THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision

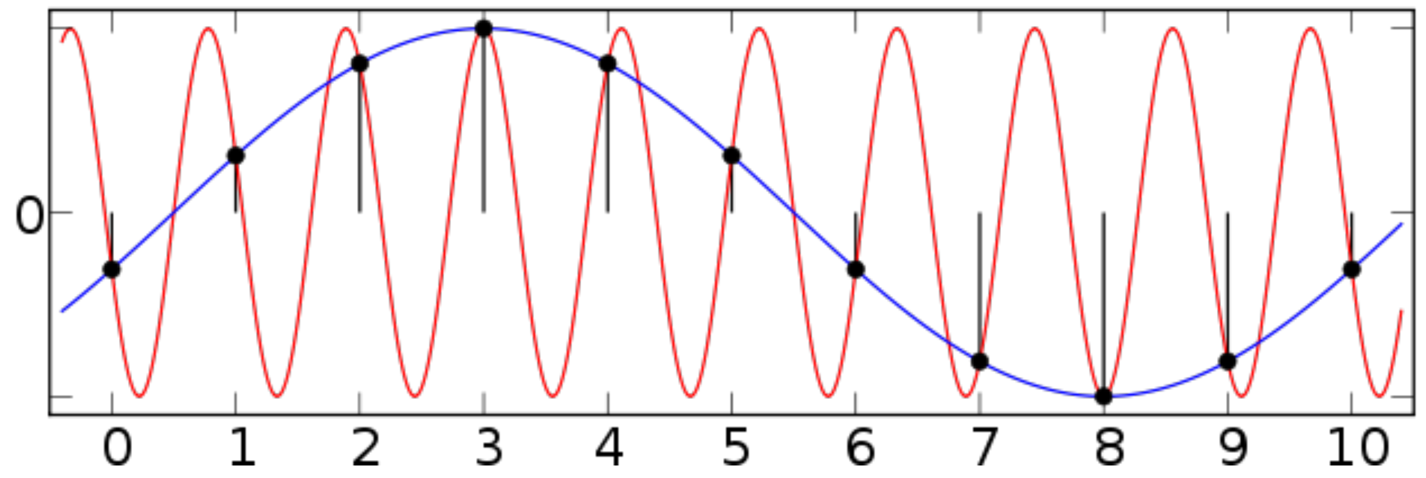


Image Credit: https://en.wikibooks.org/wiki/Analog_and_Digital_Conversion/Nyquist_Sampling_Rate

Lecture 6: Sampling

(unless otherwise stated slides are taken or adopted from **Bob Woodham, Jim Little** and **Fred Tung**)

Menu for Today

Topics:

— **Sampling** theory
— **Nyquist** rate
— Color **Filter Arrays**
— **Image** encoding

Readings:

— **Today's** Lecture: Szeliski 2.3, Forsyth & Ponce (2nd ed.) 4.5, 4.6

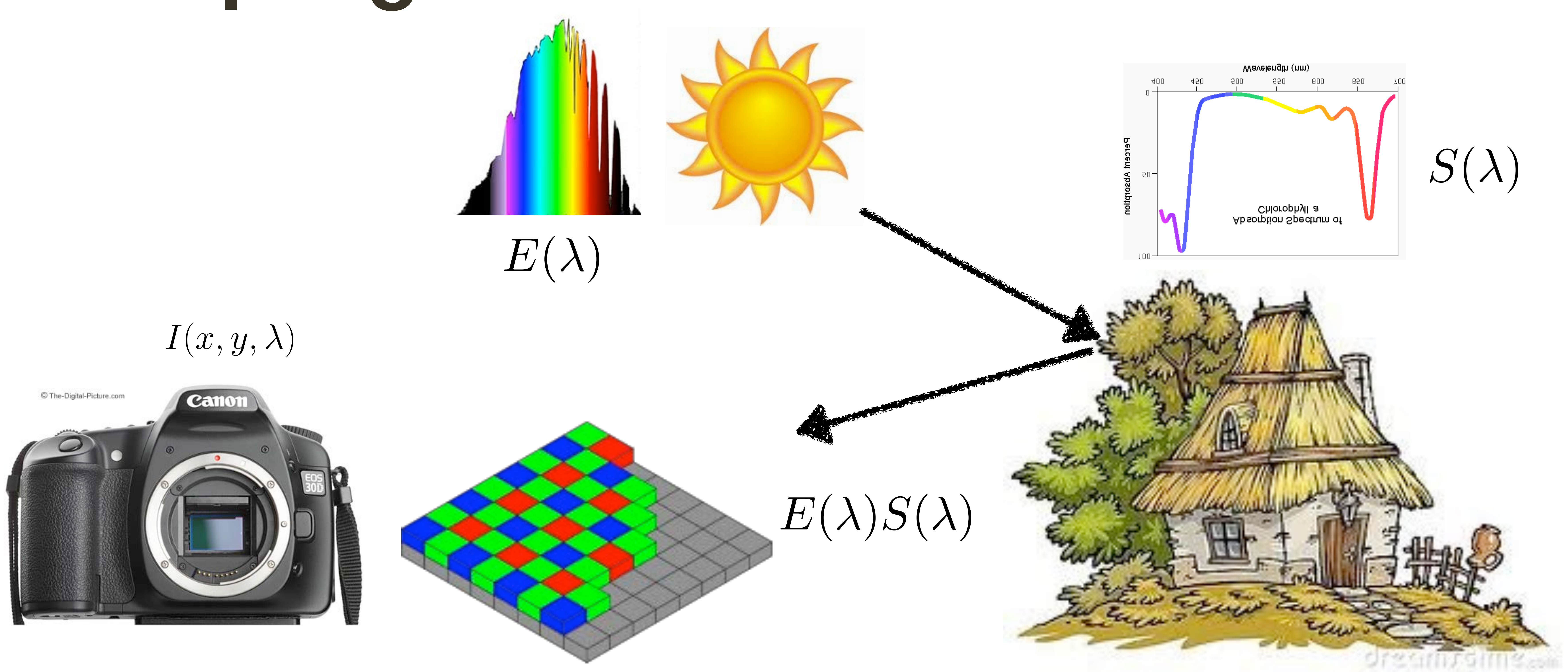
Reminders:

— **Assignment 1:** Image Filtering and Hybrid Images due **September 28th**

Goal

1. Understand discrete =/= continuous
2. How do we then deal with this?

What is **Sampling**?



A **continuous function** $I(x, y, \lambda)$ is presented at the image sensor at each time instant

How do we convert this to a **digital signal** (array of numbers)?

How can we **manipulate**, e.g., resample, this digital signal correctly?

Resampling Images

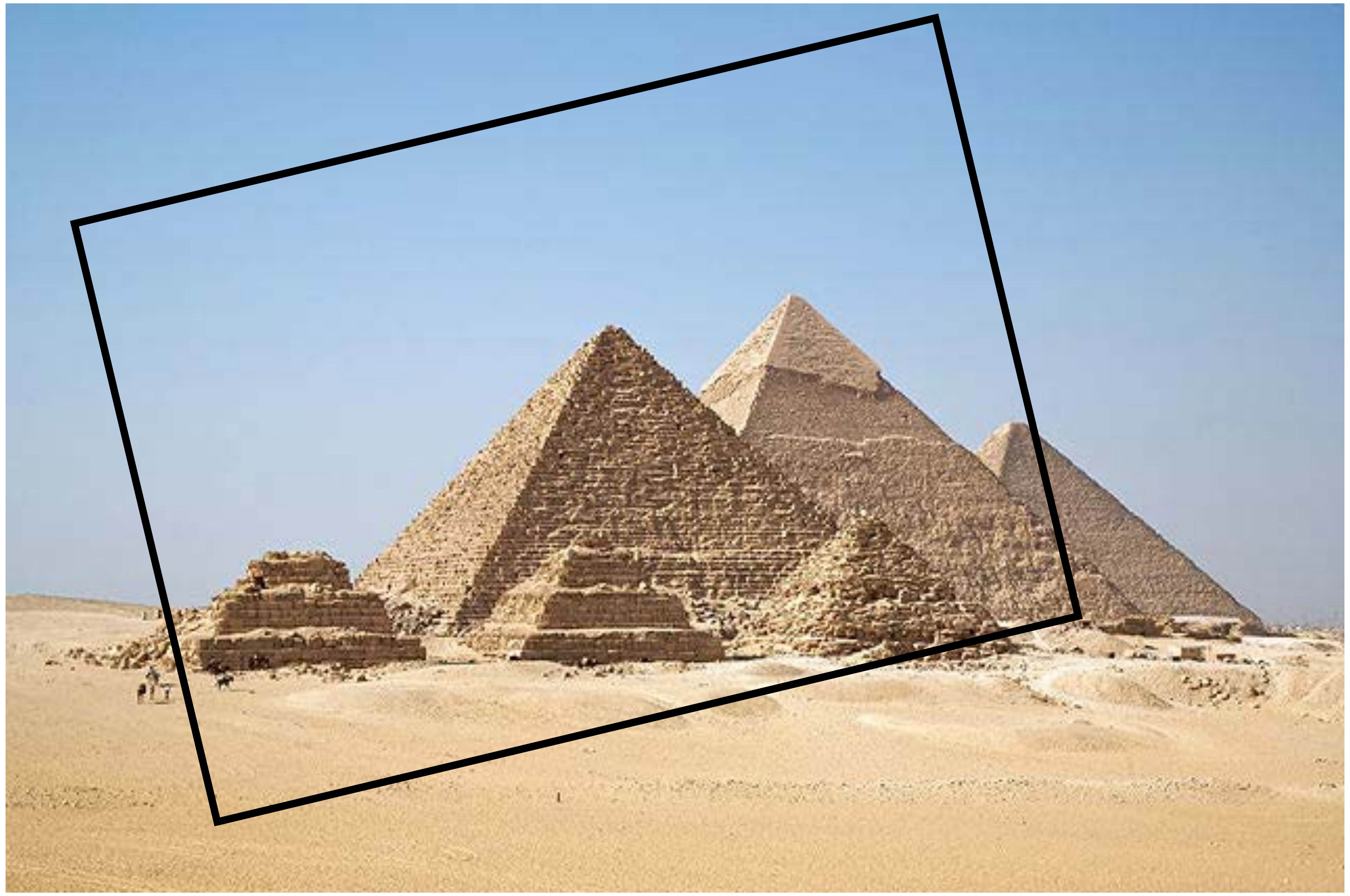
width

height

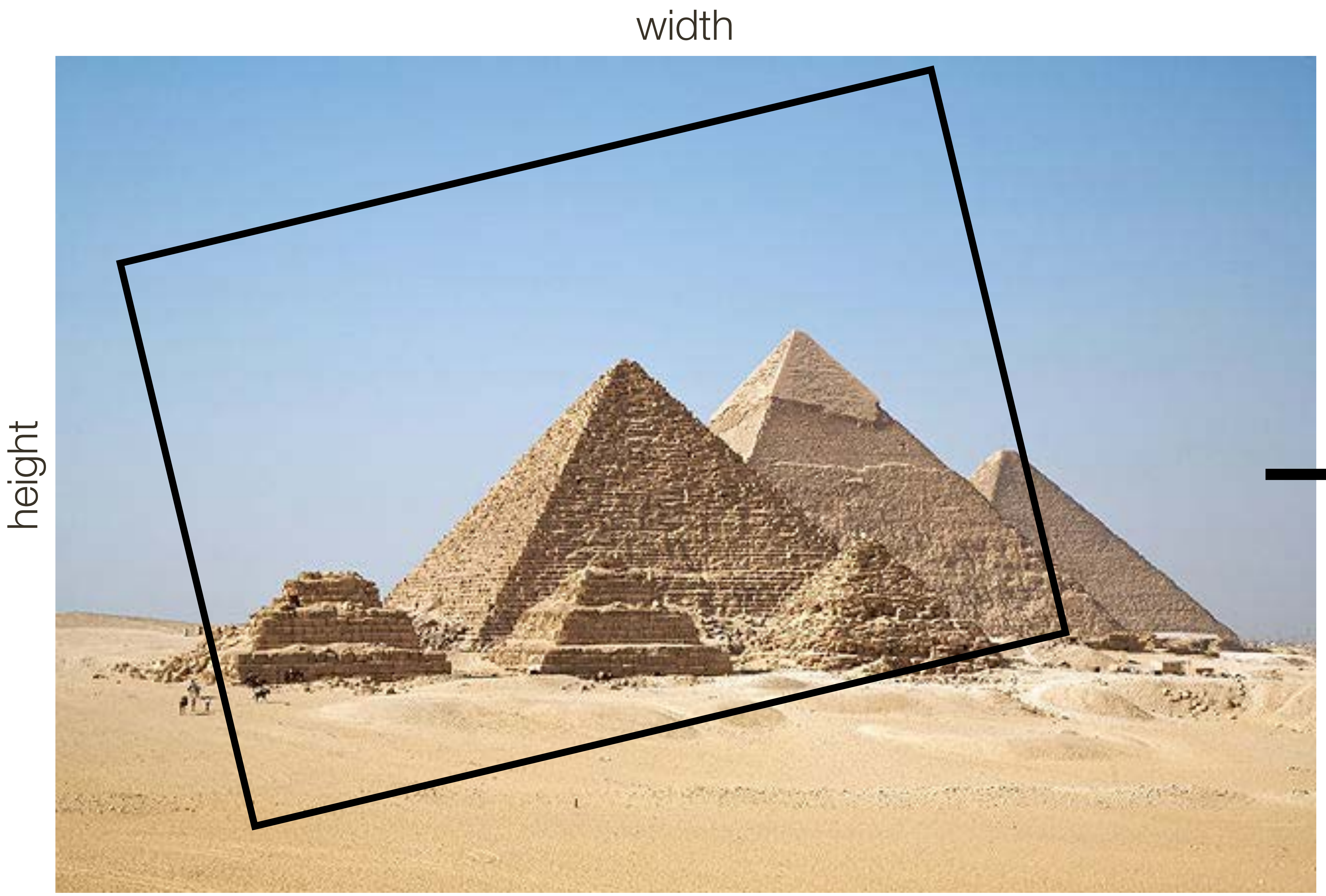
Resampling Images

width

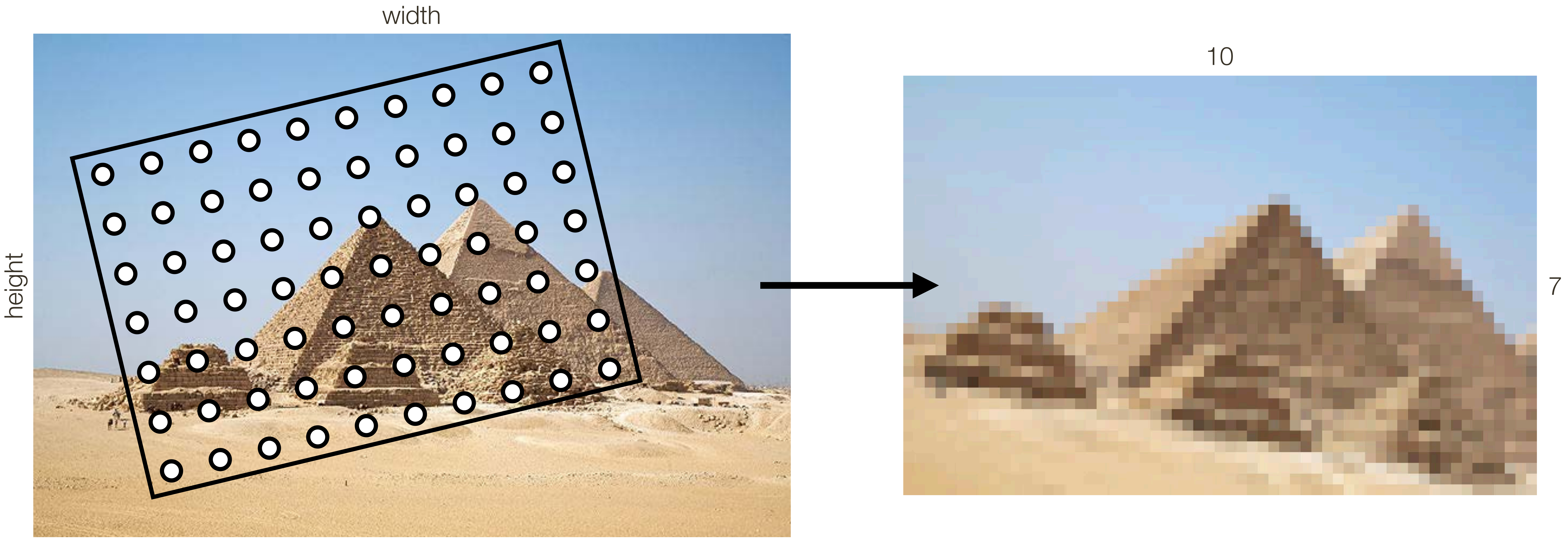
height



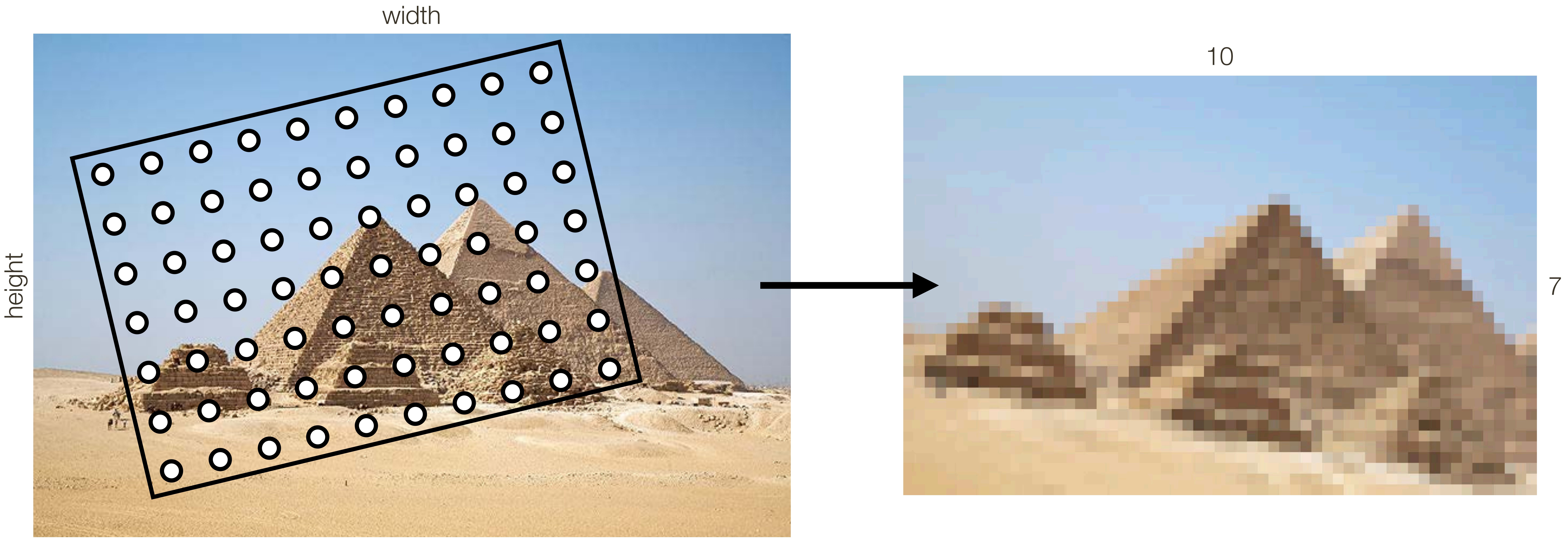
Resampling Images



Resampling Images



Resampling Images



How do we correctly **generate samples** to resample or warp an image?

What types of **transformations** can we do?

$I(X,Y)$

Filtering

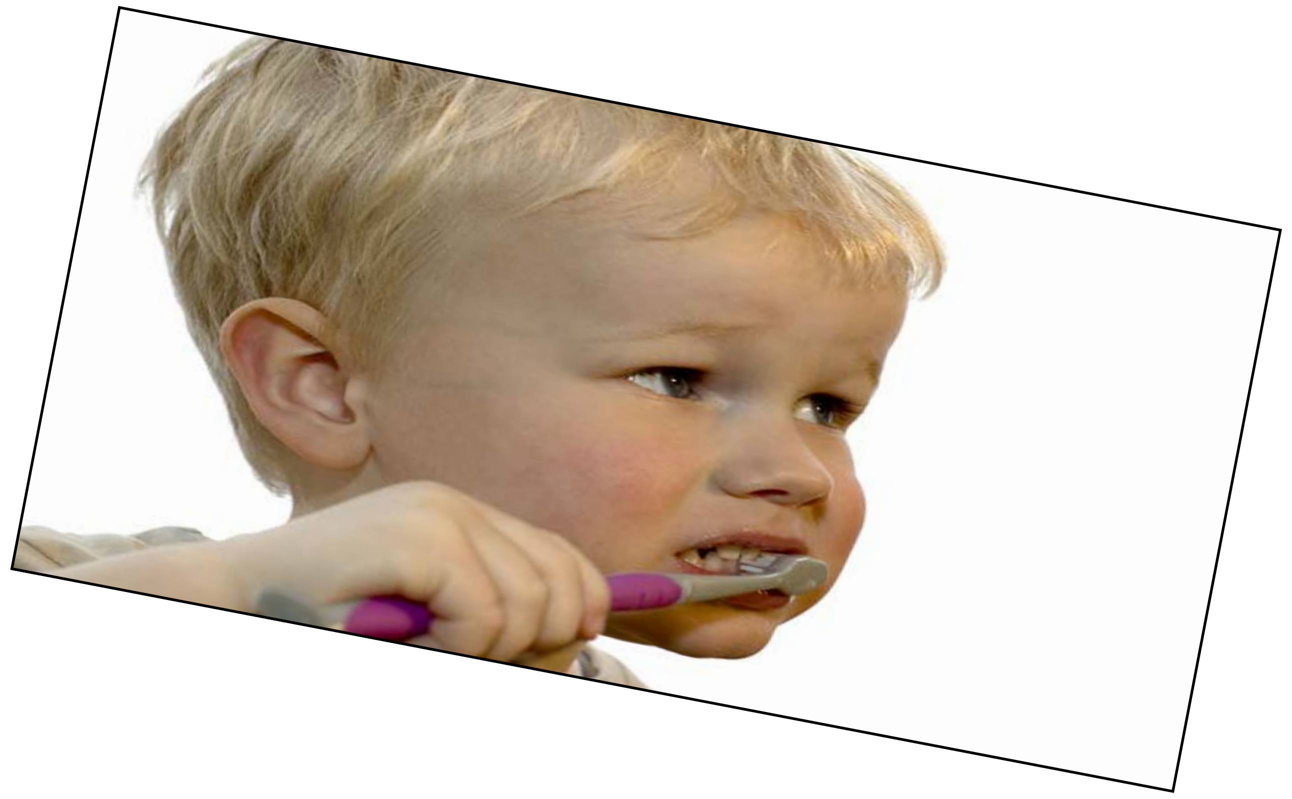
$I'(X,Y)$

changes range of image function

$I(X,Y)$

Warping

$I'(X,Y)$



changes domain of image function

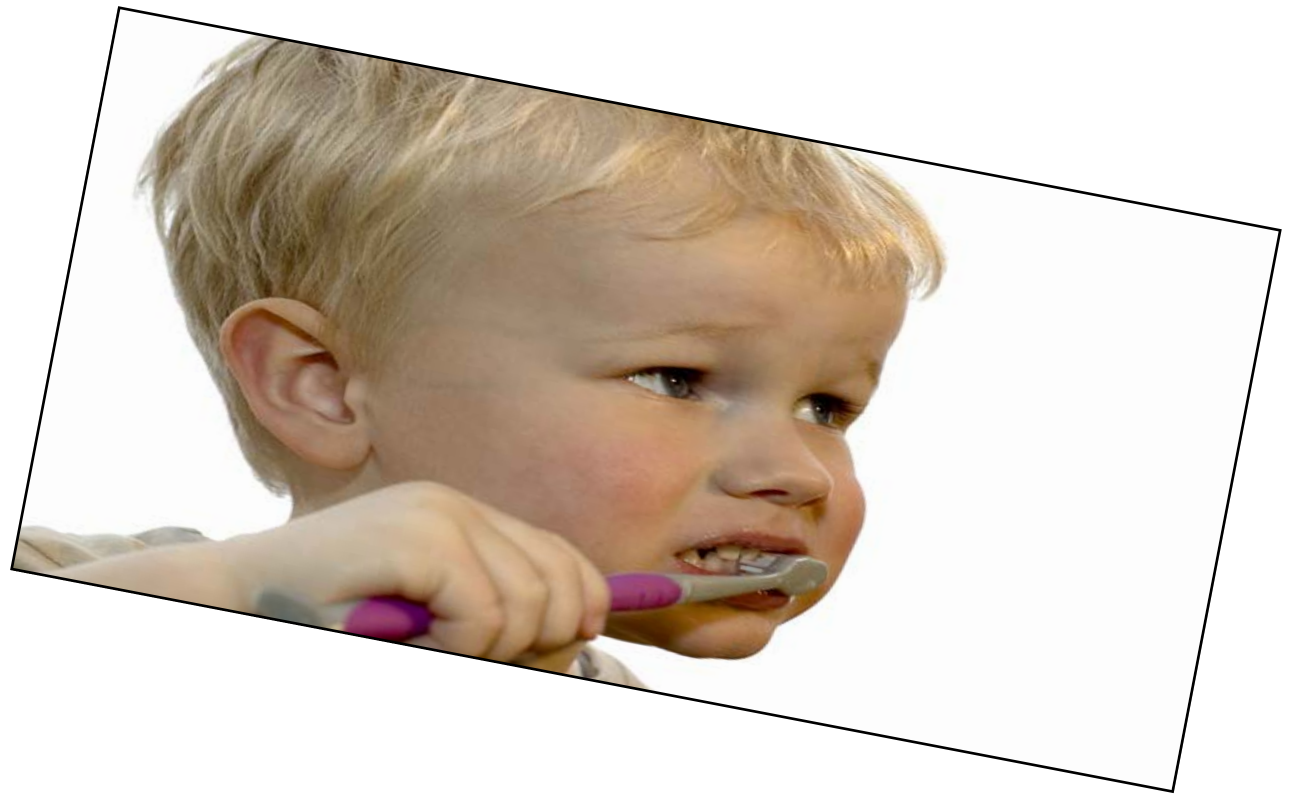
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

What types of **transformations** can we do?

