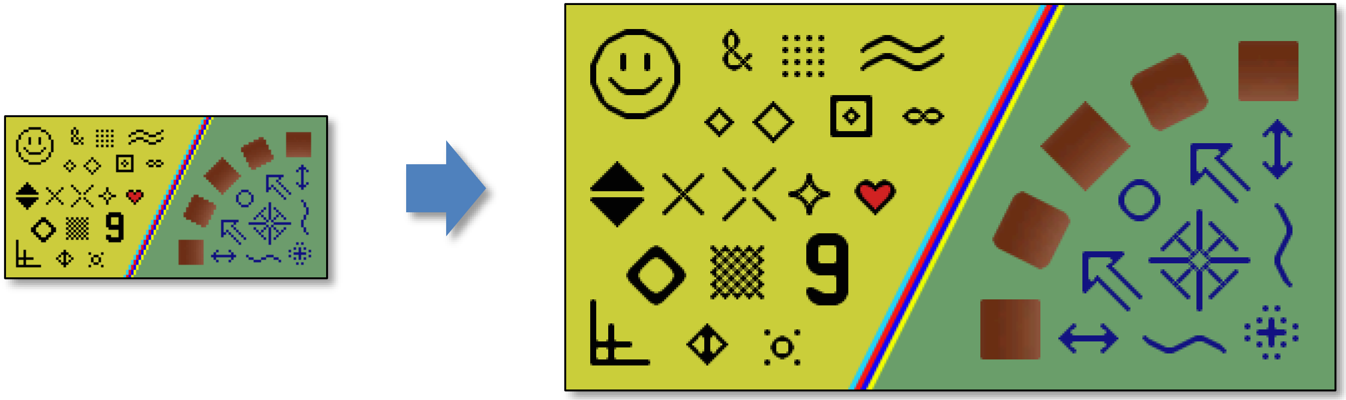
CS4670: Computer Vision

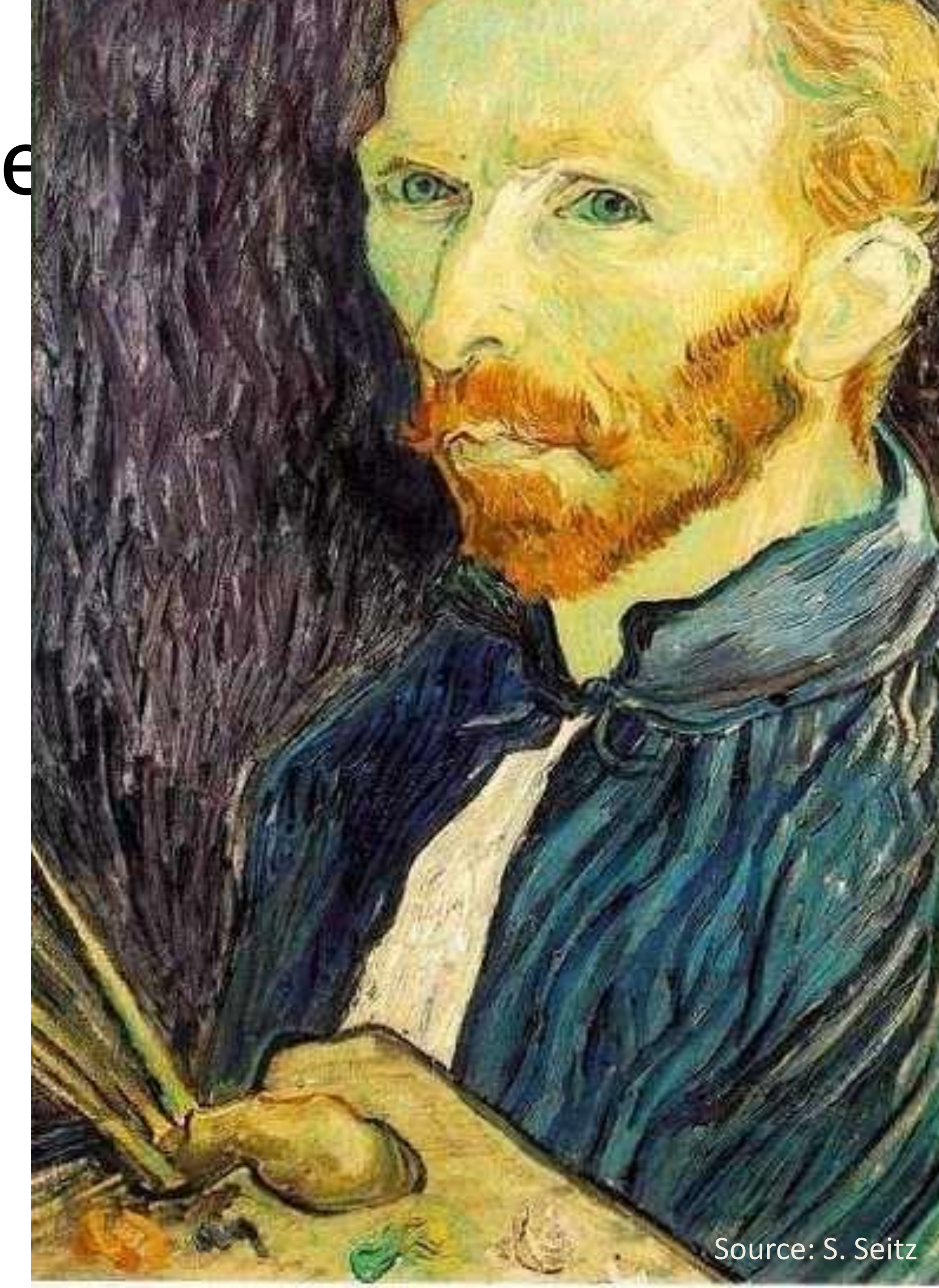
Noah Snavely

Image Resampling



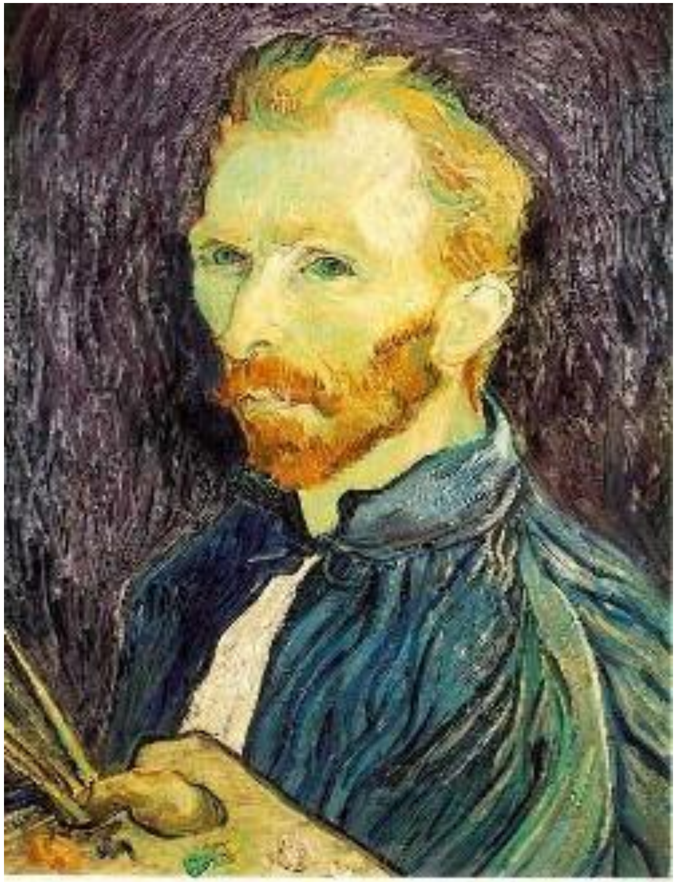
Image

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



Source: S. Seitz

Image sub-sampling



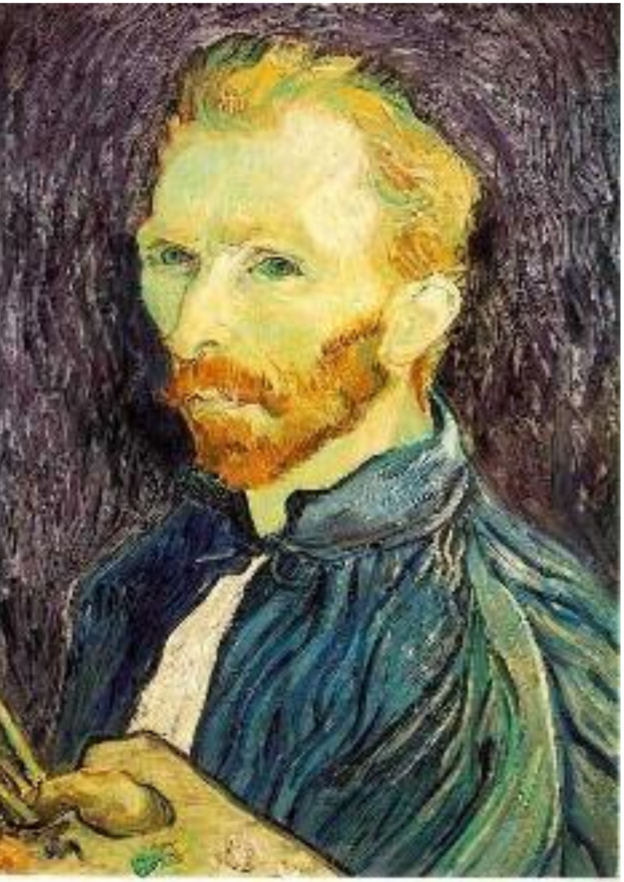
Throw away every other row and column to create a 1/2 size image
- called *image sub-sampling*

1/4

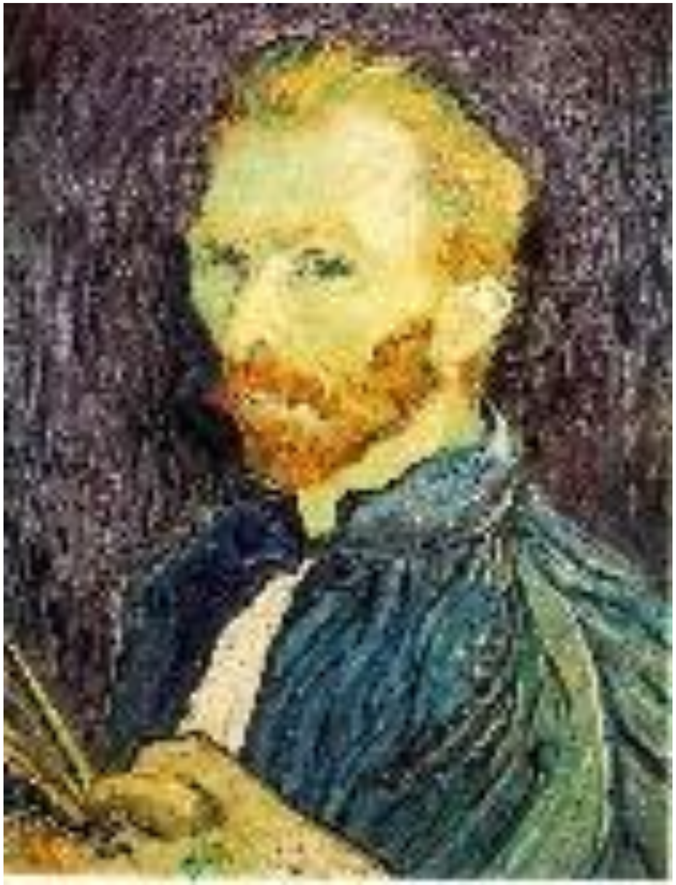
1/8

Source: S. Seitz

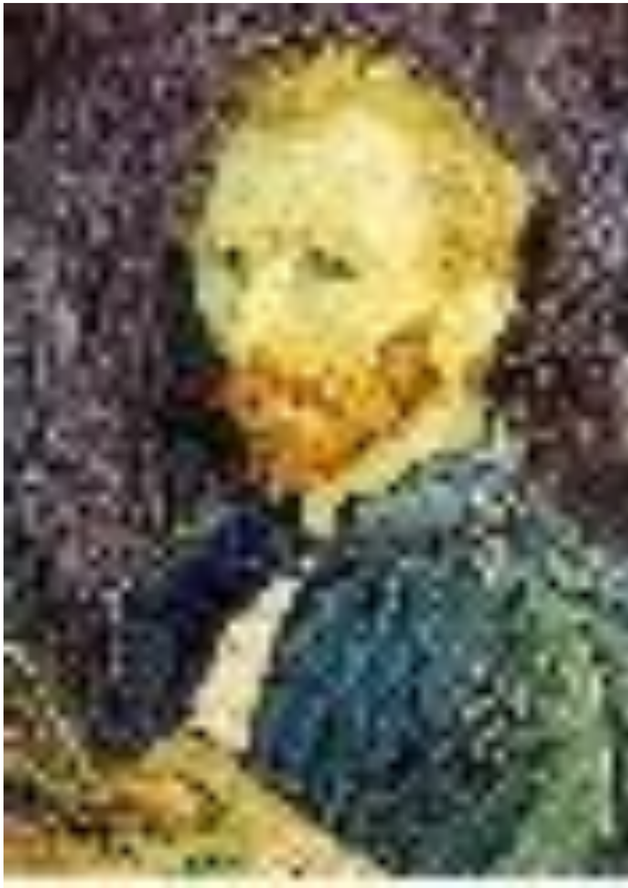
Image sub-sampling



1/2



1/4 (2x zoom)

