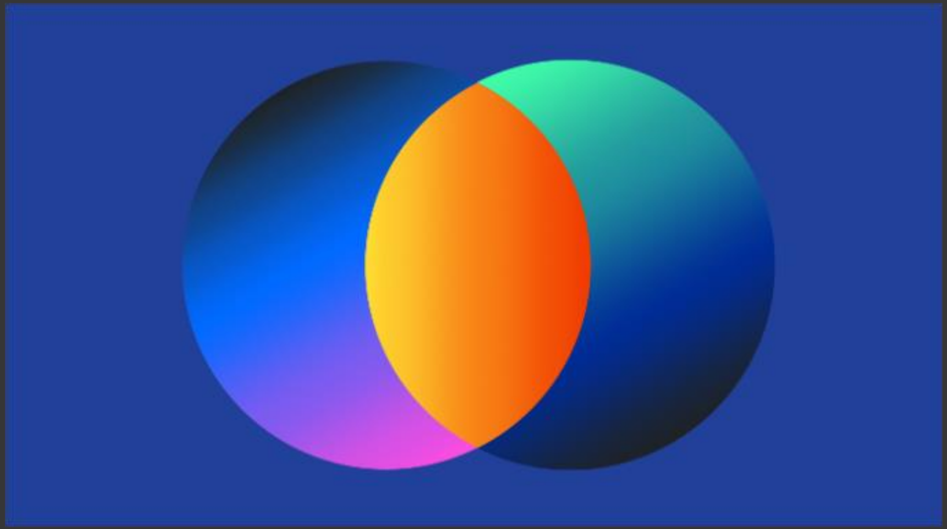
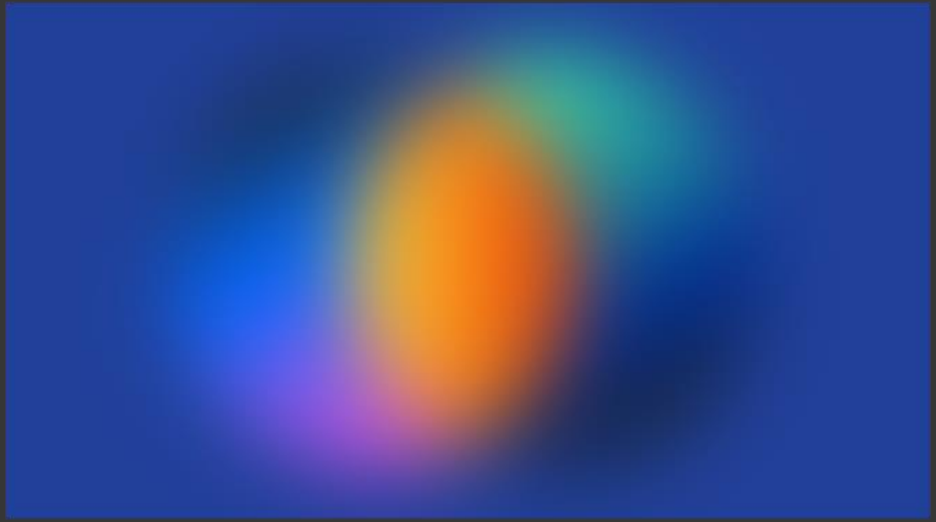
CS7GV6: Computer Vision

Image Filtering, Resampling and Interpoation



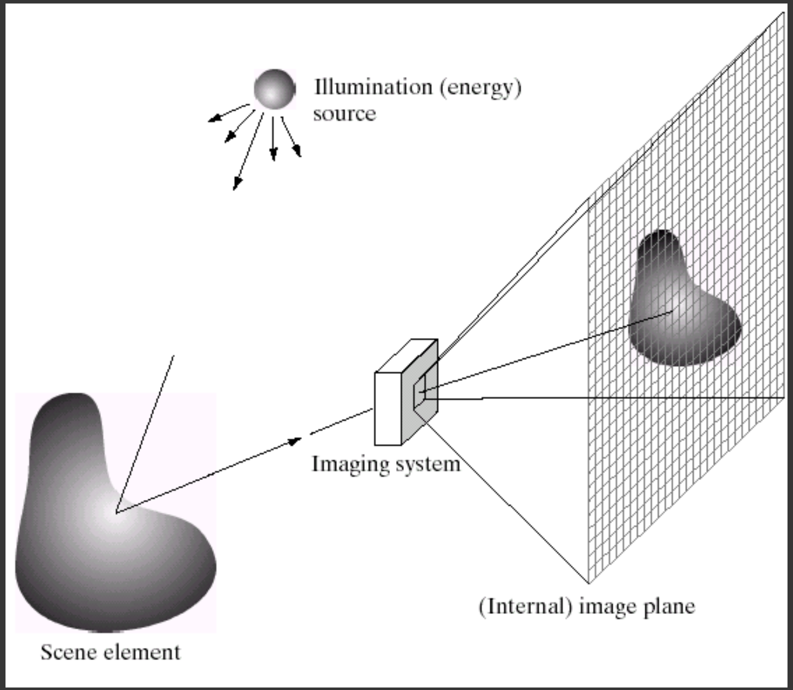


Read: Szeliski, Ch. 3

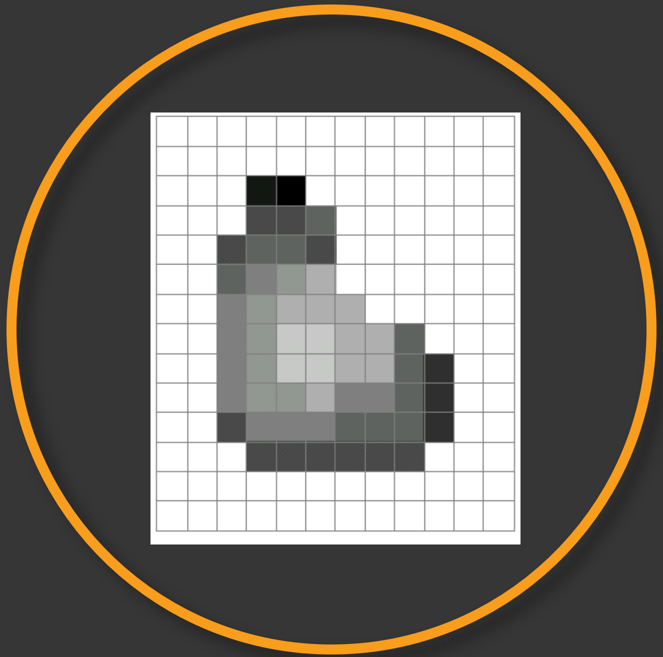
Credits: Some slides from Noah Snavely & others

What is an image?

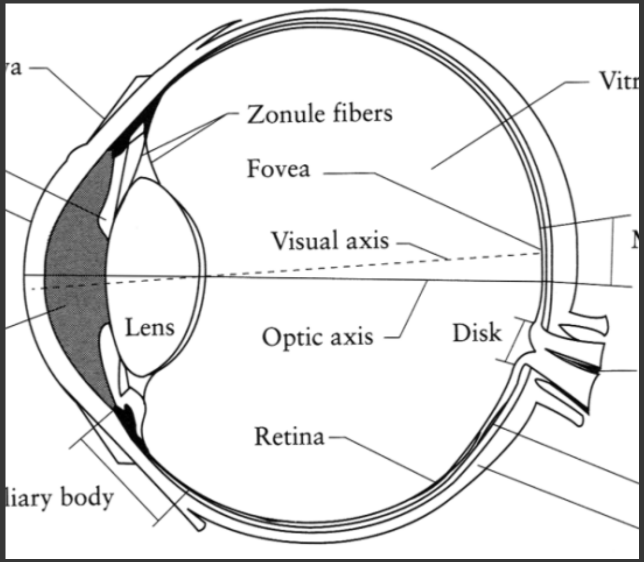
What is an image?



Record of light rays



Grid of pixel values
(digital camera)

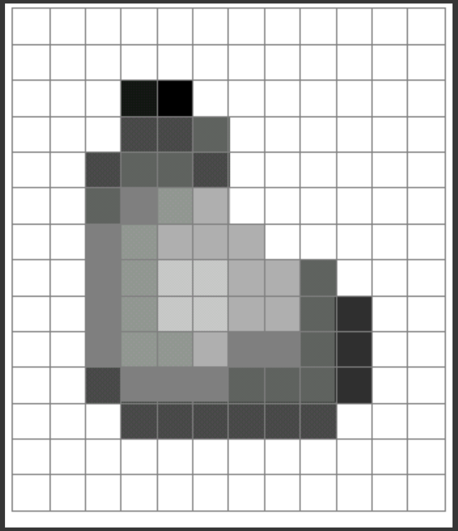


The Eye

Source: A. Efros

What is an image?

- A grid (matrix) of intensity values



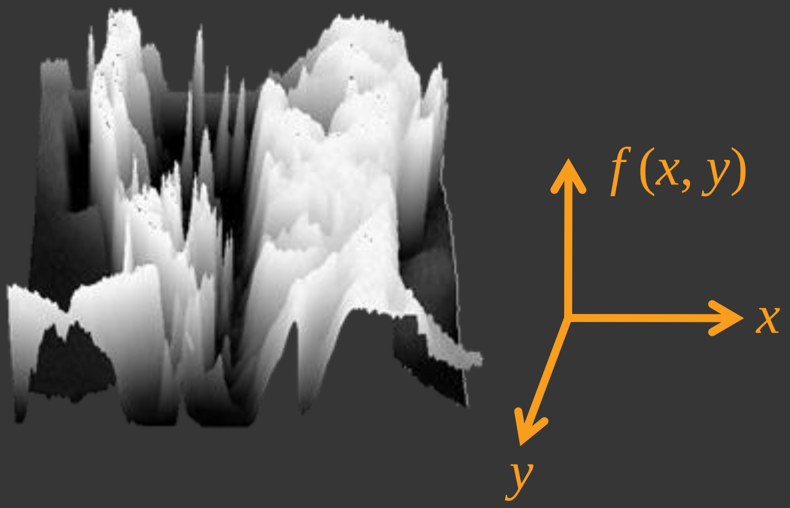
=

255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	20	0	255	255	255	255	255	255	255
255	255	255	75	75	75	255	255	255	255	255	255
255	255	75	95	95	75	255	255	255	255	255	255
255	255	96	127	145	175	255	255	255	255	255	255
255	255	127	145	175	175	175	255	255	255	255	255
255	255	127	145	200	200	175	175	95	255	255	255
255	255	127	145	200	200	175	175	95	47	255	255
255	255	127	145	145	175	127	127	95	47	255	255
255	255	74	127	127	127	95	95	95	47	255	255
255	255	255	74	74	74	74	74	74	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255

(common to use one byte per value: 0 = black, 255 = white)

What is an image?

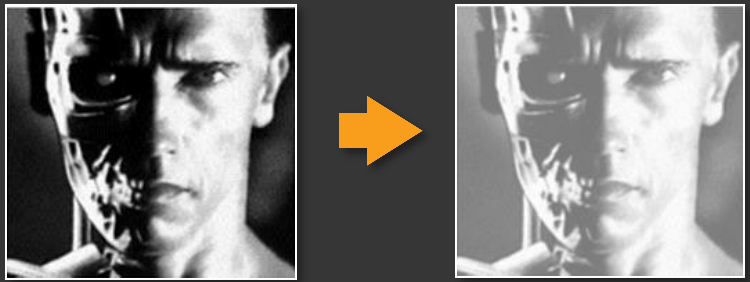
- Can think of a (grayscale) image as a **function** f from R^2 to R:
  - $f(x,y)$ gives the **intensity** at position (x,y)



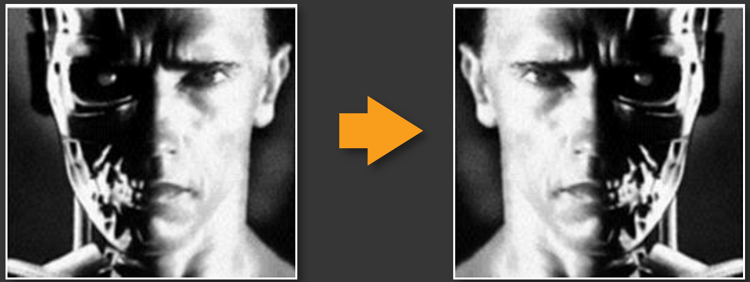
  - A **digital** image is a discrete (**sampled**, **quantized**) version of this function

Image transformations

- As with any function, we can apply operators to an image



$g(x,y) = f(x,y) + 20$



$g(x,y) = f(-x,y)$

- Today we'll talk about a special kind of operator, *convolution* (linear filtering)

Filters

- Filtering
  - Form a new image whose pixel values are a combination of the original pixel values
- Why?
  - To get useful information from images
    - E.g., extract edges or contours (to understand shape)
  - To enhance the image
    - E.g., to remove noise
    - E.g., to sharpen and "enhance image" a la CSI
  - A key operator in Convolutional Neural Networks

Filters: Thresholding

$$g(m,n) = \begin{cases} 255, & f(m,n) > A \\ 0 & otherwise \end{cases}$$

Image Processing problems

- Image Restoration
  - denoising
  - deblurring
- Image Compression
  - JPEG, HEIF, MPEG, …
- Computing Field Properties
  - optical flow
  - disparity
- Locating Structural Features
  - corners
  - edges

Question: Noise reduction

- Given a camera and a still scene, how can you reduce noise?