$I(X, Y)$

Warping

$I'(X, Y)$

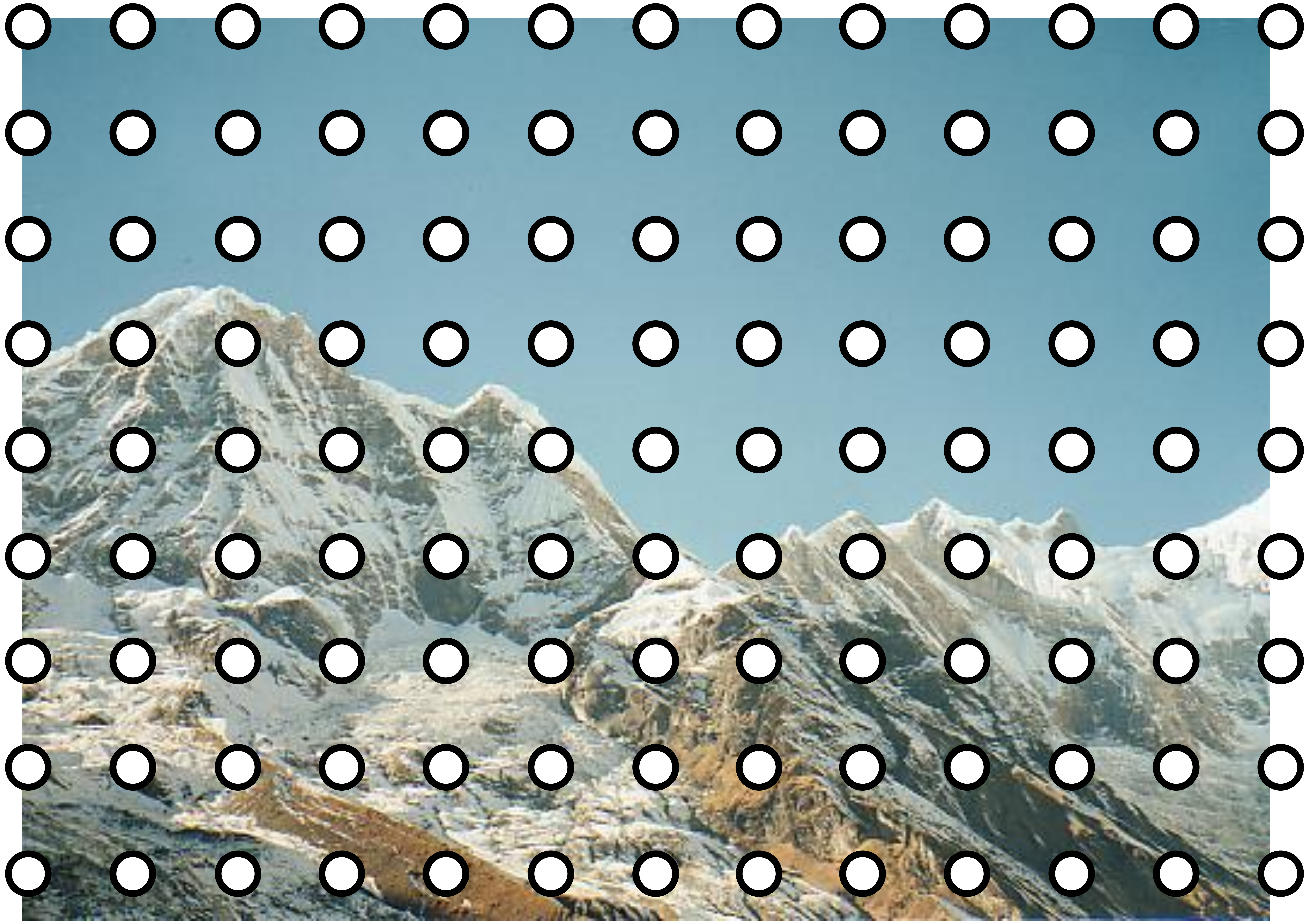


changes domain of image function

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Resampling Images

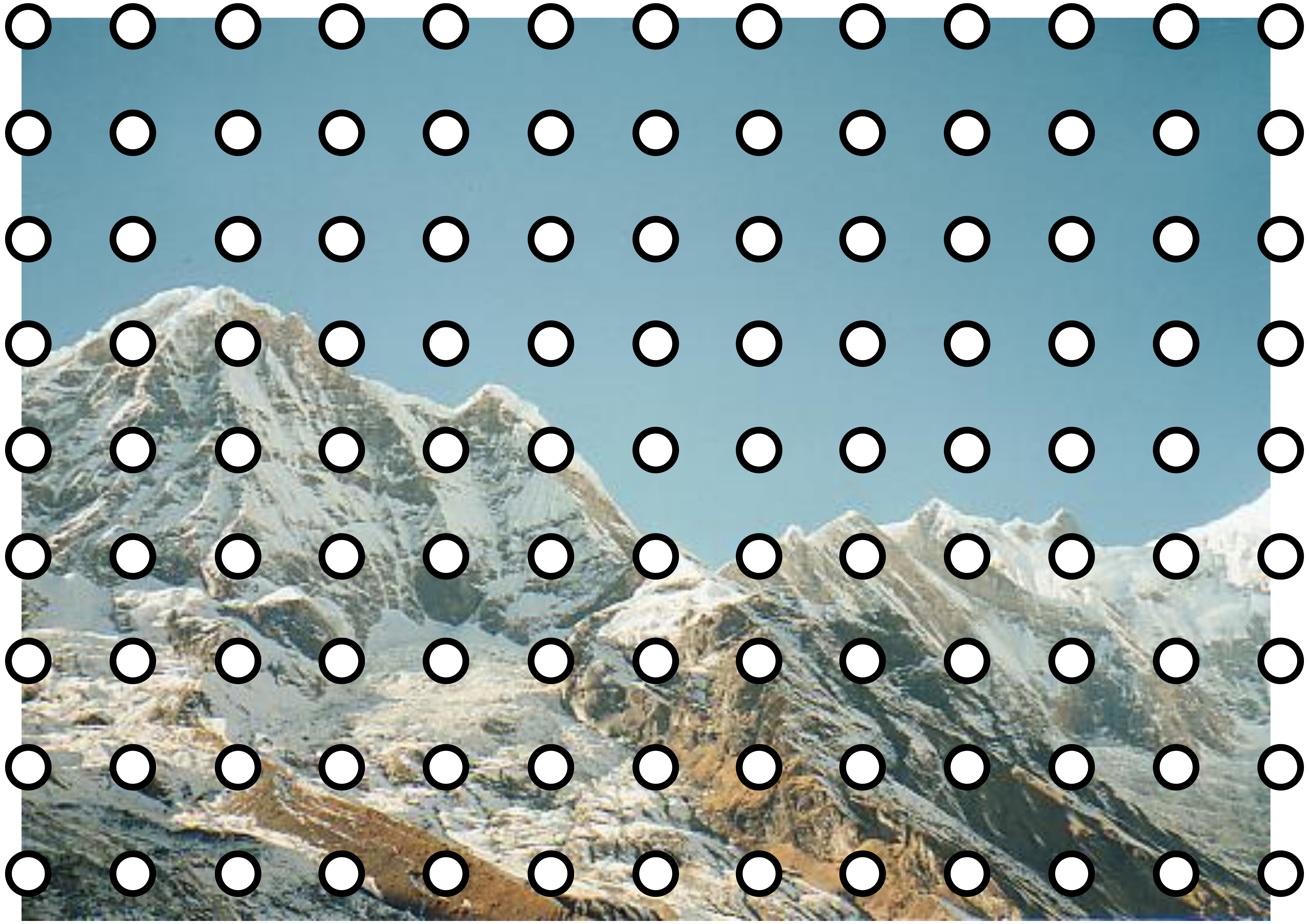
Goal: Resample the image to get a lower resolution counterpart



What is the simplest way to do this (e.g., produce image 1/5 of original size)?

Resampling Images

Goal: Resample the image to get a lower resolution counterpart



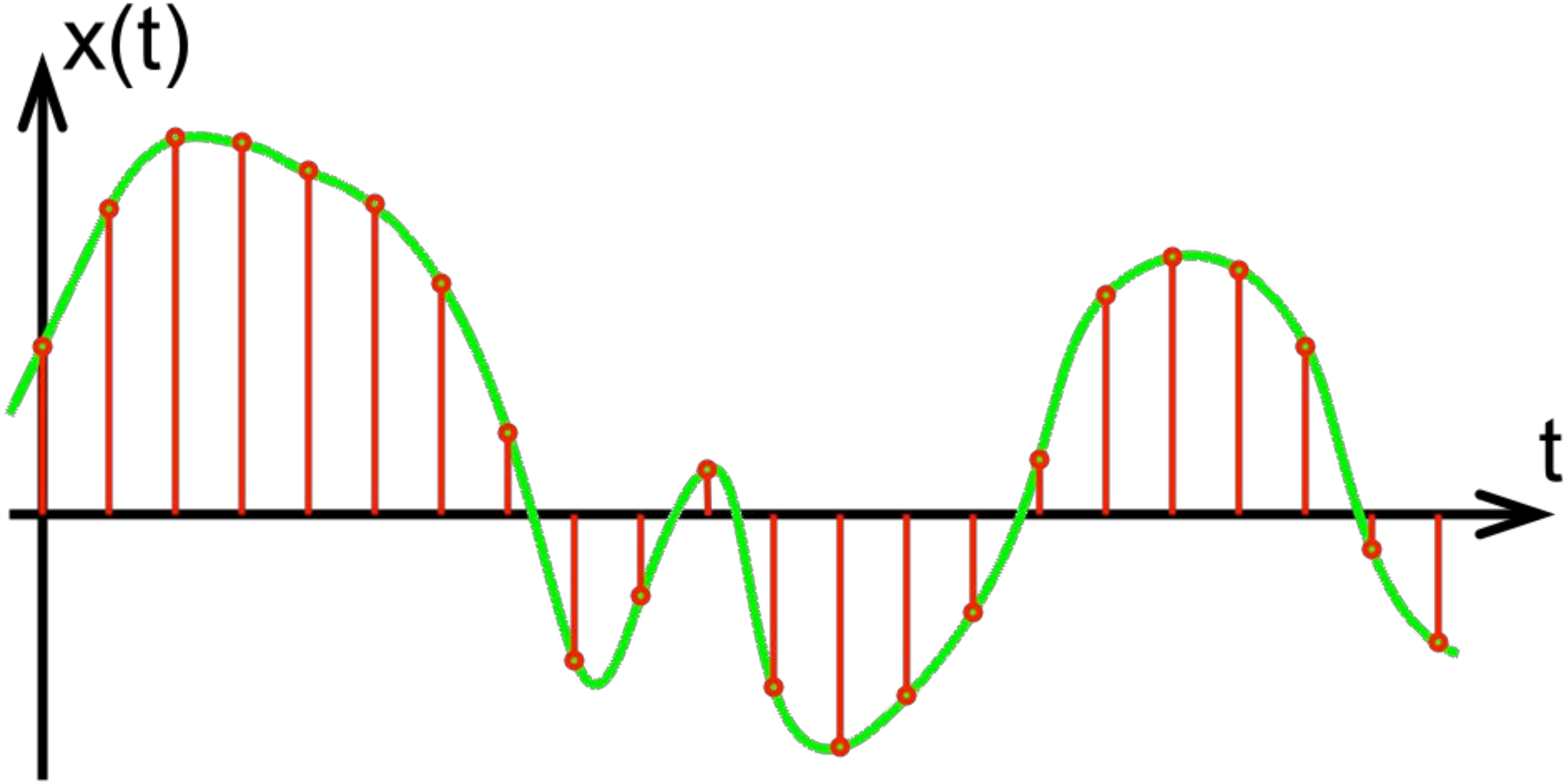
Naive Method: Form new image by taking every n-th pixel of the original image

Resampling Images

Sampling every 5-th pixel, while shifting rightwards one pixel at a time

What's wrong with this method?

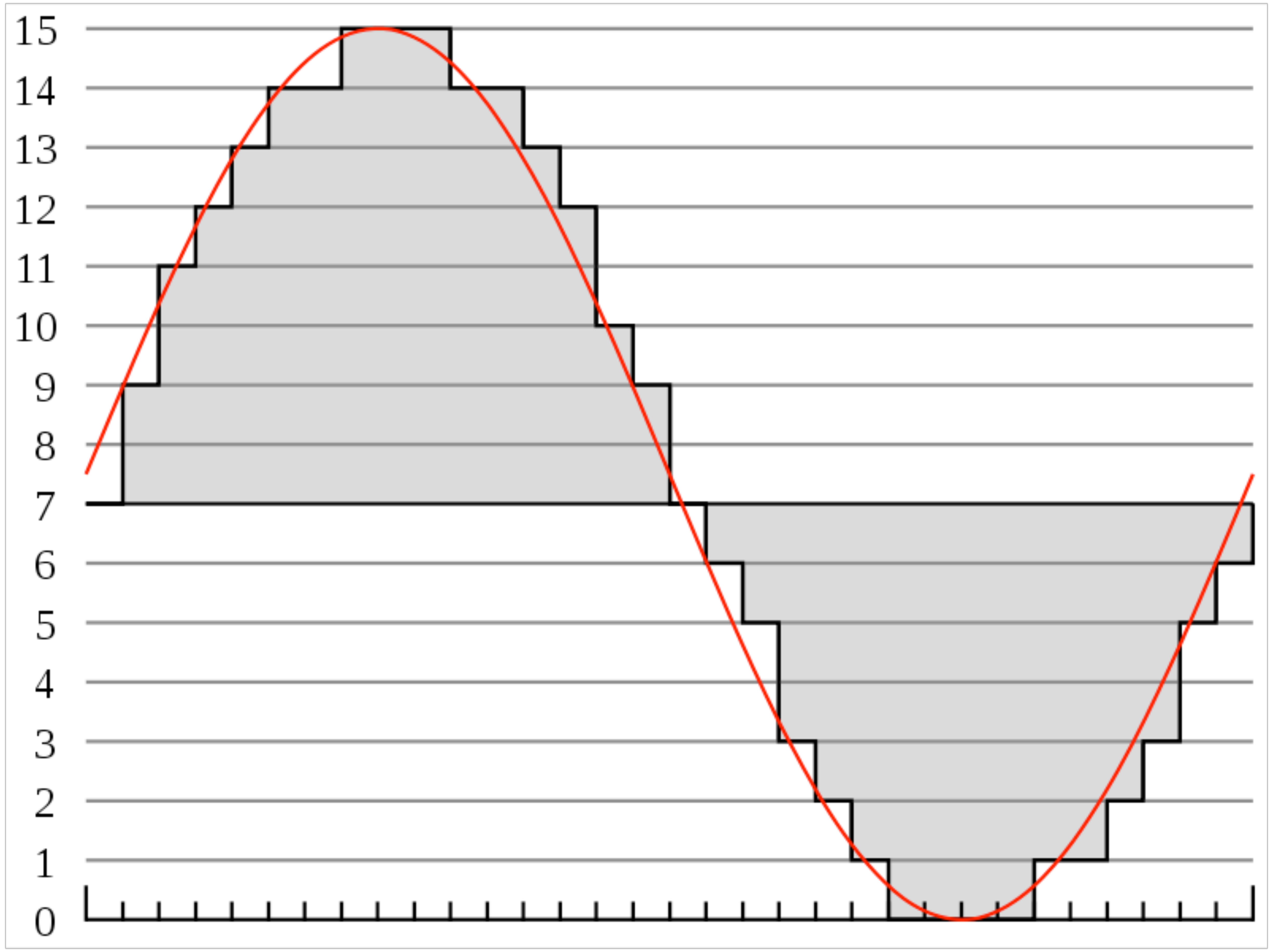
Example: Audio Sampling



Question: What choice/parameters do we have when sampling audio signal?

Sampling rate and bit depth, e.g., 44.1 kHz (samples/second), 16 bits/sample

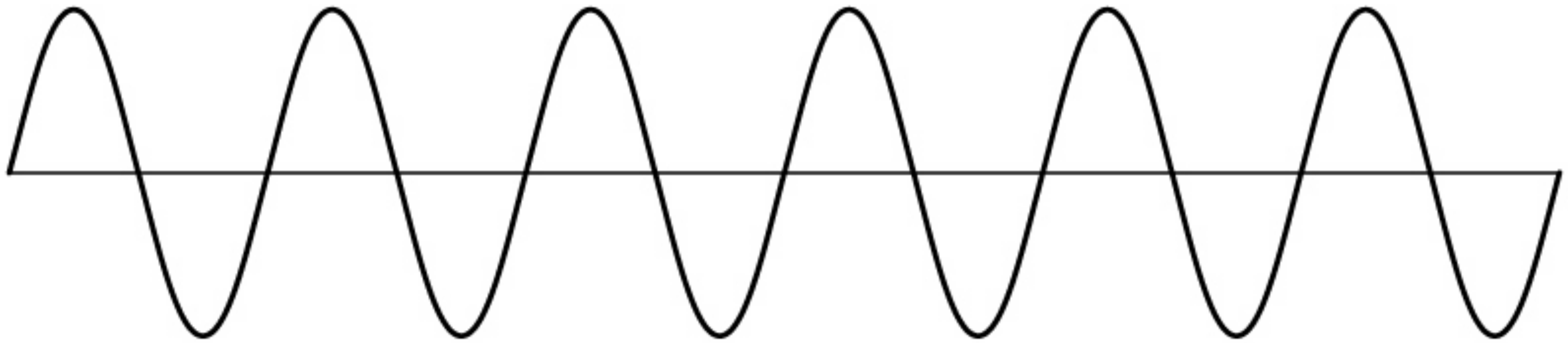
Example: Audio Sampling



Quantization noise / error is the difference between black and red curves

Example: A Simple Sine Wave

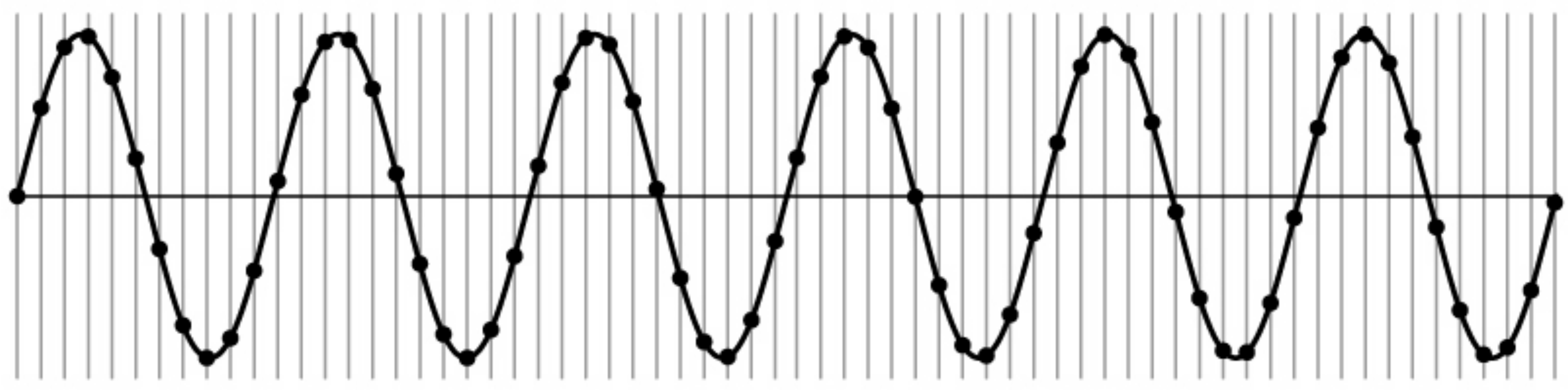
How do we discretize the signal?



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Example: A Simple Sine Wave

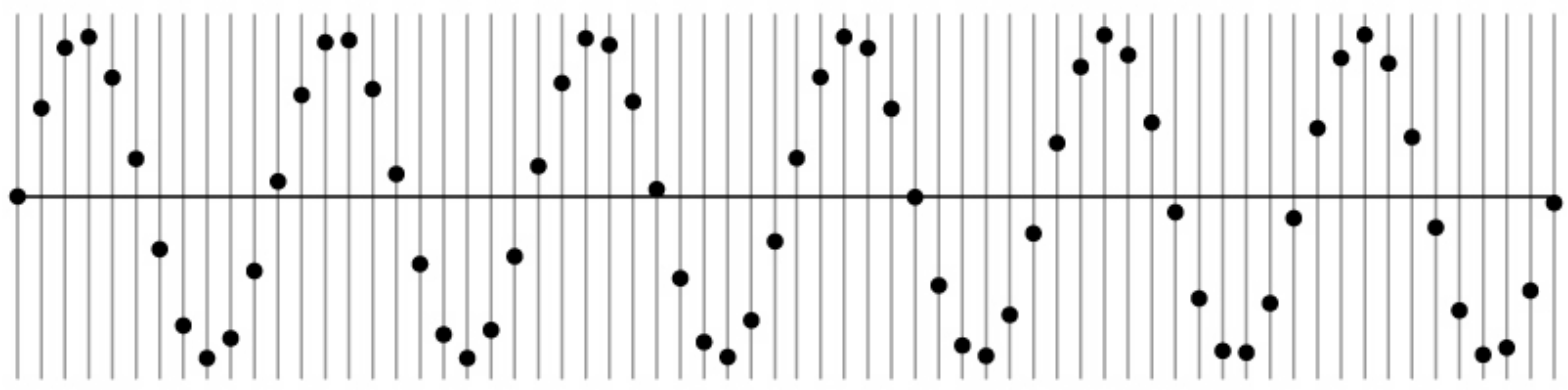
How do we discretize the signal?