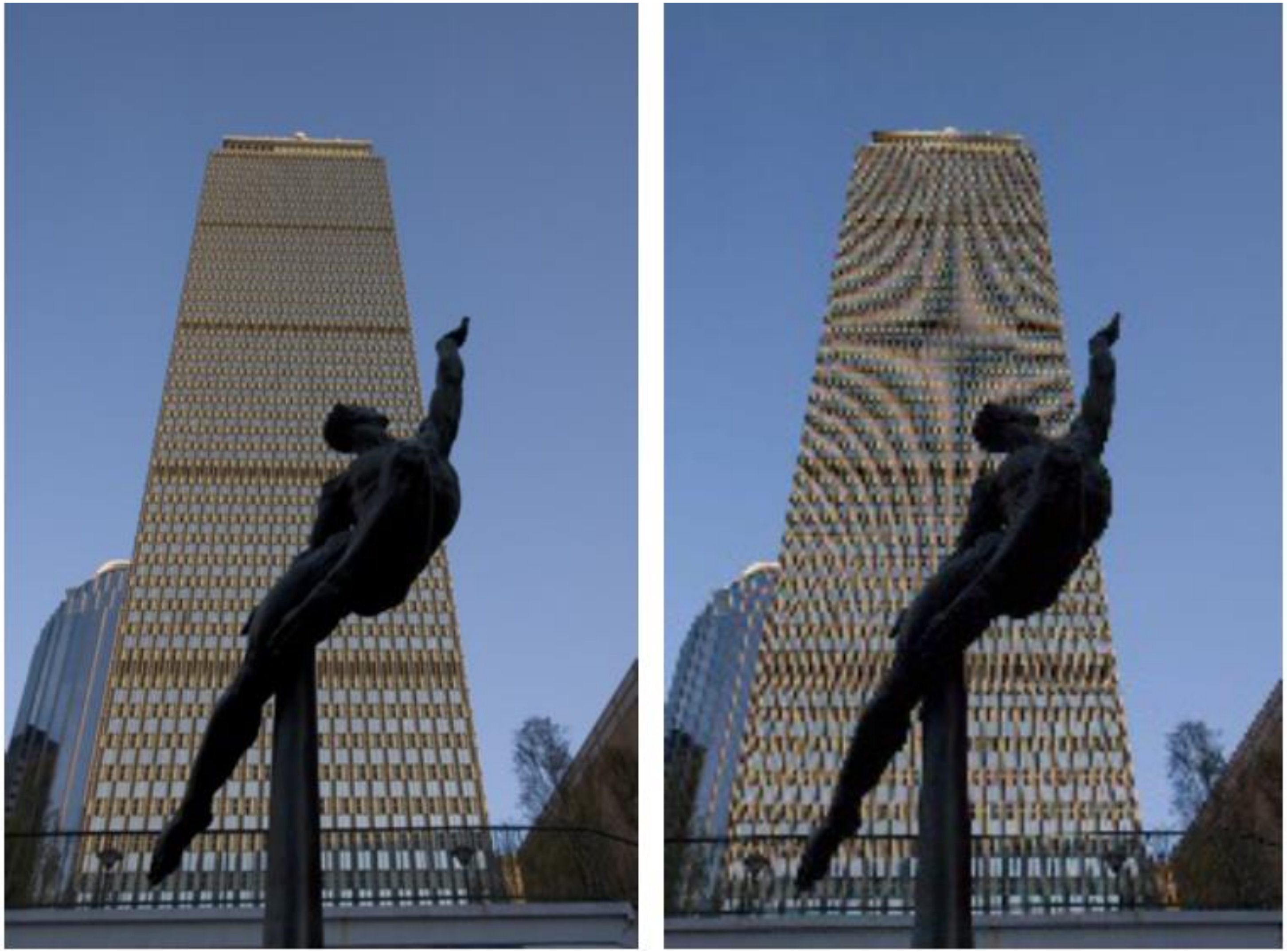


1/8 (4x zoom)

Why does this look so crufty?

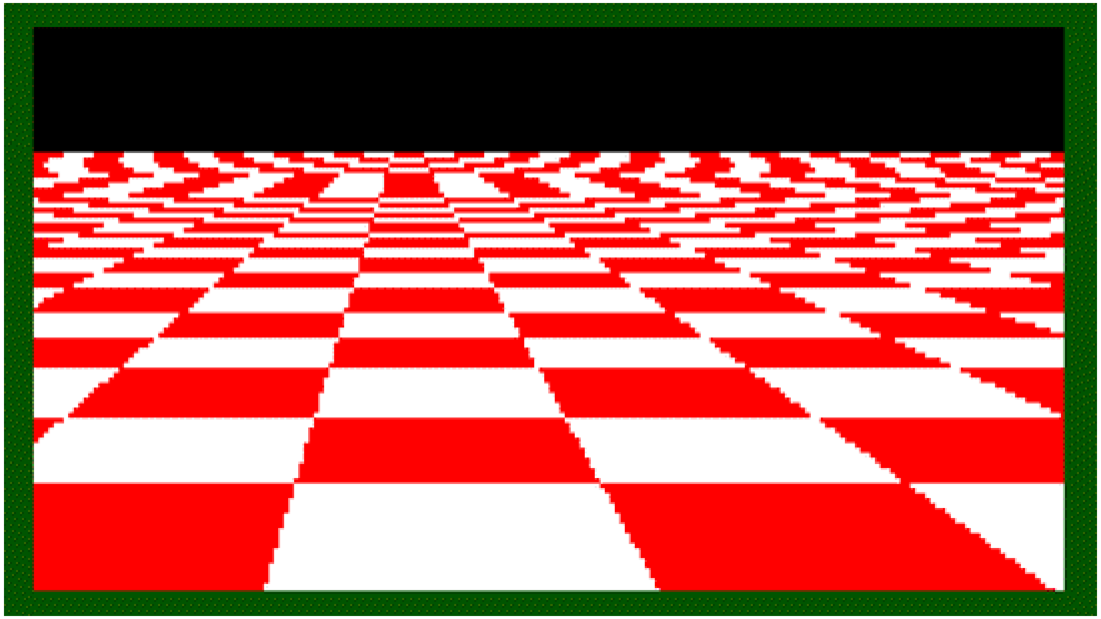
Source: S. Seitz

Image sub-sampling



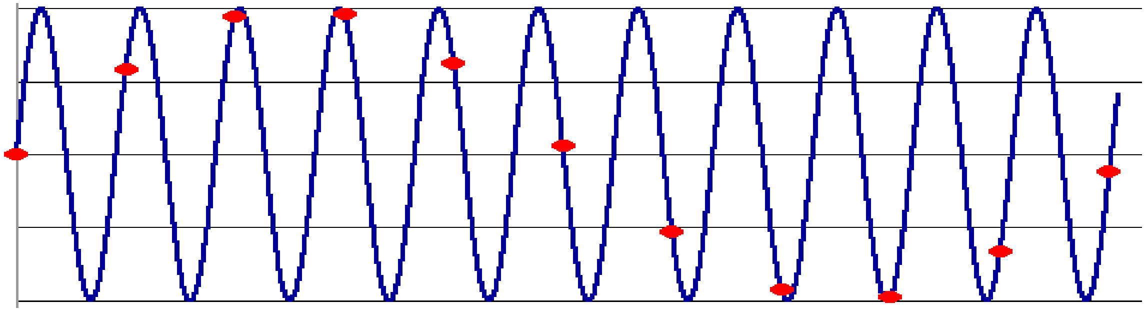
Source: F. Durand

Even worse for synthetic images



Source: L. Zhang

Aliasing



- Occurs when your sampling rate is not high enough to capture the amount of detail in your image
- Can give you the wrong signal/image—an *alias*

- To do sampling right, need to understand the structure of your signal/image
- Enter Monsieur Fourier…

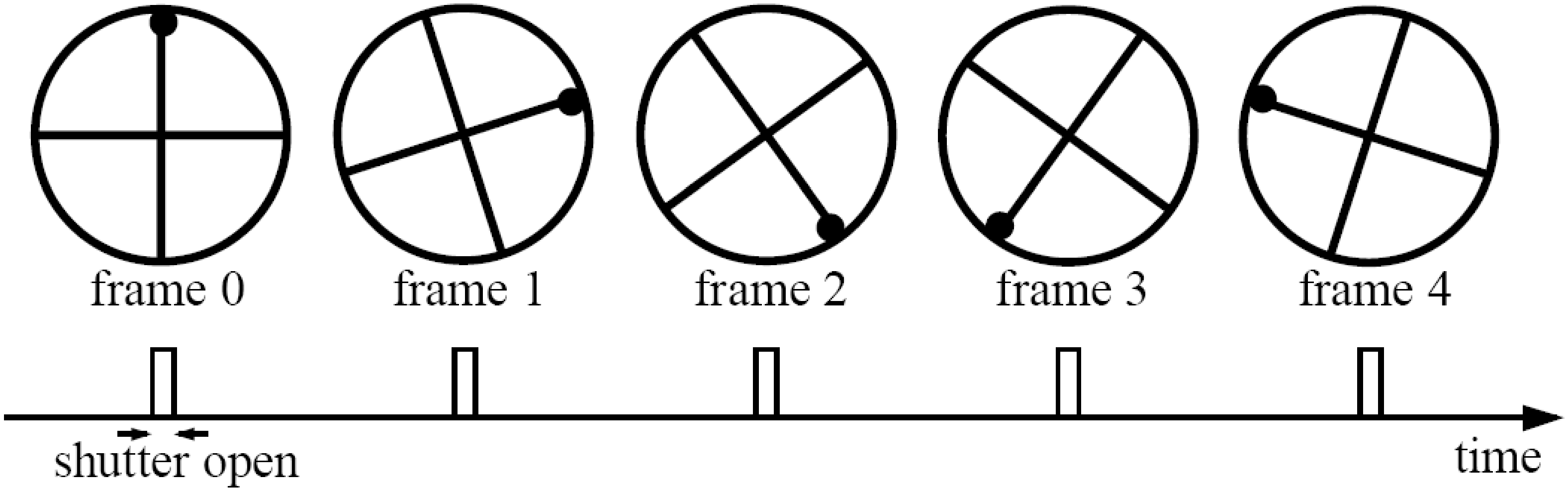
- To avoid aliasing:
  - sampling rate ≥ 2 * max frequency in the image
    - said another way: ≥ two samples per cycle
  - This minimum sampling rate is called the **Nyquist rate**

Source: L. Zhang

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):

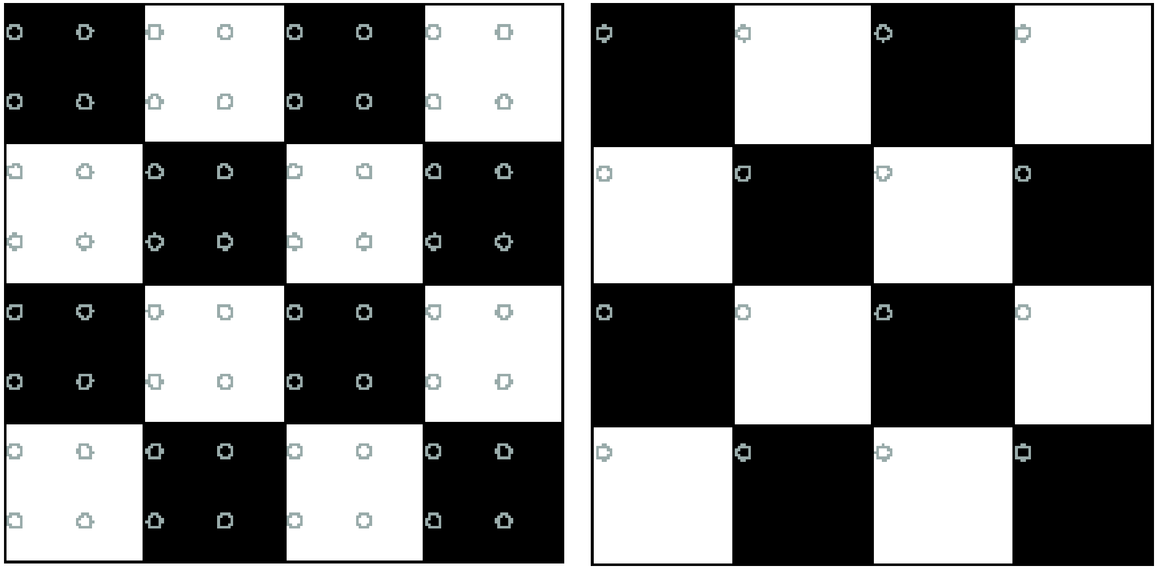


Without dot, wheel appears to be rotating slowly backwards! (counterclockwise)

