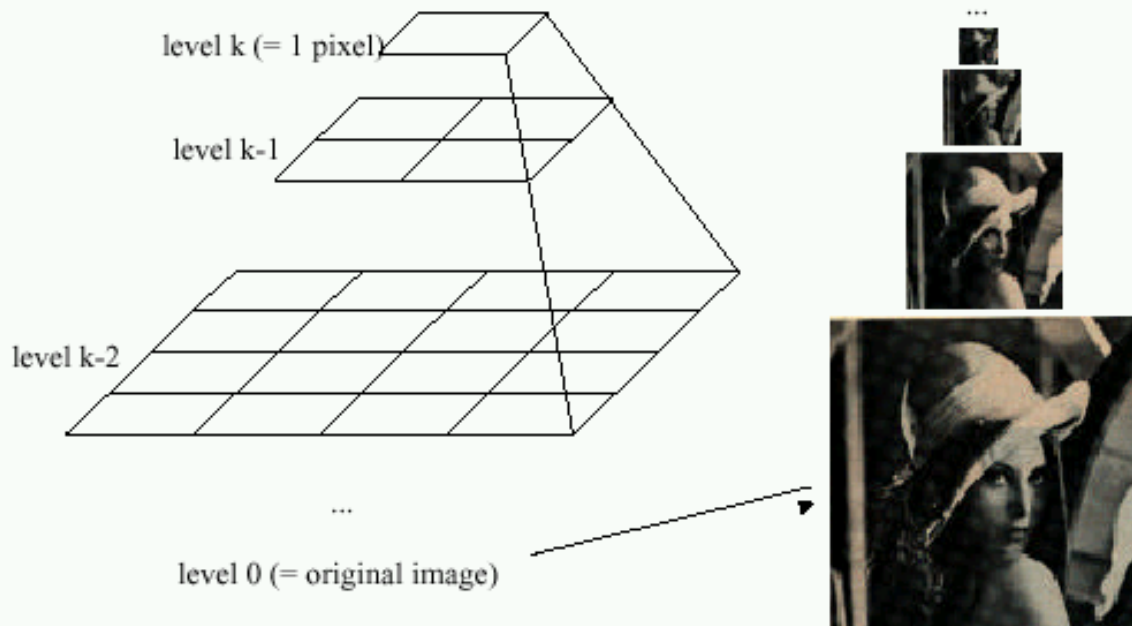
...



Gaussian pyramids

[Burt and Adelson, 1983]

Idea: Represent $N \times N$ image as a “pyramid” of $1 \times 1, 2 \times 2, 4 \times 4, \dots, 2^k \times 2^k$ images (assuming $N=2^k$)



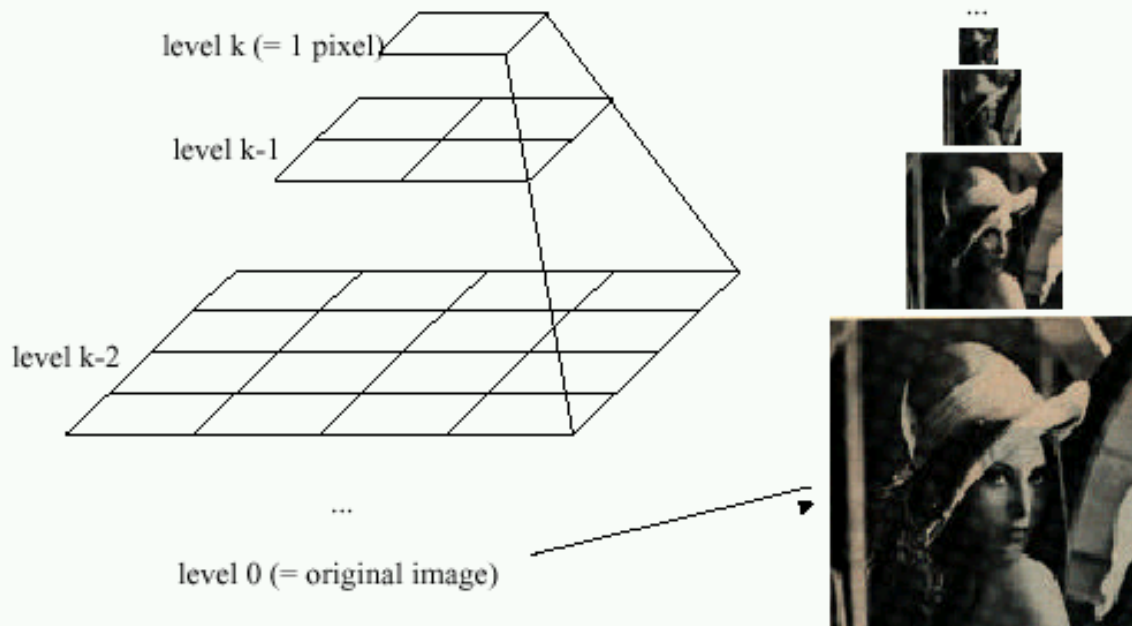
- In computer graphics, a *mip map* [Williams, 1983]
- A precursor to *wavelet transform*