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Example: A Simple Sine Wave

How do we discretize the signal?



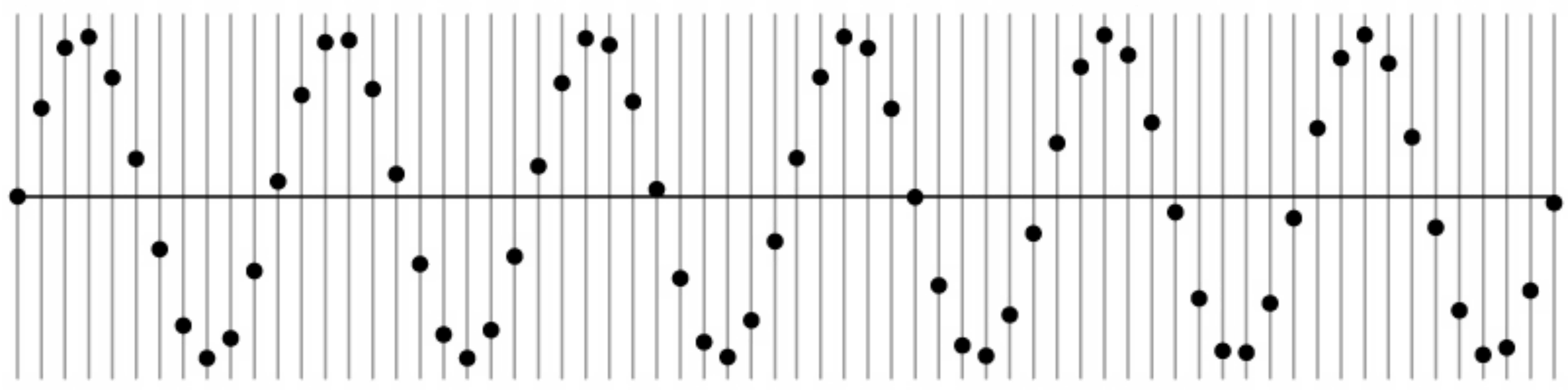
How many samples should I take?

Can I take as many samples as I want?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Example: A Simple Sine Wave

How do we discretize the signal?



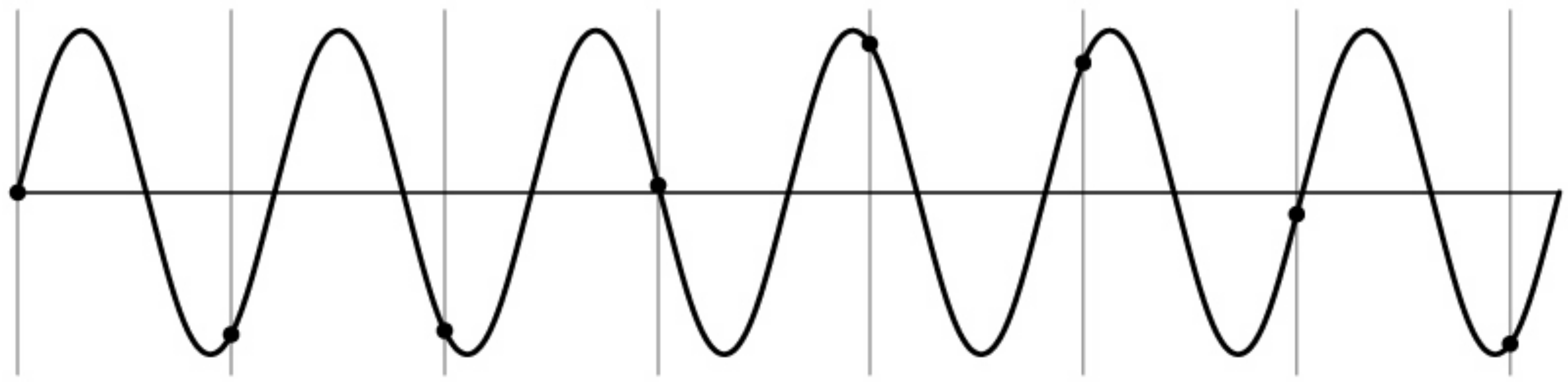
How many samples should I take?

Can I take as few samples as I want?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Example: A Simple Sine Wave

How do we discretize the signal?

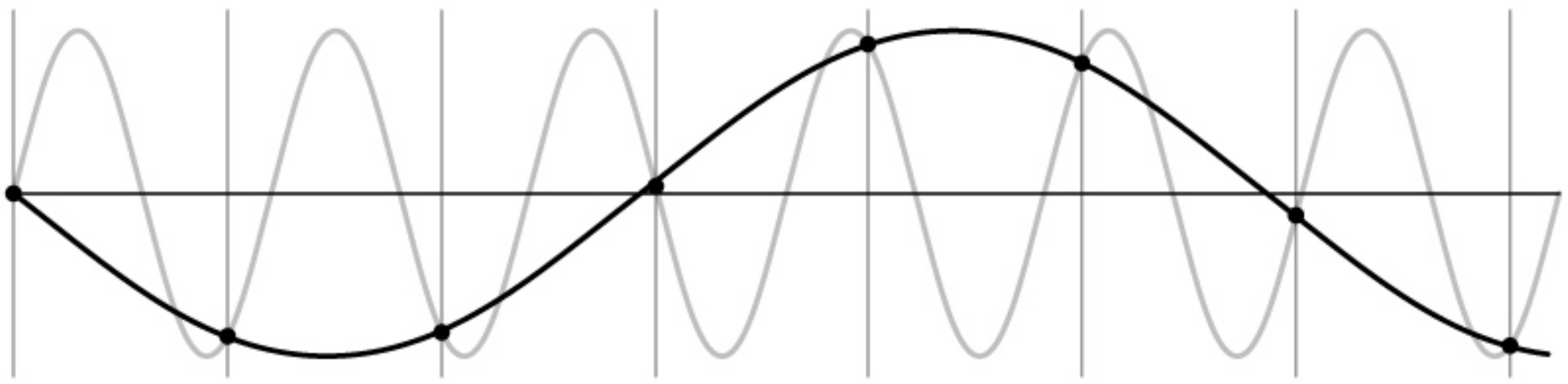


Signal can be confused with one at lower frequency

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Example: A Simple Sine Wave

How do we discretize the signal?



Signal can be confused with one at lower frequency
— This is called “Aliasing”

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Audio **Aliasing**

- Aliasing causes undesirable artifacts in audio reproduction
- e.g., if we take an audio signal and simply drop every second sample, the highest frequencies will be aliased… we hear robotic sounding distortion

```
import scipy.io.wavfile as wavfile

rate, signal = wavfile.read("stevie.wav")

data=signal[0:(rate*10),:] # 10 seconds of audio

data_2=data[0:-1:2,:] # select every 2nd sample
data_4=data[0:-1:4,:] # select every 4th sample
data_8=data[0:-1:8,:] # select every 8th sample

wavfile.write('test2.wav', int(rate/2), data_2)
wavfile.write('test4.wav', int(rate/4), data_4)
wavfile.write('test8.wav', int(rate/8), data_8)
```

Original

$\downarrow 2$ $\downarrow 4$ $\downarrow 8$

Audio **Aliasing**

- We can reduce the aliasing artifacts by **pre-filtering** with a low pass filter
- e.g., if we apply smoothing with a Gaussian filter standard deviation 2.0 for each octave (factor 2) of downsampling we get a better result:

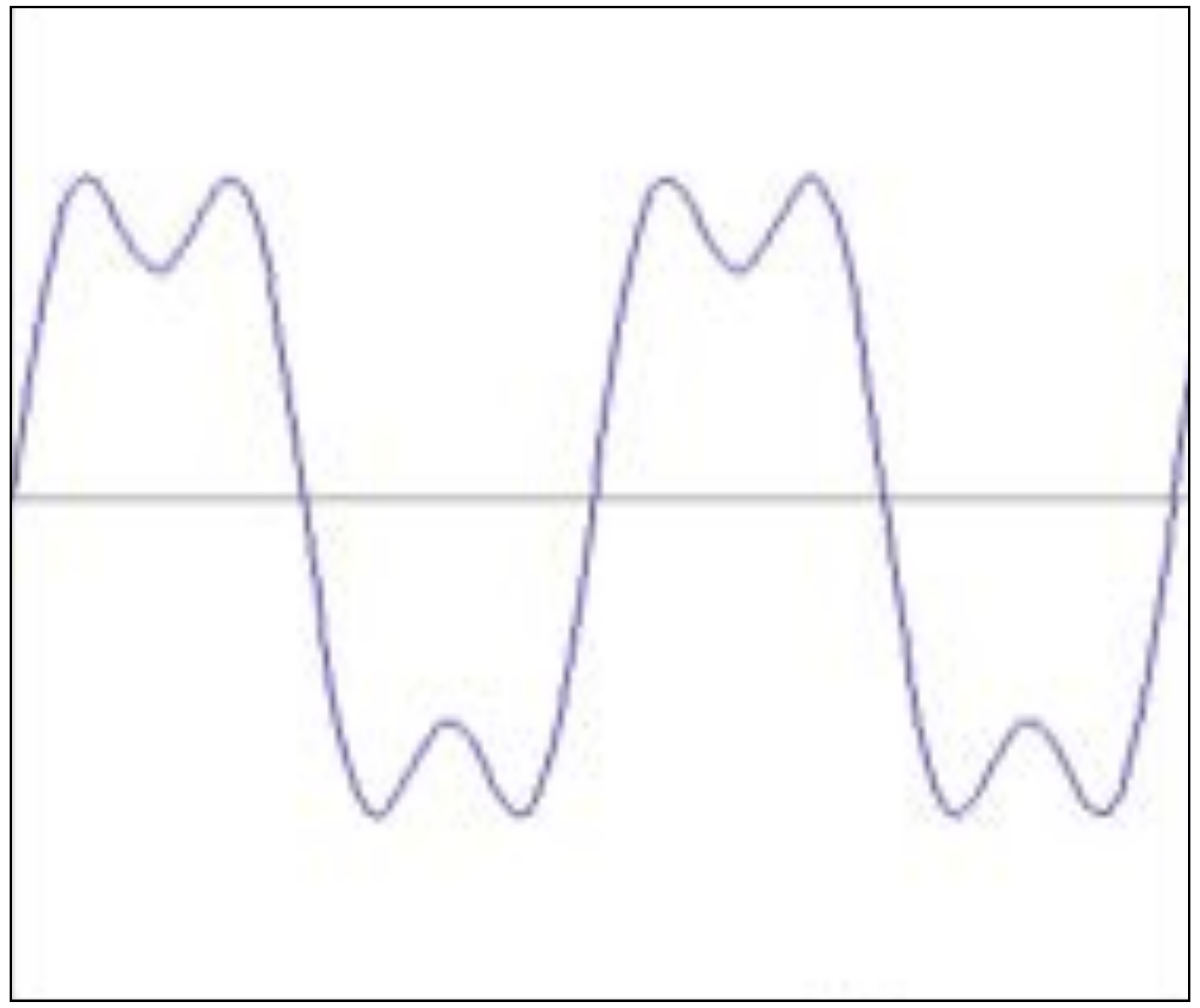
$\downarrow 8$

$\downarrow 8$ with pre-filtering

- Note we have still lost some of the high frequency content, but the crunchy sounding distortion due to aliasing has now gone

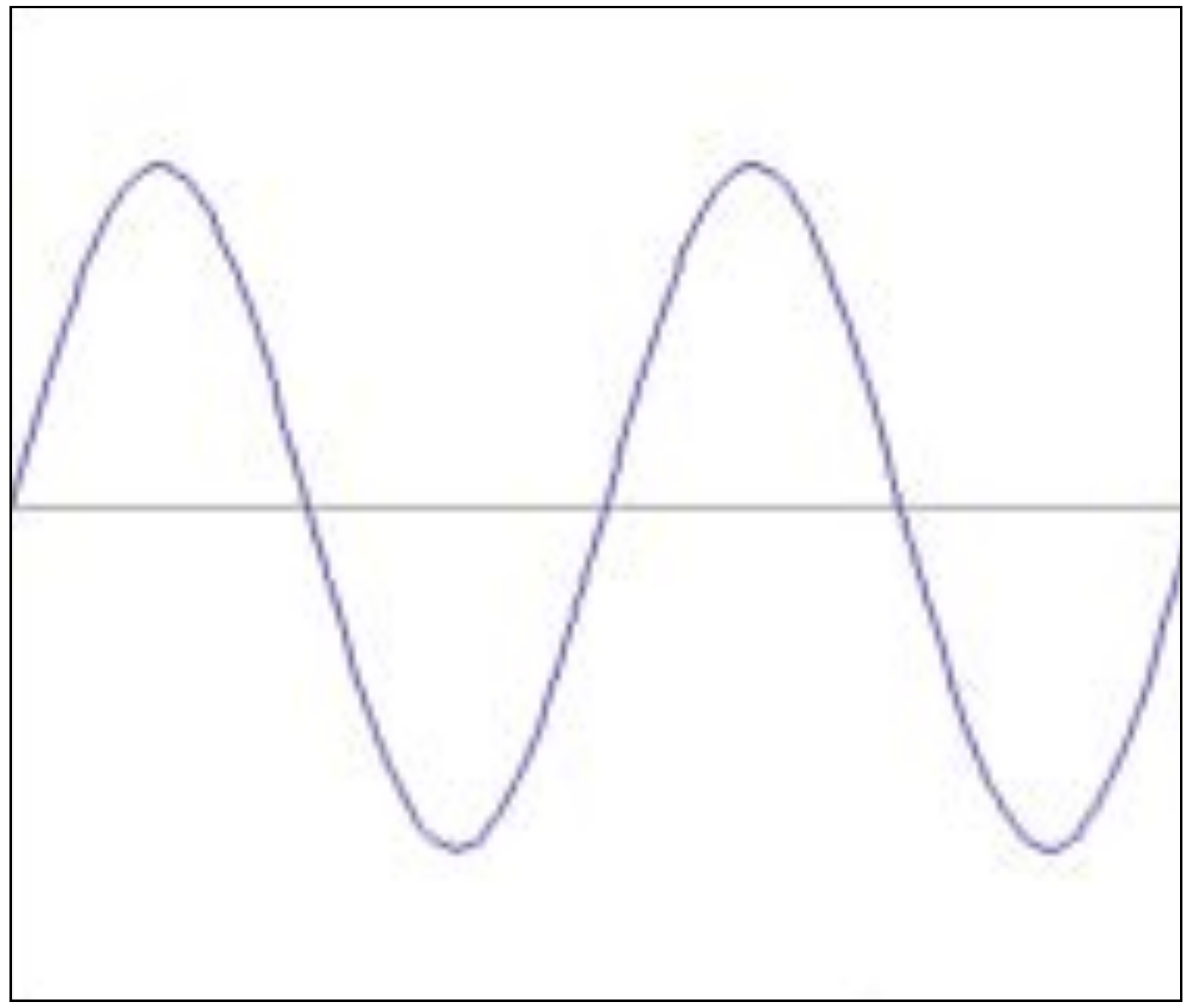
Recall: **Fourier** Representation

Any signal can be written as a sum of sinusoidal functions



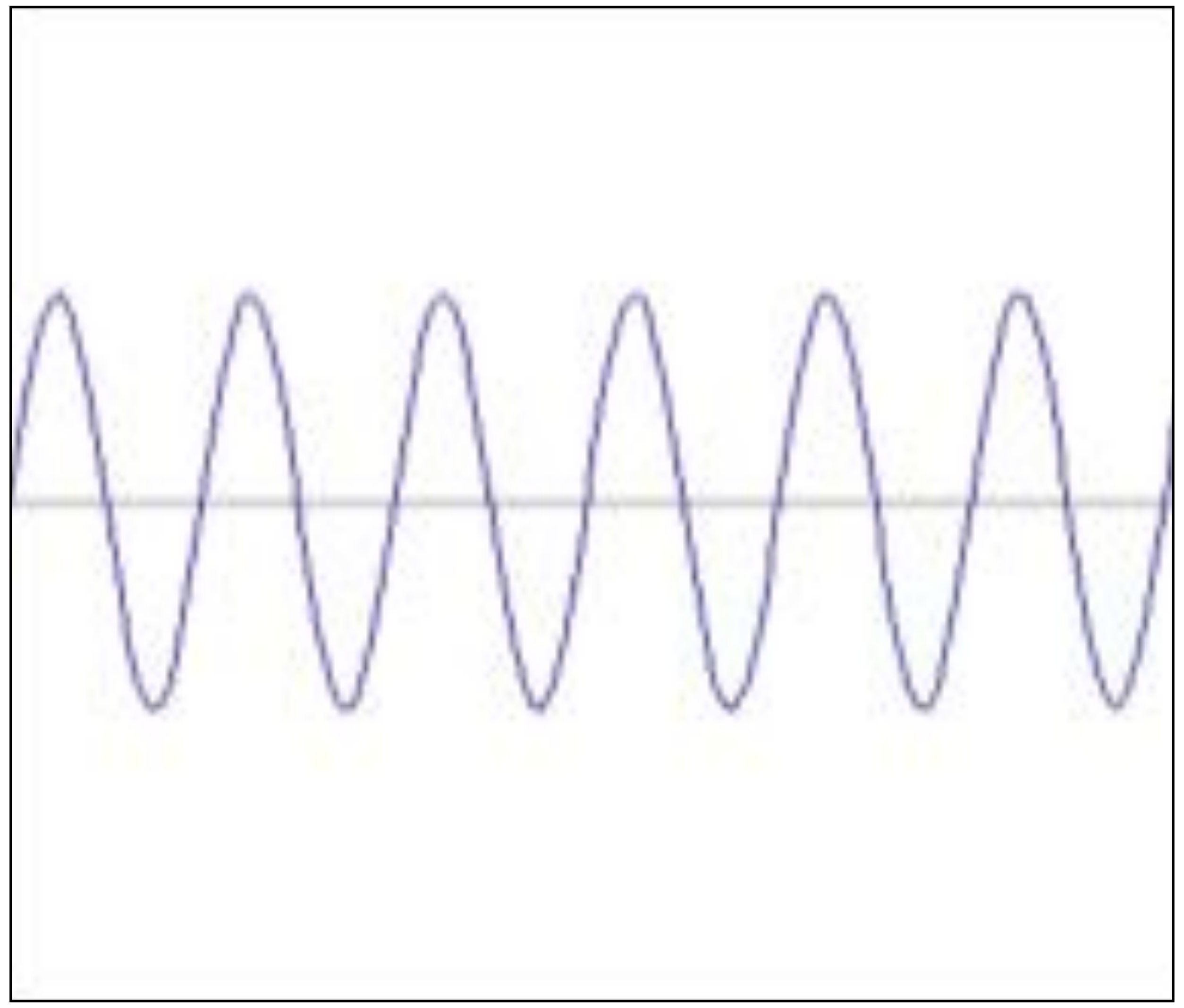
$$f(x) = \sin(2\pi x) + \frac{1}{3}\sin(2\pi 3x)$$

=



$$\sin(2\pi x)$$

+



$$\frac{1}{3}\sin(2\pi 3x)$$

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Nyquist Sampling Theorem

To avoid aliasing a signal must be sampled at twice the maximum frequency:

$$f_s > 2 \times f_{max}$$

where f_s is the sampling frequency, and f_{max} is the maximum frequency present in the signal

Futhermore, Nyquist's theorem states that a signal is **exactly recoverable** from its **samples** if sampled at the **Nyquist rate** (or higher)

Note: that a signal must be **bandlimited** for this to apply (i.e., it has a maximum frequency)

6.1

Exact **Reconstruction** from **Samples**

Question: When is $I(X, Y)$ an exact characterization of $i(x, y)$?

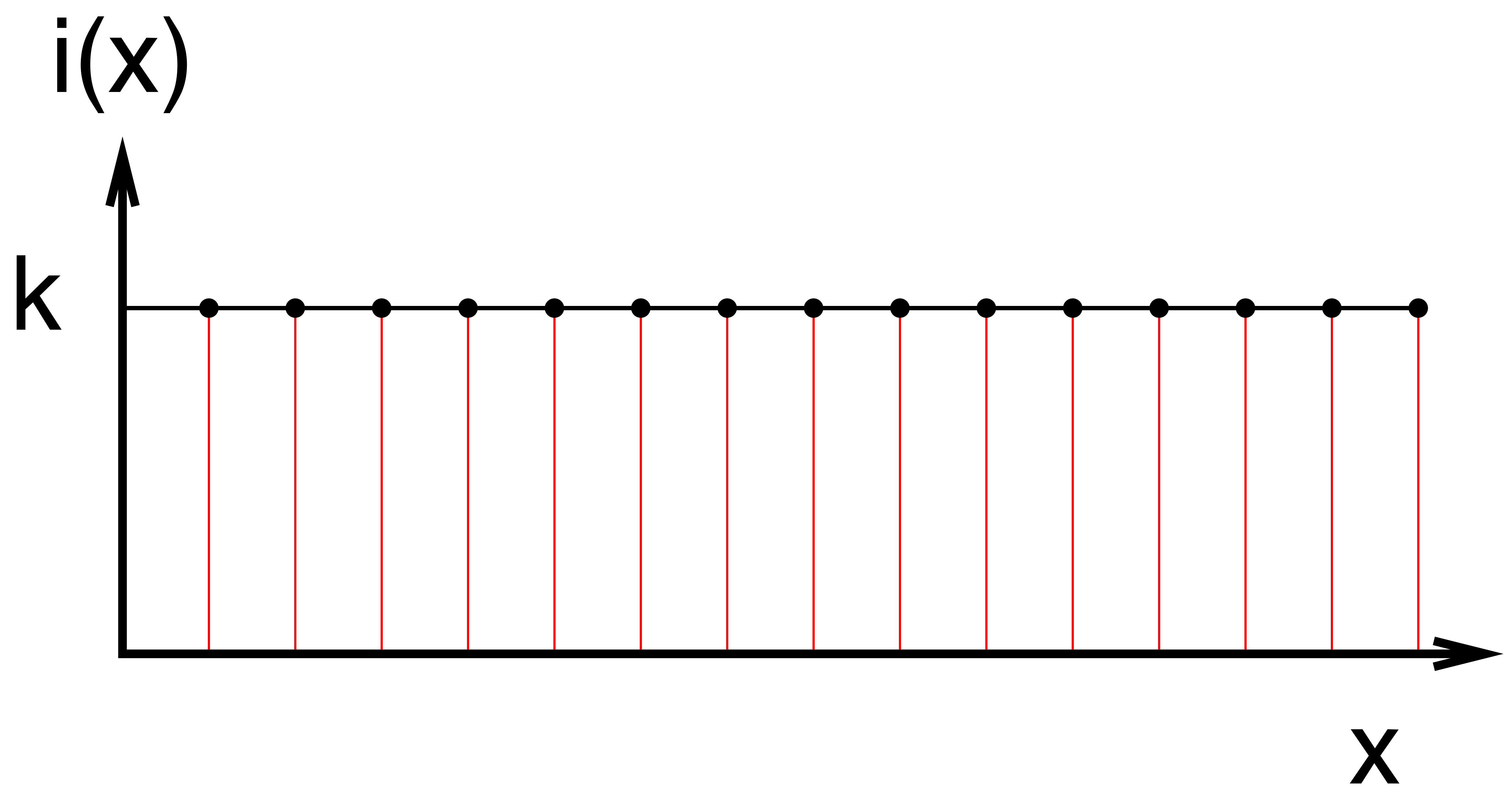
Question (modified): When can we reconstruct $i(x, y)$ exactly from $I(X, Y)$?