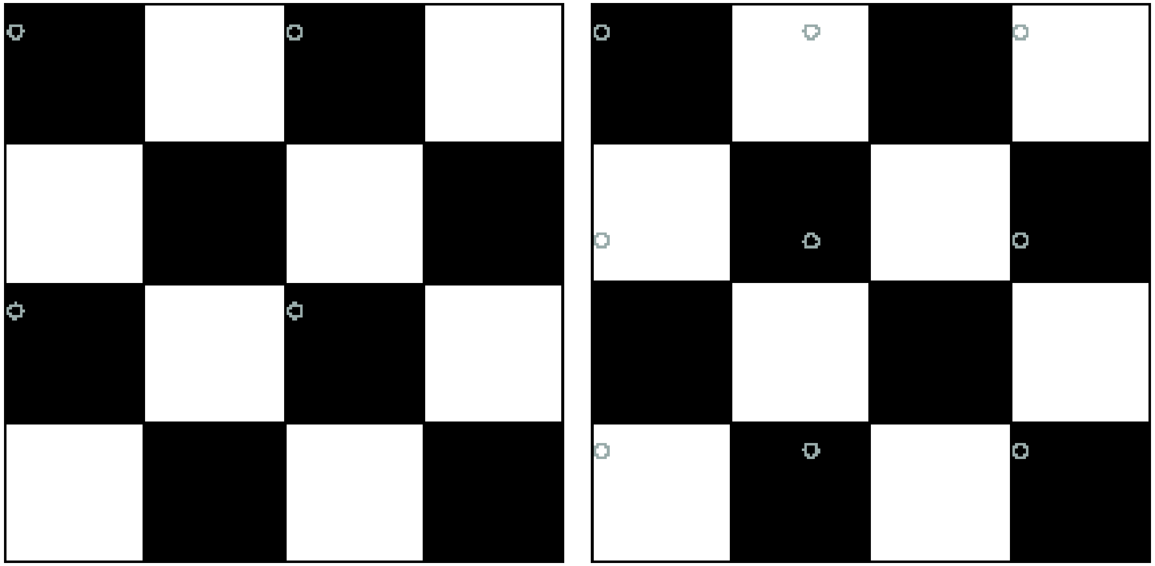
(See http://www.michaelbach.de/ot/mot_wagonWheel/index.html)

Source: L. Zhang

Nyquist limit – 2D example



Good sampling



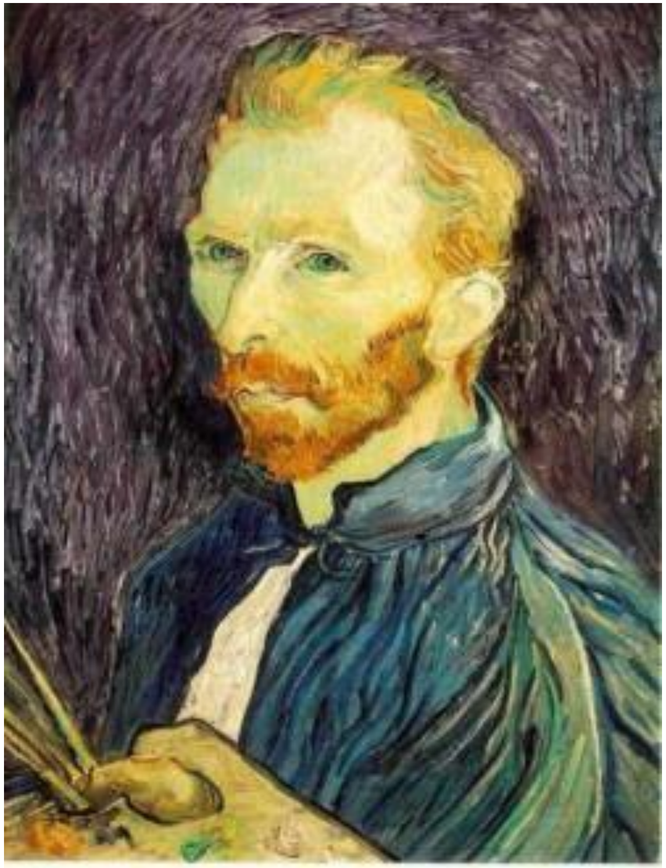
Bad sampling

Aliasing

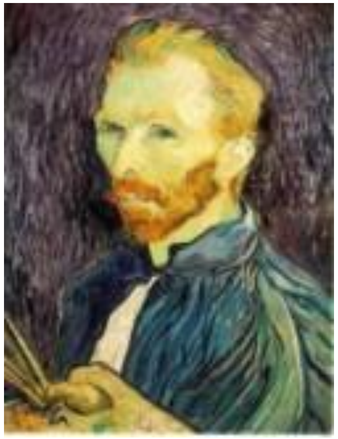
- When downsampling by a factor of two
  - Original image has frequencies that are too high

- How can we fix this?

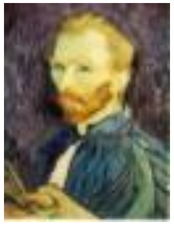
Gaussian pre-filtering



Gaussian 1/2



G 1/4

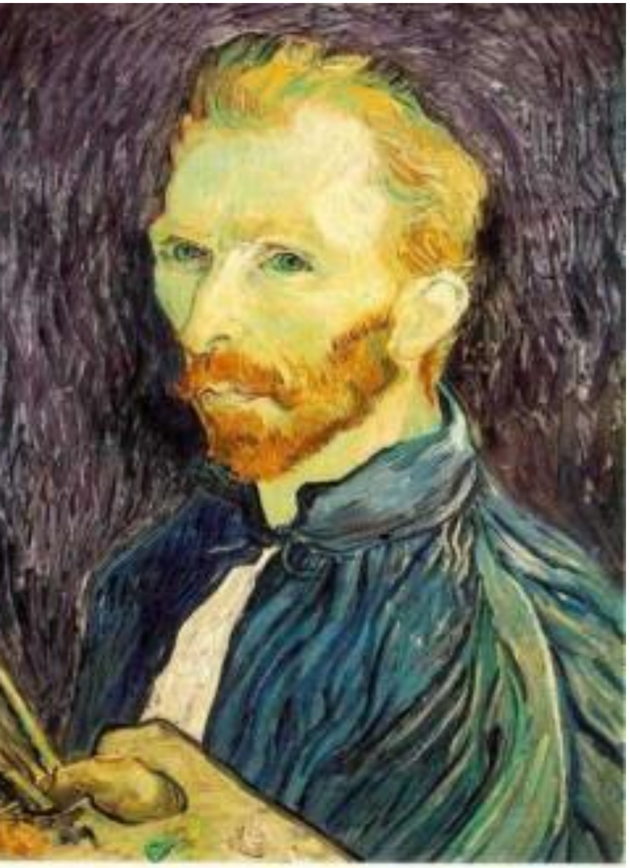


G 1/8

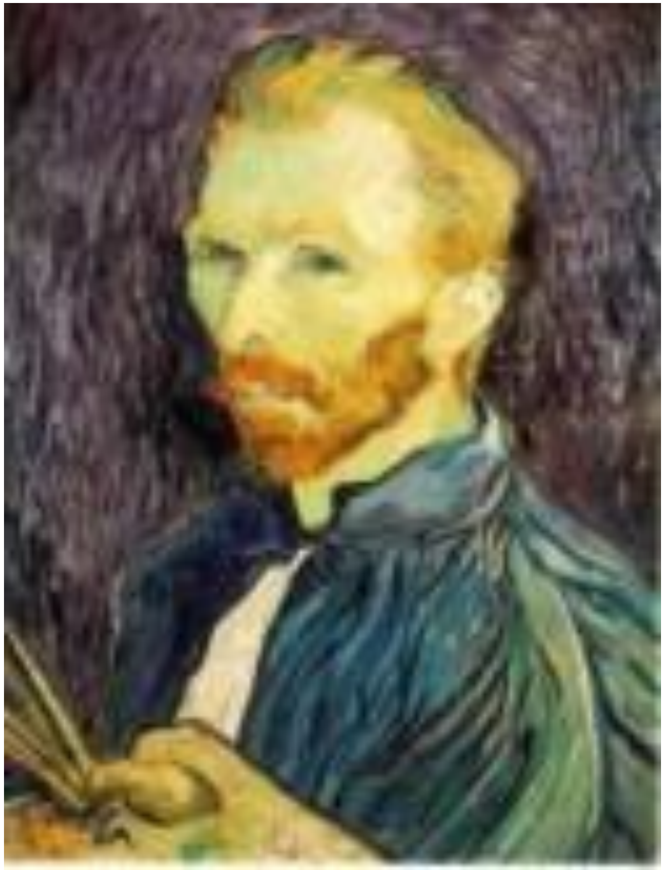
- Solution: filter the image, *then* subsample