Gaussian Pyramids have all sorts of applications in computer vision

Gaussian pyramids

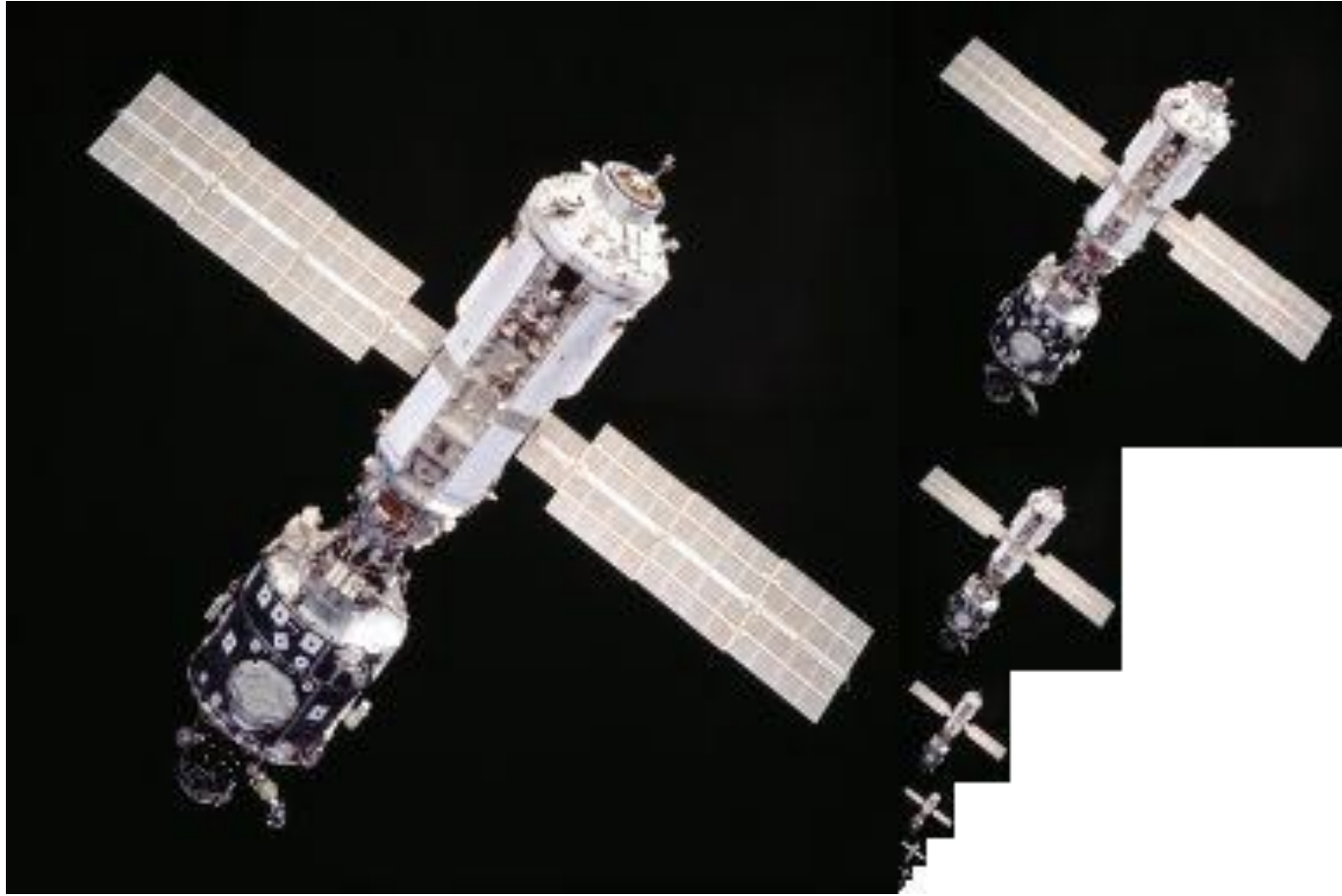
[Burt and Adelson, 1983]