Take lots of images and average them!

What's the next best thing?

Source: S. Seitz

Image filtering

- Modify the pixels in an image based on some function of a local neighborhood of each pixel

10	5	3
4	5	1
1	1	7

Local image data

Some function →

	7	

Modified image data

Source: L. Zhang

Linear filtering

- One simple version of filtering: linear filtering (cross-correlation, convolution)
  - Replace each pixel by a linear combination (a weighted sum) of its neighbors
- The prescription for the linear combination is called the “kernel” (or “mask”, “filter”)

10	5	3
4	6	1
1	1	8

Local image data

0	0	0
0	0.5	0
0	1	0.5

kernel

	8	

Modified image data

Source: L. Zhang

Cross-correlation

Let F be the image, H be the kernel (of size 2*k*+1 x 2*k*+1), and G be the output image

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v] F[i+u, j+v]$$

This is called a **cross-correlation** operation:

$$G = H \otimes F$$

- Can think of as a “dot product” between local neighborhood and kernel for each pixel

Convolution

- Same as cross-correlation, except that the kernel is "flipped" (horizontally and vertically)

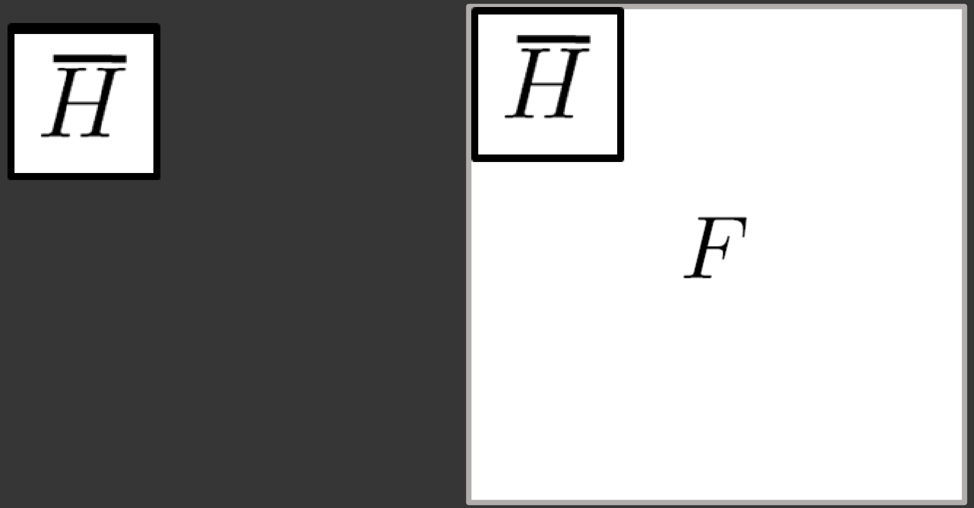
$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i-u,j-v]$$

This is called a **convolution** operation:

$$G = H * F$$

- Convolution is **commutative** and **associative**

Convolution



Adapted from F. Durand

Mean filtering

H

$*$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

F

$=$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

G

Mean filtering/Moving average

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G[x, y]$

Mean filtering/Moving average

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G[x, y]$

	0	10							

Mean filtering/Moving average

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G[x, y]$

	0	10	20	30					

Mean filtering/Moving average

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G[x, y]$

	0	10	20	30	30				

Mean filtering/Moving average

$F[x, y]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G[x, y]$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

Linear filters: examples

Original

$$* \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Source: D. Lowe

Linear filters: Question

What image operation does filtering with this kernel perform?

[0 0 0; 0 1 0; 0 0 0]

Source: D. Lowe

Linear filters: examples

Original

*

0	0	0
0	1	0
0	0	0

=

Identical image

Source: D. Lowe

Linear filters: examples

Original

$*$

0	0	0
1	0	0
0	0	0

Source: D. Lowe

Linear filters: Question

What image operation does filtering with this kernel perform?

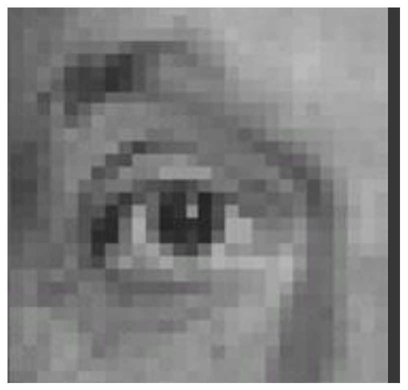
[0 0 0; 1 0 0; 0 0 0]

Source: D. Lowe

Linear filters: examples

Original

$$* \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} =$$



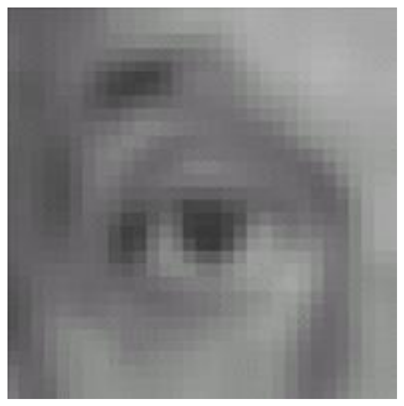
Shifted left by 1 pixel

Source: D. Lowe

Linear filters: examples

Original

$$* \ \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} =$$



Blur (with a mean filter) *a.k.a. Box Filter)*

Source: D. Lowe

Linear filters: examples

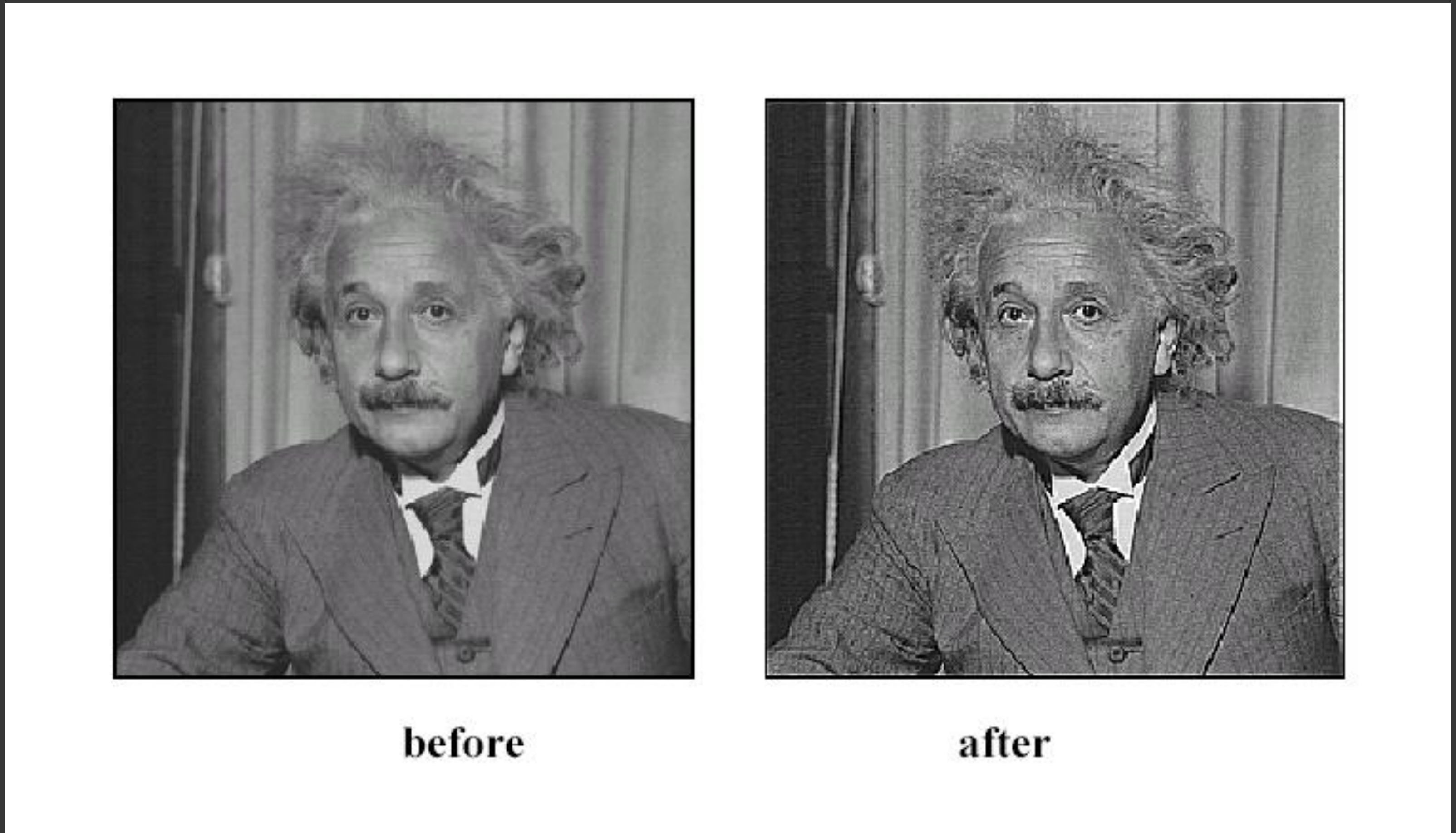
Original

$$* \left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right) =$$

Sharpening filter
(accentuates edges)