Source: S. Seitz

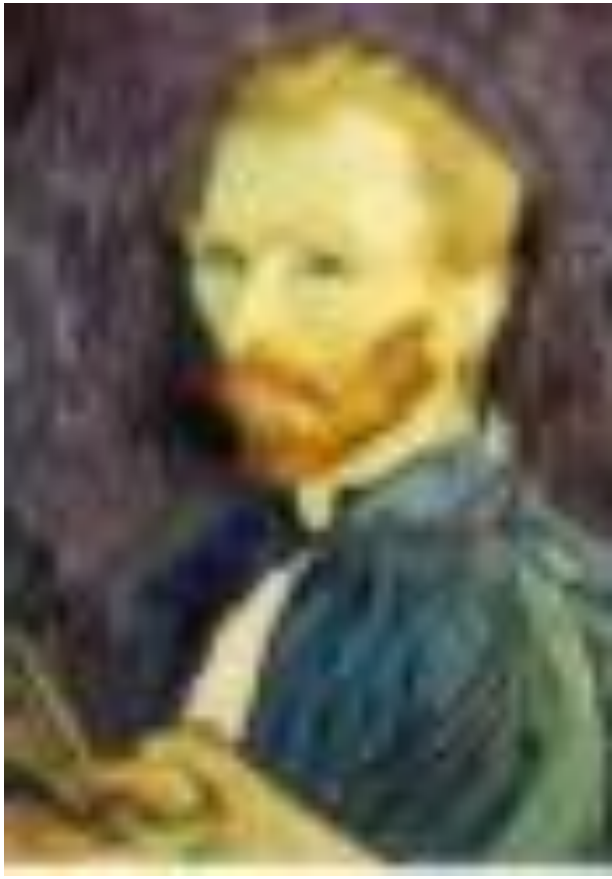
Subsampling with Gaussian pre-filtering



Gaussian 1/2



G 1/4

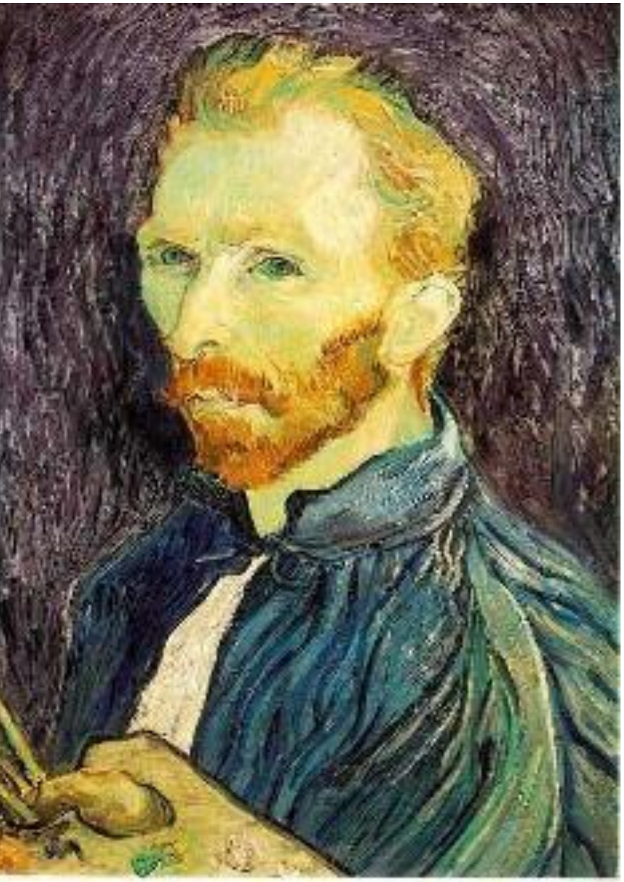


G 1/8

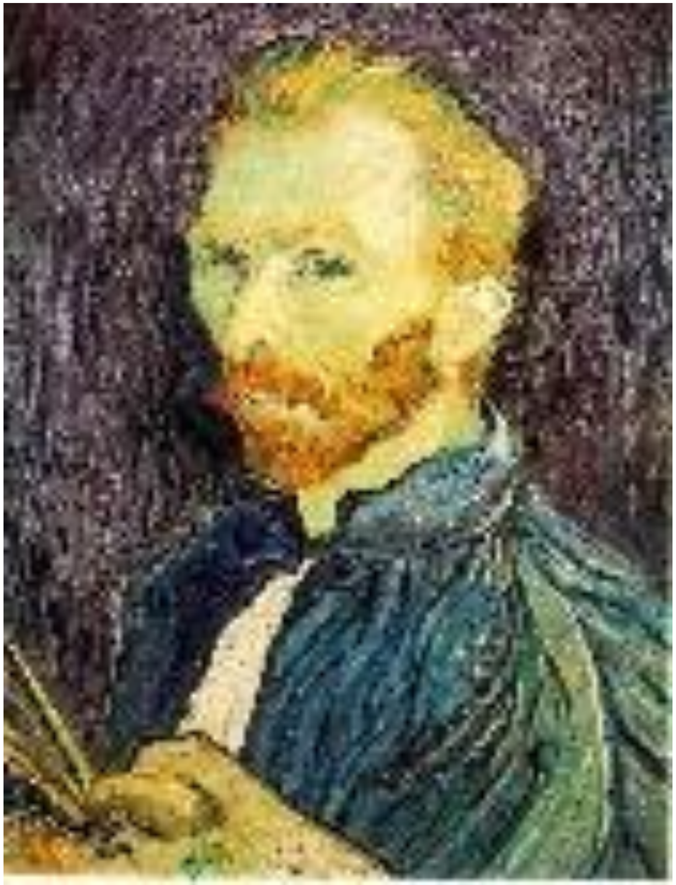
- Solution: filter the image, *then* subsample

Source: S. Seitz

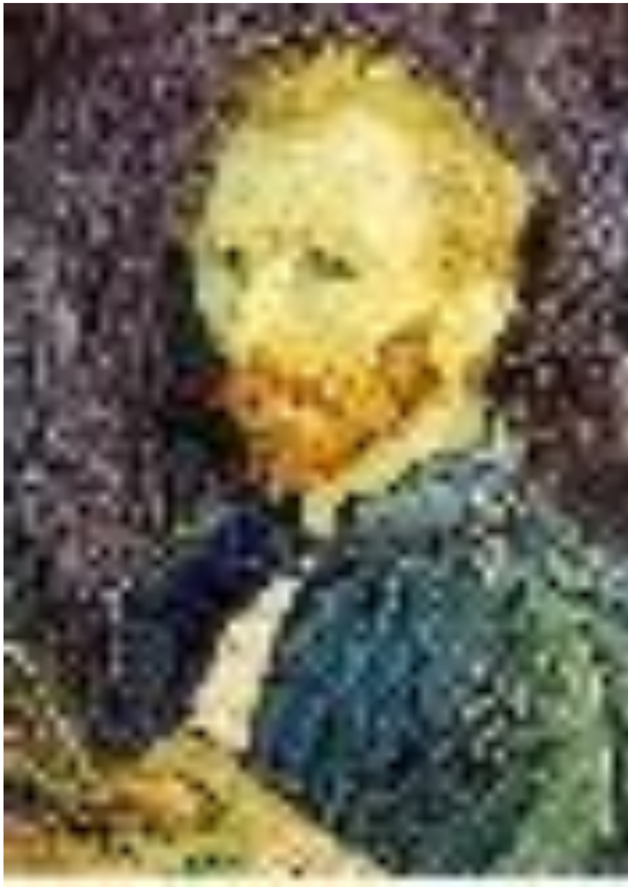
Compare with...



1/2



1/4 (2x zoom)

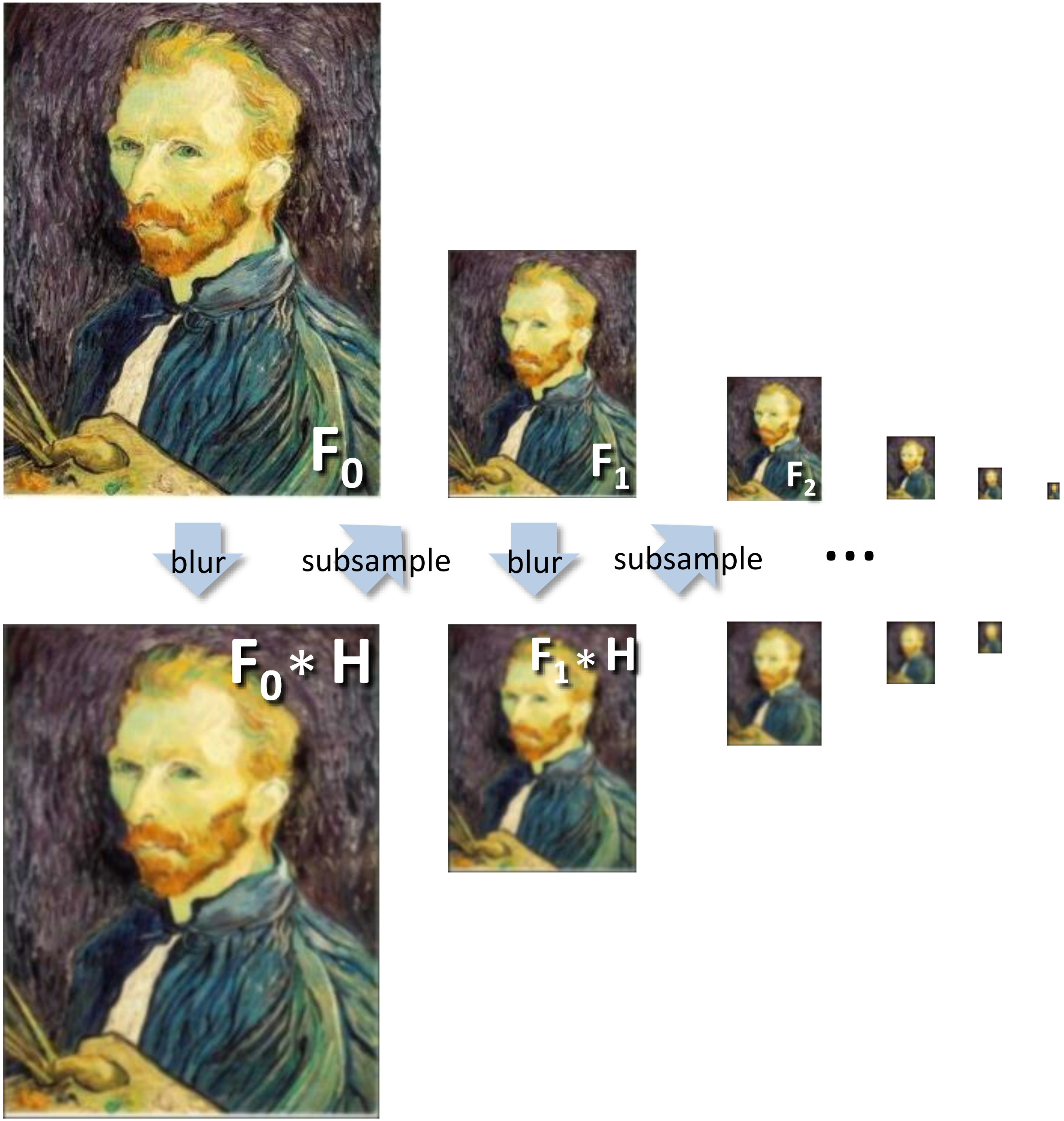


1/8 (4x zoom)