Intuition: Reconstruction involves some kind of **interpolation**

Heuristic: When in doubt, consider simple cases

Sampling Theory (informal)

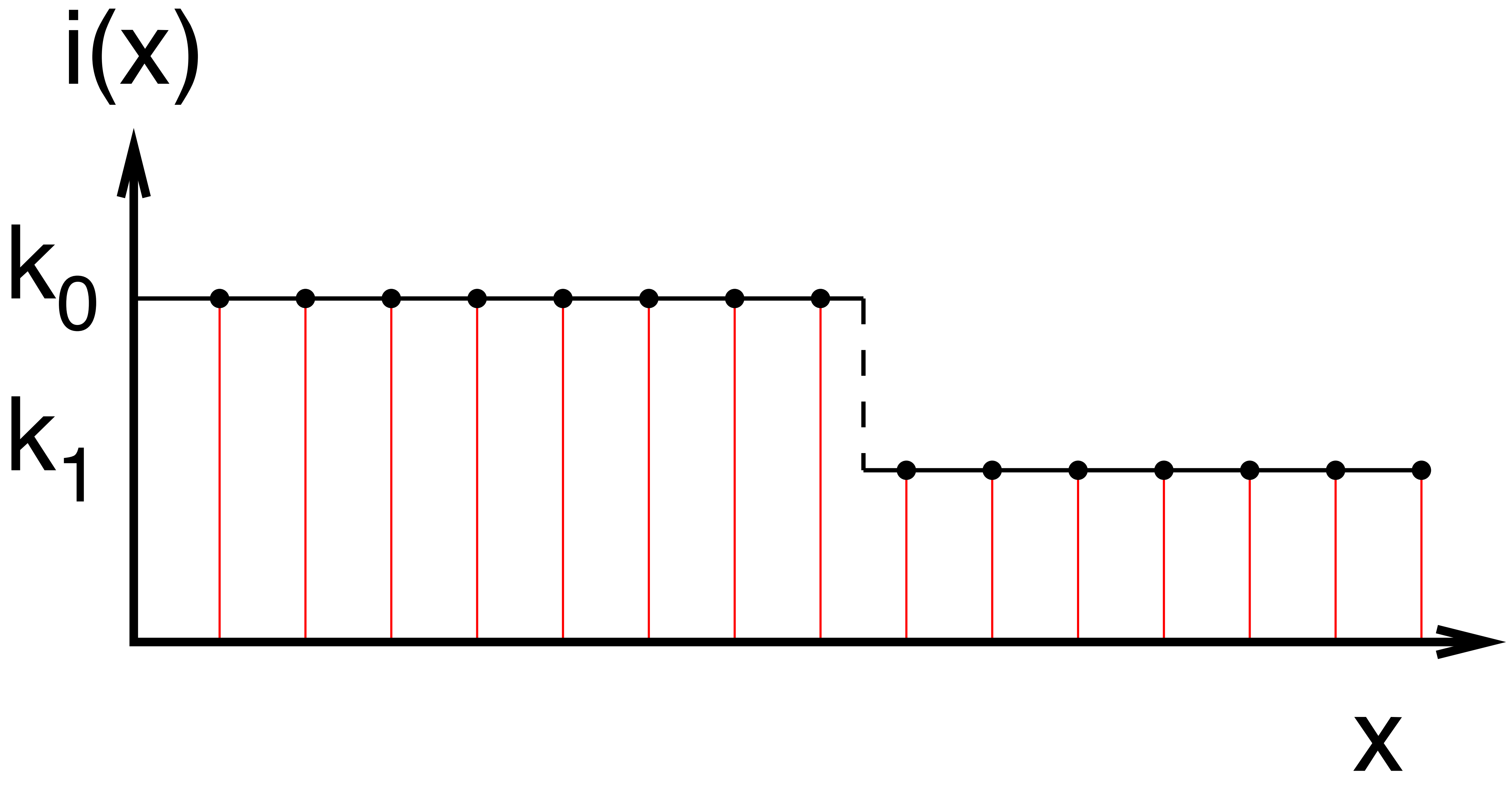
Case 0: Suppose $i(x, y) = k$ (with k being one of our gray levels)



$I(X, Y) = k$. Any standard interpolation function would give $i(x, y) = k$ for non-integer x and y (irrespective oh how coarse the sampling is)

Sampling Theory (informal)

Case 0: Suppose $i(x, y)$ has a discontinuity not falling precisely at integer x, y

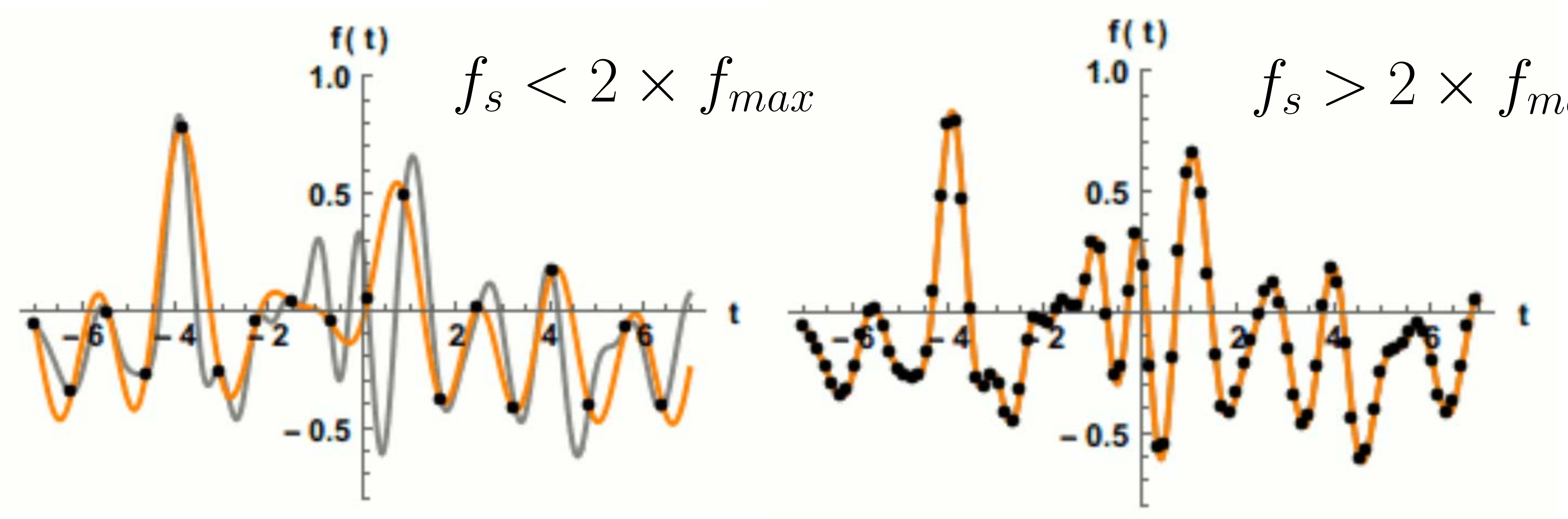


We cannot reconstruct $i(x, y)$ exactly because we can never know exactly where the discontinuity lies

Reconstruction with **Bandlimited** Signal

It can be shown that a bandlimited and correctly sampled signal can be reconstructed exactly via interpolation with a **sinc** function (sin(x)/x)

(This is the Fourier Transform pair of a box filter, which in frequency domain is a pure low-pass filter)



https://en.wikipedia.org/wiki/Whittaker%E2%80%93Shannon_interpolation_formula

Sampling Theory (informal)

Exact reconstruction requires constraint on the rate at which i(x,y) can change between samples

— "rate of change" means derivative

— the formal concept is **bandlimited signal**

— "bandlimit" and "constraint on derivative" are linked

Think of music

— bandlimited if it has some maximum **temporal frequency**

— the upper limit of human hearing is about 20 kHz

Think of imaging systems. Resolving power is measured in

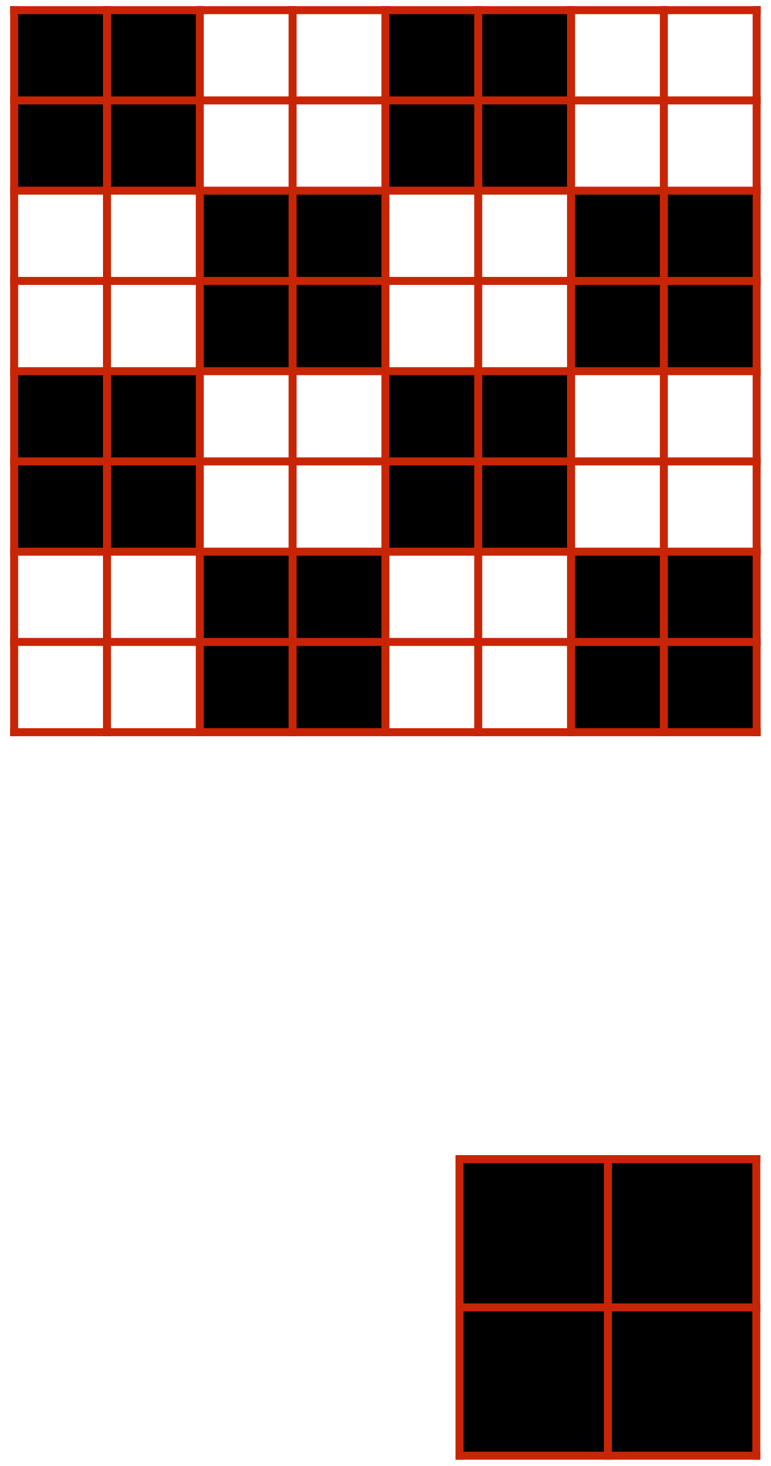
— "line pairs per mm" (for a bar test pattern)

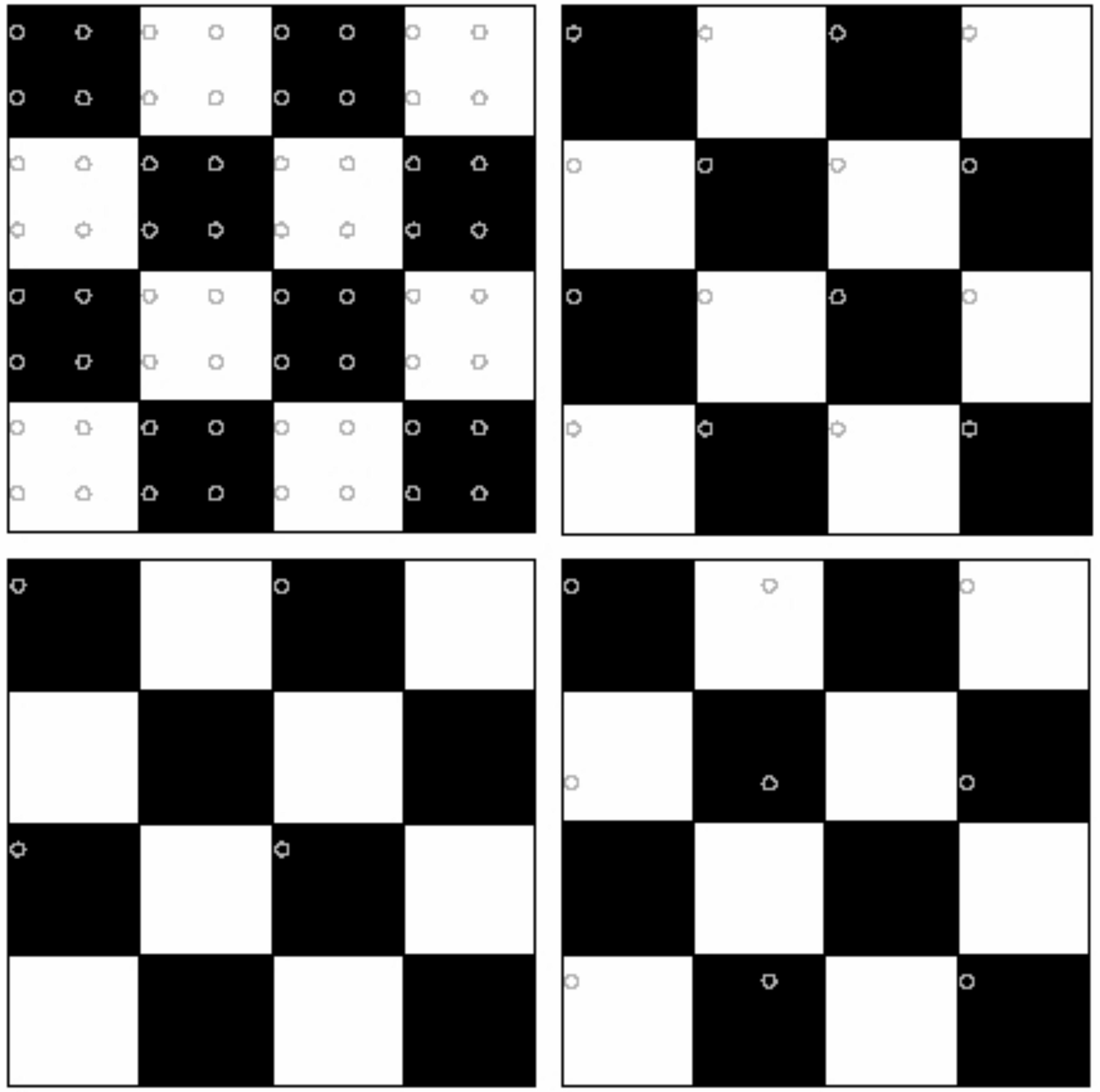
— "cycles per mm" (for a sine wave test pattern)

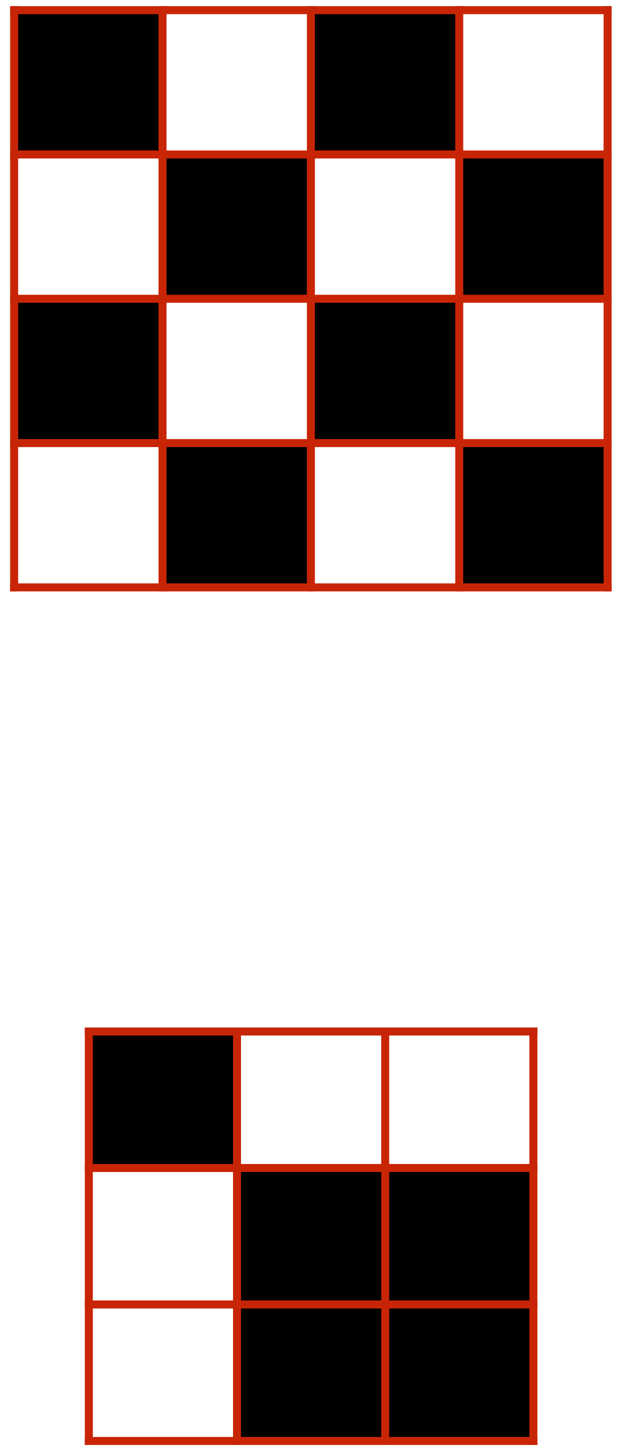
An image is bandlimited if it has some maximum **spatial frequency**

Sampling

It is clear that *some* information may be lost when we work on a discrete pixel grid.





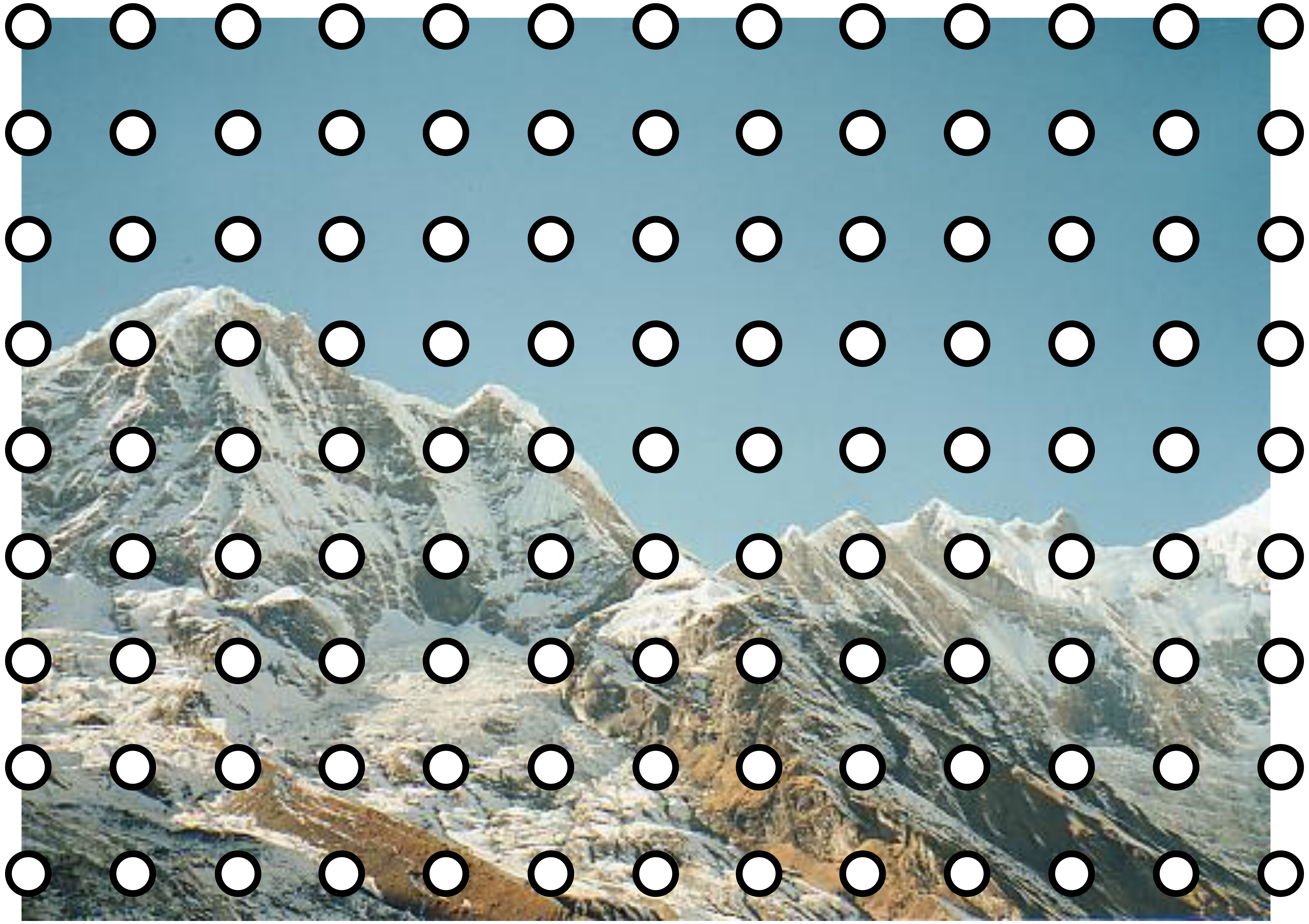


6.2

Forsyth & Ponce (2nd ed.) Figure 4.7

Resampling Images

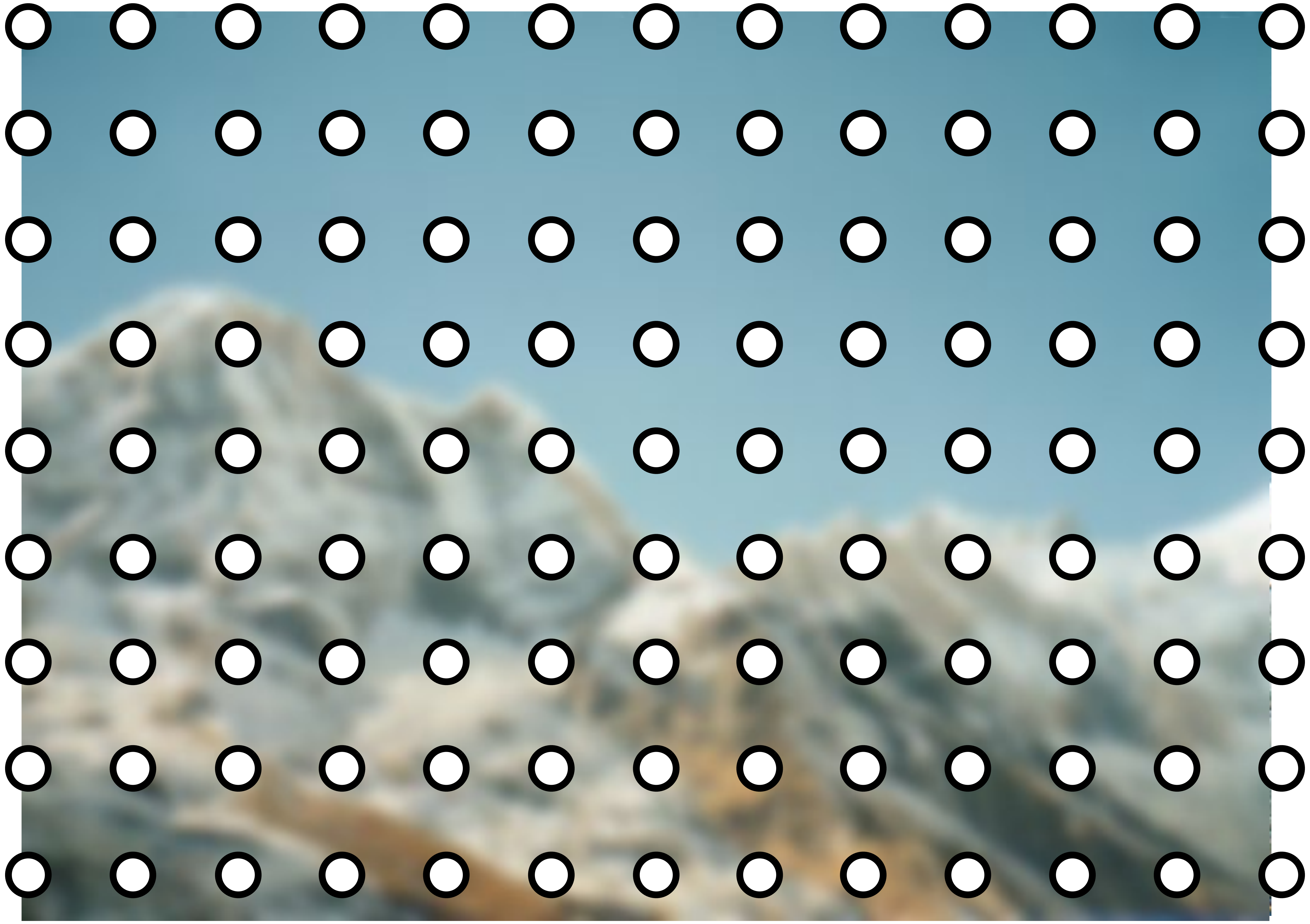
Goal: Resample the image to get a lower resolution counterpart