Idea: Represent $N \times N$ image as a “pyramid” of $1 \times 1, 2 \times 2, 4 \times 4, \dots, 2^k \times 2^k$ images (assuming $N=2^k$)



- How much space does a Gaussian pyramid take compared to the original image?

Gaussian Pyramid



Questions?

Upsampling


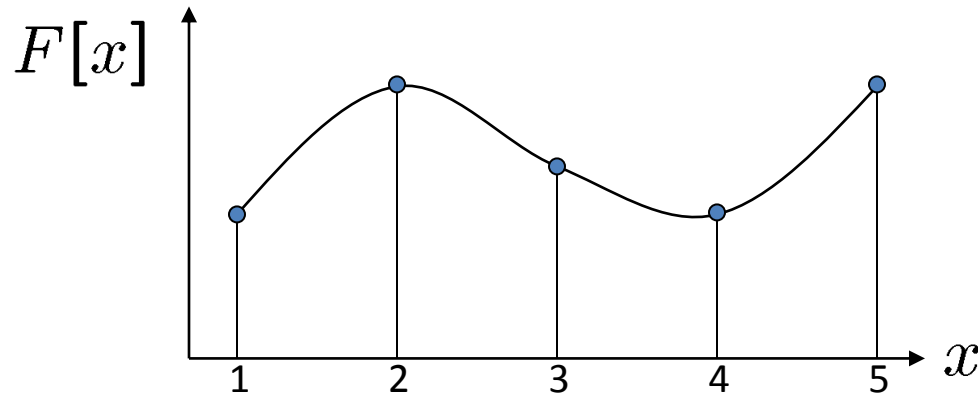
- This image is too small for this screen: 
- How can we make it 10 times as big?
- Simplest approach:
 - repeat each row
 - and column 10 times
- (“Nearest neighbor interpolation”)