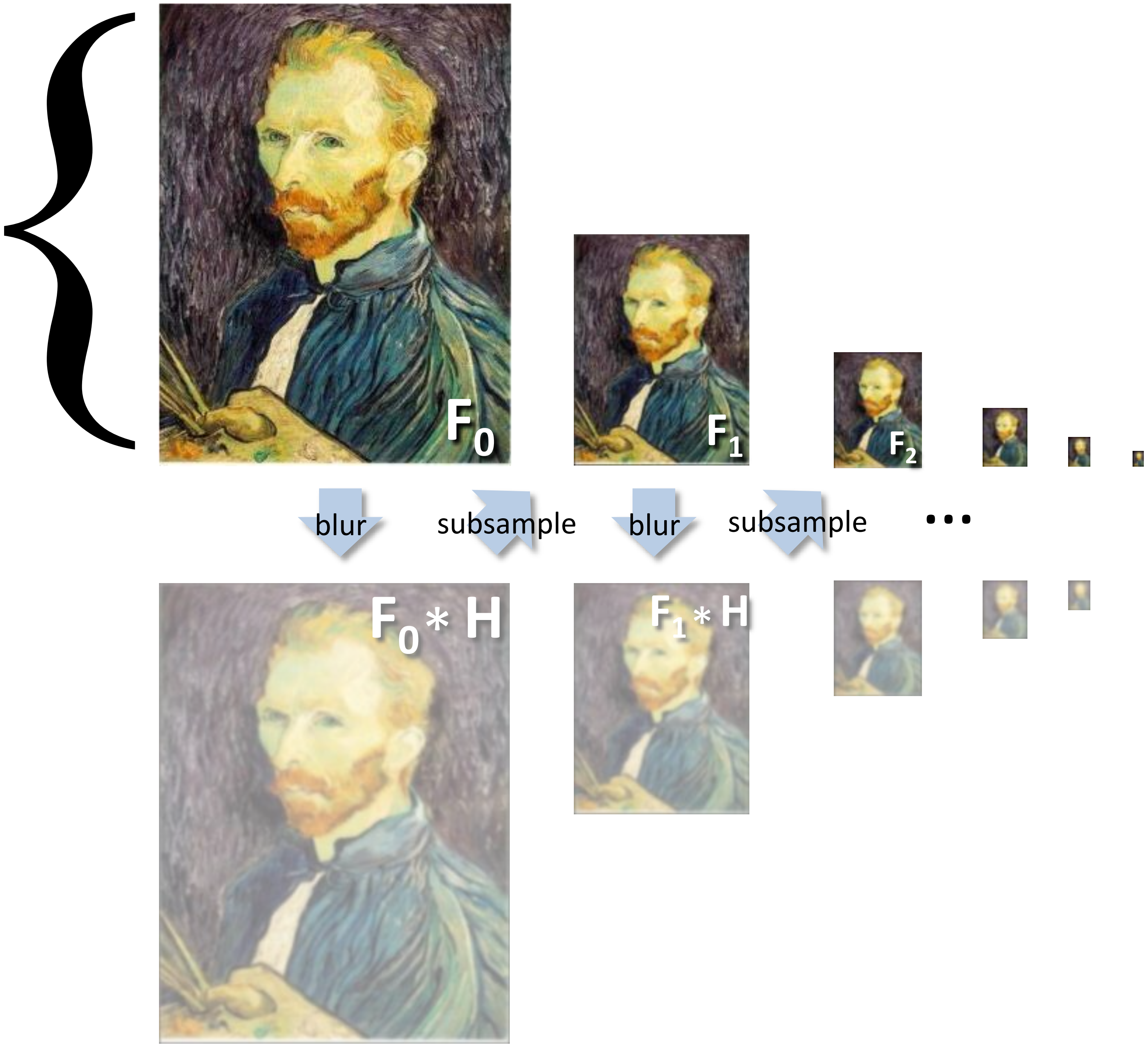
Source: S. Seitz

Gaussian pre-filtering

- Solution: filter the image, *then* subsample



Gaussian pyramid

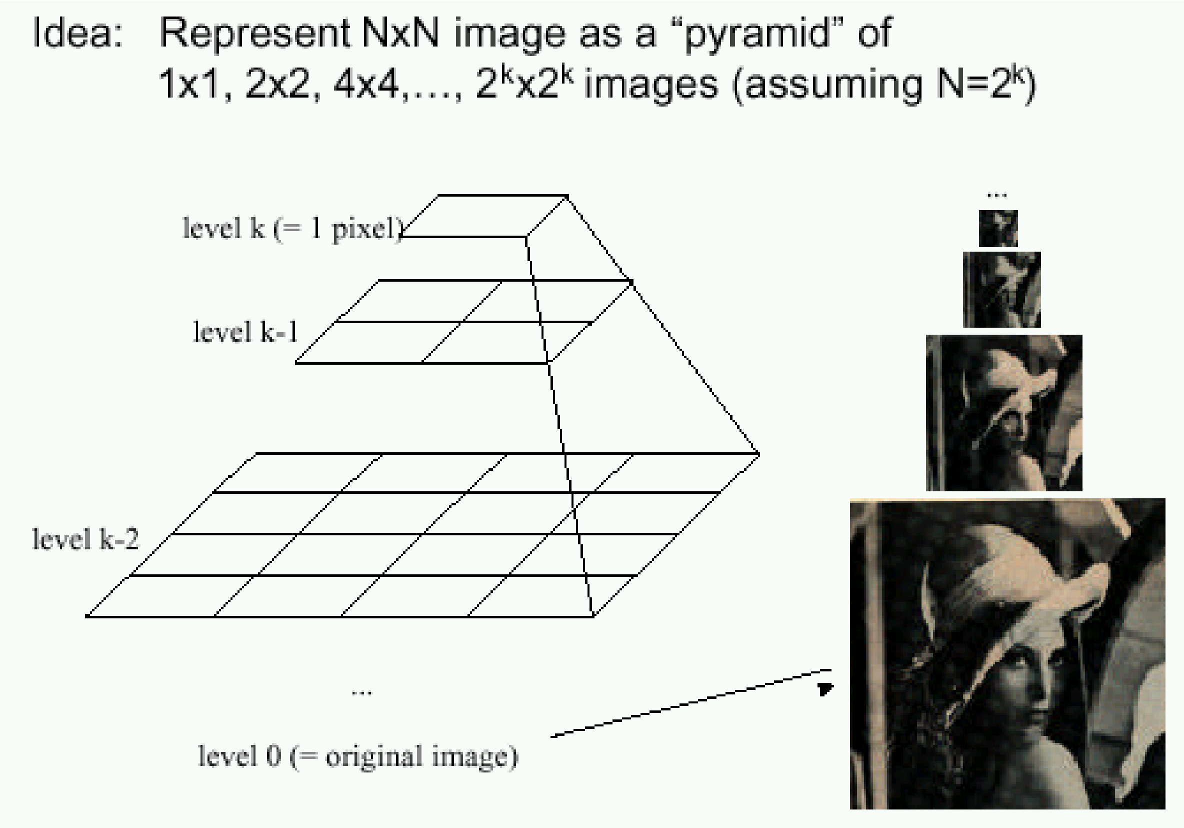


F_0 F_1 F_2

blur subsample blur subsample …

$F_0 * H$ $F_1 * H$

Gaussian pyramids
[Burt and Adelson, 1983]



Idea: Represent NxN image as a “pyramid” of 1x1, 2x2, 4x4,…, $2^k x 2^k$ images (assuming $N=2^k$)

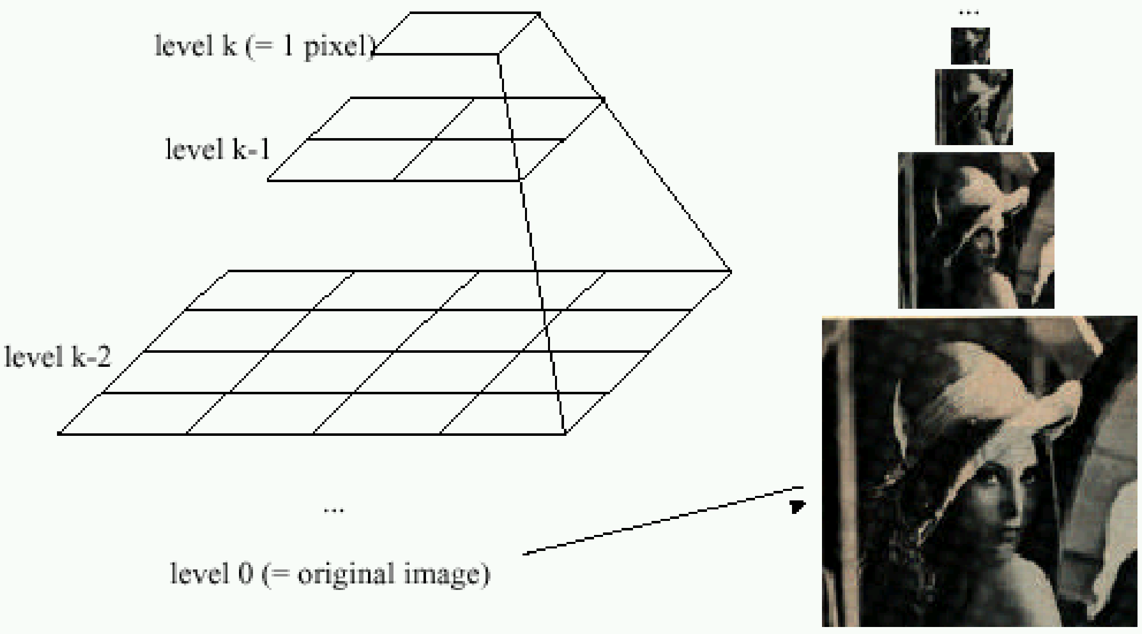
- In computer graphics, a *mip map* [Williams, 1983]
- A precursor to *wavelet transform*

Gaussian Pyramids have all sorts of applications in computer vision

Source: S. Seitz

Gaussian pyramids [Burt and Adelson, 1983]

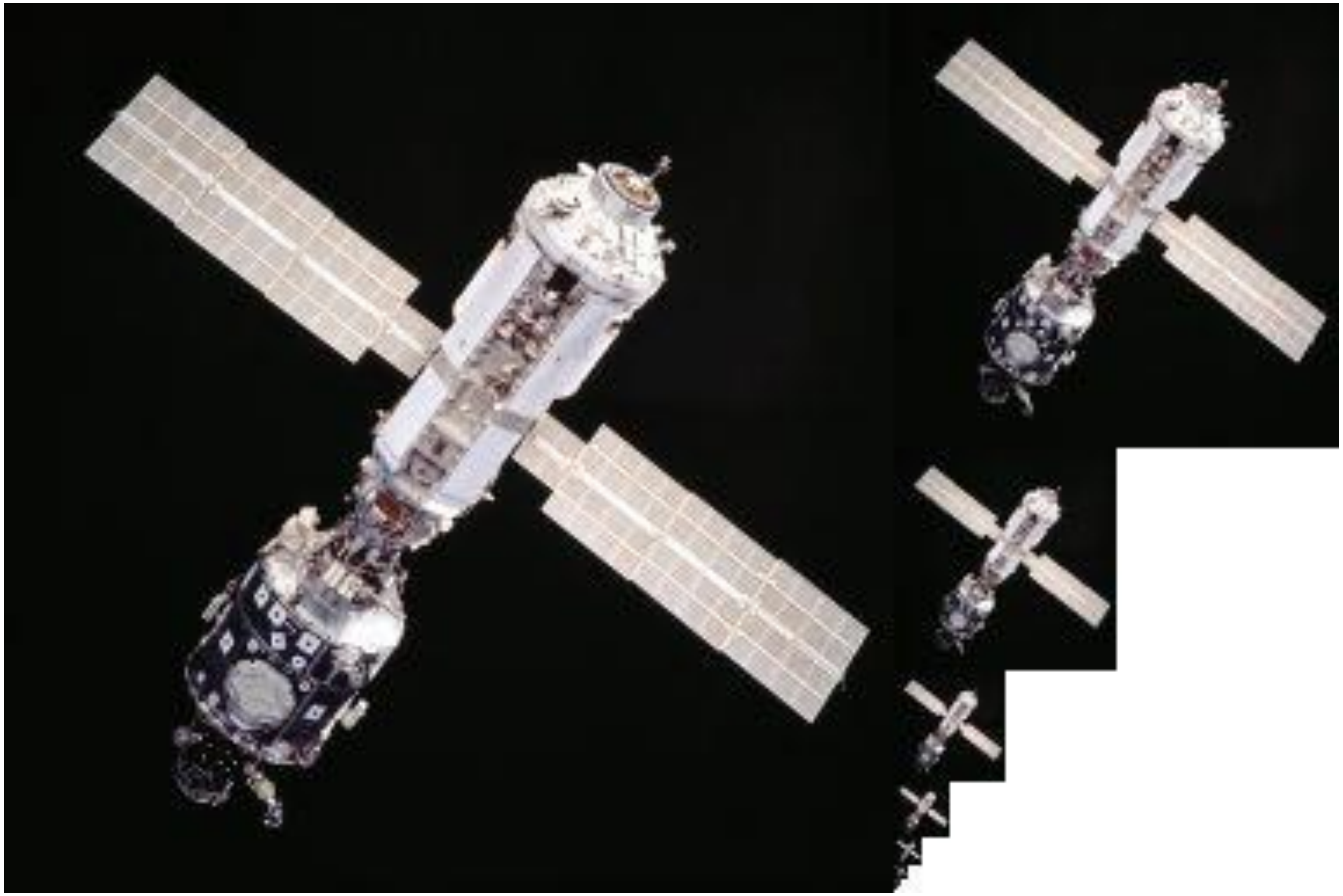
Idea: Represent NxN image as a “pyramid” of 1x1, 2x2, 4x4,…, 2^kx2^k images (assuming N=2^k)