Naive Method: Form new image by taking every n-th pixel of the original image

Resampling Images

With correct sigma value for a Gaussian, no information is lost



Improved Method: First blur the image (with low-pass) then take n-th pixel

Aliasing Example

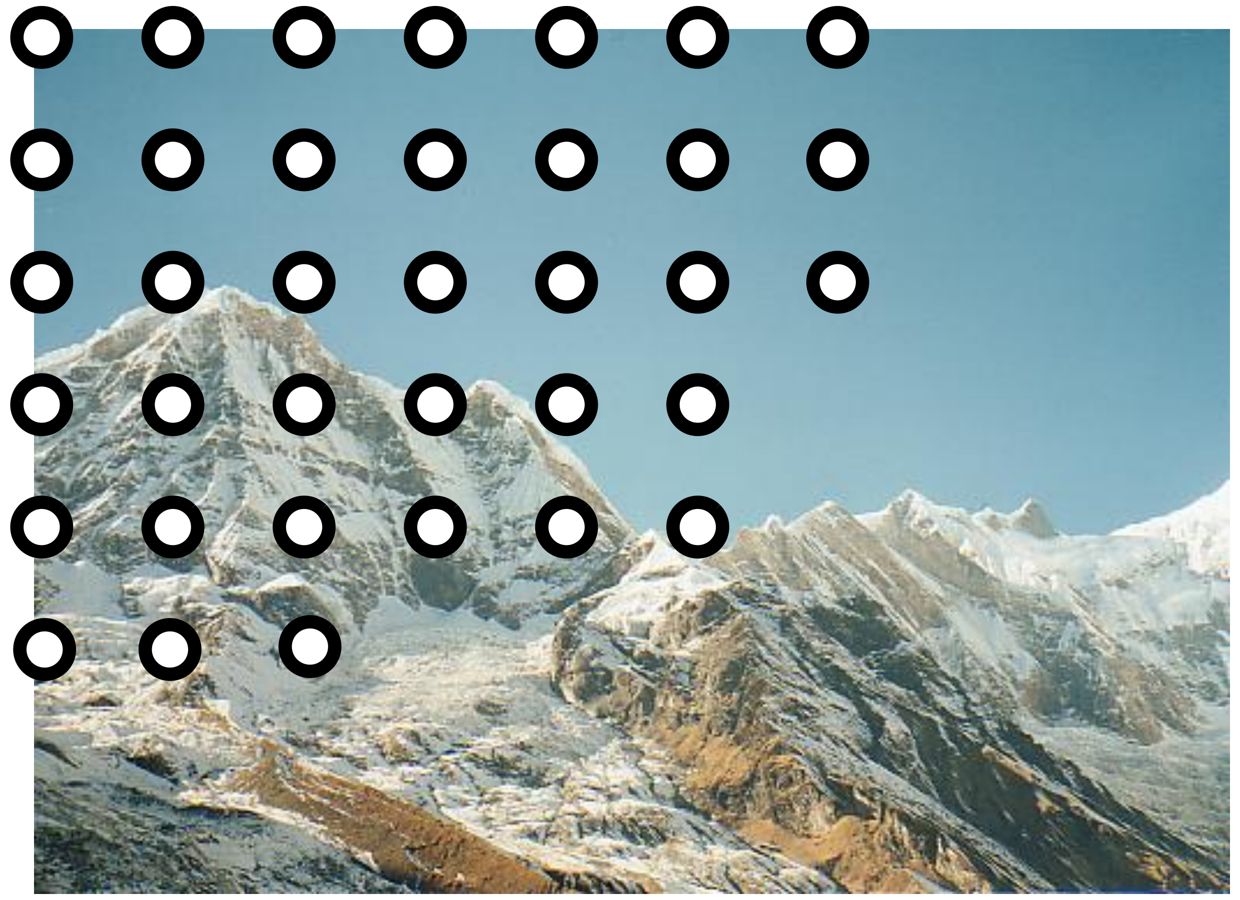
Sampling every 5th pixel with and without low-pass blur

No filtering

Gaussian Blur $\sigma = 3.0$

$\sigma = 1/(2s)$

Resampling Images

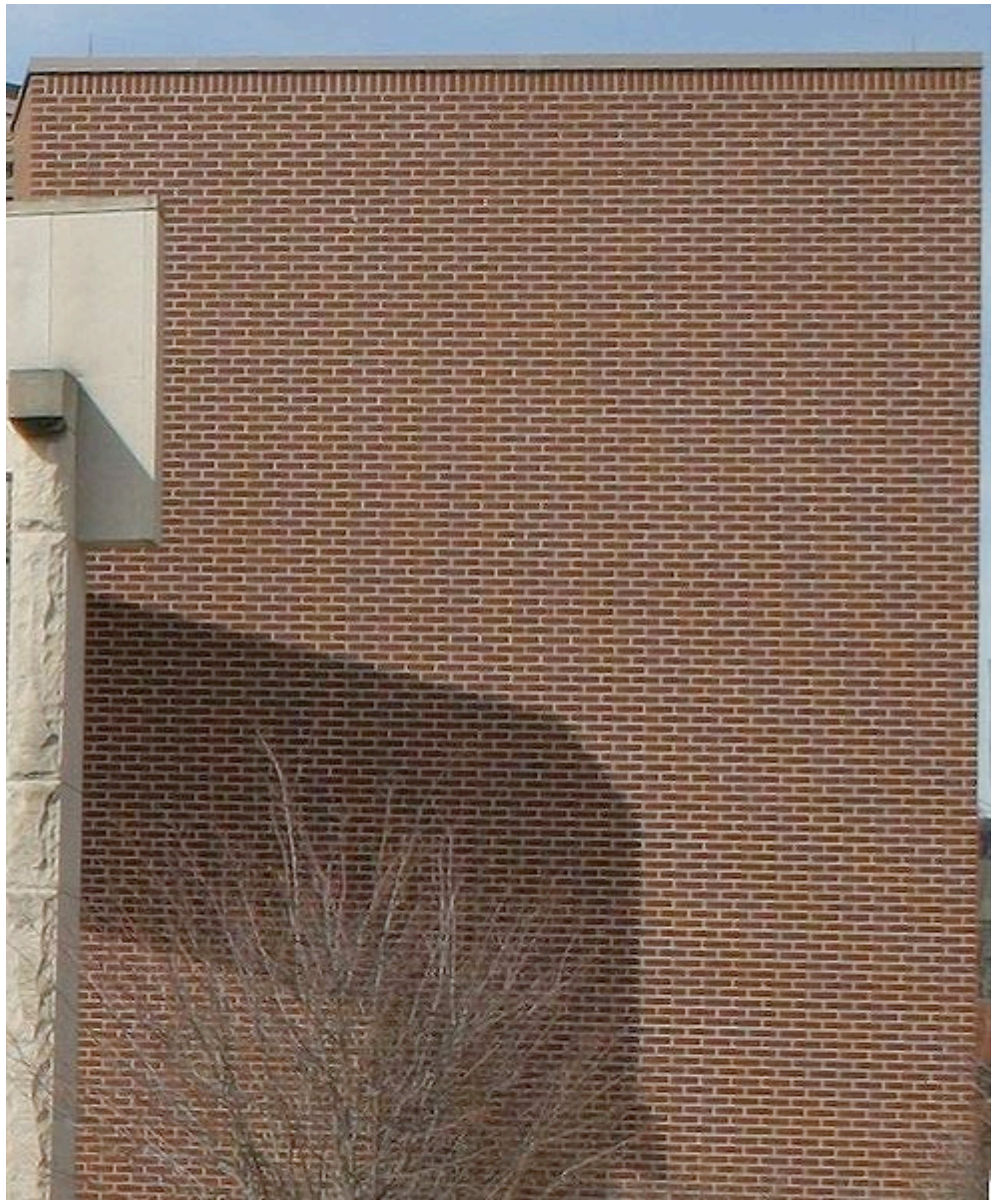


every 10th pixel
(aliased)

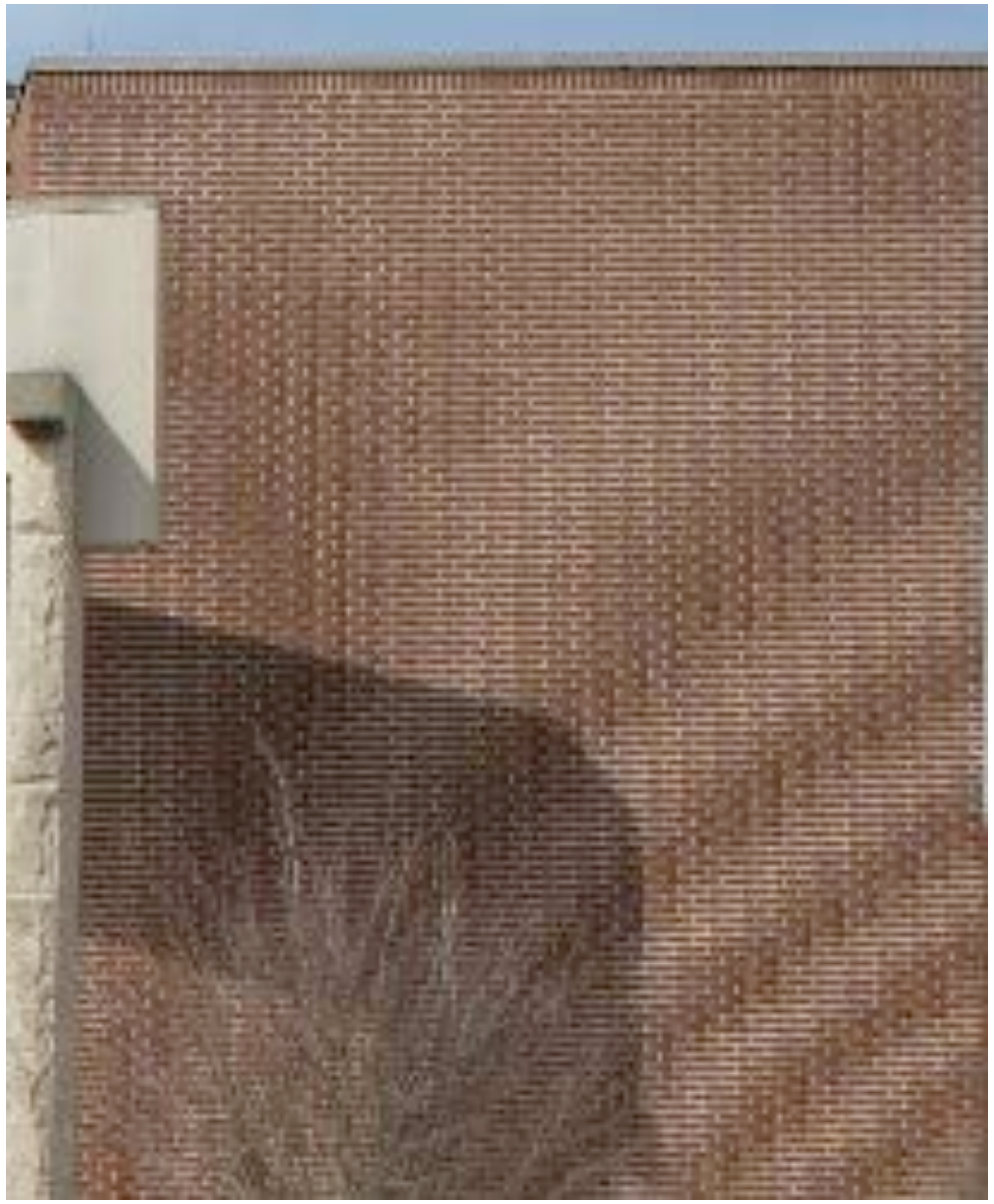
low pass filtered
(correct sampling)

- Note that selecting every 10th pixel ignores the intervening information, whereas the low-pass filter (blur) smoothly combines it
- If we shifted the original image 1 pixel to the right, the aliased image would look completely different, but the low pass filtered image would look almost the same