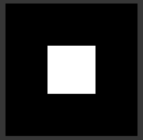
Source: D. Lowe

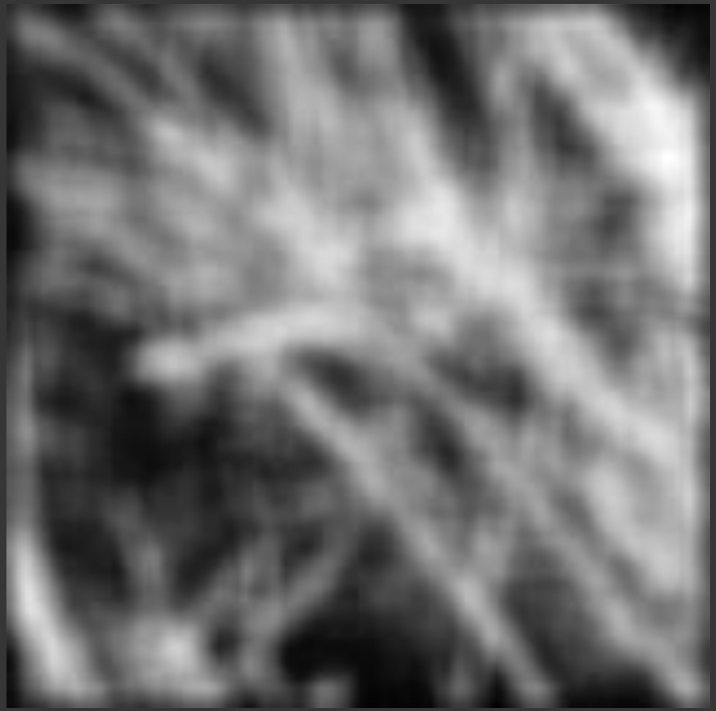
Sharpening



Source: D. Lowe

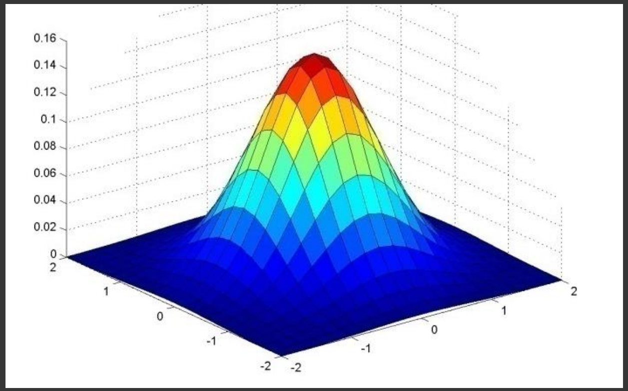
Smoothing with box filter

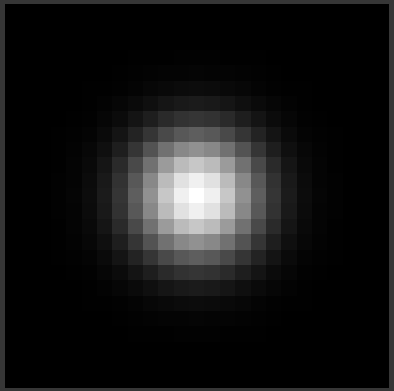




Source: D. Forsyth

Gaussian kernel

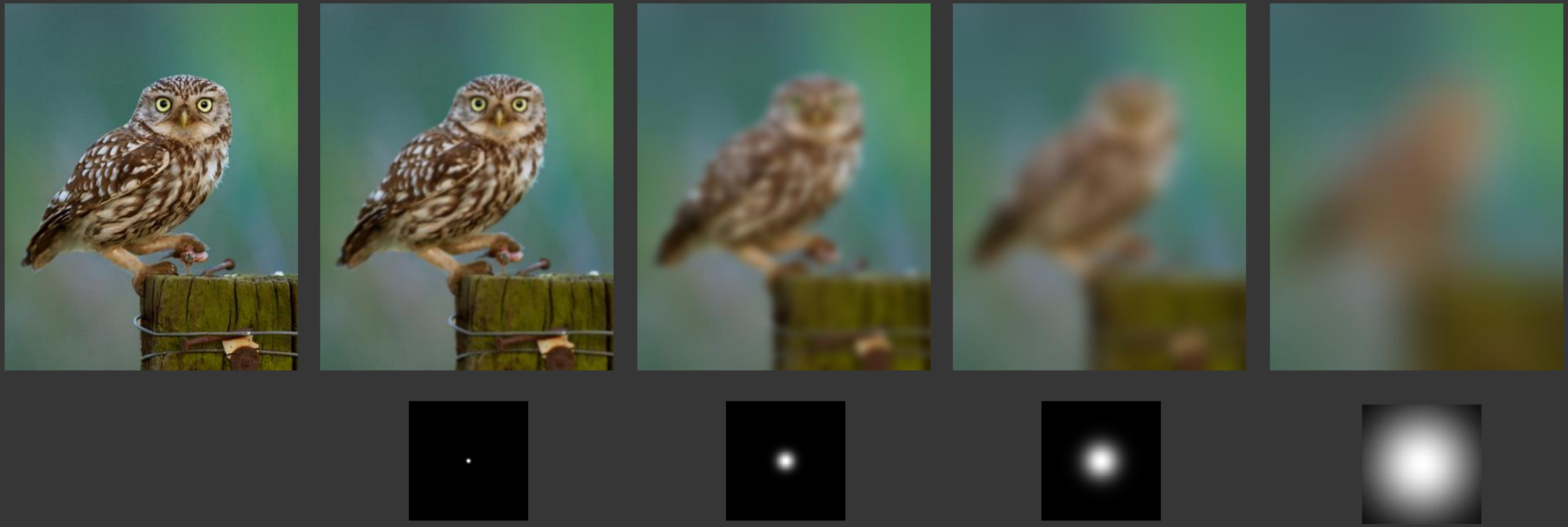




$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

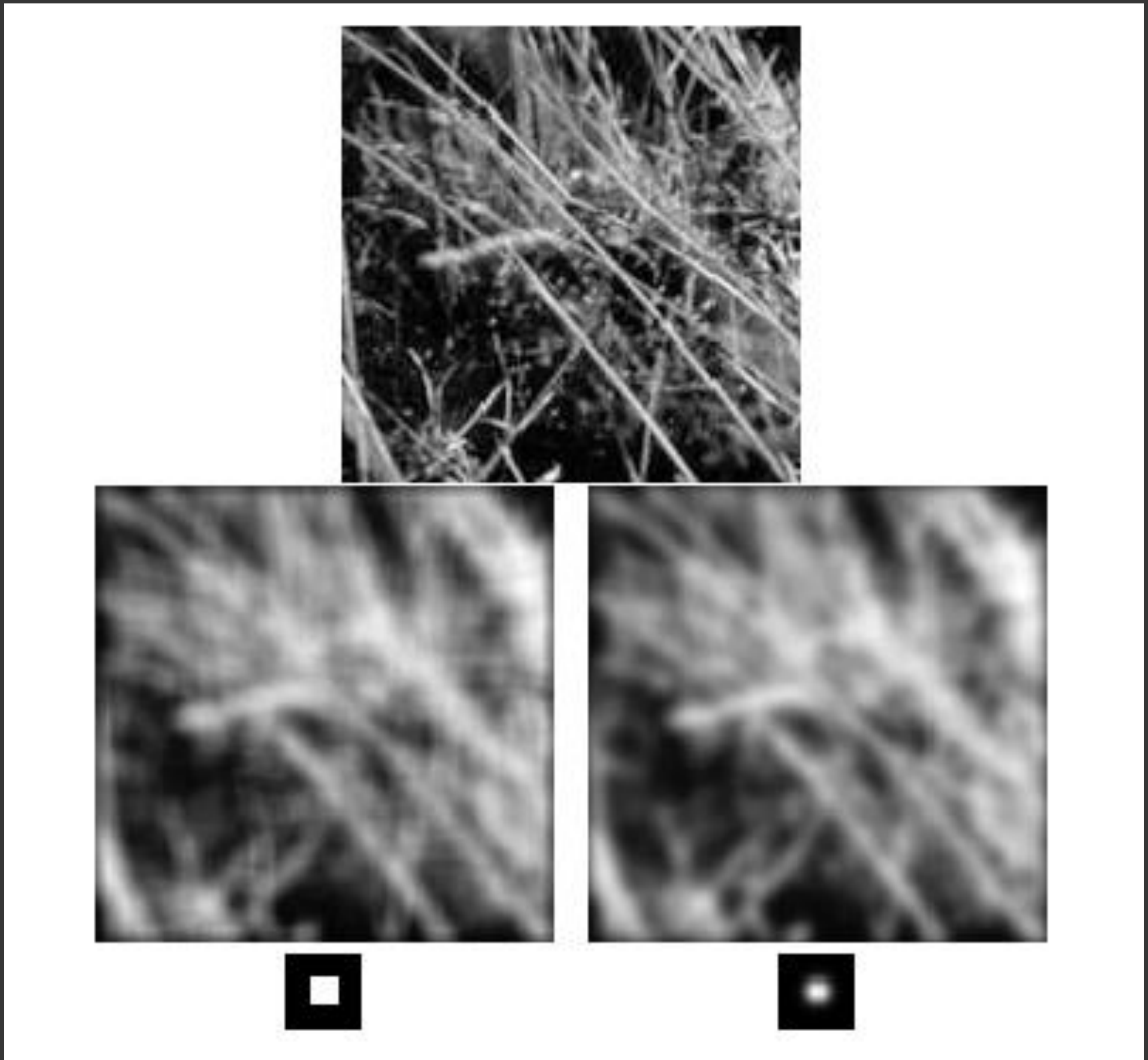
Source: C. Rasmussen

Gaussian filters



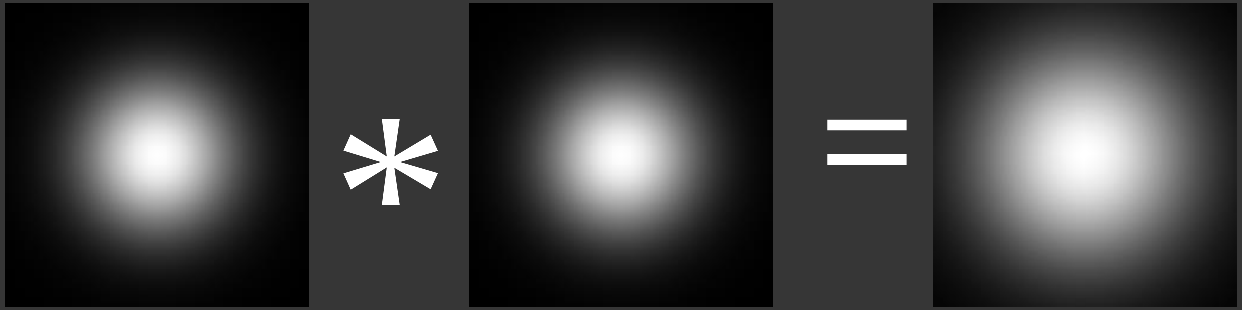
σ= 1 pixel σ= 5 pixels σ= 10 pixels σ= 30 pixels

Mean vs. Gaussian filtering



Gaussian filter

- Removes "high-frequency" components from the image (low-pass filter)
- Convolution with self is another Gaussian



  – Convolving twice with Gaussian kernel of width σ
  = convolving once with kernel of width $\sigma\sqrt{2}$

Source: K. Grauman

Sharpening revisited

- What does blurring take away?

−

=

(This "detail extraction" operation is also called a ***high-pass filter***)

Let's add it back:

+ a

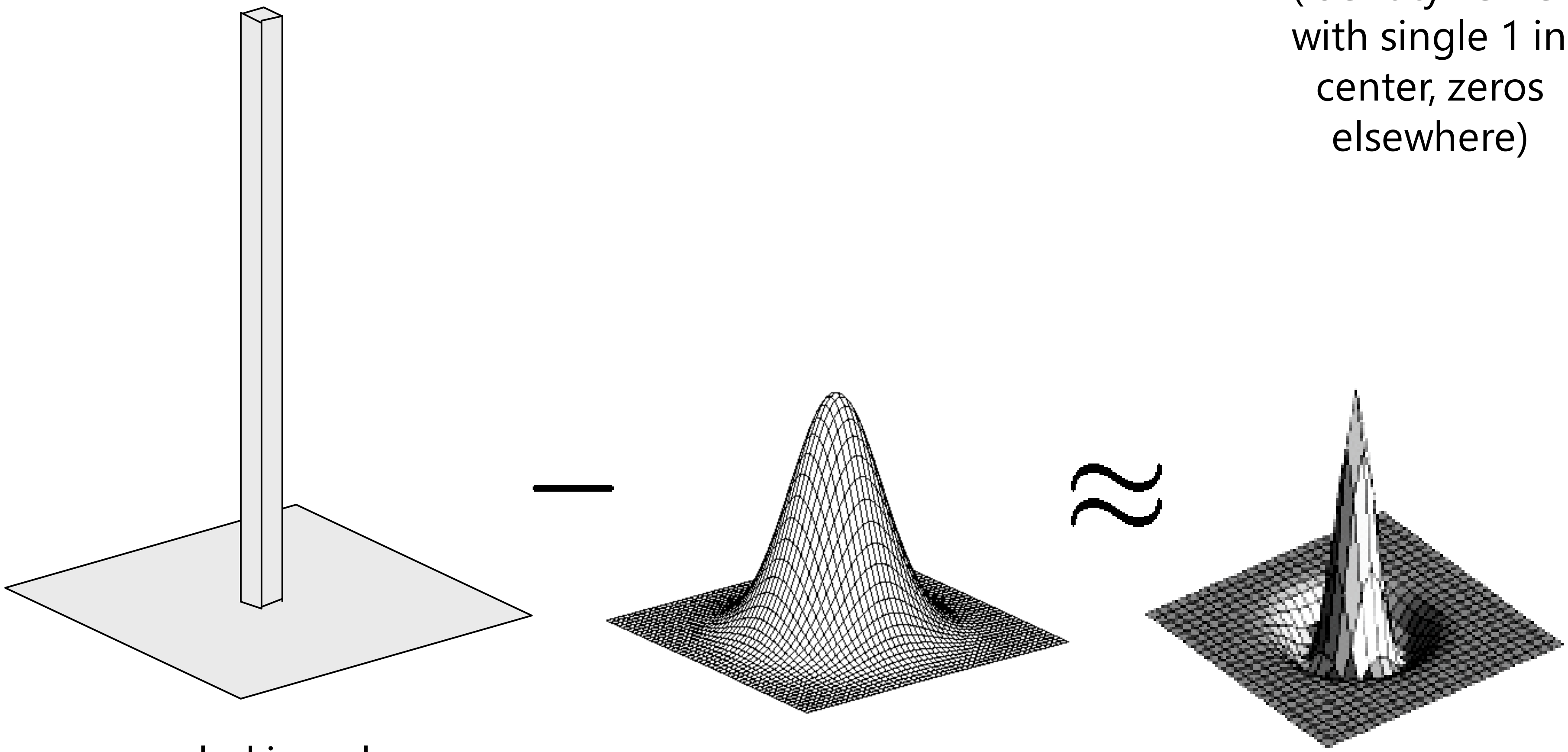
=

Photo credit: https://www.flickr.com/photos/geezaweezer/16089096376/

Sharpen filter

$$\underset{\substack{\uparrow\\ \text{image}}}{F} + \alpha\left(F - \underbrace{F * H}_{\text{blurred image}}\right) =$$

unit impulse (identity kernel with single 1 in center, zeros elsewhere)