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

Source: S. Seitz

Gaussian Pyramid



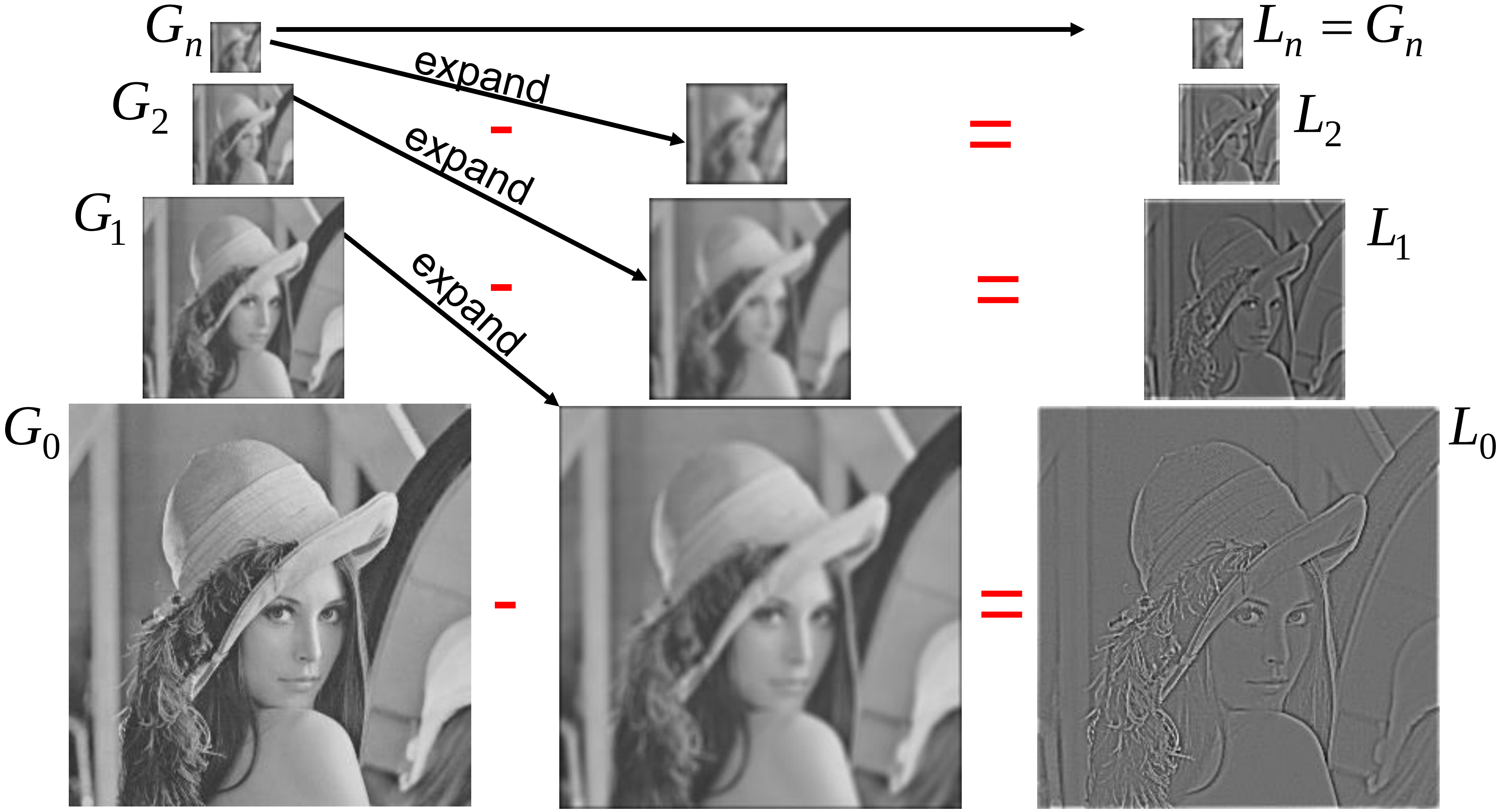
The Laplacian Pyramid

$$L_i = G_i - \mathrm{expand}(G_{i+1})$$

Gaussian Pyramid

$$G_i = L_i + \mathrm{expand}(G_{i+1})$$

Laplacian Pyramid



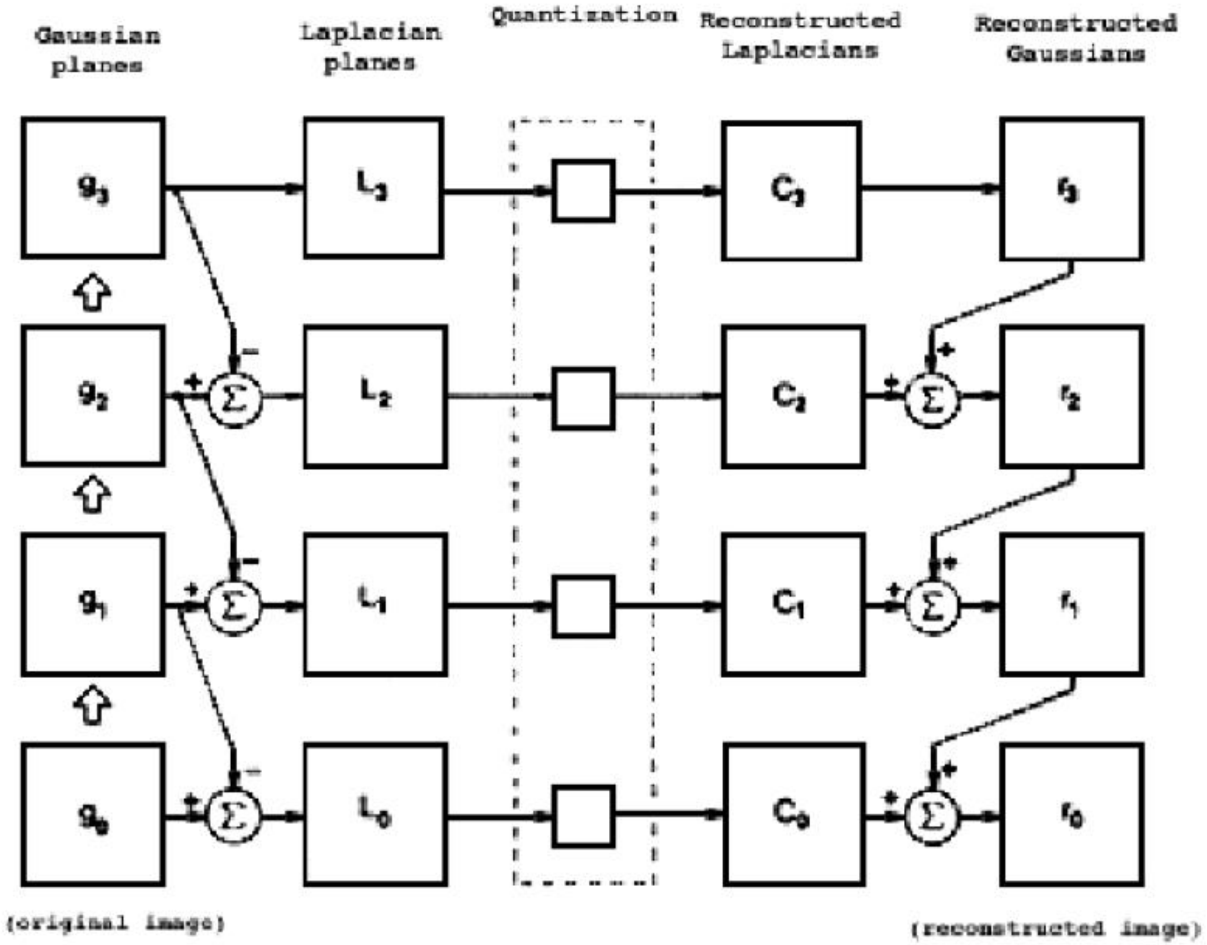


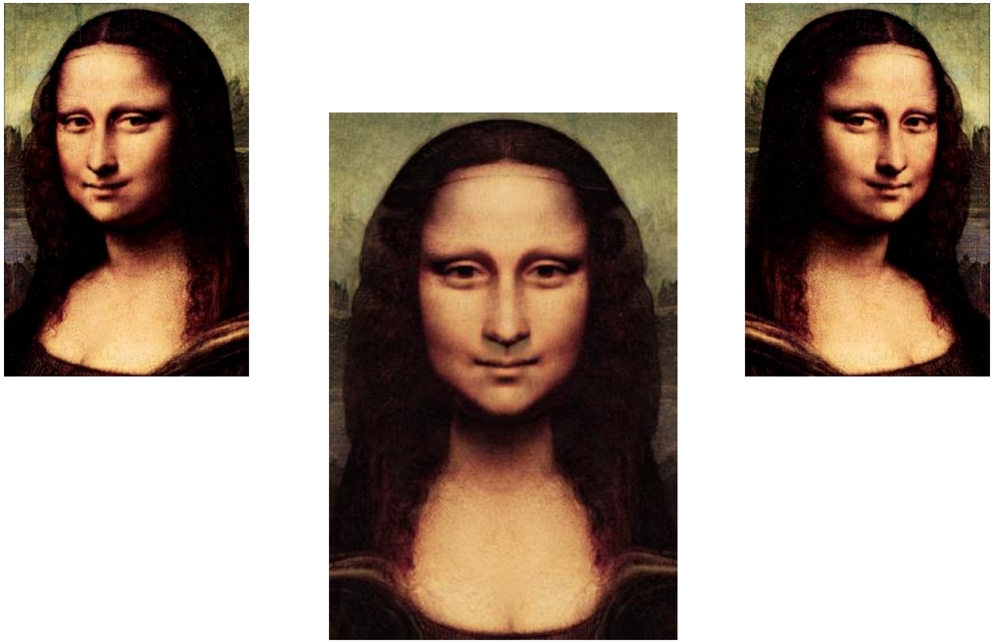
Fig. 10. A summary of the steps in Laplacian pyramid coding and decoding. First, the original image g_0 (lower left) is used to generate Gaussian pyramid levels g_1, g_2, … through repeated local averaging. Levels of the Laplacian pyramid L_0, L_1, … are then computed as the differences between adjacent Gaussian levels. Laplacian pyramid elements are quantized to yield the Laplacian pyramid code C_0, C_1, C_2, …. Finally, a reconstructed image r_0 is generated by summing levels of the code pyramid.

Questions?

CS4670: Computer Vision

Noah Snavely

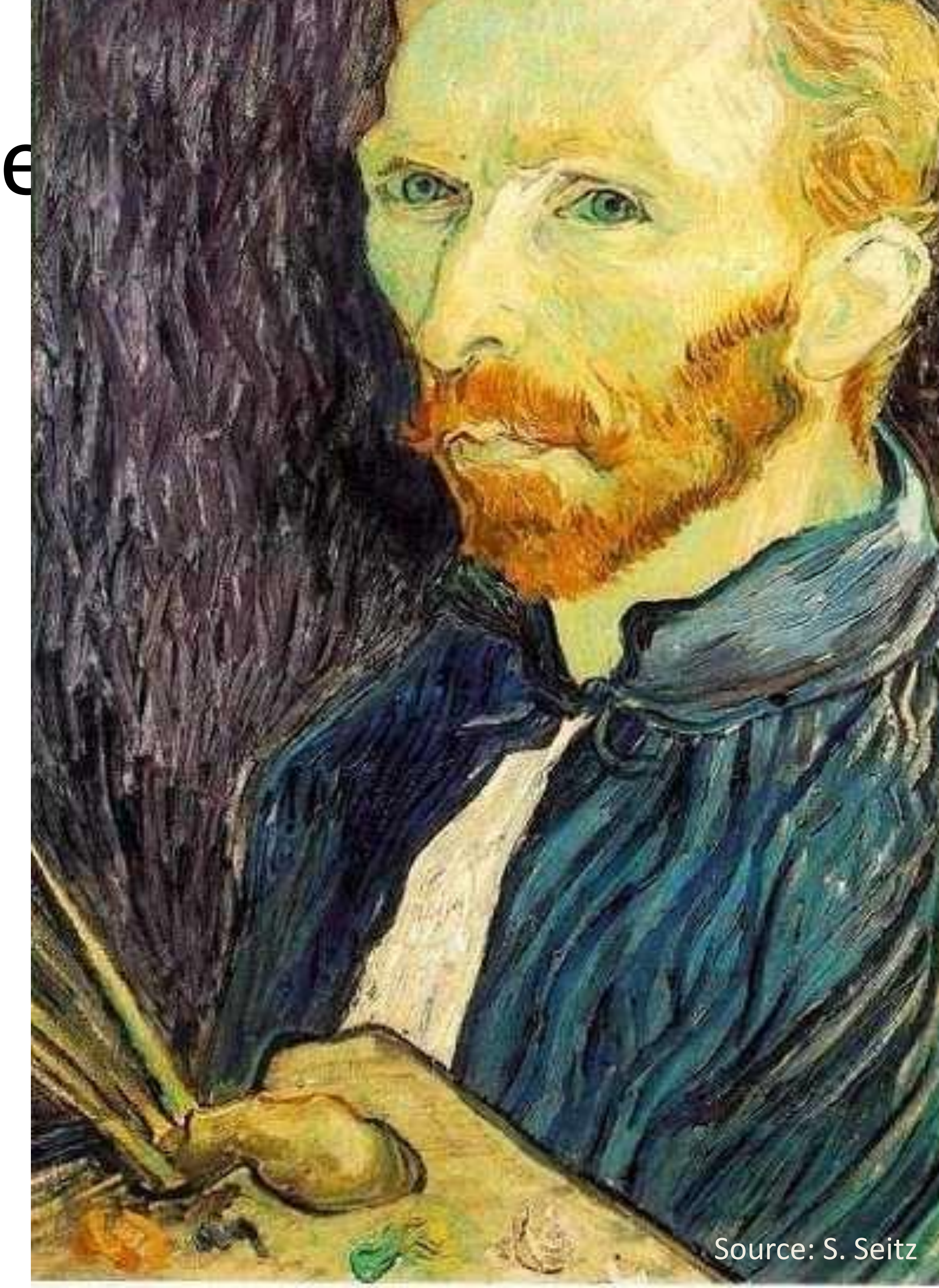
Image Interpolation



Image

Last time:

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



Source: S. Seitz

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”)