Image Sampling and Aliasing

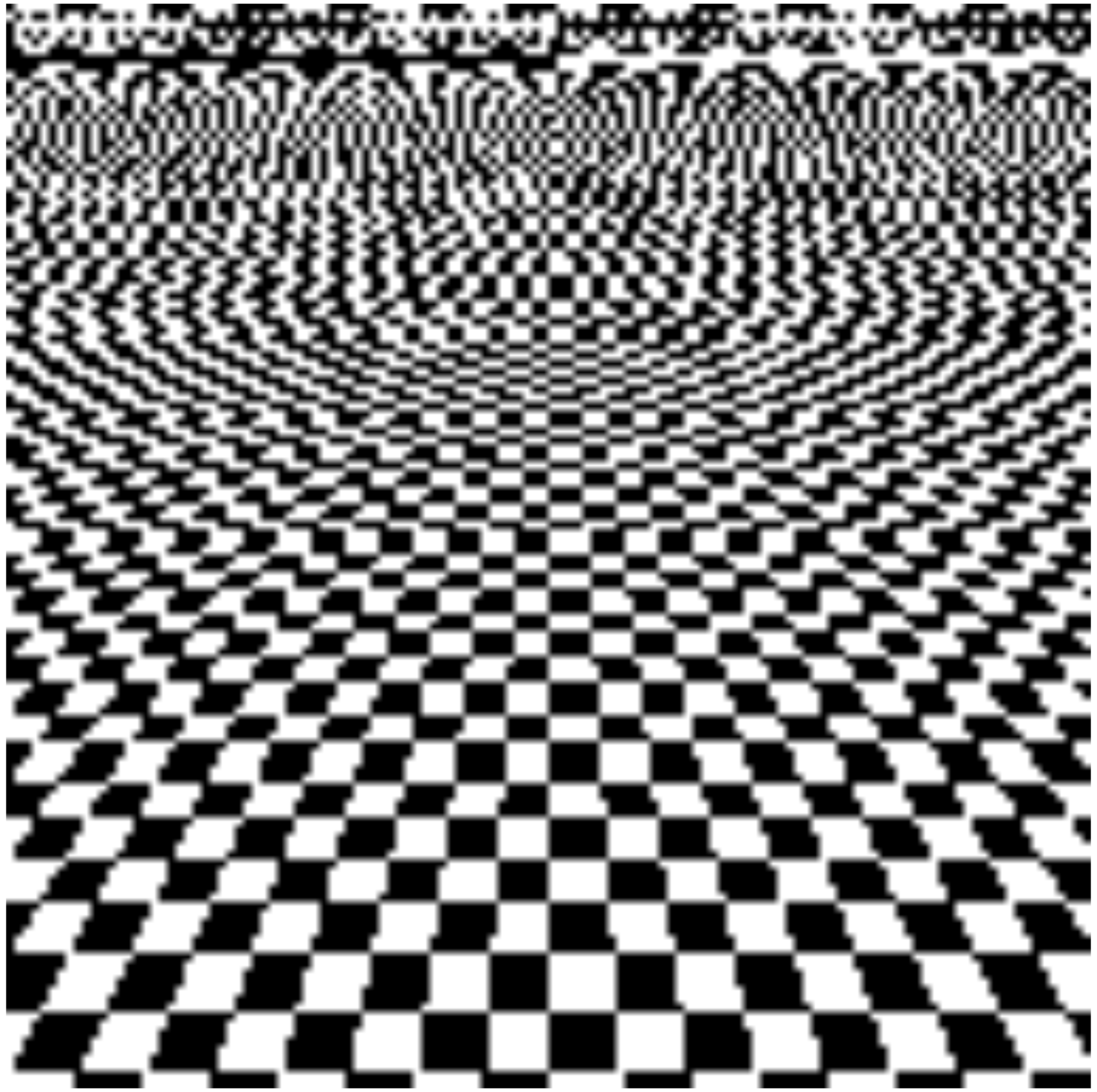


$$f_s > 2 \times f_{max}$$

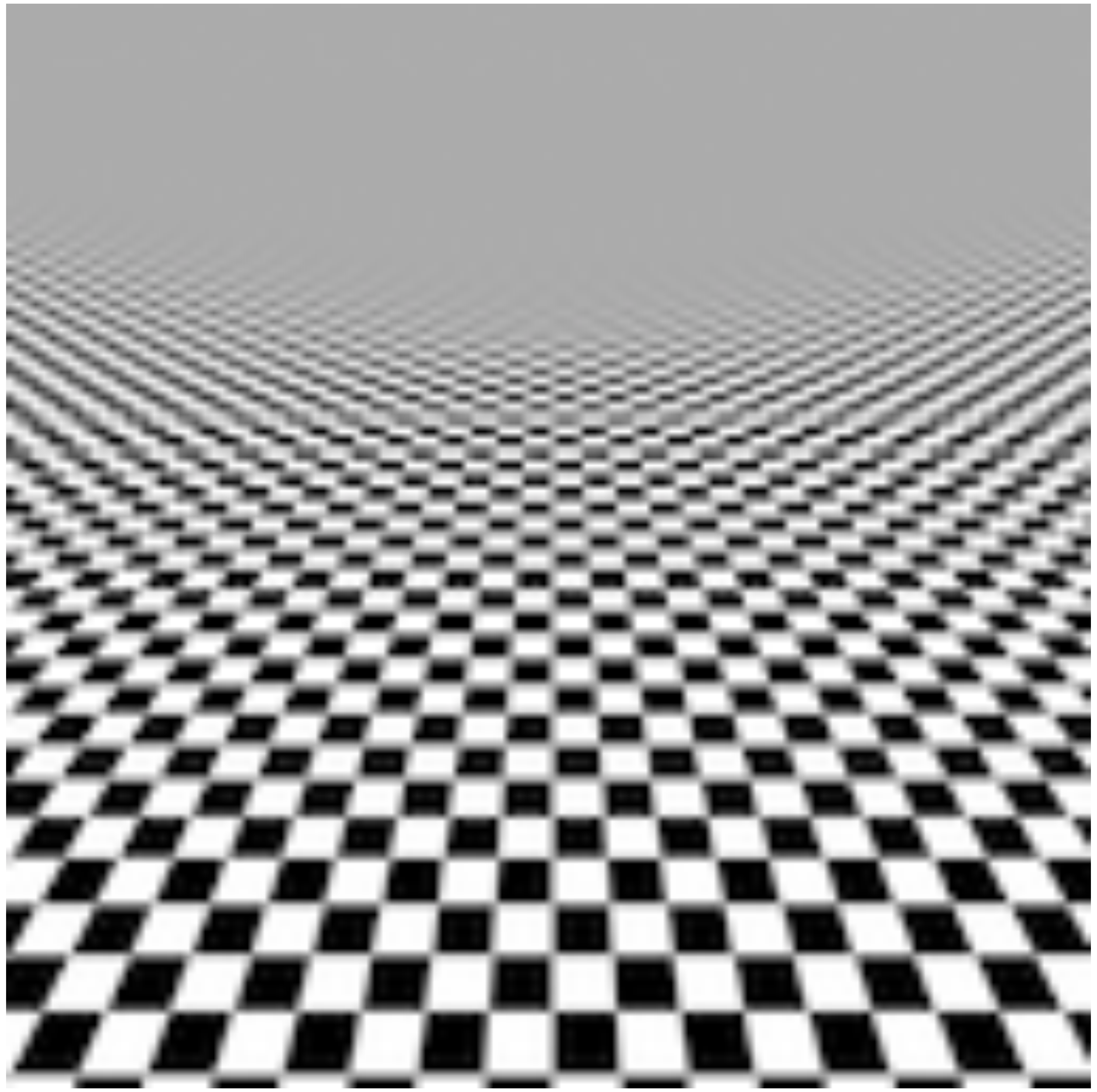


$$f_s < 2 \times f_{max}$$

Another example of aliasing



Aliased



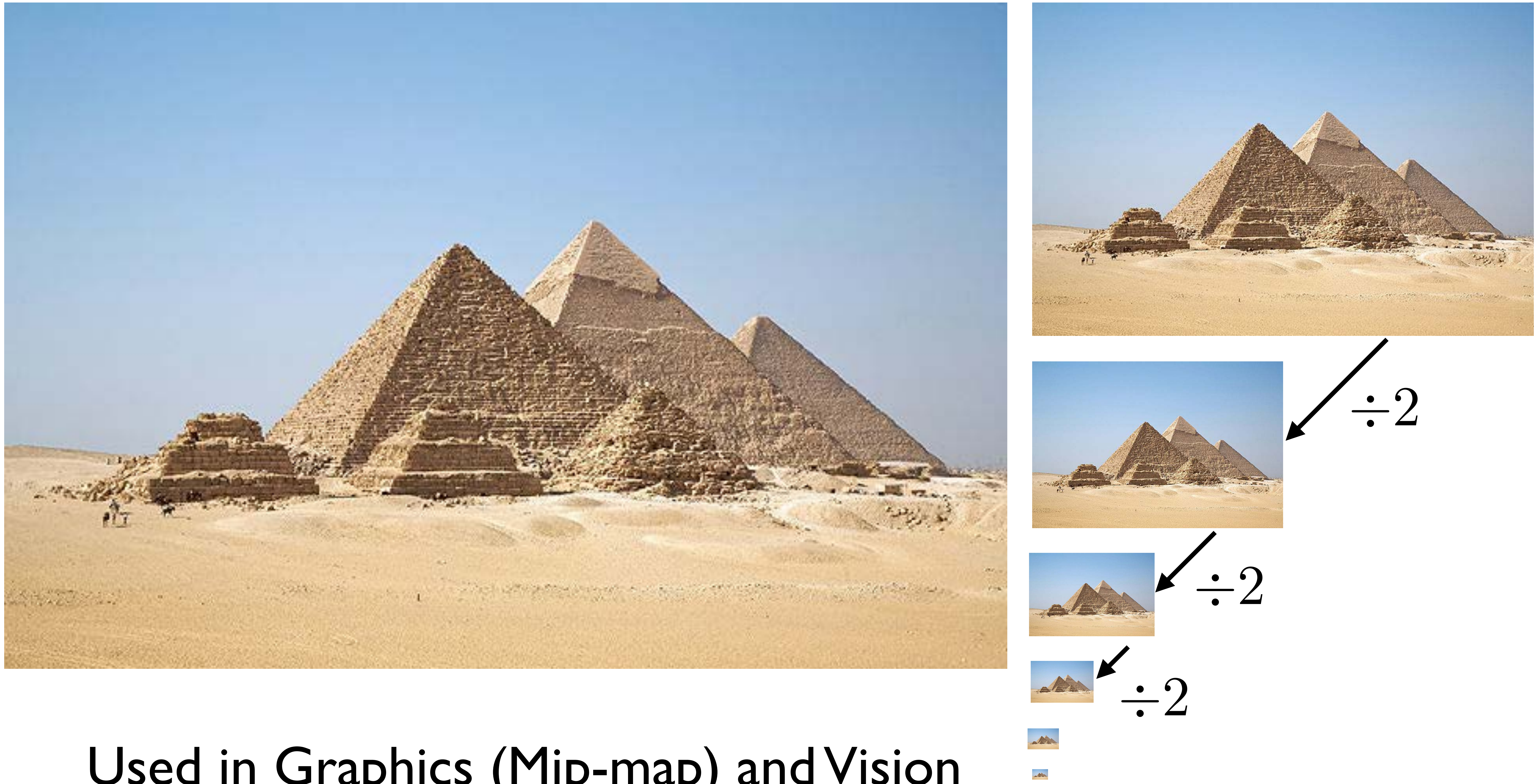
Correctly sampled

Aliasing in Photographs

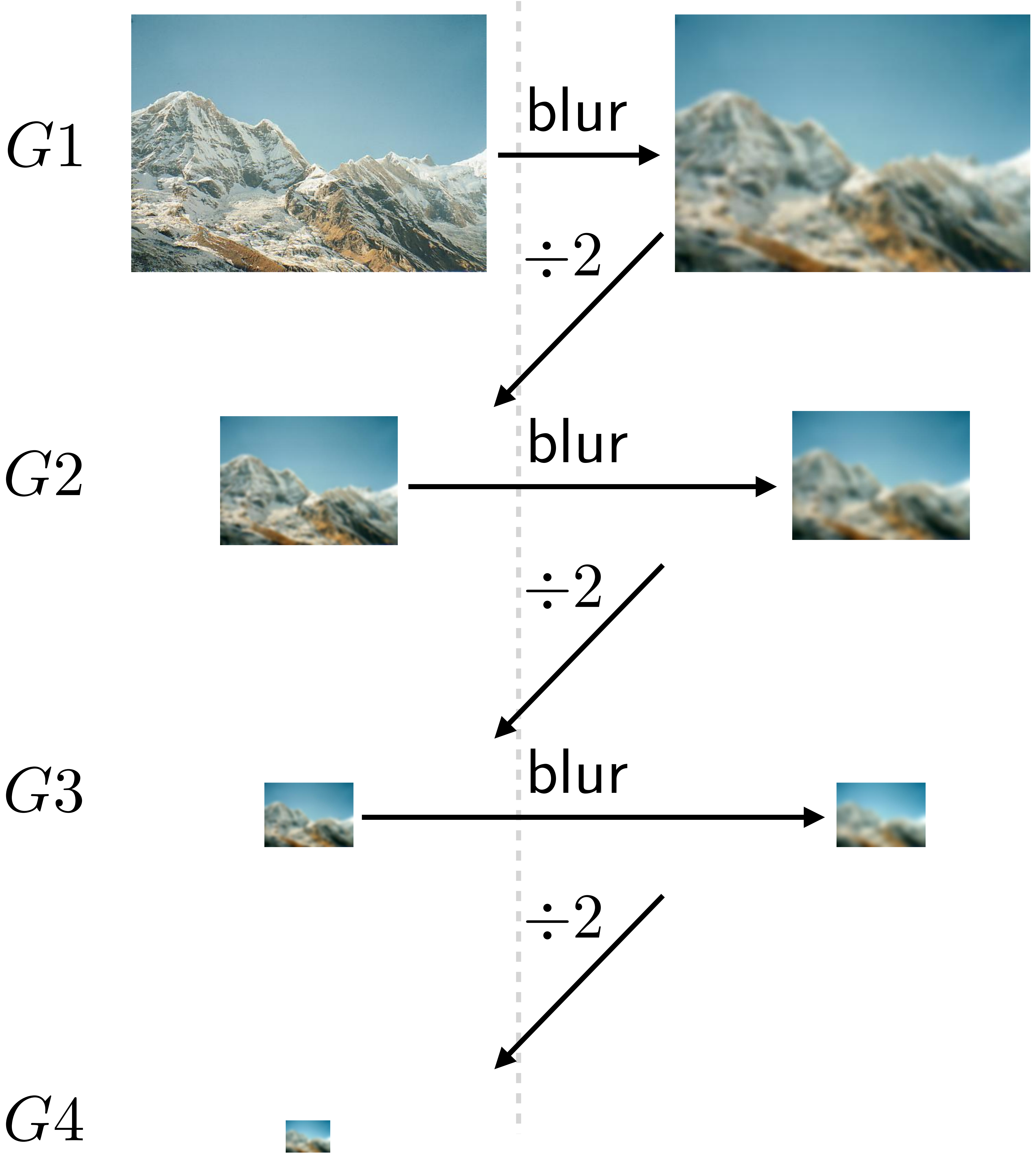
This is also known as “moire”

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Image Pyramids



Used in Graphics (Mip-map) and Vision
(for **multi-scale** processing)



Gaussian Pyramid

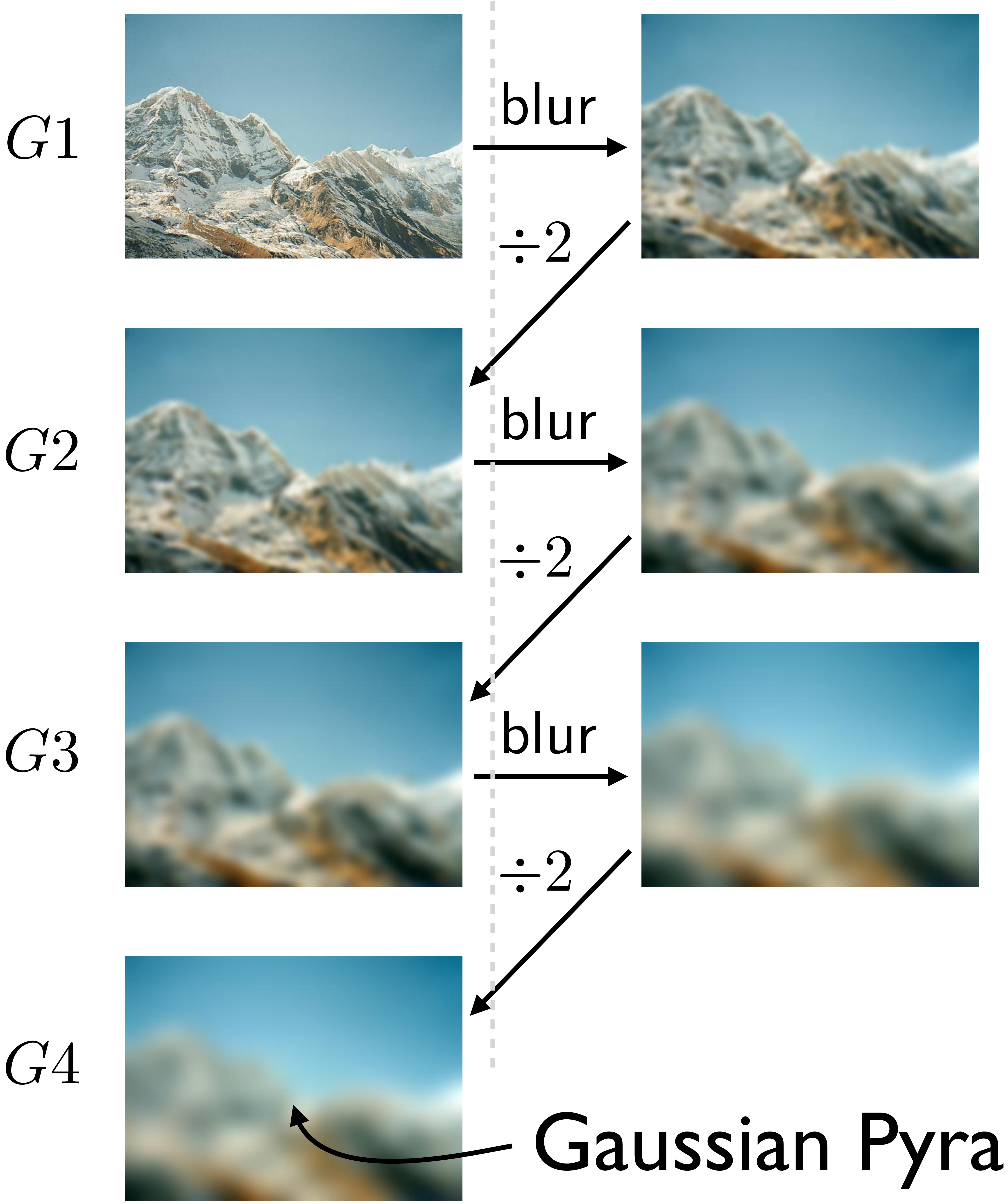
Blur with a Gaussian kernel, then select every 2nd pixel

$$I_s(x, y) = I(x, y) * g_\sigma(x, y)$$

Often approximations to the Gaussian kernel are used, e.g.,

$$\frac{1}{16}\begin{bmatrix}1 & 4 & 6 & 4 & 1\end{bmatrix}$$

[Assignment 2]



Blur with a Gaussian kernel, then select every 2nd pixel

$$I_s(x, y) = I(x, y) * g_\sigma(x, y)$$

Often approximations to the Gaussian kernel are used, e.g.,

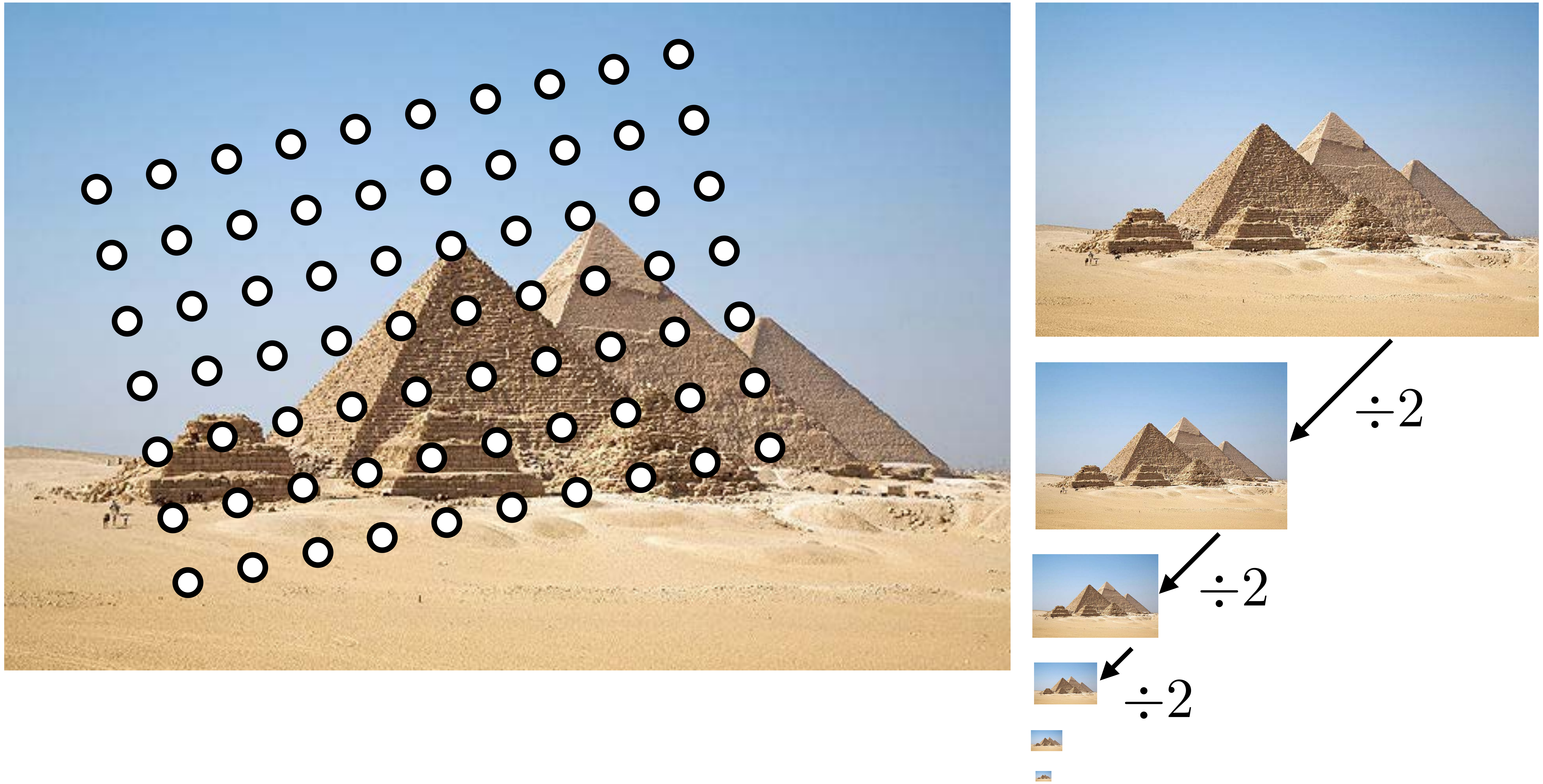
$$\frac{1}{16}\begin{bmatrix}1 & 4 & 6 & 4 & 1\end{bmatrix}$$

Gaussian Pyramid

[Assignment 2]

Sampling with Pyramids

Why are image pyramids important?



Find the level where the sample spacing is between 1 and 2 pixels, apply extra fraction of inter-octave blur as needed

Oversampling and Undersampling

Question: For a bandlimited signal, what if you **oversample** (i.e., sample at greater than the Nyquist rate)

Answer: Nothing bad happens! Samples are redundant and there are wasted bits

Question: For a bandlimited signal, what if you **undersample** (i.e., sample at less than the Nyquist rate)

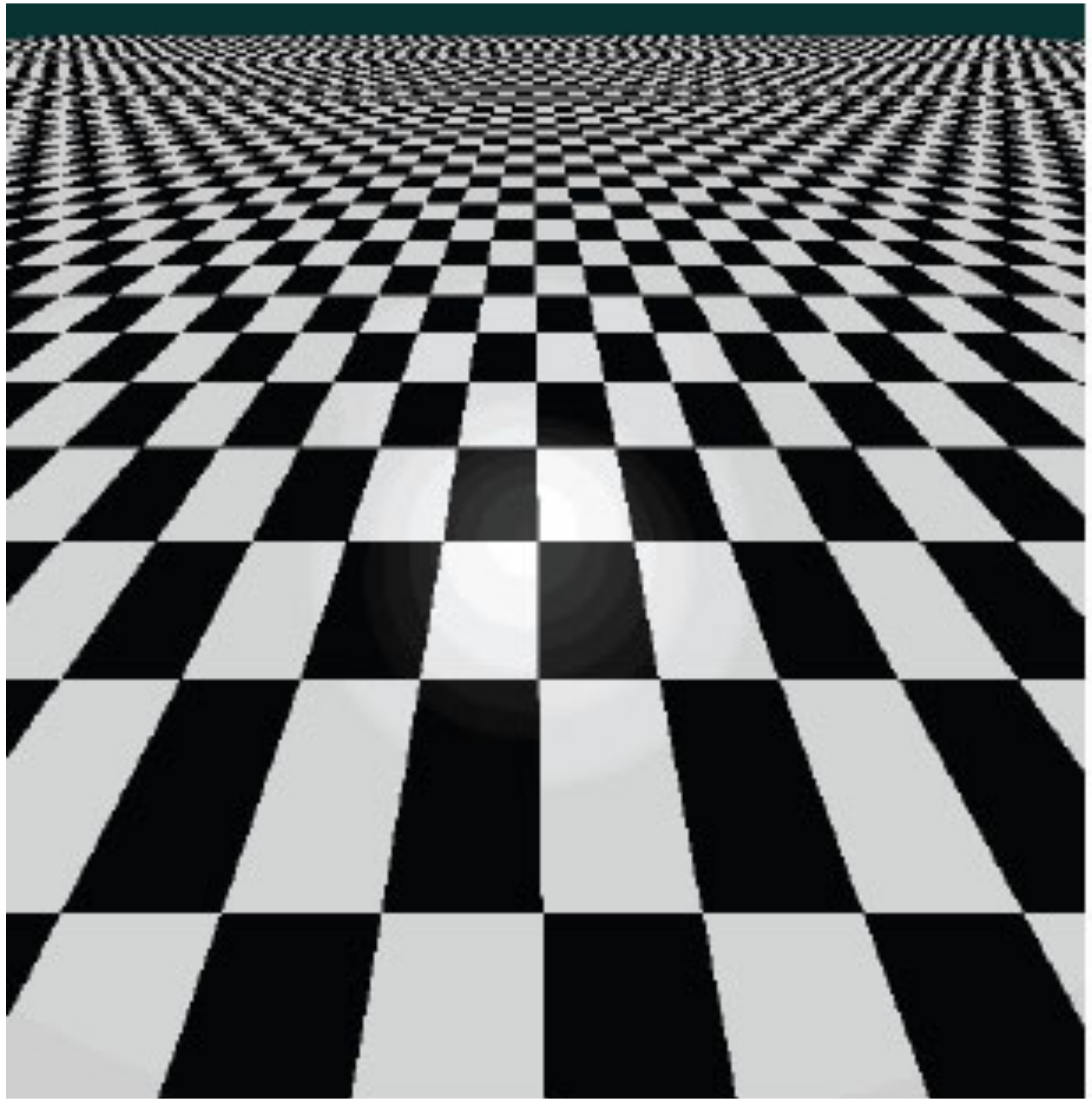
Answer: Two bad things happen! Things are missing (i.e., things that should be there aren’t). There are artifacts (i.e., things that shouldn’t be there are)

How to Prevent **Aliasing?**

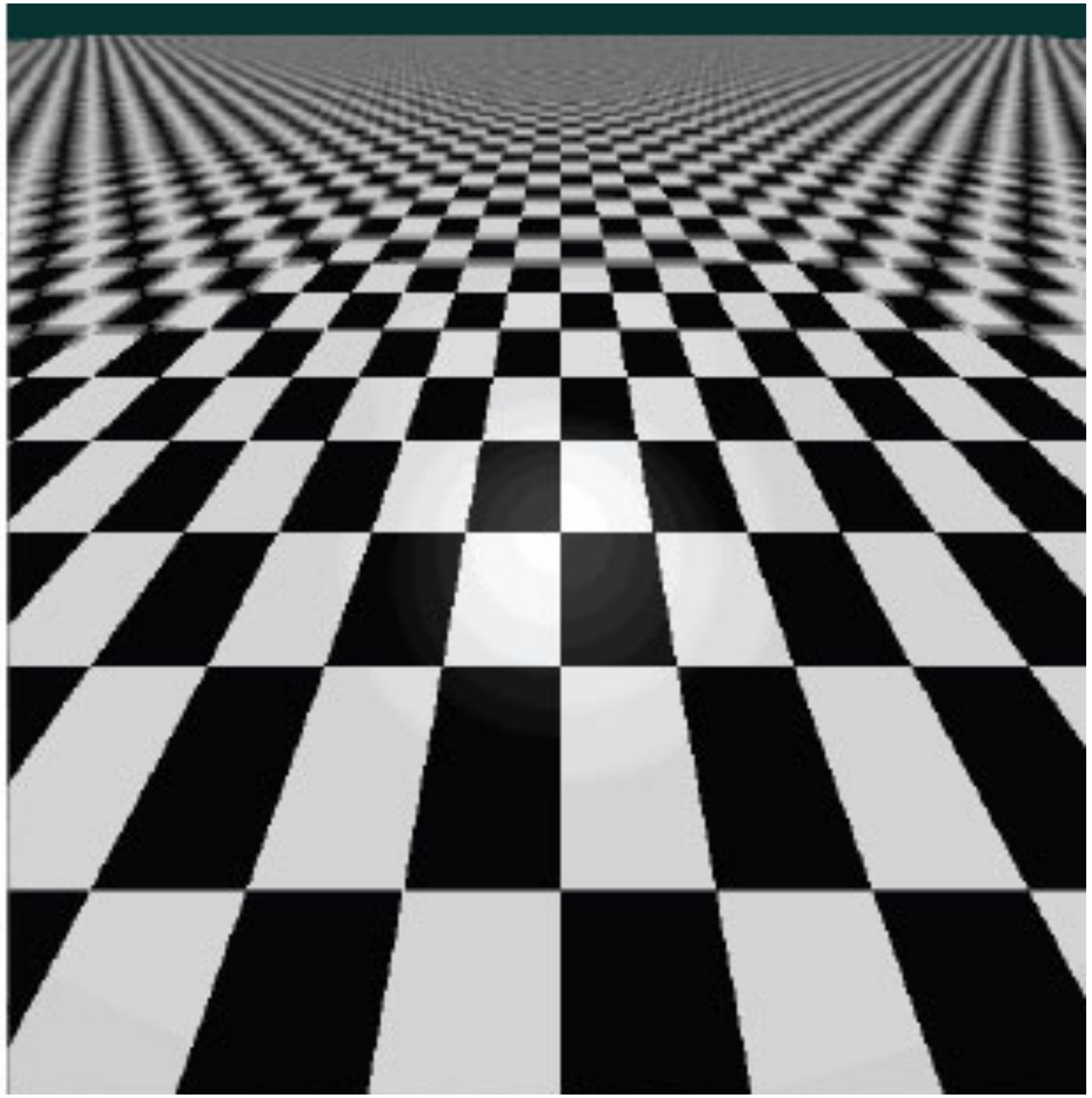
1. **Sample more frequently** i.e., oversampling — sample more than you think you need and average (i.e., area sampling)

2. **Reduce the maximum frequency**, by low pass filtering i.e., **Smoothing** before sampling.