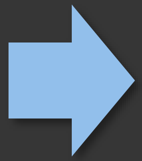
scaled impulse

Gaussian

Sharpen filter

Sharpen filter

Image Resampling & Interpolation



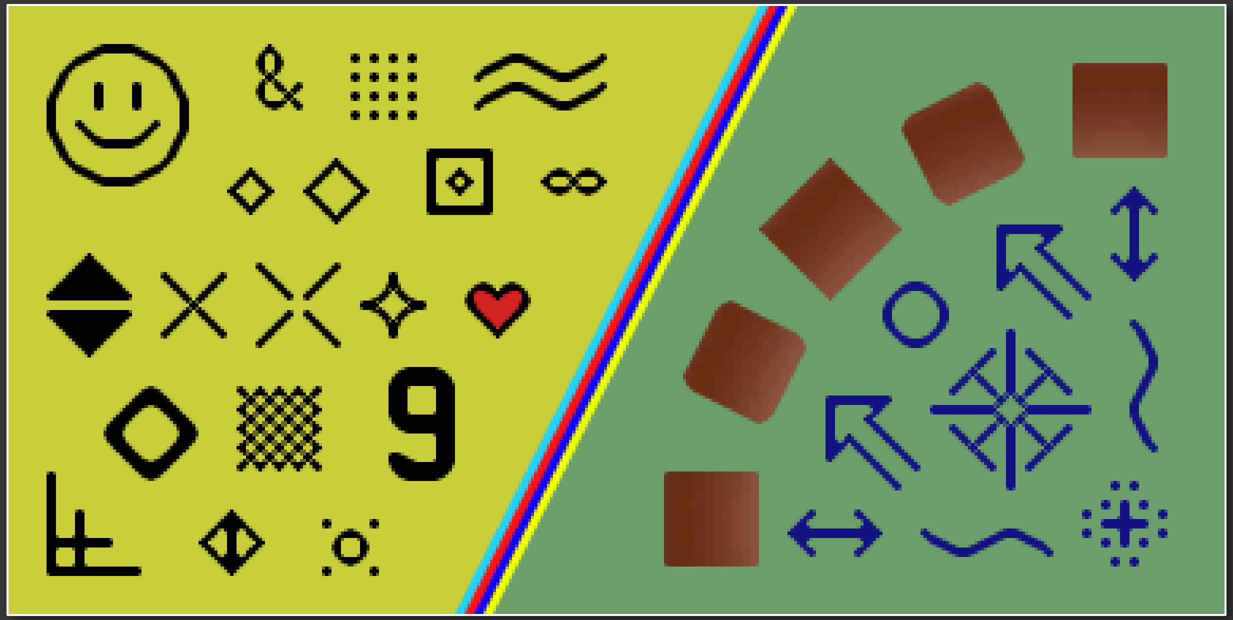


Image scaling

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

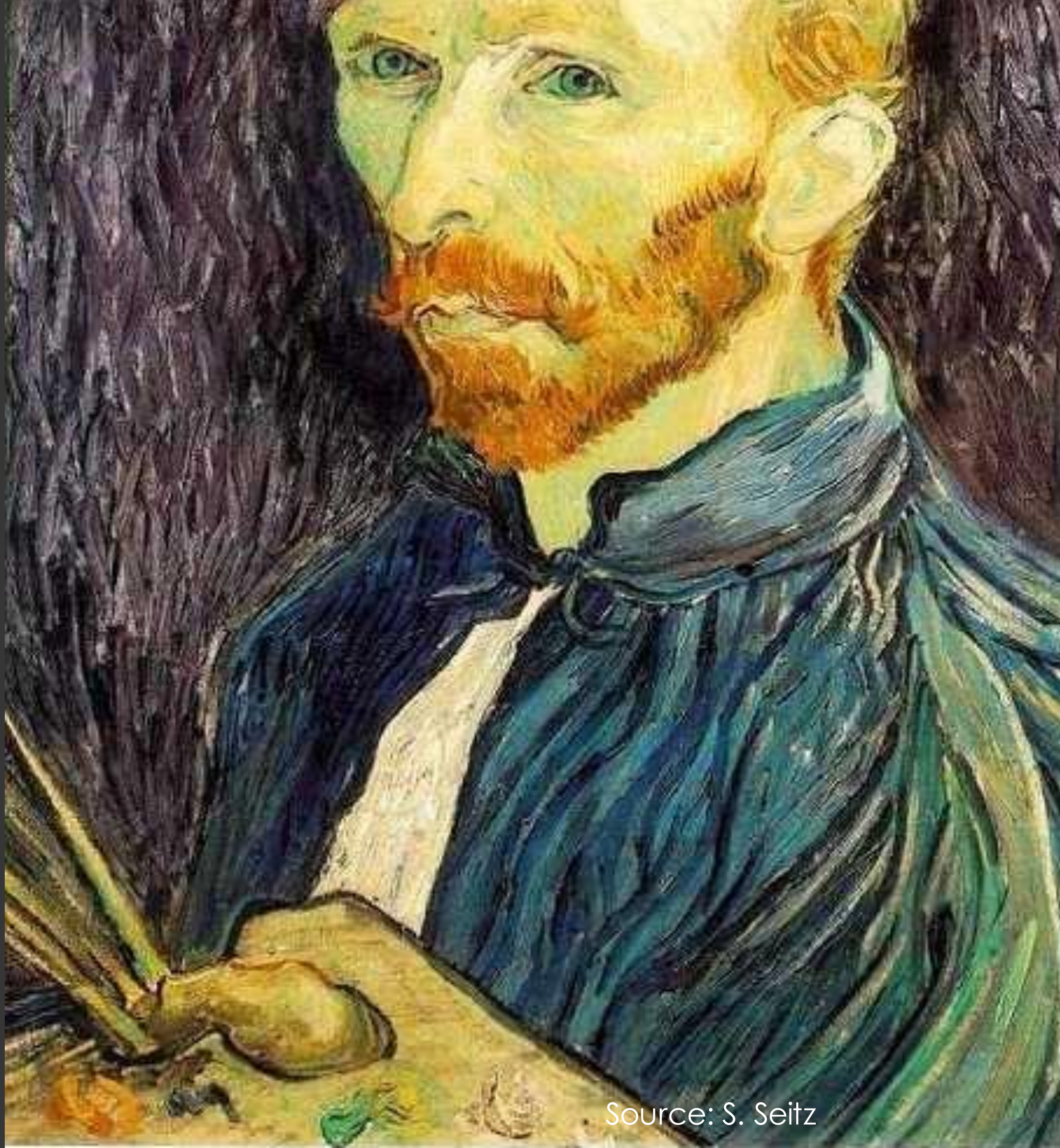
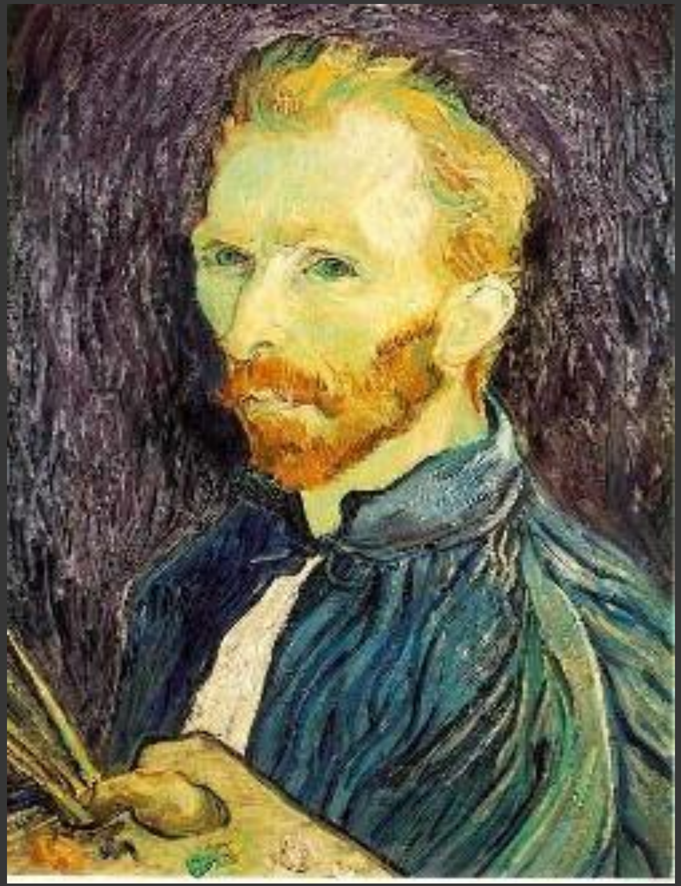


Image sub-sampling



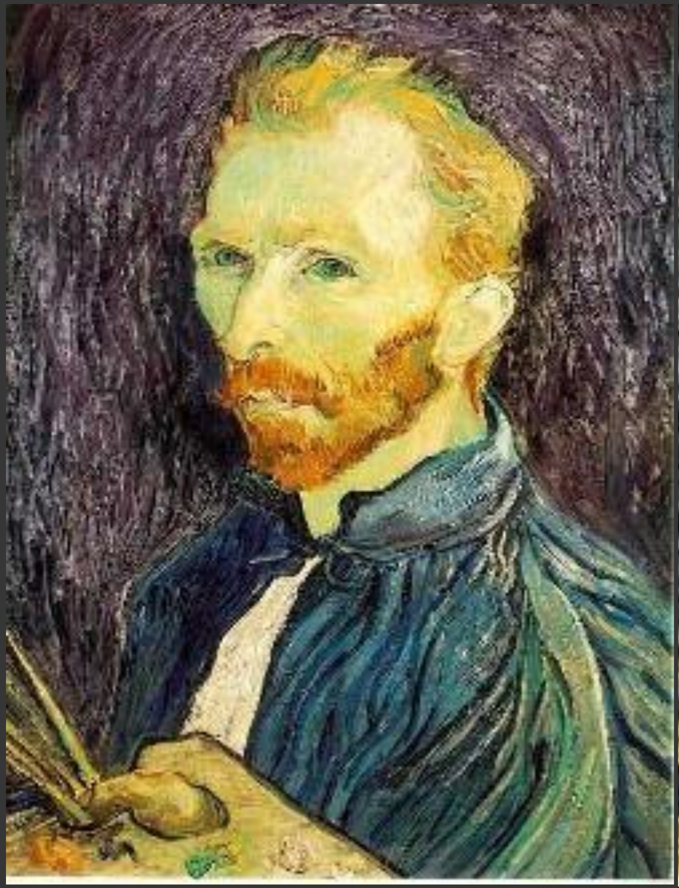
1/4

1/8

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

Source: S. Seitz

Image sub-sampling

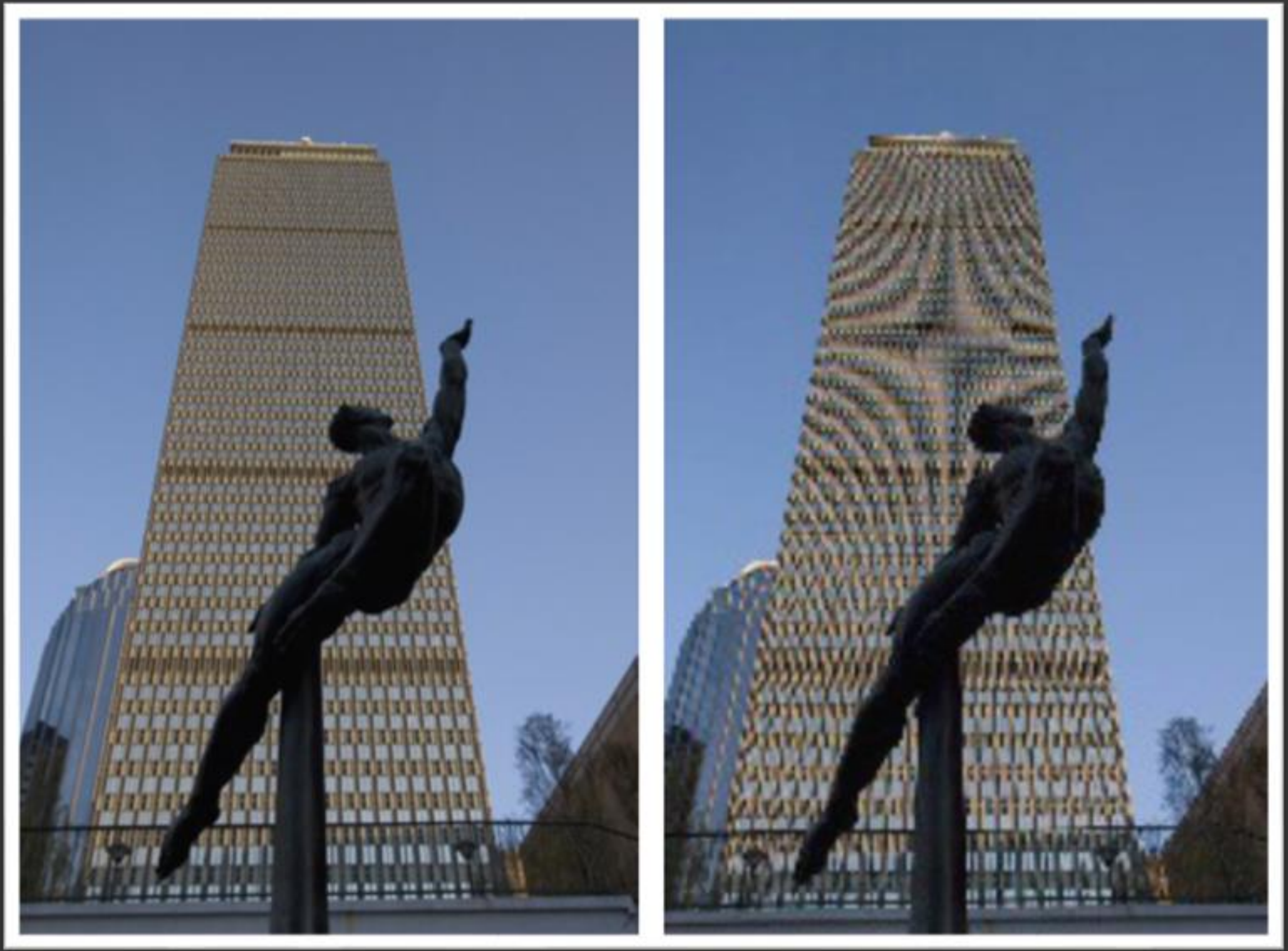


1/2 1/4 (2x zoom) 1/8 (4x zoom)

Why does this look so bad?