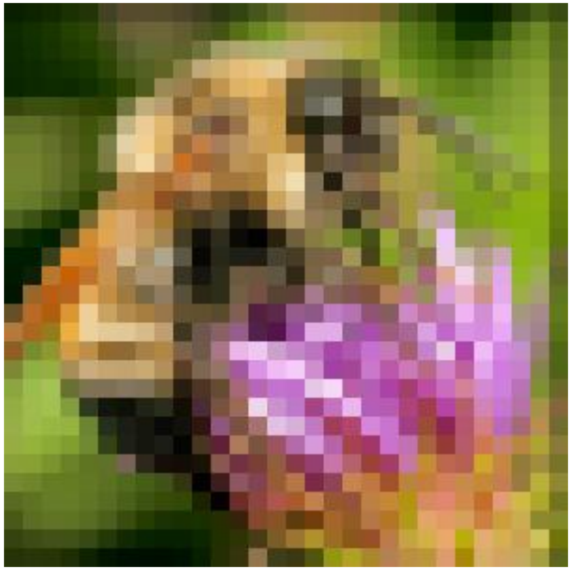
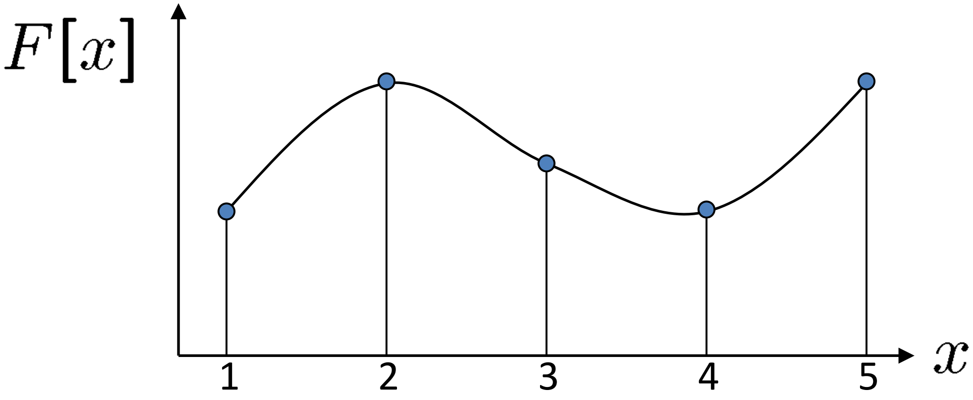


Image interpolation



d = 1 in this example

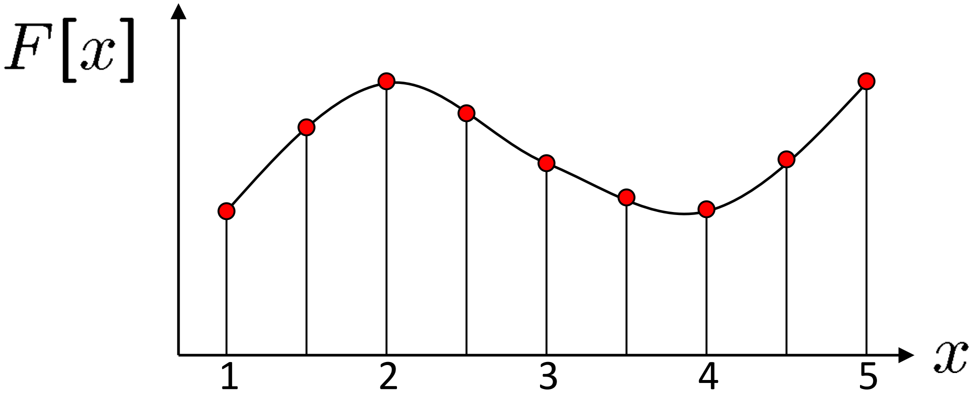
Recall how a digital image is formed

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

- 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

Adapted from: S. Seitz

Image interpolation



d = 1 in this example

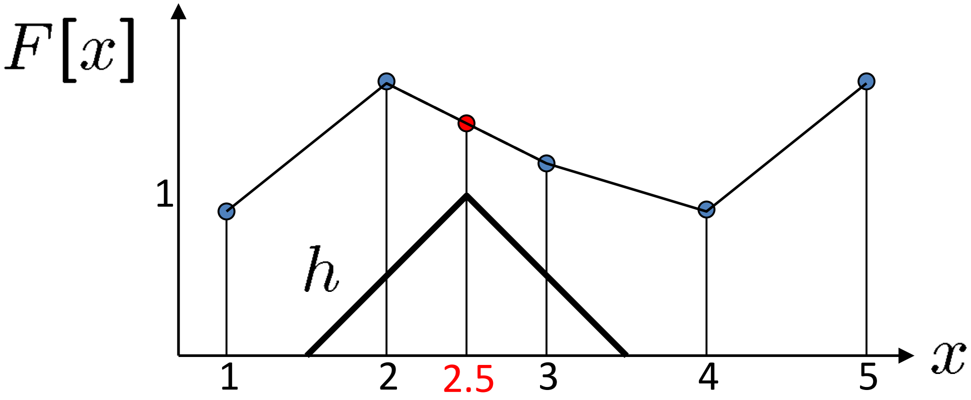
Recall how a digital image is formed

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

- 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

Adapted from: S. Seitz

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:

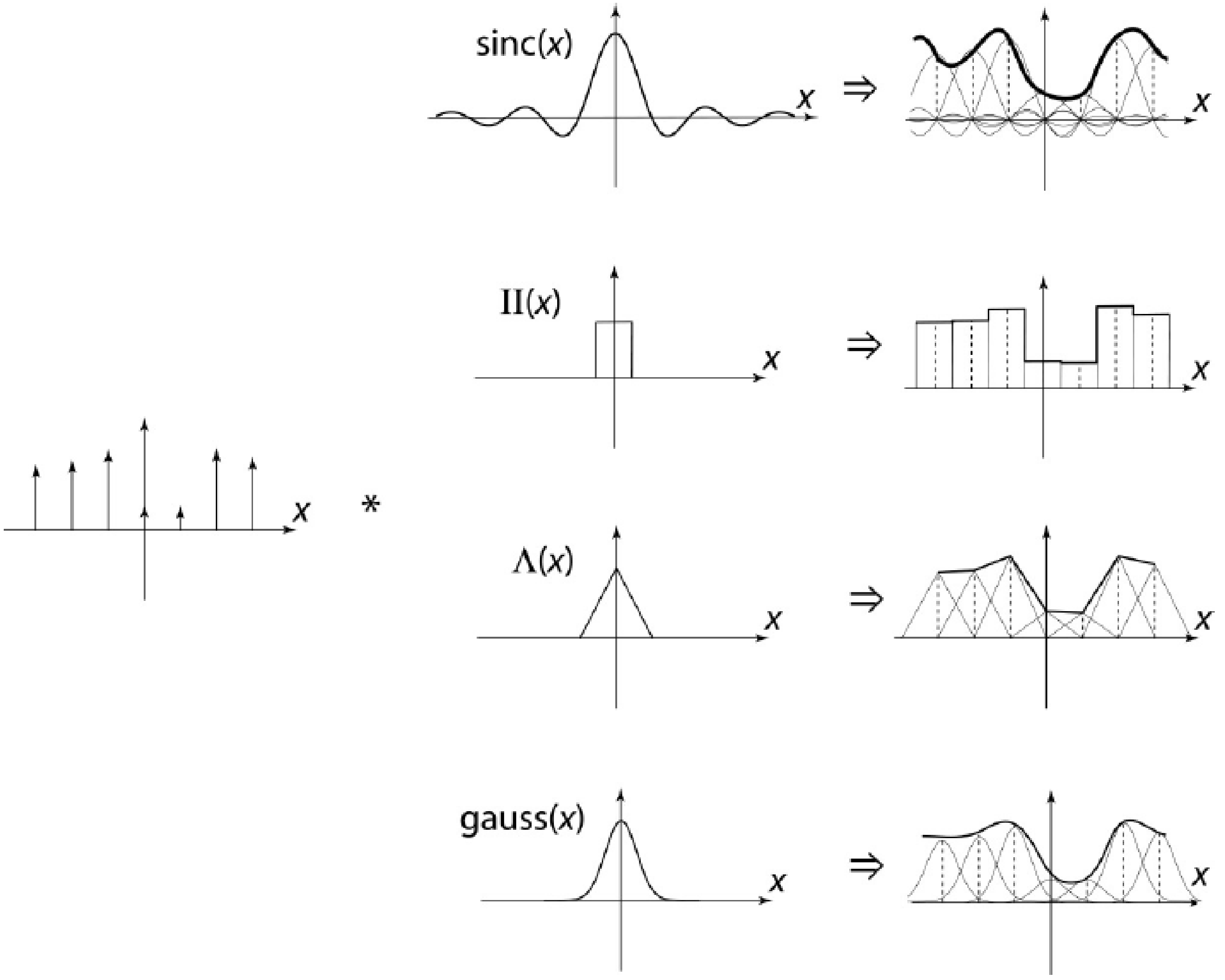
$$f_F(x) = F(\tfrac{x}{d}) \text{ when } \tfrac{x}{d} \text{ is an integer, 0 otherwise}$$

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

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

Adapted from: S. Seitz

Image interpolation



“Ideal” reconstruction

Nearest-neighbor interpolation

Linear interpolation

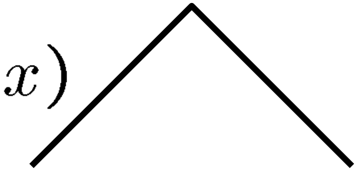
Gaussian reconstruction

Source: B. Curless

Reconstruction filters

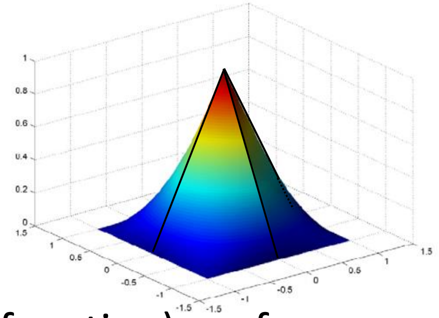
- What does the 2D version of this hat function look like?

$h(x)$



performs
linear interpolation

$h(x, y)$



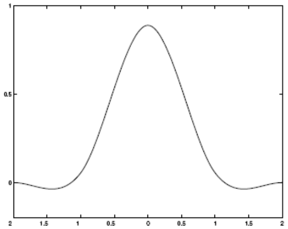
(tent function) performs
bilinear interpolation

Often implemented without cross-correlation

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

Better filters give better resampled images

- **Bicubic** is common choice

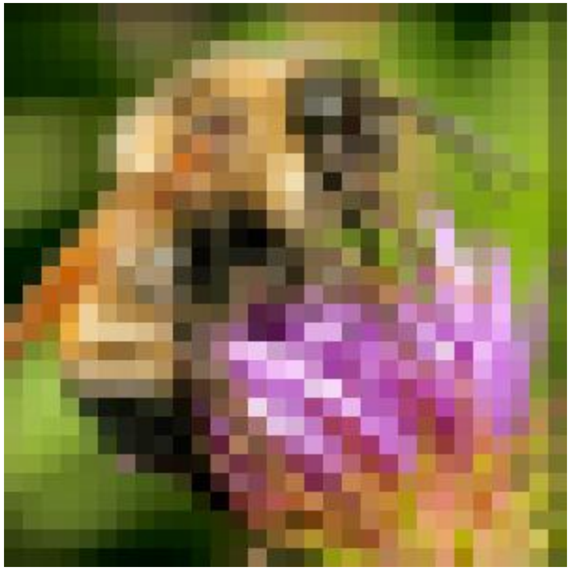


Cubic reconstruction filter

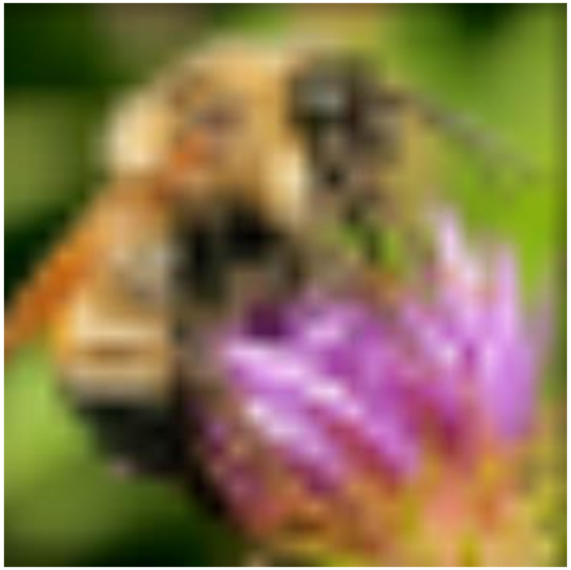
$$r(x) = \frac{1}{6}\begin{cases} (12-9B-6C)|x|^3 + (-18+12B+6C)|x|^2 + (6-2B) & |x| < 1 \\ ((-B-6C)|x|^3 + (6B+30C)|x|^2 + (-12B-48C)|x| + (8B+24C) & 1 \le |x| < 2 \\ 0 & \textit{otherwise} \end{cases}$$