Aliasing



aliasing artifacts



anti-aliasing by oversampling

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Temporal Aliasing

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

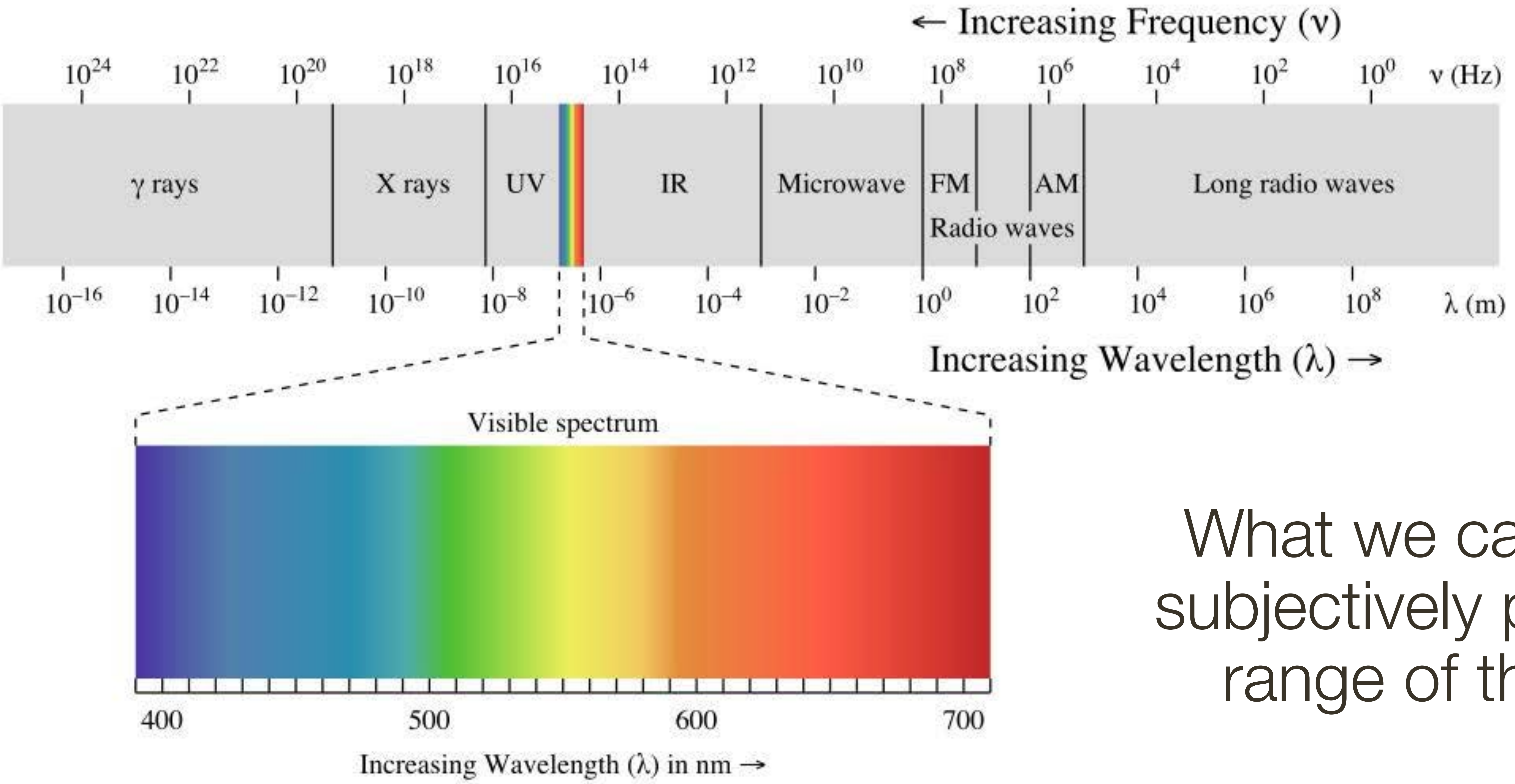
Temporal Aliasing

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Color is an Artifact of Human Perception

"Color" is **not** an objective physical property of light (electromagnetic radiation).

Instead, light is characterized by its wavelength.

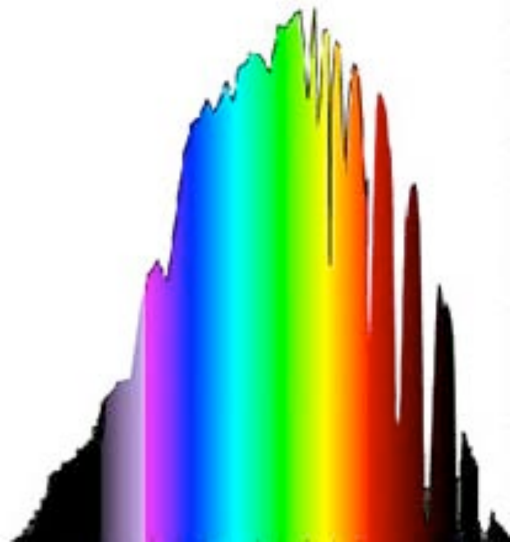


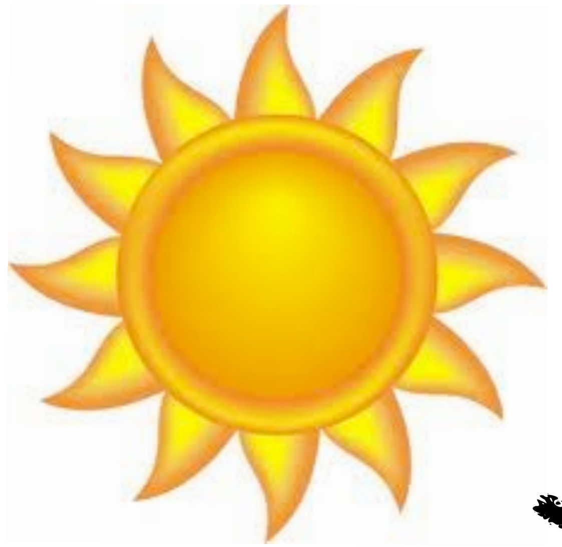
electromagnetic spectrum

What we call "color" is how we subjectively perceive a very small range of these wavelengths.

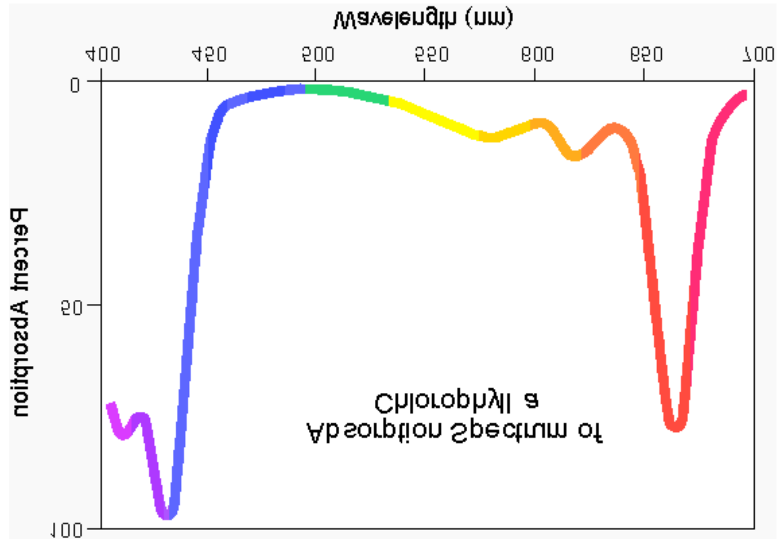
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Colour Perception





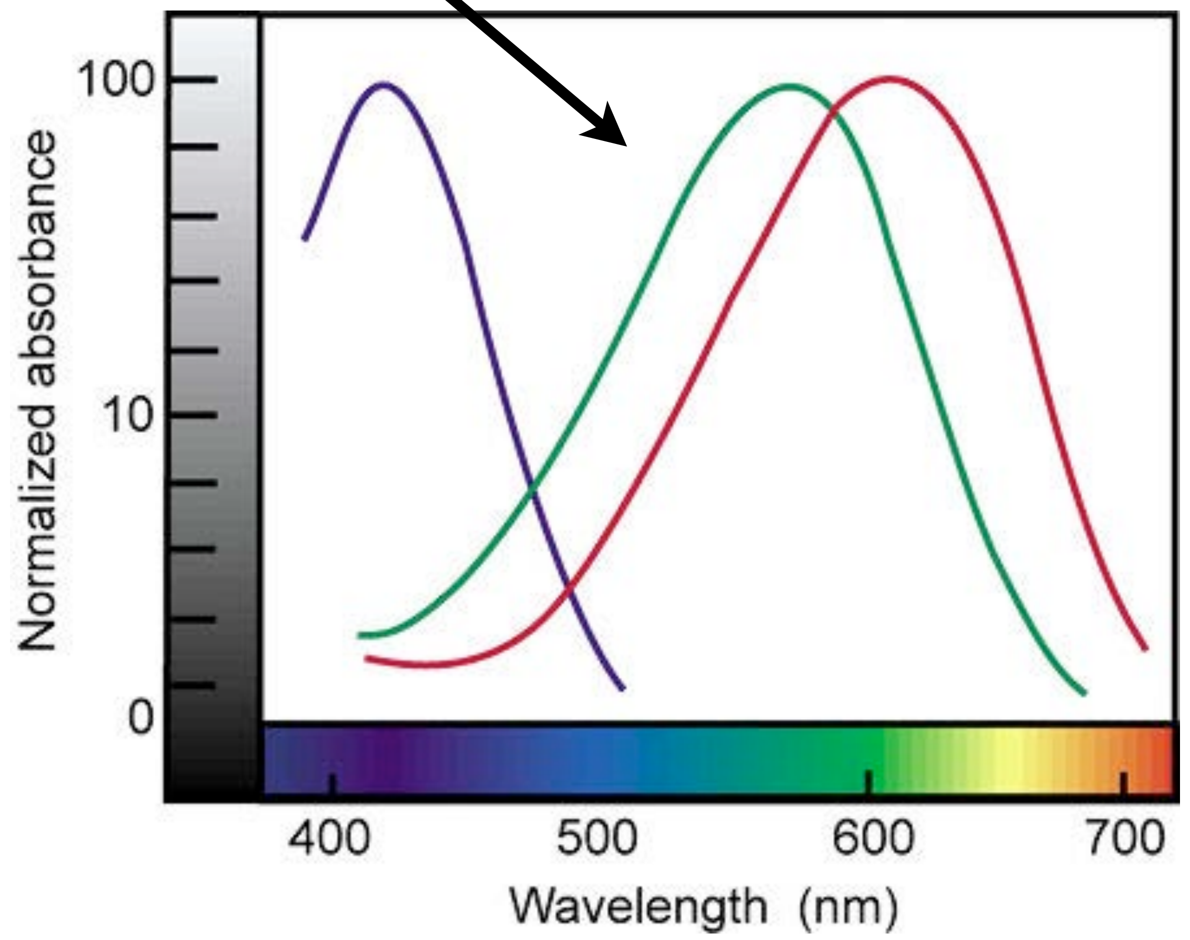
$E(\lambda)$



$S(\lambda)$

Cone responses

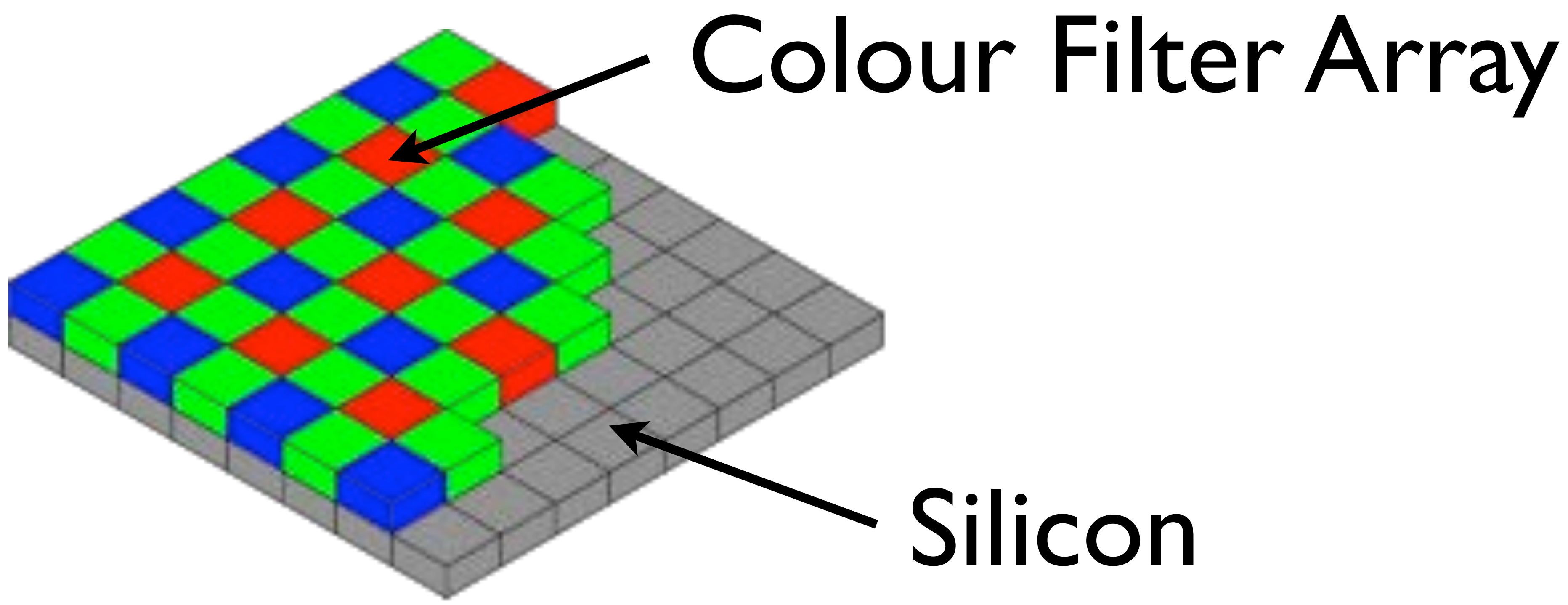
$R_{red}(\lambda)$



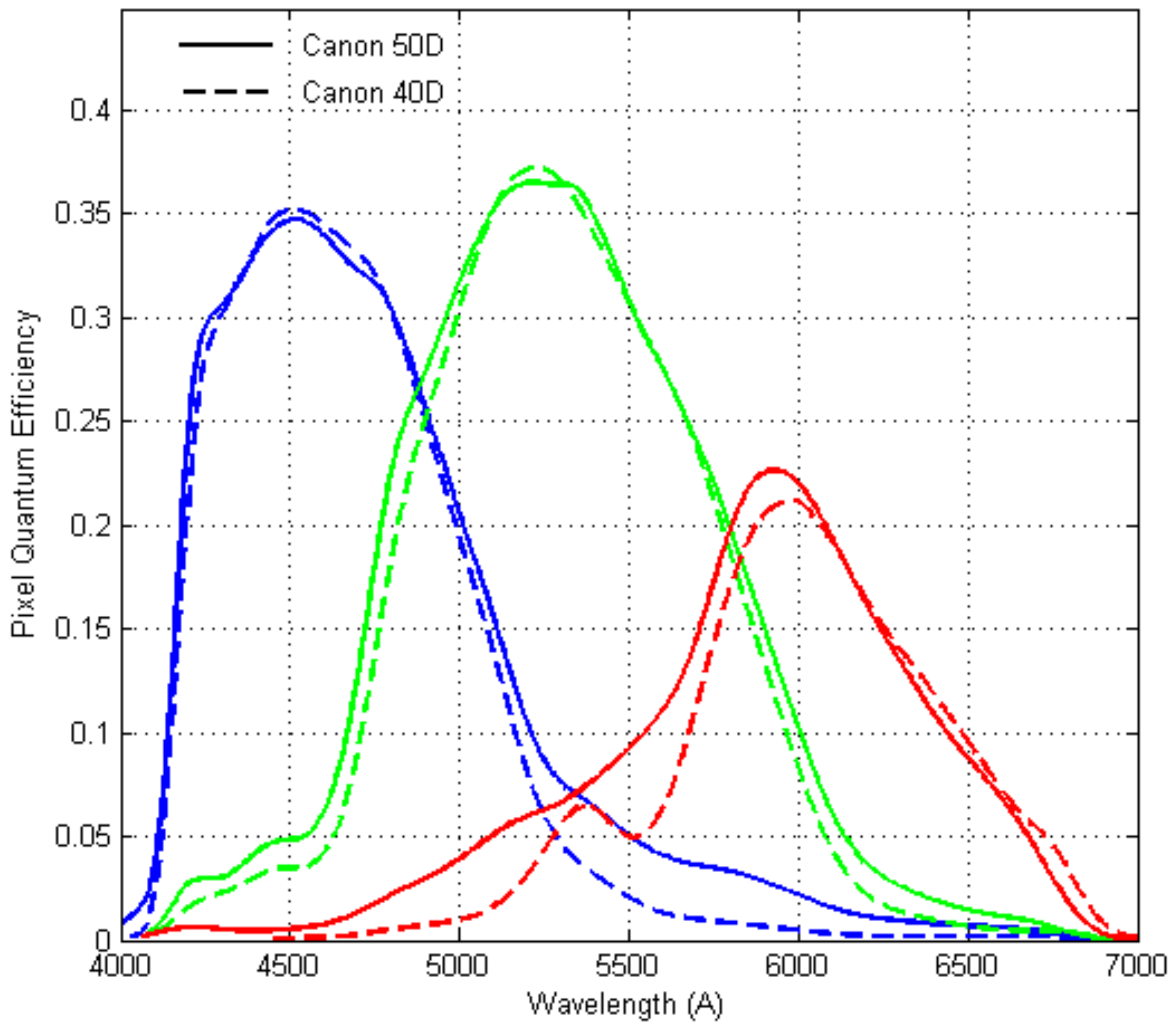
Cone excitation (multiply and add):

$$\rho_{red} = \int R_{red}(\lambda) E(\lambda) S(\lambda) \, d\lambda$$

Digital Sensor



Canon 50D

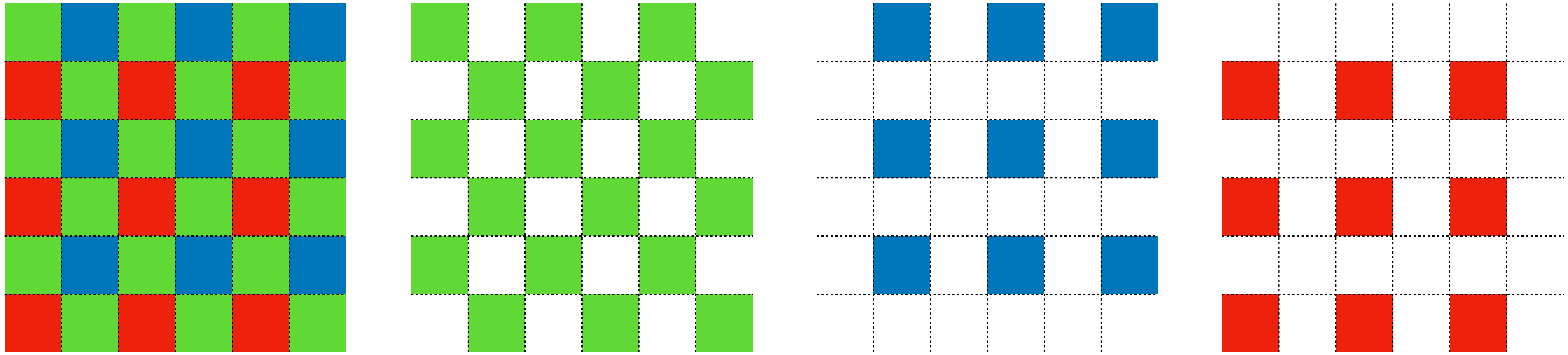


$f(\lambda)$

- Analogue image is sampled by a CMOS (or CCD) sensor
- RGB colour filters arranged in a “Bayer” pattern
- Spectral response of R,G,B filters = Quantum Efficiency
- Counts from this sensor are camera RAW

Demosaicing

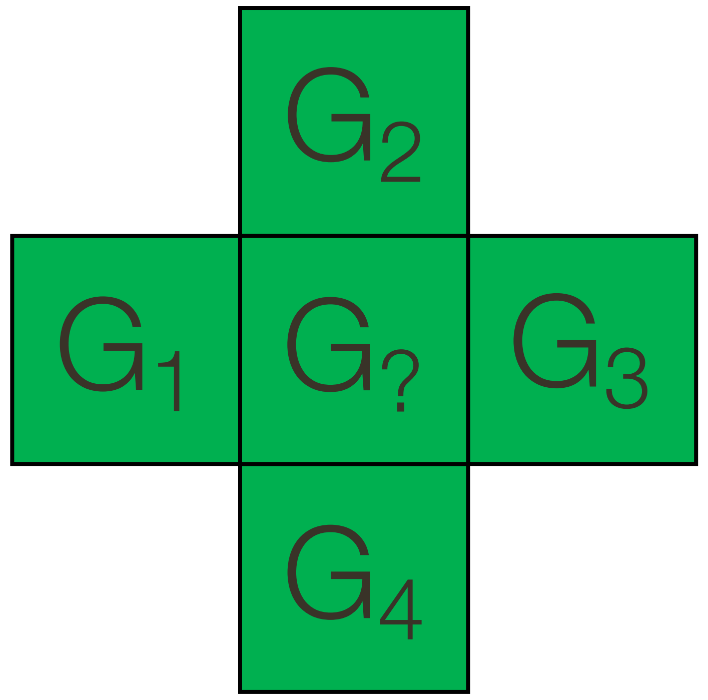
Each colour channel has different information:



How can we fill in the missing information?