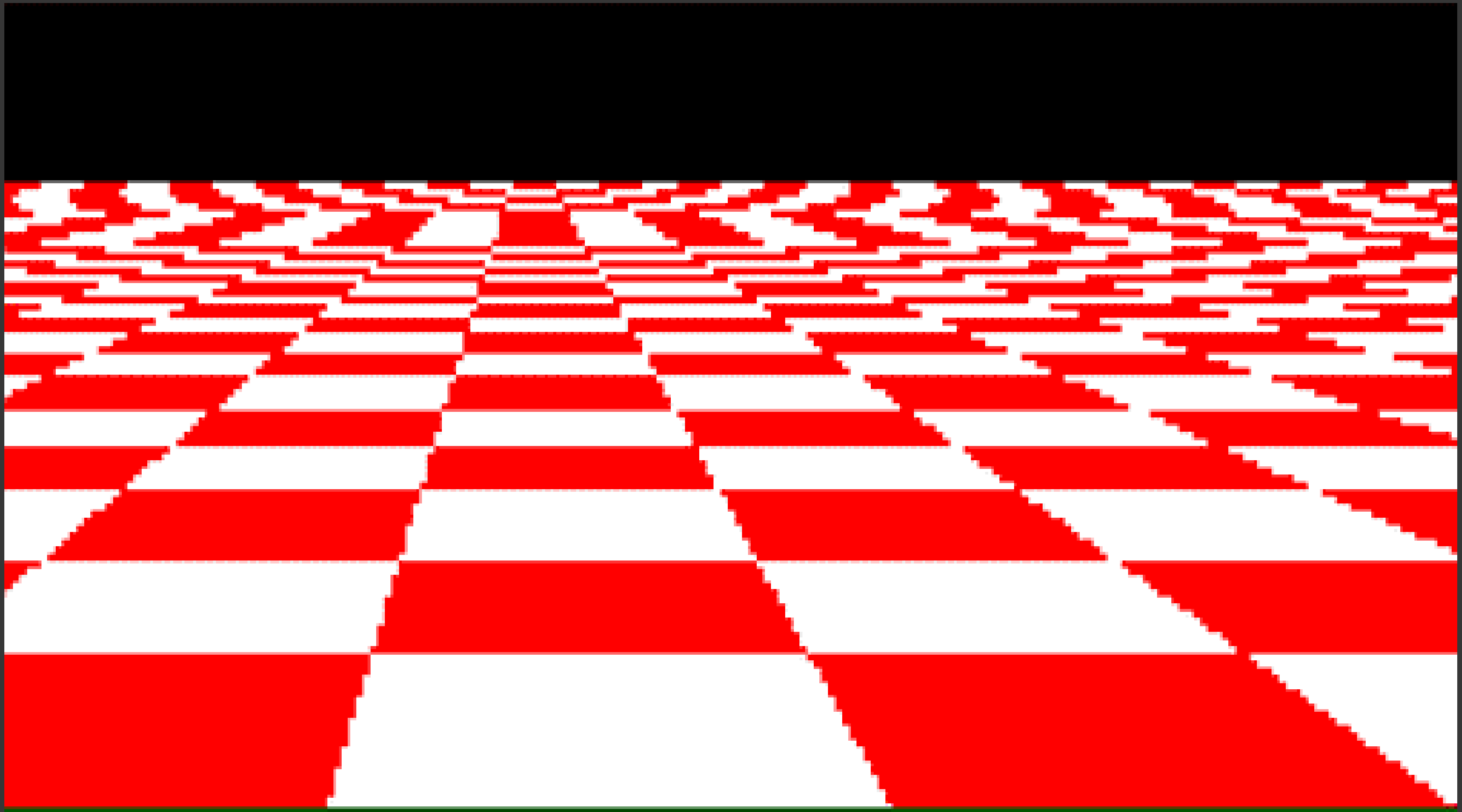
Source: S. Seitz

Image sub-sampling – another example



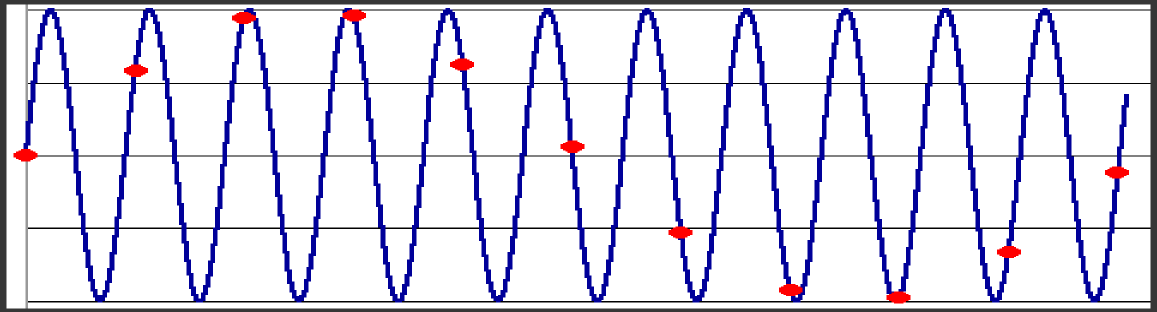
Source: F. Durand

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*

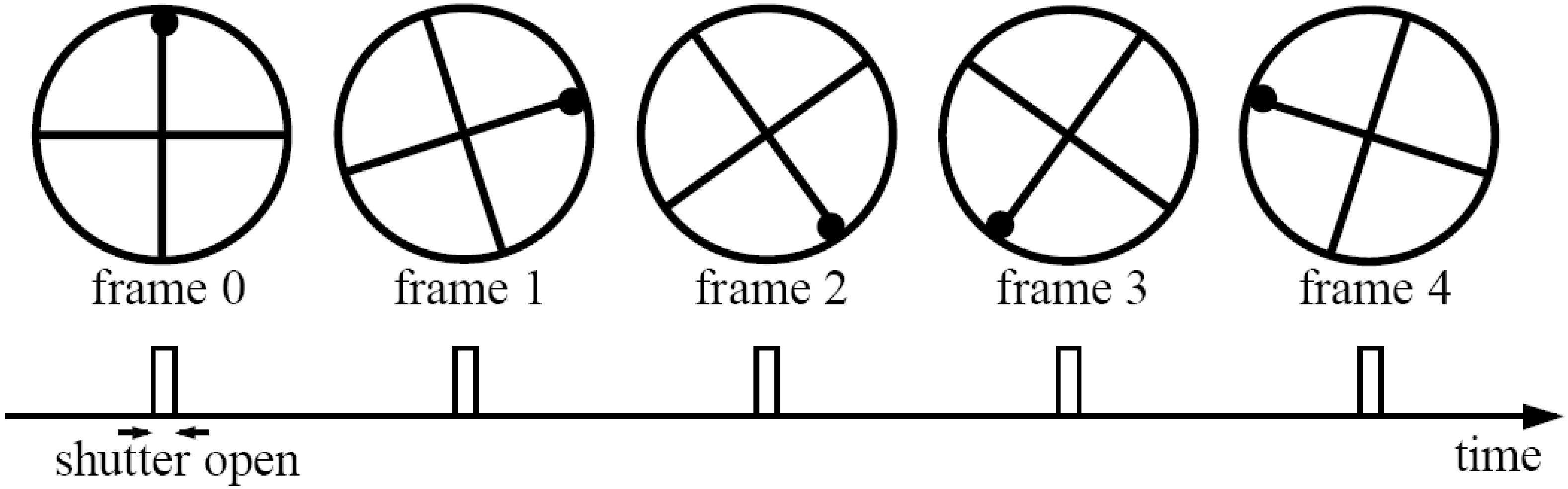
- To do sampling right, need to understand the structure of your signal/image
- Enter Monsieur Fourier…
  - "But what is the Fourier Transform? A visual introduction." https://www.youtube.com/watch?v=spUNpyF58BY
- To avoid aliasing:
  - sampling rate ≥ 2 * max frequency in the image
    - said another 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):

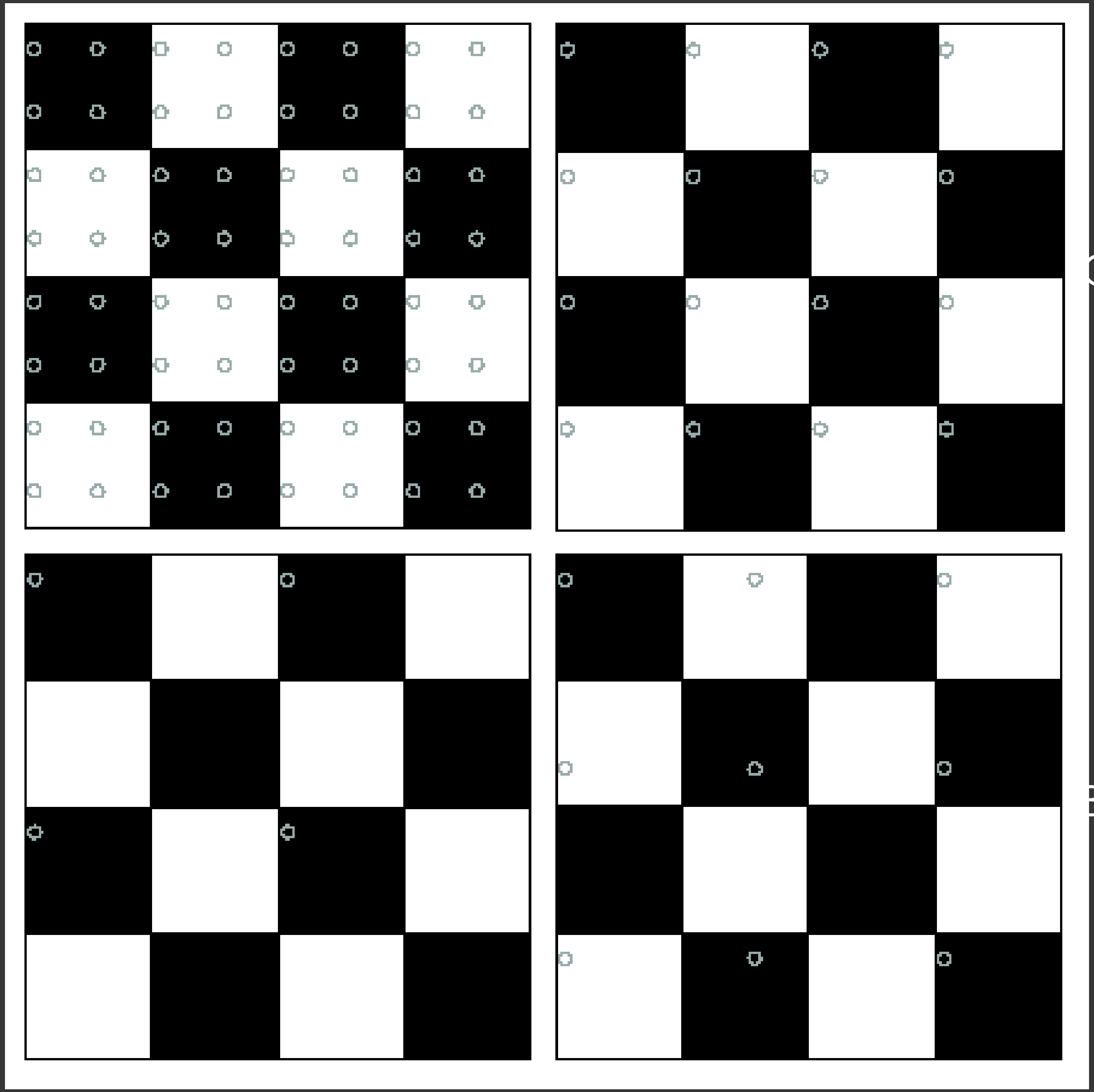


Without dot, wheel appears to be rotating slowly backwards! (counterclockwise)

