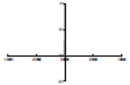
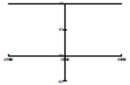
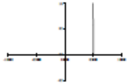
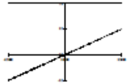
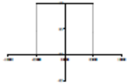
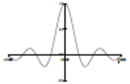
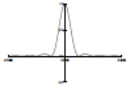
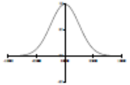
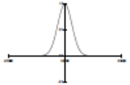
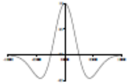
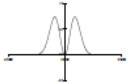
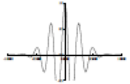
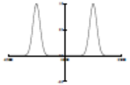
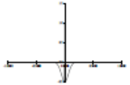
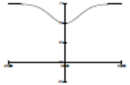
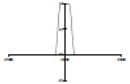
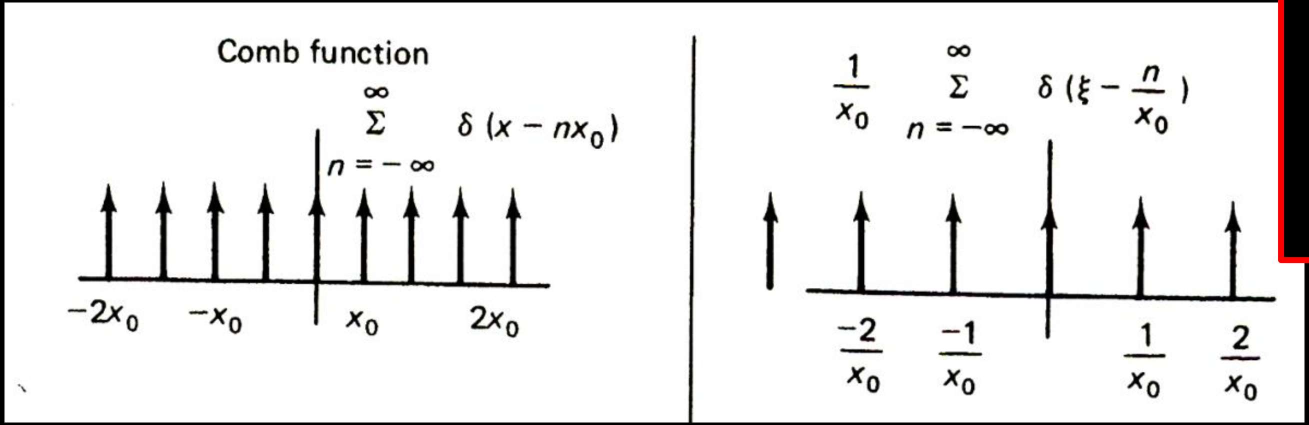
CS4495/6495
Introduction to Computer Vision

2C-L3 *Aliasing*

Recall: Fourier Pairs (from Szeliski)

Name	Signal			Transform	
impulse		$\delta(x)$	$\Leftrightarrow$	1	
shifted impulse		$\delta(x-u)$	$\Leftrightarrow$	$e^{-j\omega u}$	
box filter		$\mathrm{box}(x/a)$	$\Leftrightarrow$	$a\mathrm{sinc}(a\omega)$	
tent		$\mathrm{tent}(x/a)$	$\Leftrightarrow$	$a\mathrm{sinc}^2(a\omega)$	
Gaussian		$G(x;\sigma)$	$\Leftrightarrow$	$\frac{\sqrt{2\pi}}{\sigma}G(\omega;\sigma^{-1})$	
Laplacian of Gaussian		$(\frac{x^2}{\sigma^4}-\frac{1}{\sigma^2})G(x;\sigma)$	$\Leftrightarrow$	$-\frac{\sqrt{2\pi}}{\sigma}\omega^2 G(\omega;\sigma^{-1})$	
Gabor		$\cos(\omega_0 x)G(x;\sigma)$	$\Leftrightarrow$	$\frac{\sqrt{2\pi}}{\sigma}G(\omega\pm\omega_0;\sigma^{-1})$	
unsharp mask		$(1+\gamma)\delta(x)$ $-\gamma G(x;\sigma)$	$\Leftrightarrow$	$(1+\gamma)-$ $\frac{\sqrt{2\pi}\gamma}{\sigma}G(\omega;\sigma^{-1})$	
windowed sinc		$\mathrm{rcos}(x/(aW))$ $\mathrm{sinc}(x/a)$	$\Leftrightarrow$	(see Figure 3.29)	