Image interpolation

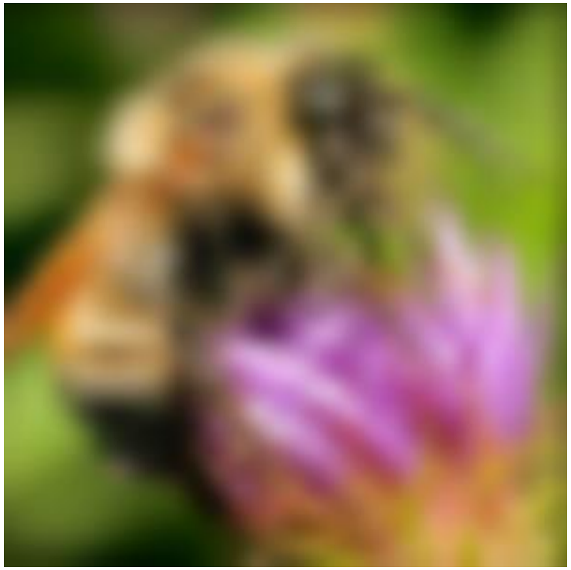
Original image: x 10



Nearest-neighbor interpolation



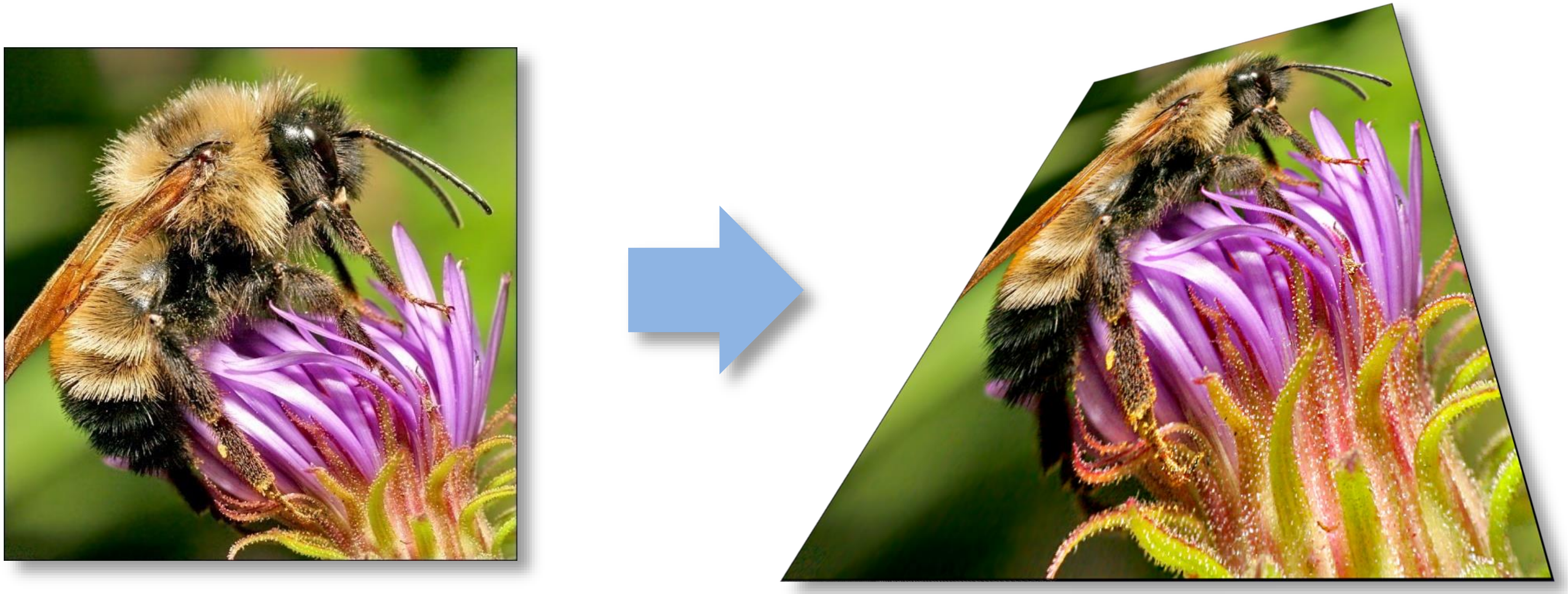
Bilinear interpolation



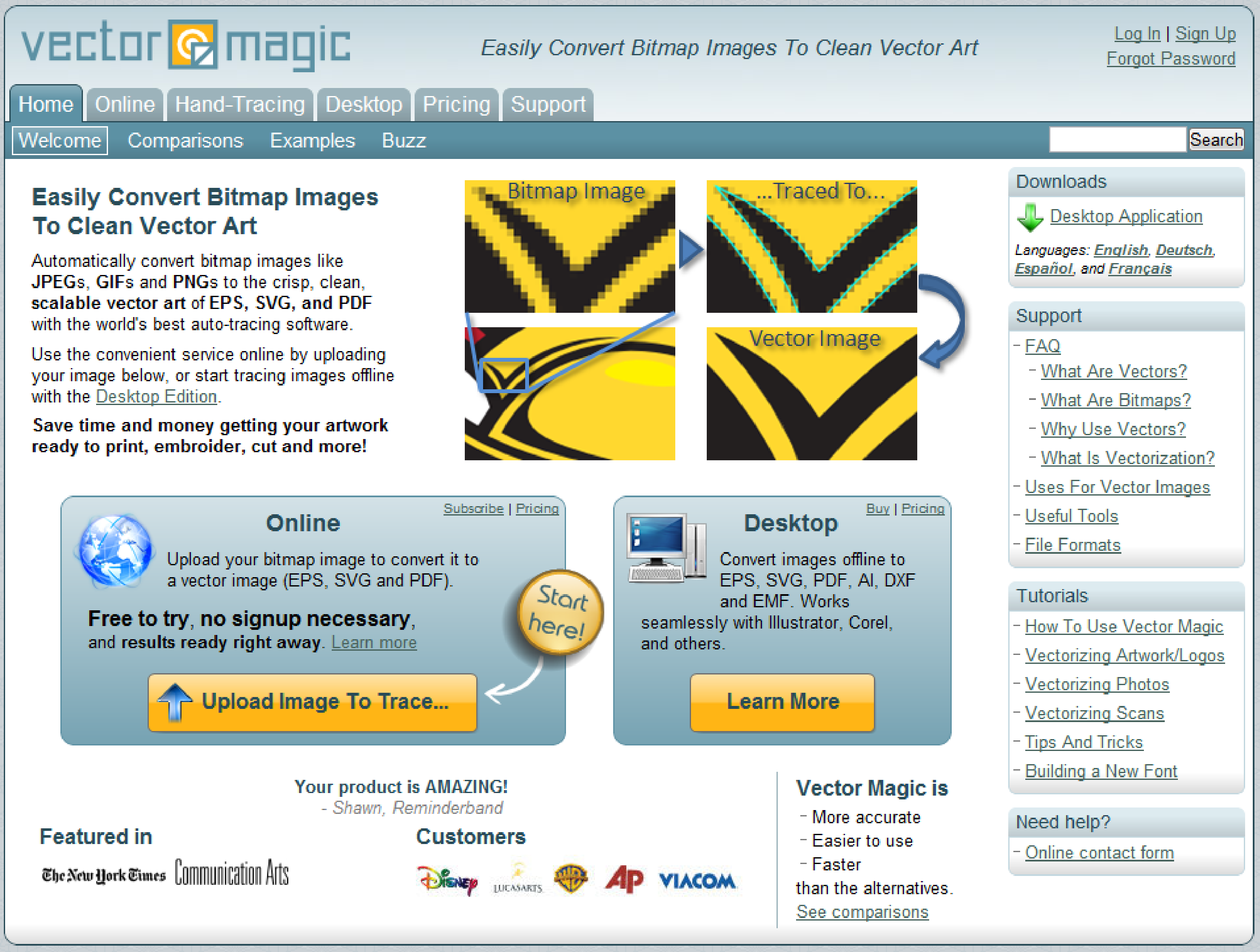
Bicubic interpolation

Image interpolation

Also used for *resampling*



Raster to Vector Graphics



Easily Convert Bitmap Images To Clean Vector Art

Easily Convert Bitmap Images To Clean Vector Art

Automatically convert bitmap images like **JPEGs**, **GIFs** and **PNGs** to the crisp, clean, **scalable vector art** of **EPS, SVG, and PDF** with the world's best auto-tracing software.

Use the convenient service online by uploading your image below, or start tracing images offline with the Desktop Edition.

Save time and money getting your artwork ready to print, embroider, cut and more!

Online

Upload your bitmap image to convert it to a vector image (EPS, SVG and PDF).

Free to try, no signup necessary, **and results ready right away**. Learn more

Desktop

Convert images offline to EPS, SVG, PDF, AI, DXF and EMF. Works seamlessly with Illustrator, Corel, and others.

Your product is AMAZING!
- Shawn, Reminderband

Featured in

The New York Times Communication Arts

Customers

Disney LucasArts WB AP VIACOM

Vector Magic is

- More accurate
- Easier to use
- Faster

than the alternatives.
See comparisons

Downloads

Desktop Application

Languages: English, Deutsch, Español, and Français

Support

- FAQ
  - What Are Vectors?
  - What Are Bitmaps?
  - Why Use Vectors?
  - What Is Vectorization?
- Uses For Vector Images
- Useful Tools
- File Formats

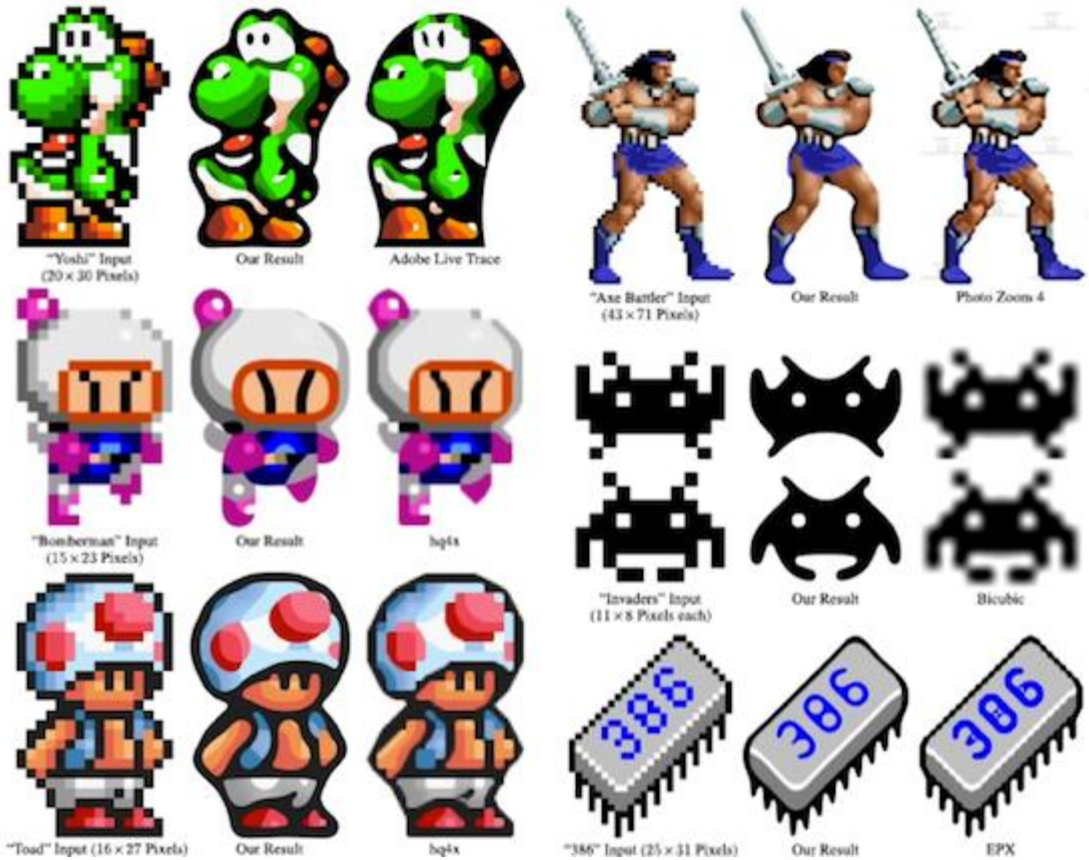
Tutorials

- How To Use Vector Magic
- Vectorizing Artwork/Logos
- Vectorizing Photos
- Vectorizing Scans
- Tips And Tricks
- Building a New Font

Need help?

- Online contact form

Depixelating Pixel Art



"Yoshi" Input (20 × 30 Pixels)

Our Result

Adobe Live Trace

"Axe Battler" Input (43 × 71 Pixels)

Our Result

Photo Zoom 4

"Bomberman" Input (15 × 23 Pixels)

Our Result

hq4x

"Invaders" Input (11 × 8 Pixels each)

Our Result

Bicubic

"Toad" Input (16 × 27 Pixels)

Our Result

hq4x

"386" Input (25 × 31 Pixels)

Our Result

EPX

Questions?