(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

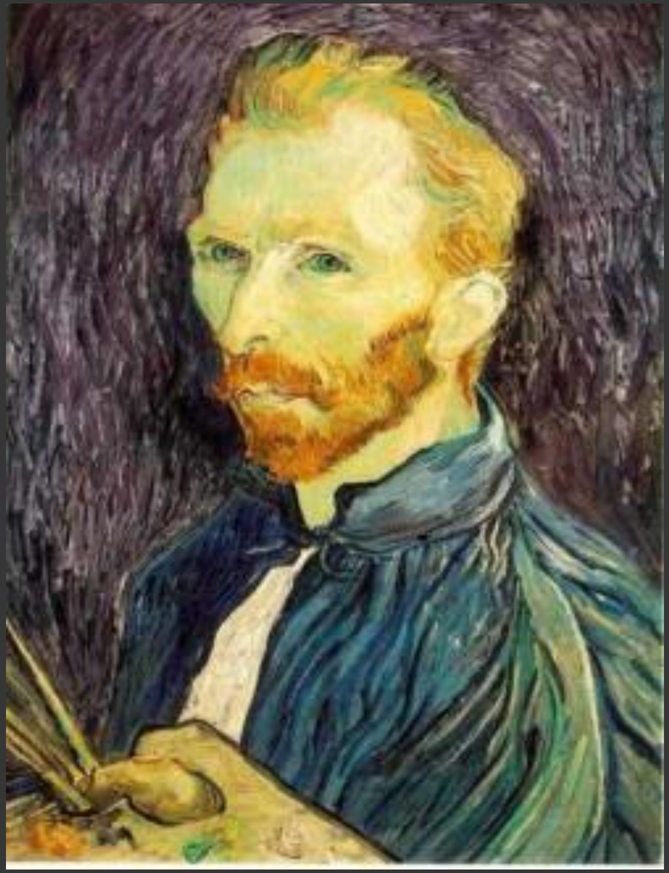
G 1/4

G 1/8

- Solution: filter the image, *then* subsample

Source: S. Seitz

Subsampling with Gaussian pre-filtering



Gaussian 1/2

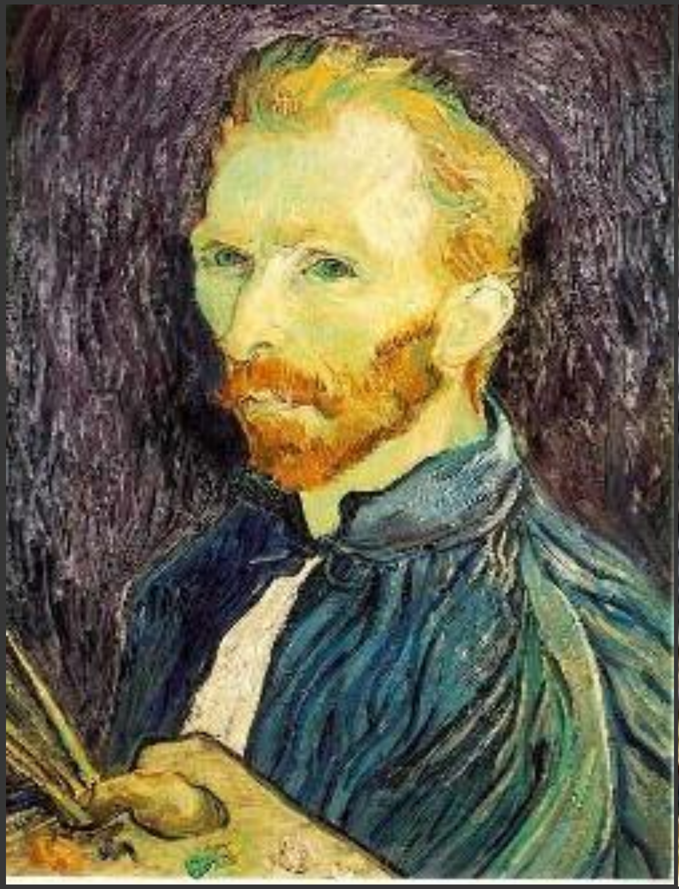
G 1/4

G 1/8

- Solution: filter the image, *then* subsample

Source: S. Seitz

Compare with...


1/2

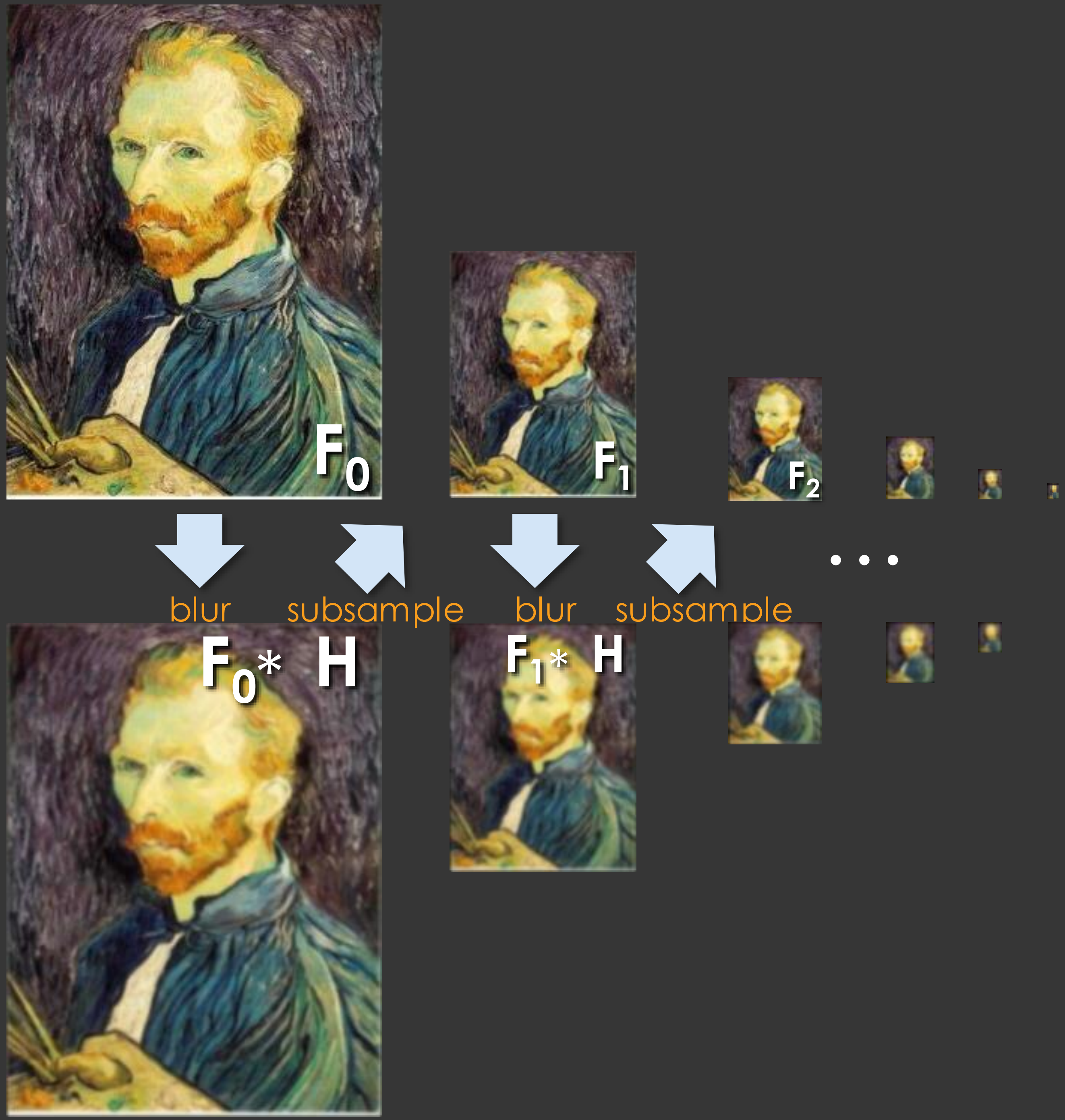

1/4 (2x zoom)

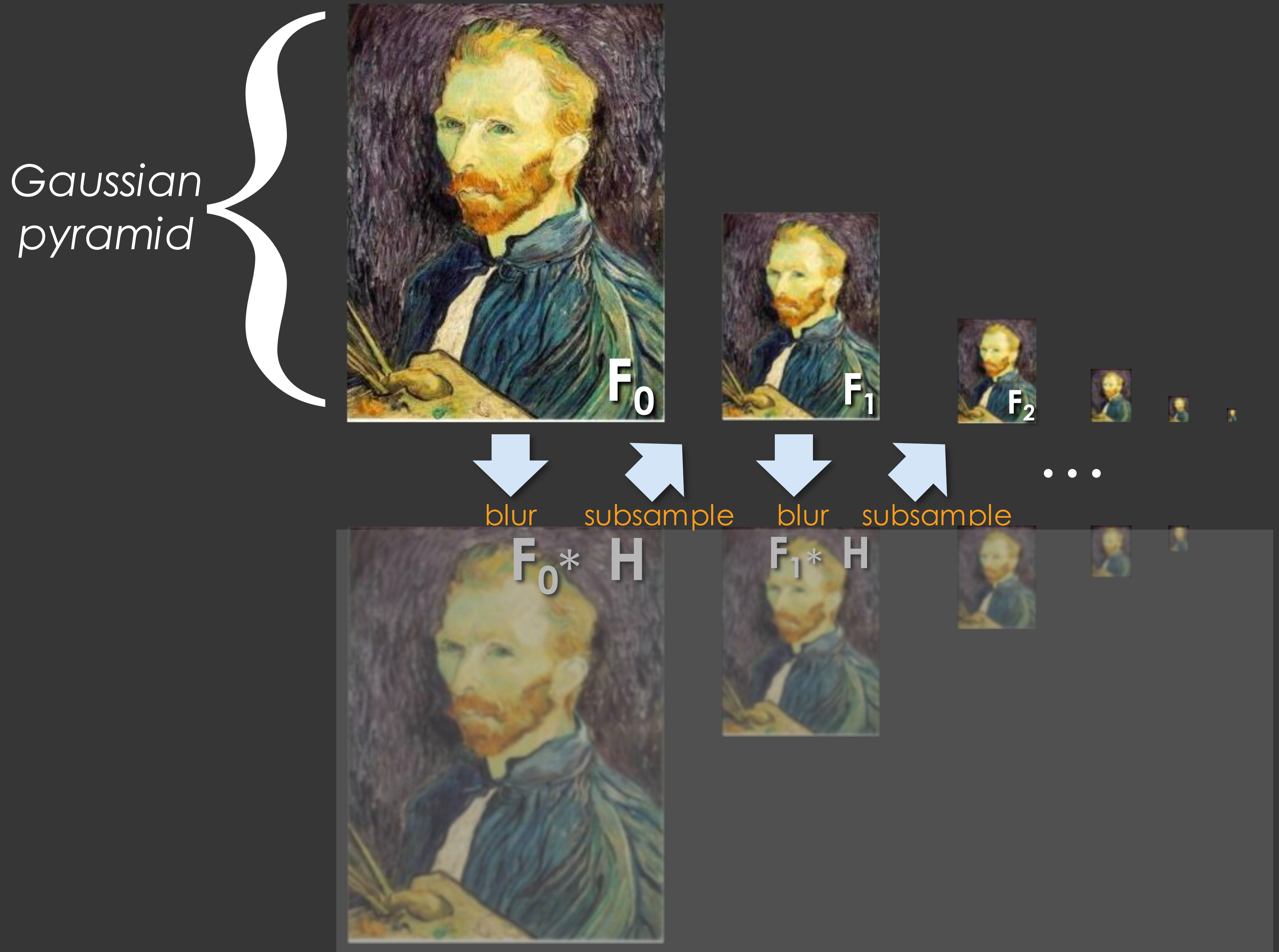

1/8 (4x zoom)