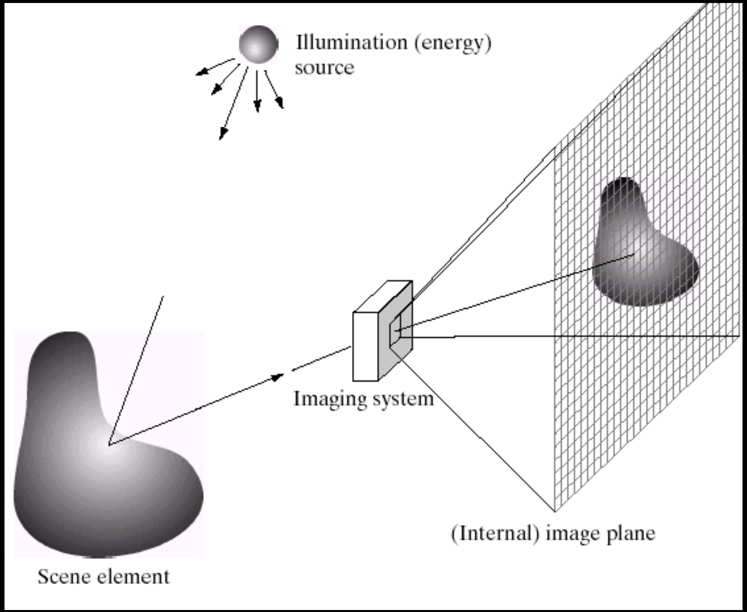
Fourier Transform Sampling Pairs

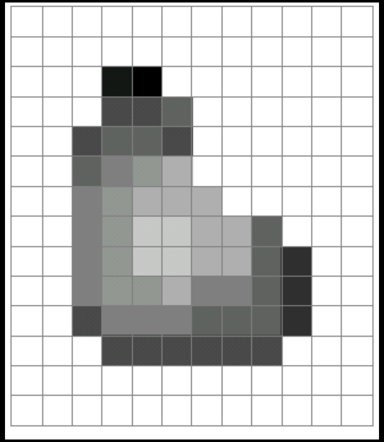


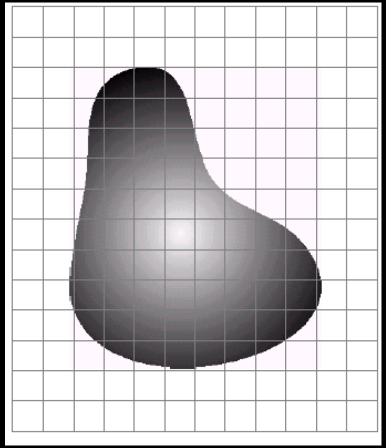
FT of an "impulse train" is an impulse train