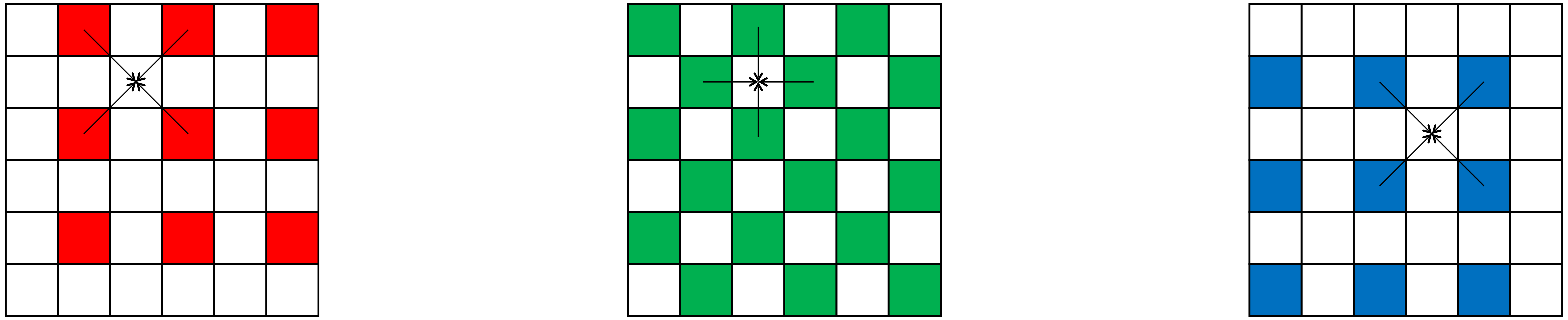
Demosaicing by Bilinear Interpolation

Bilinear interpolation: Simply average your 4 neighbors.



$$G_? = \frac{G_1 + G_2 + G_3 + G_4}{4}$$

Neighborhood changes for different channels:

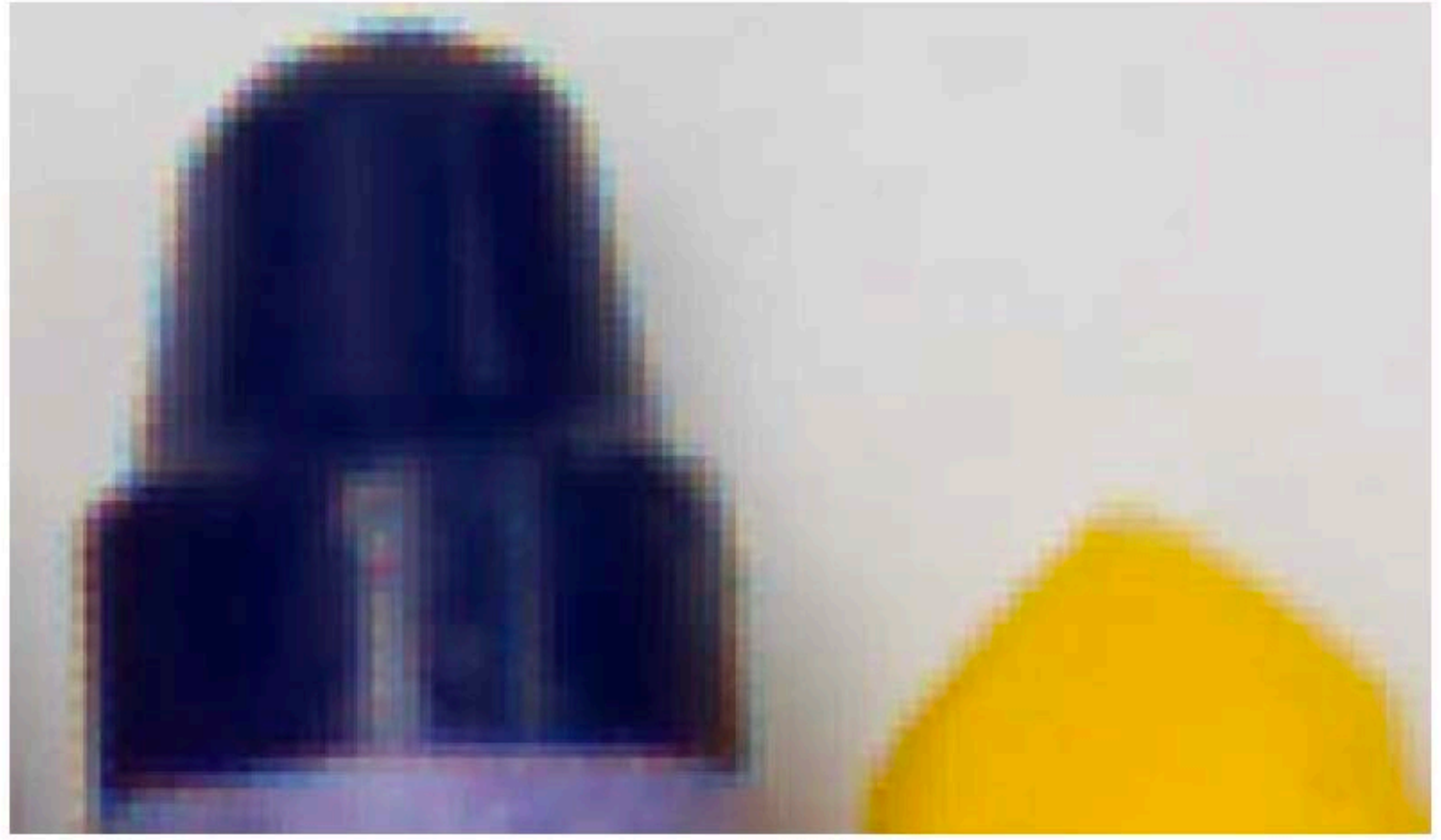


Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

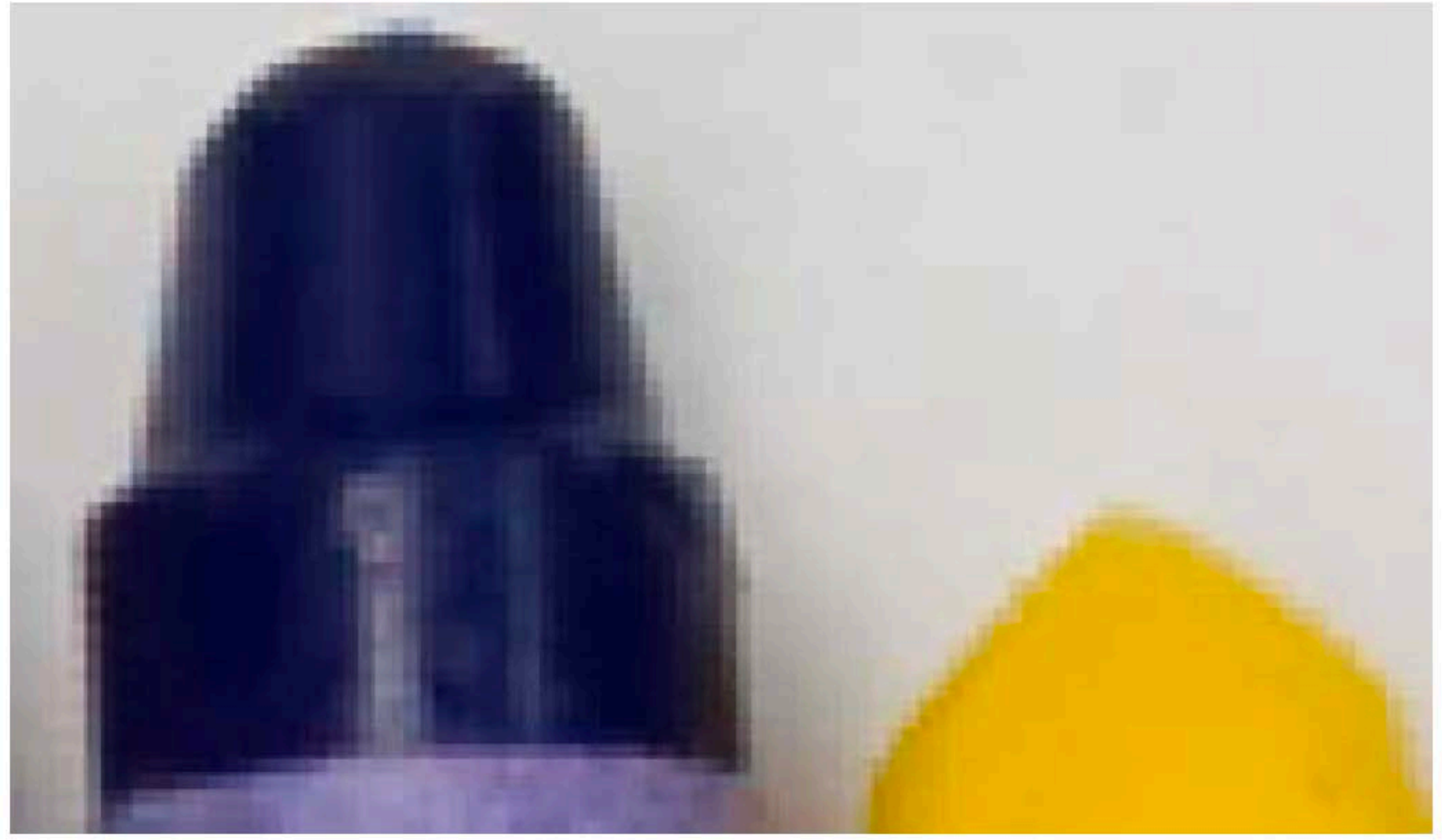
Demosaicing

Optional subtitle

- Simple interpolation causes colour errors



Bilinear interpolation

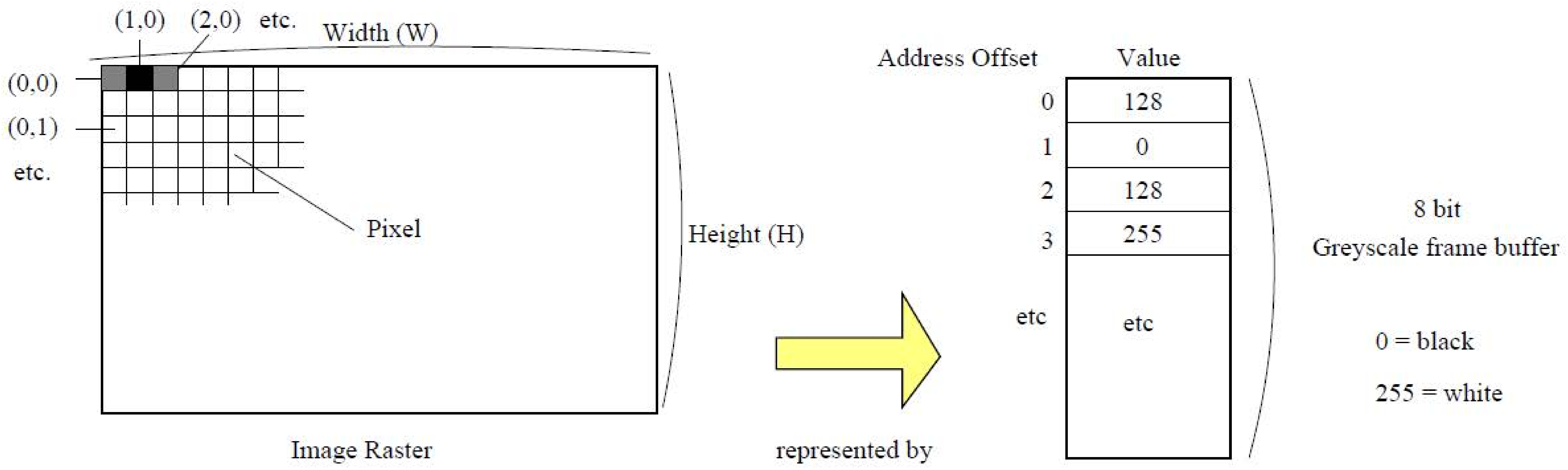


Bennet et al 2006
(local 2 colour prior)

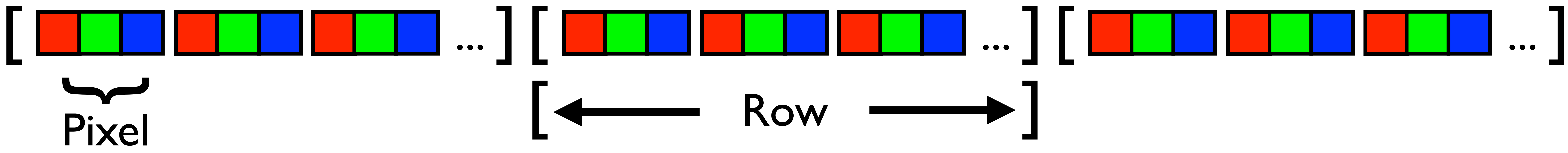
- Many techniques use edge information from the densely sampled green channel, and some form of image prior
- It can also been tackled via a data-driven approach, e.g., [Gharbi et al. 2016]

The Digital Image

Optional subtitle



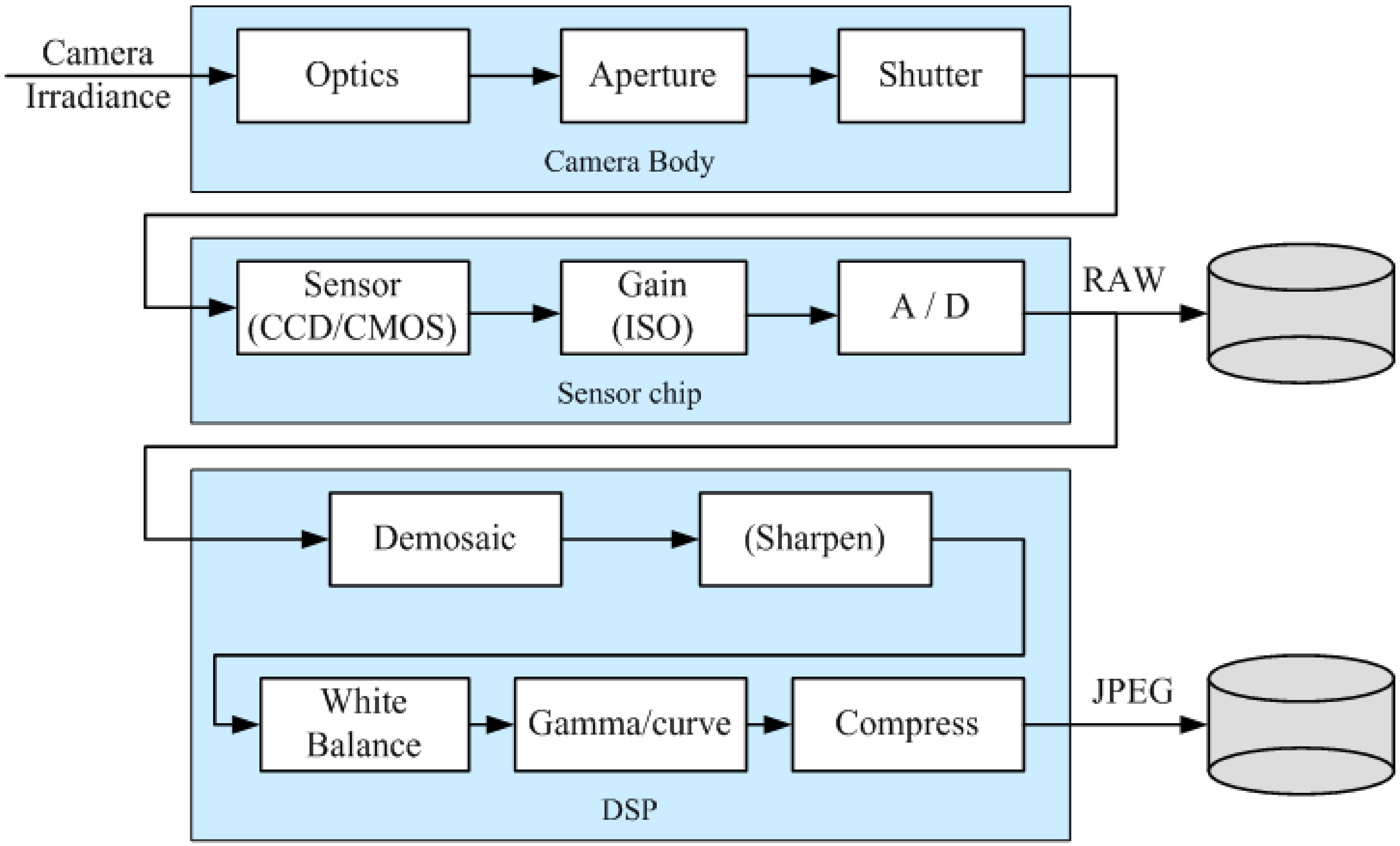
e.g., arranged in memory with RGB pixels stored in rows:



Many other possibilities, e.g., BGR, RGBA pixels, row/column major ordering, and rows or columns aligned to power of 2 boundaries

Digital Camera Processing

Main stages in a digital camera



[Szeliski 2.3]

White Balance

Optional subtitle

- Humans are good at adapting to global illumination conditions: you would still describe a white object as white whether under blue sky or candle light.
- However, when the picture is viewed later, the viewer is no longer correcting for the environment and the illuminant colour typically appears too strong.
- **White balancing** is the process of correcting for the illuminant

7.1

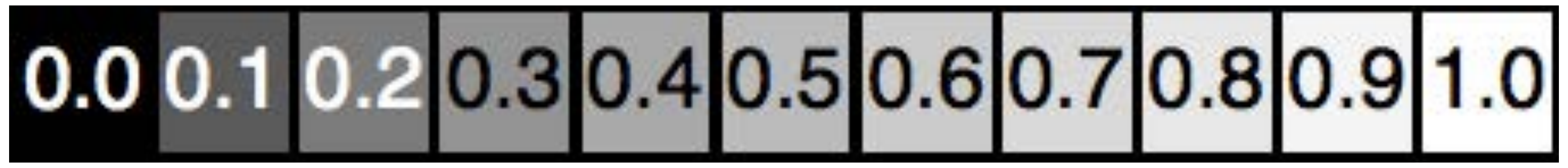
- A simple white balance algorithm is to assume the scene is grey on average "greyworld", state of the art methods use learning, e.g., Barron ICCV 2015

Gamma Correction

Optional subtitle

- Equal steps in luminance ≠ equal in perceived brightness

linear luminance (raw)



equal brightness steps

- Equal steps in human perceived brightness are achieved by increasingly large steps in luminance (sensor counts)
- So we encode pixel values V using a power law:

7.2

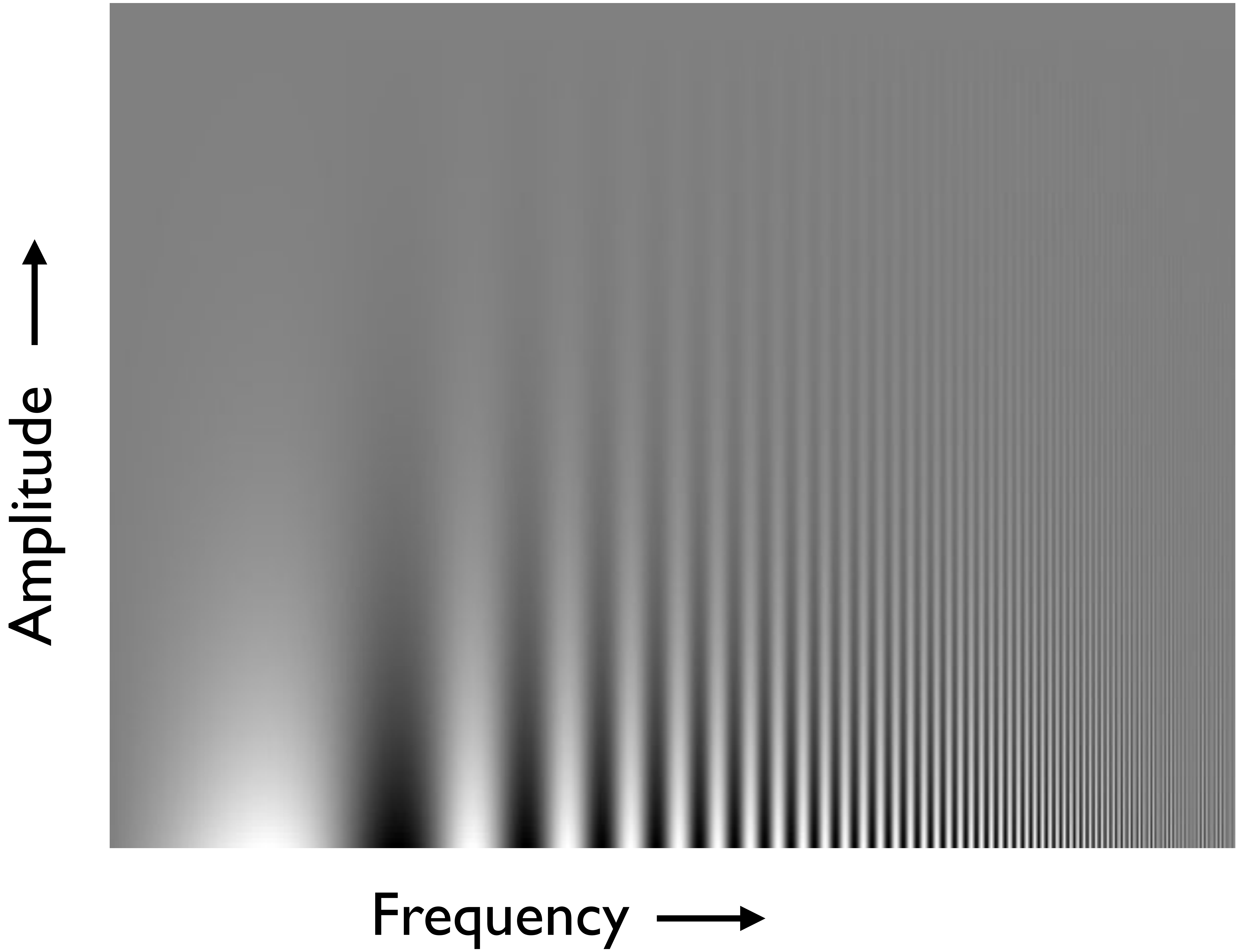
$$L = V^{\gamma}$$

- Using raw sensor counts wastes bits as we can't differentiate the large values → use **gamma corrected encoding (V)** that allocates more bits to smaller

Contrast Sensitivity

Optional subtitle

Human visual system is most sensitive to mid-frequencies



Discrete Cosine Transform

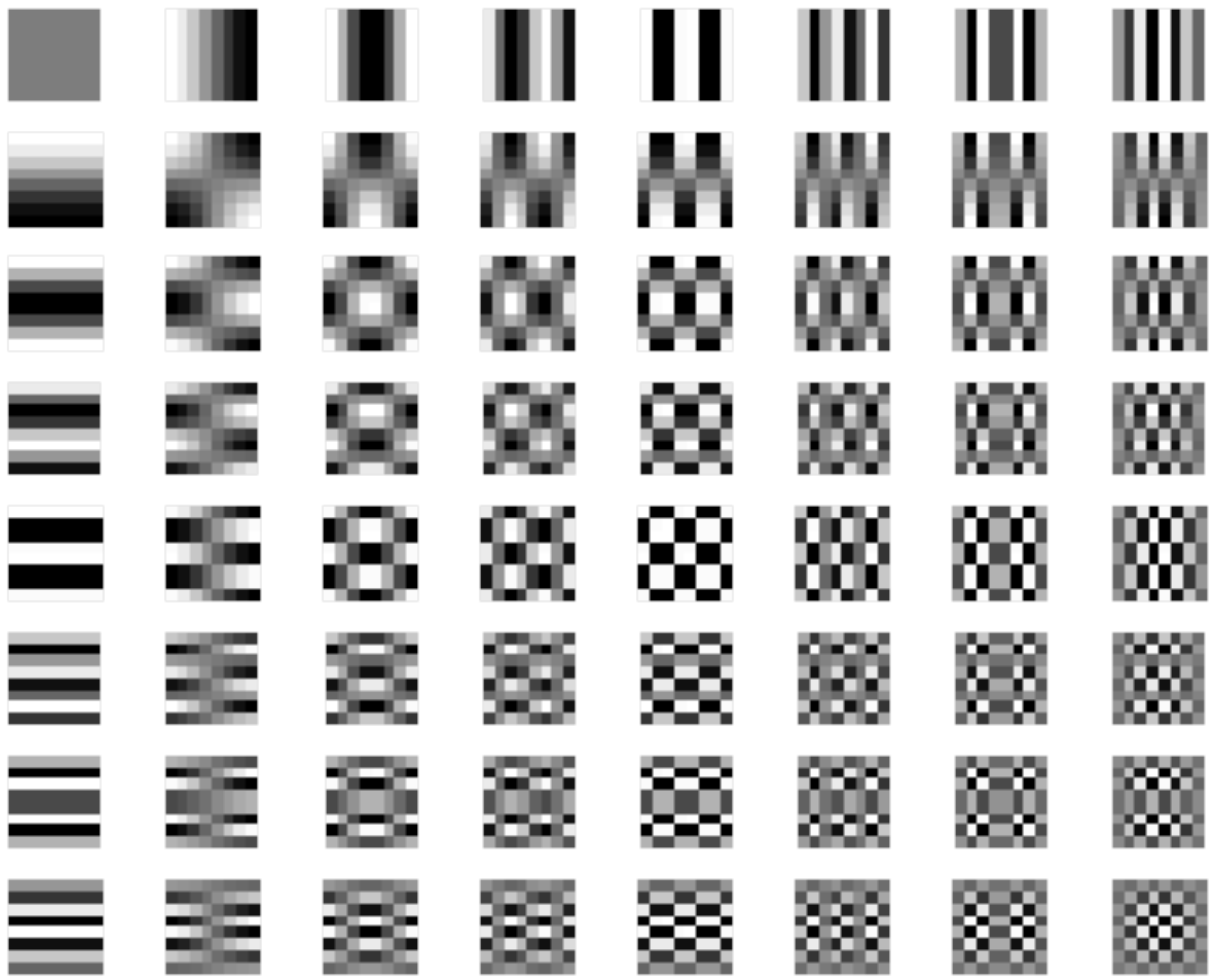
Optional subtitle

- Basis functions used in JPEG

$$X(m,n) = \alpha_m \alpha_n \sum_{k=0}^{K-1} \sum_{l=0}^{L-1} x(k,l) \cos\left[\frac{(2k+1)m\pi}{2K}\right] \cos\left[\frac{(2l+1)n\pi}{2L}\right]$$

- Energy is concentrated in the low-frequency components

- Efficient algorithm to compute (similar to FFT)



8x8 basis functions

Summary

In the continuous case, images are functions of two spatial variables, x and y.

The discrete case is obtained from the continuous case via sampling (i.e. tessellation, quantization).

If a signal is **bandlimited** then it is possible to design a sampling strategy such that the sampled signal captures the underlying continuous signal exactly (**Nyquist Sampling**).

Human **trichromatic colour** perception, and other perceptual sensitivities such as contrast sensitivity influence the image coding pipeline.