Image interpolation



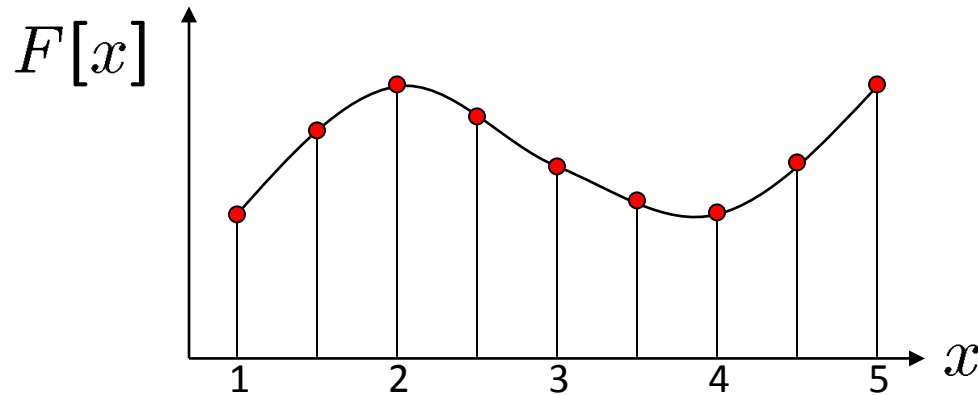
$d = 1$ in this example

Recall how a digital image is formed

$$F[x, y] = \text{quantize}\{f(xd, yd)\}$$

- It is a discrete point-sampling of a continuous function
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale

Image interpolation



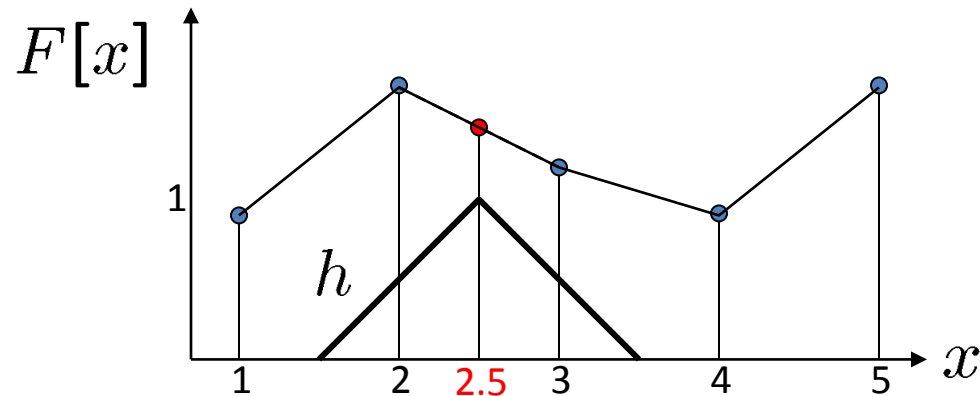
$d = 1$ in this example

Recall how a digital image is formed

$$F[x, y] = \text{quantize}\{f(xd, yd)\}$$

- It is a discrete point-sampling of a continuous function
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale

Image interpolation



$d = 1$ in this example

- What if we don't know f ?

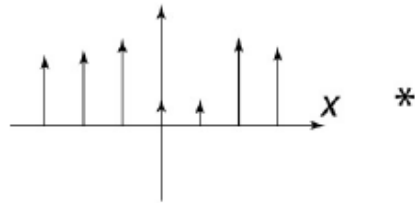
- Guess an approximation: \tilde{f}
- Can be done in a principled way: filtering
- Convert F to a continuous function:

$$f_F(x) = F\left(\frac{x}{d}\right) \text{ when } \frac{x}{d} \text{ is an integer, } 0 \text{ otherwise}$$

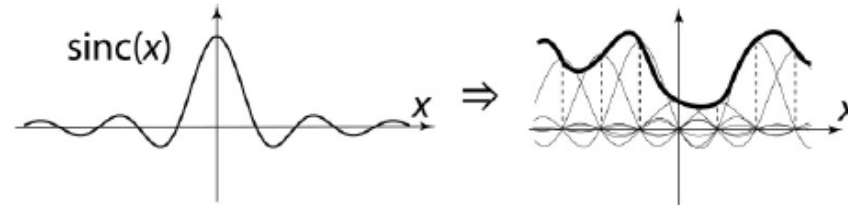
- Reconstruct by convolution with a *reconstruction filter*, h