Sampling and Aliasing

Sampling and Reconstruction

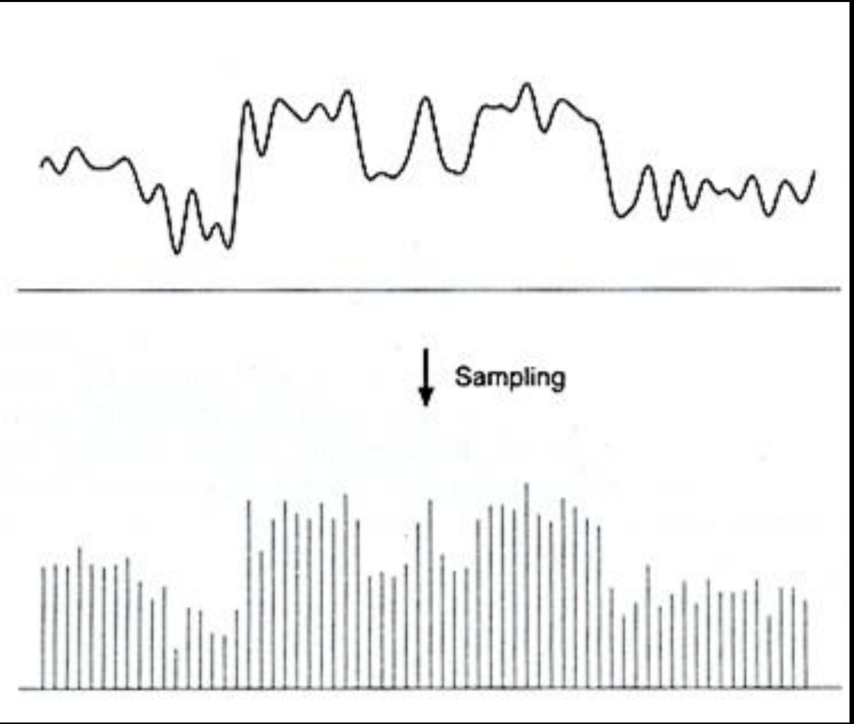






Sampled representations

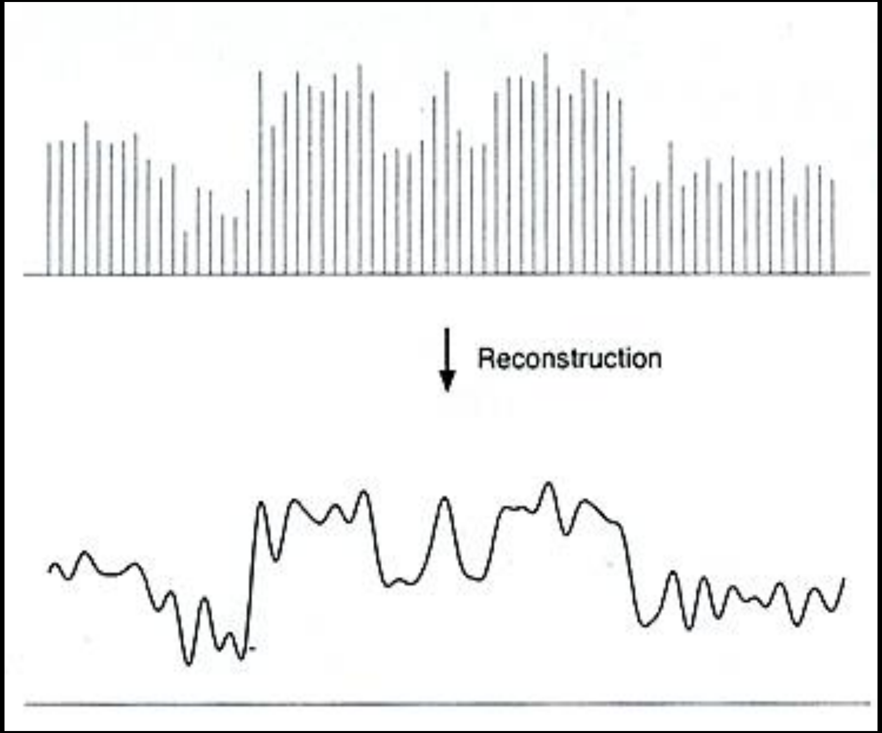
- How to store and compute with continuous functions?
- Common scheme for representation: *samples*