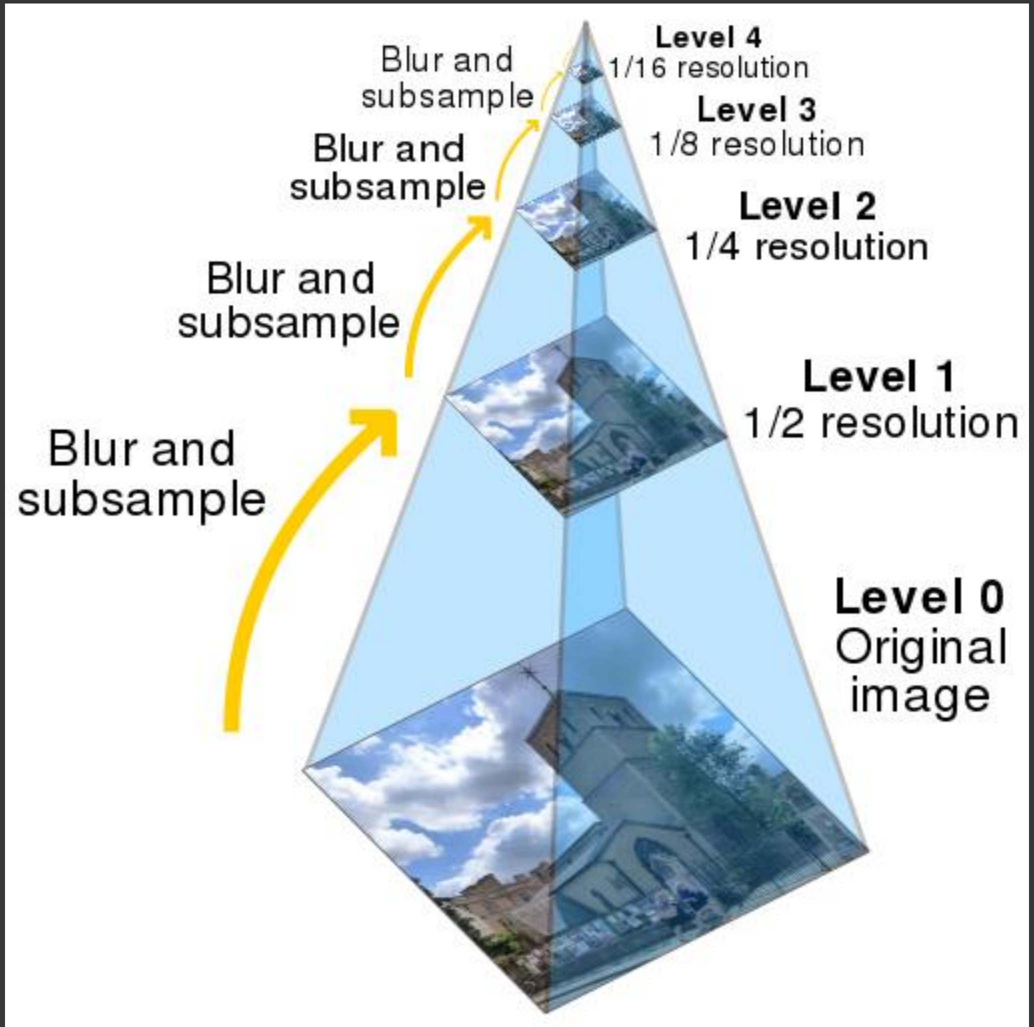
Source: S. Seitz

Gaussian pre-filtering

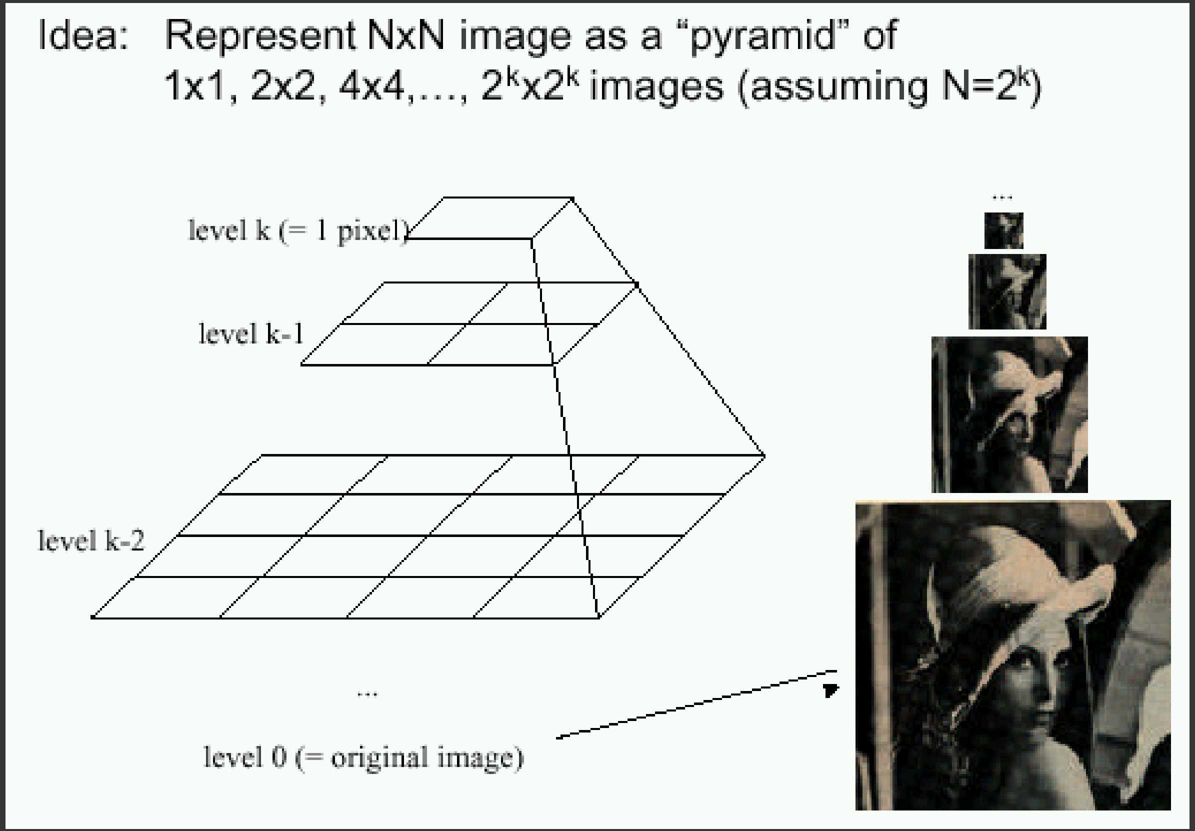
- Solution: filter the image, *then* subsample




Gaussian pyramid
F0
F1
F2
blur
subsample
blur
subsample
F0* H
F1* H


Level 4
1/16 resolution
Blur and subsample
Level 3
1/8 resolution
Blur and subsample
Level 2
1/4 resolution
Blur and subsample
Level 1
1/2 resolution
Blur and subsample
Level 0
Original image

Gaussian pyramids [Burt and Adelson, 1983]

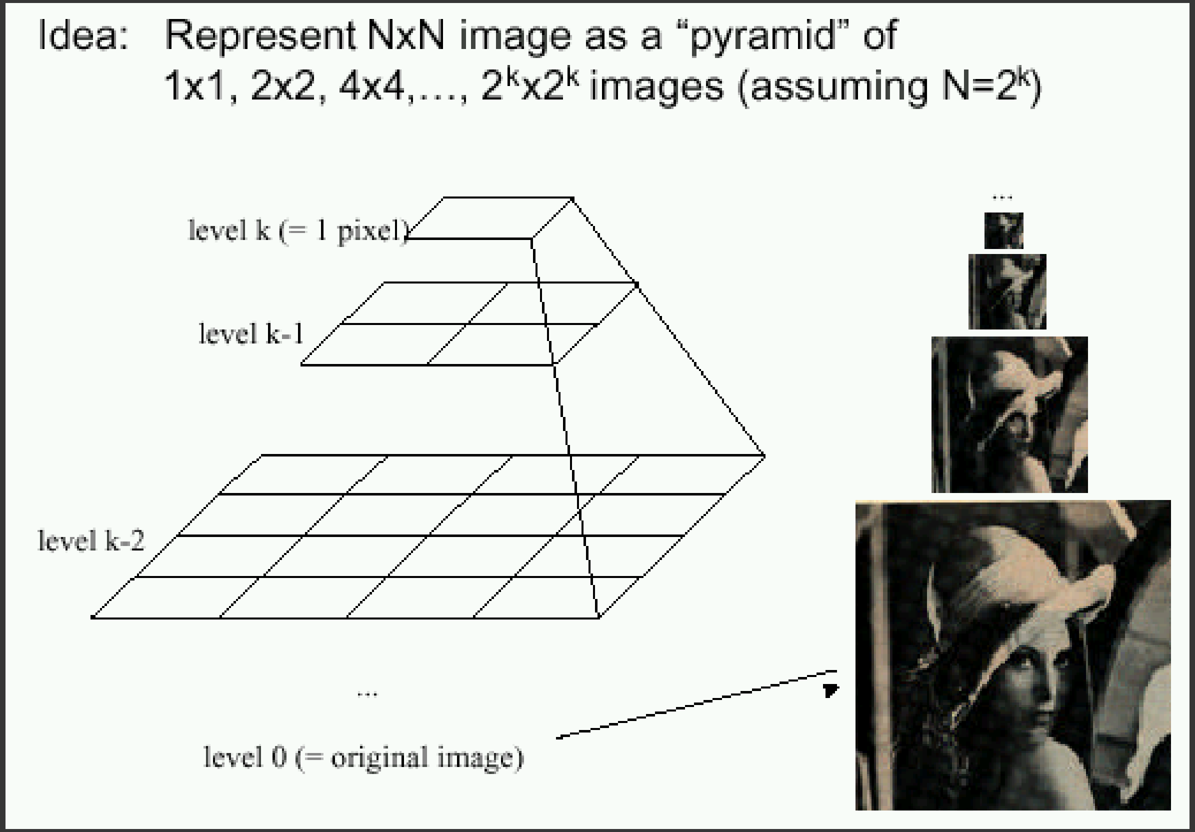


- In computer graphics, a *mip map* [Williams, 1983]
- A precursor to *wavelet transform*

Gaussian Pyramids have all sorts of applications in computer vision

Source: S. Seitz

Gaussian pyramids [Burt and Adelson, 1983]

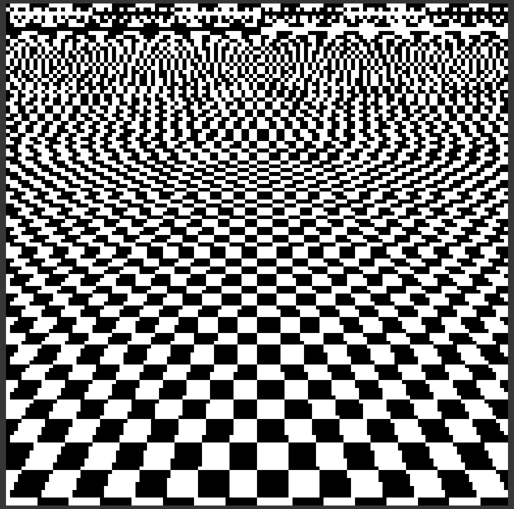


- How much space does a Gaussian pyramid take compared to the original image?

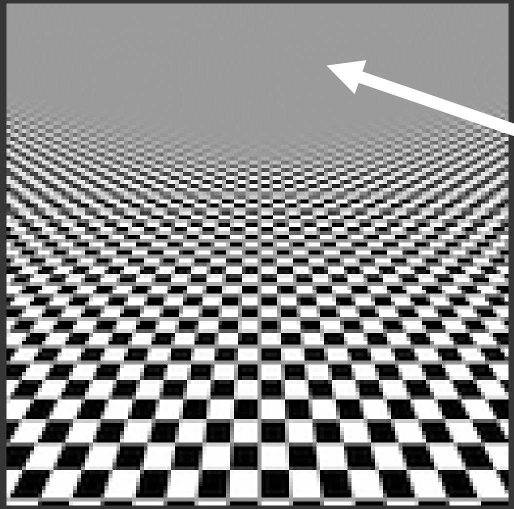
Source: S. Seitz

Back to the checkerboard

- What should happen when you make the checkerboard smaller and smaller?



Naïve subsampling



Proper prefiltering
("antialiasing")

Image turns grey! (Average of black and white squares, because each pixel contains both.)

Upsampling

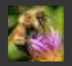
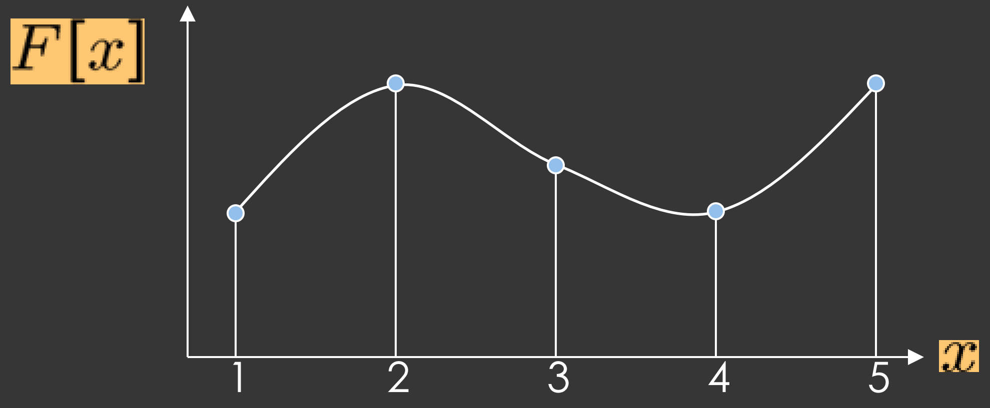
- This image is too small for this screen: 
- How can we make it 10 times as big?
- Simplest approach: repeat each row and column 10 times
- ("Nearest neighbor interpolation")