$$\tilde{f} = h * f_F$$

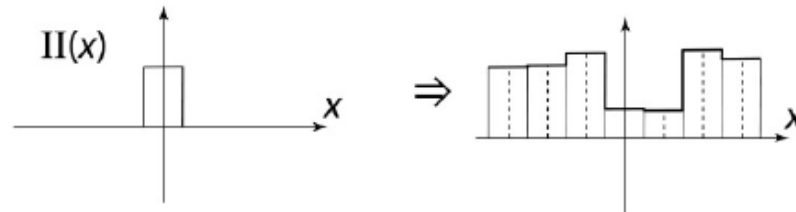
Image interpolation



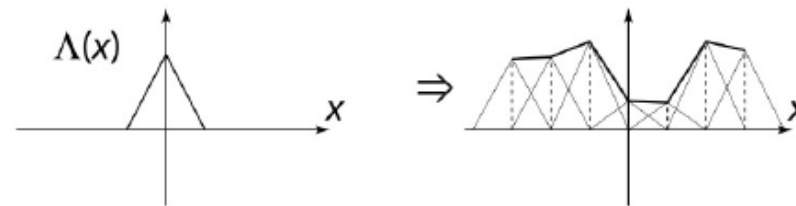
*



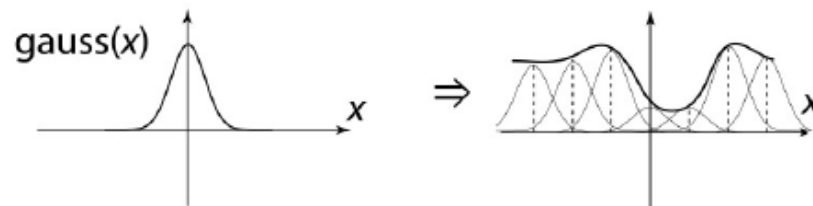
“Ideal” reconstruction



Nearest-neighbor interpolation



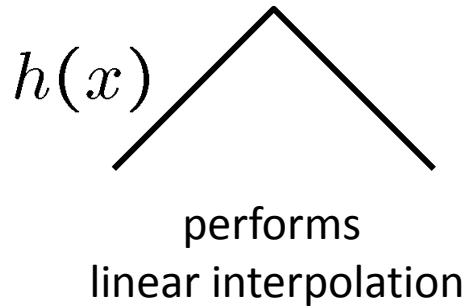
Linear interpolation



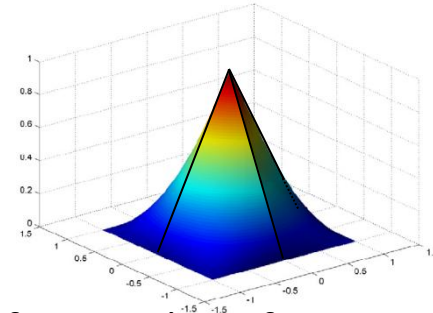
Gaussian reconstruction

Reconstruction filters

- What does the 2D version of this hat function look like?



$h(x, y)$



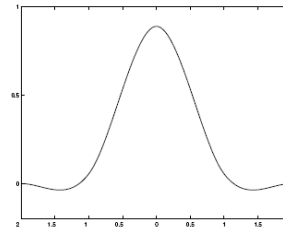
(tent function) performs
bilinear interpolation

Often implemented without cross-correlation

- E.g., http://en.wikipedia.org/wiki/Bilinear_interpolation

Better filters give better resampled images

- **Bicubic** is common choice



Cubic reconstruction filter

$$r(x) = \frac{1}{6} \begin{cases} (12 - 9B - 6C)|x|^3 + (-18 + 12B + 6C)|x|^2 + (6 - 2B) & |x| < 1 \\ ((-B - 6C)|x|^3 + (6B + 30C)|x|^2 + (-12B - 48C)|x| + (8B + 24C)) & 1 \leq |x| < 2 \\ 0 & \text{otherwise} \end{cases}$$

Image interpolation

Original image:  x 10



Nearest-neighbor interpolation



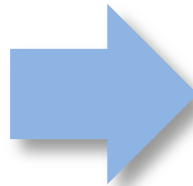
Bilinear interpolation



Bicubic interpolation

Image interpolation

Also used for *resampling*



Questions?