S. Marschner

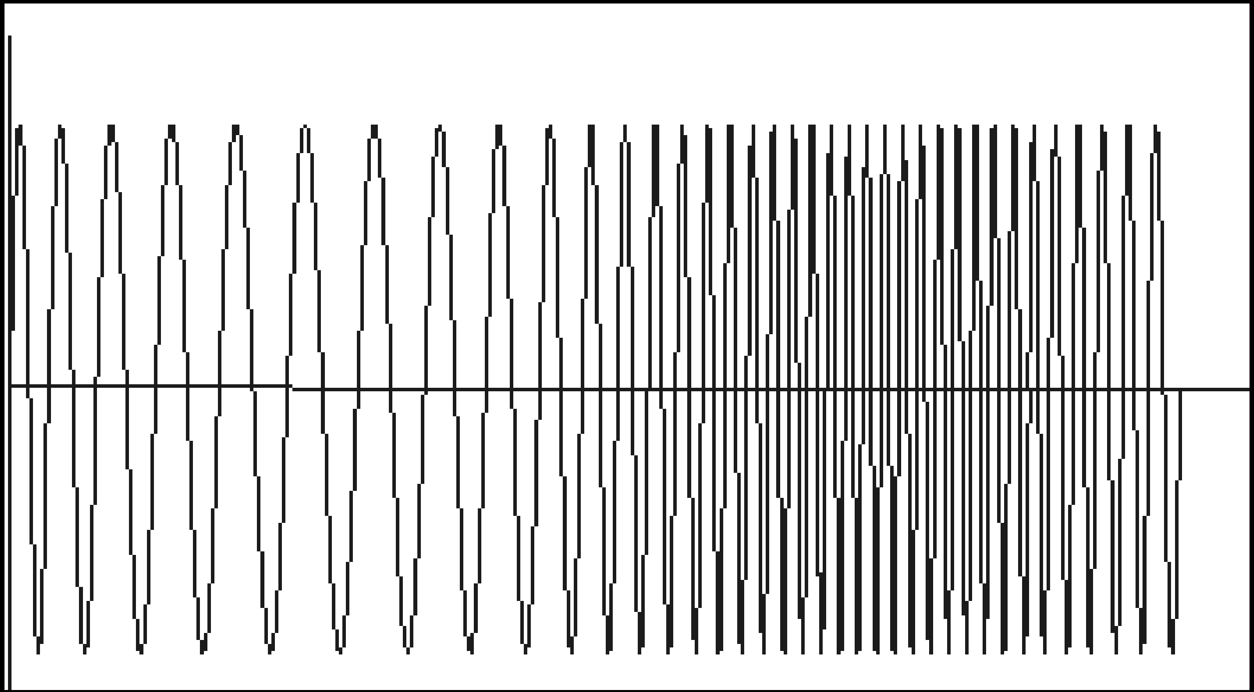
Reconstruction

- Making samples back into a continuous function
  - for output (need realizable method)
  - for analysis or processing (need mathematical method)
- Amounts to “guessing” what the function did in between