Image interpolation



d = 1 in this example

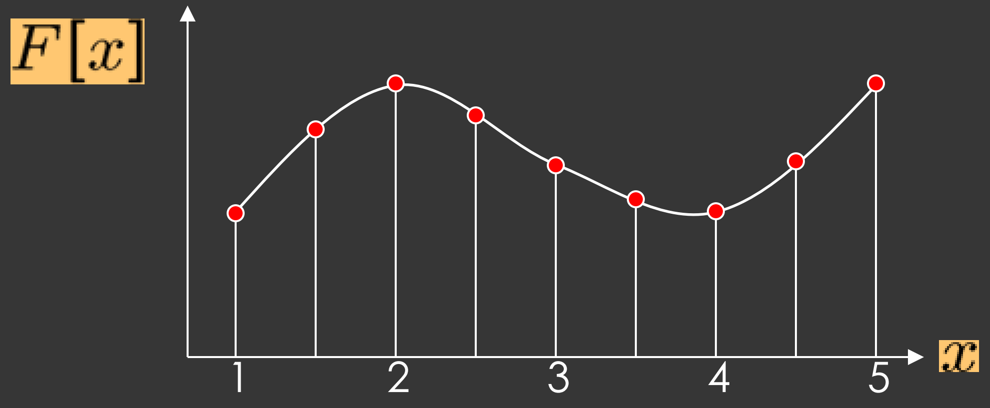
Recall that a digital images is formed as follows:

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

- 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

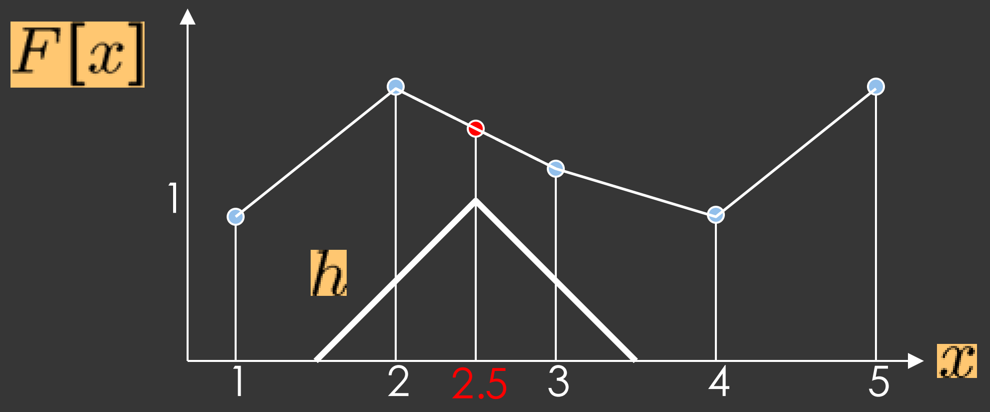
Recall that a digital images is formed as follows:

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

- 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:

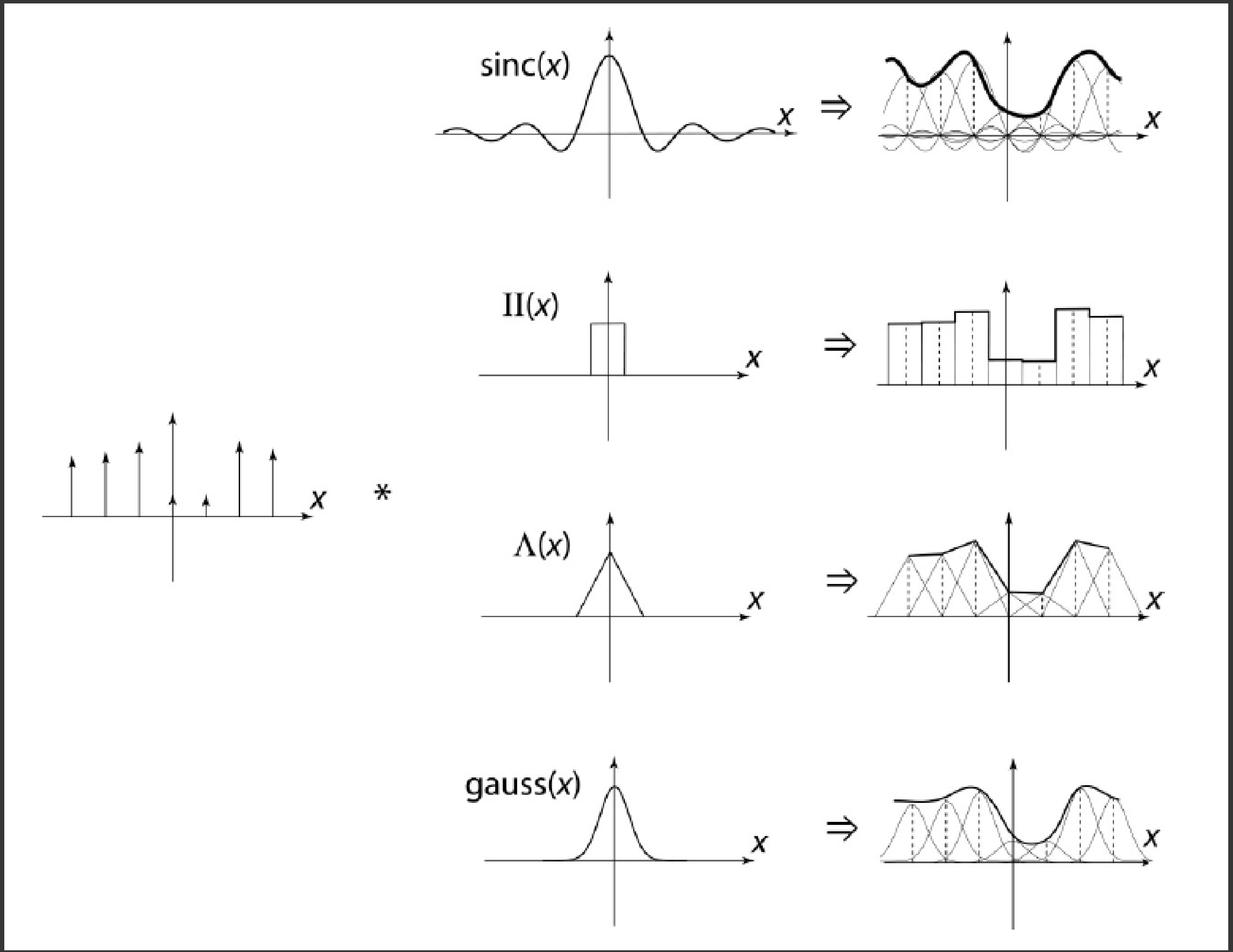
$$f_F(x) = F(\tfrac{x}{d}) \text{ when } \tfrac{x}{d} \text{ is an integer, 0 otherwise}$$

- 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

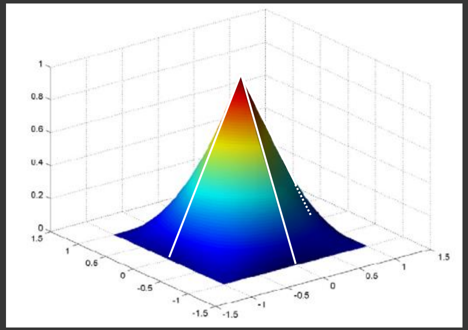
Reconstruction filters

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

$h(x)$

performs
linear interpolation

$h(x, y)$



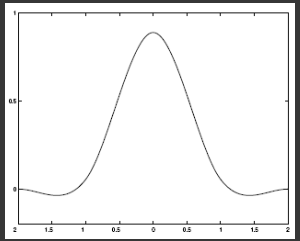
(tent function) performs
bilinear interpolation

Often implemented without cross-correlation

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

Better filters give better resampled images

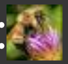
- **Bicubic** is common choice



Cubic reconstruction filter

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

Image interpolation

Original image:  x 10

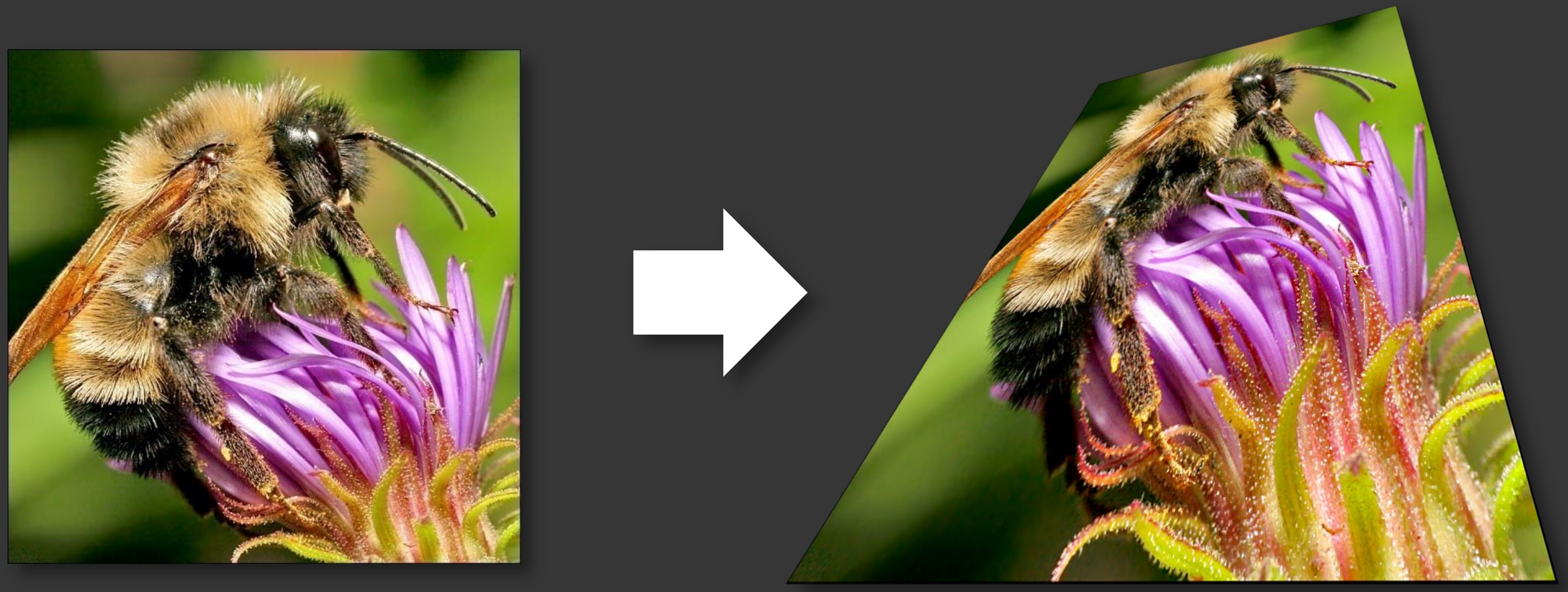
Nearest-neighbor interpolation

Bilinear interpolation

Bicubic interpolation

Image interpolation

Also used for *resampling*



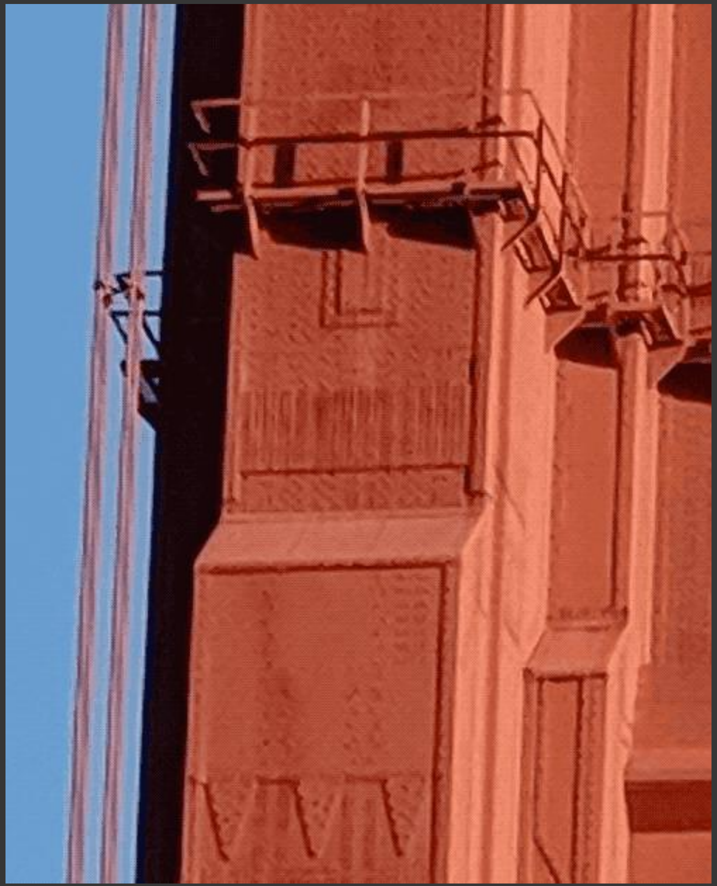
Depixelating Pixel Art

Super-resolution with multiple images

- Can do better upsampling if you have multiple images of the scene taken with small (subpixel) shifts
- Some cellphone cameras (like the Google Pixel line) capture a **burst** of photos
- Can we use that burst for upsampling?

Google Pixel 3 Super Res Zoom

Effect of hand tremor as seen in a cropped burst of photos, after global alignment



Example photo with and without super res zoom (smart burst align and merge)

https://ai.googleblog.com/2018/10/see-better-and-further-with-super-res.html