S. Marschner

1D Example: Audio

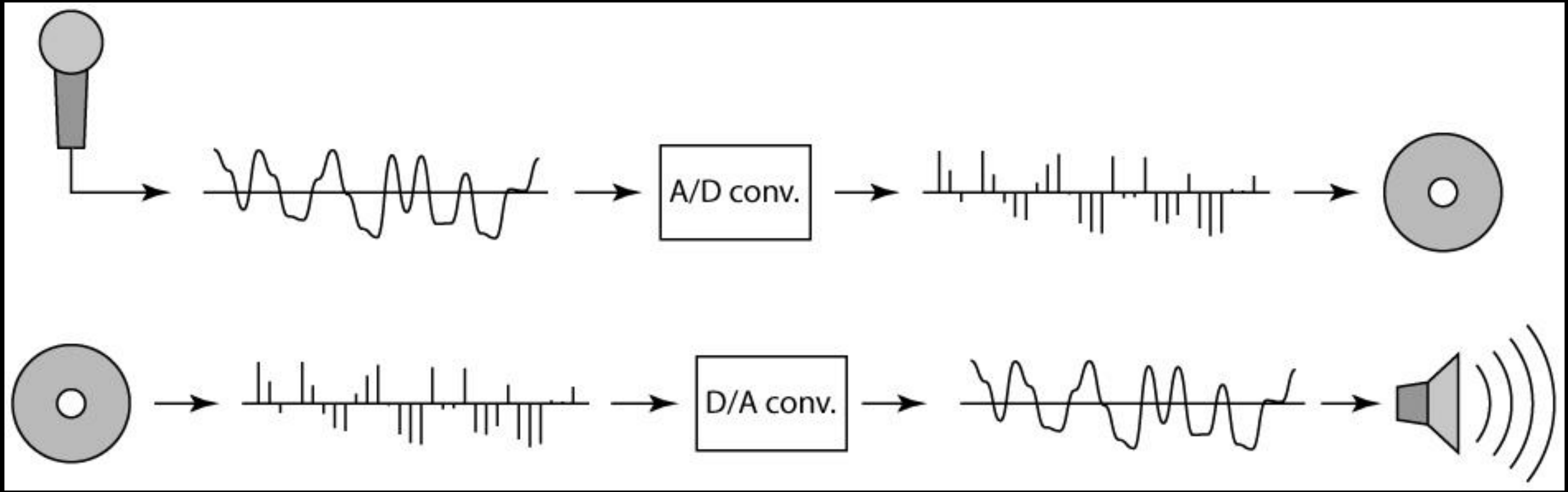


low high

frequencies

Sampling in digital audio

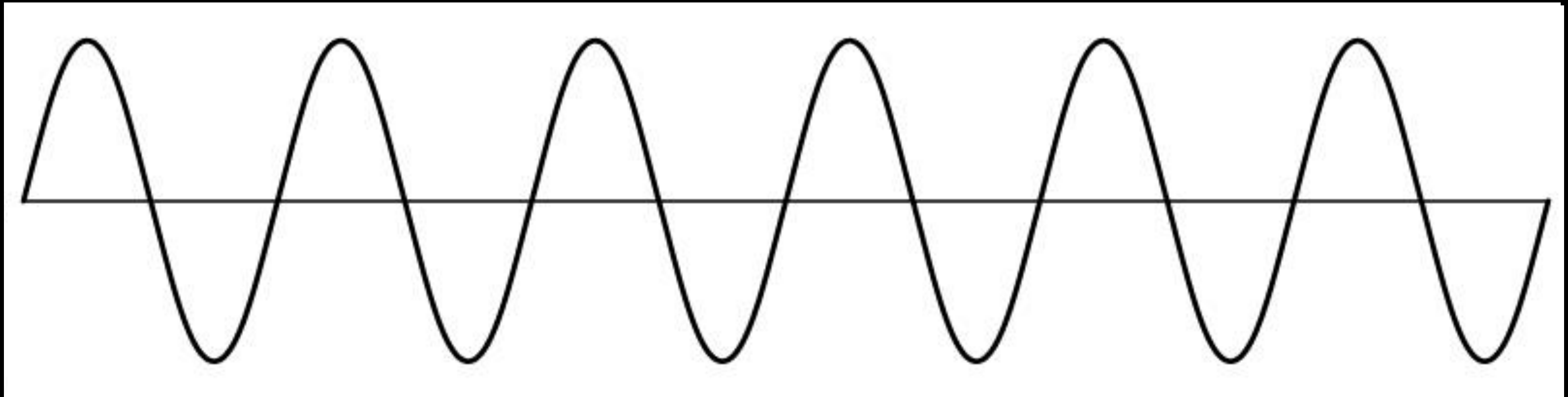
- Recording: sound to analog to samples to disc
- Playback: disc to samples to analog to sound again



S. Marschner

Sampling and Reconstruction

- Simple example: a sign wave



S. Marschner

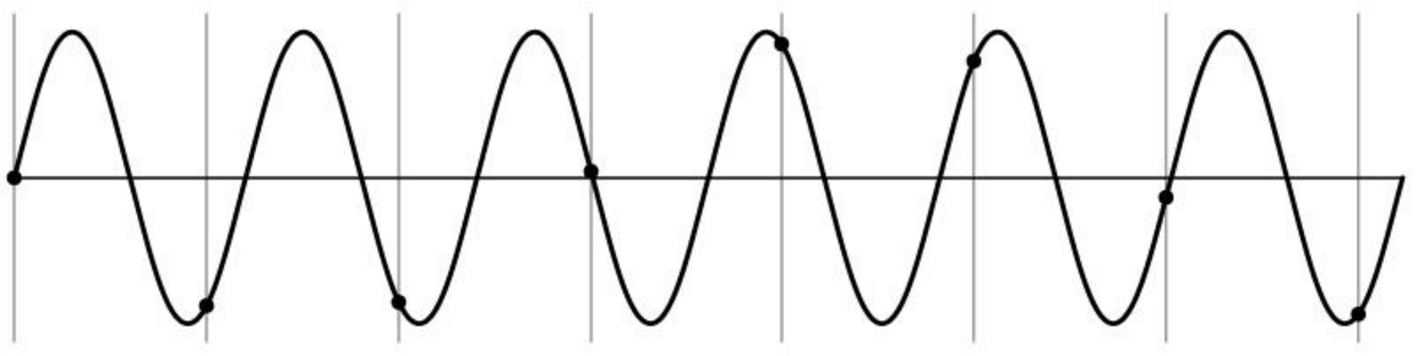
Undersampling

- What if we "missed" things between the samples?

S. Marschner

Undersampling

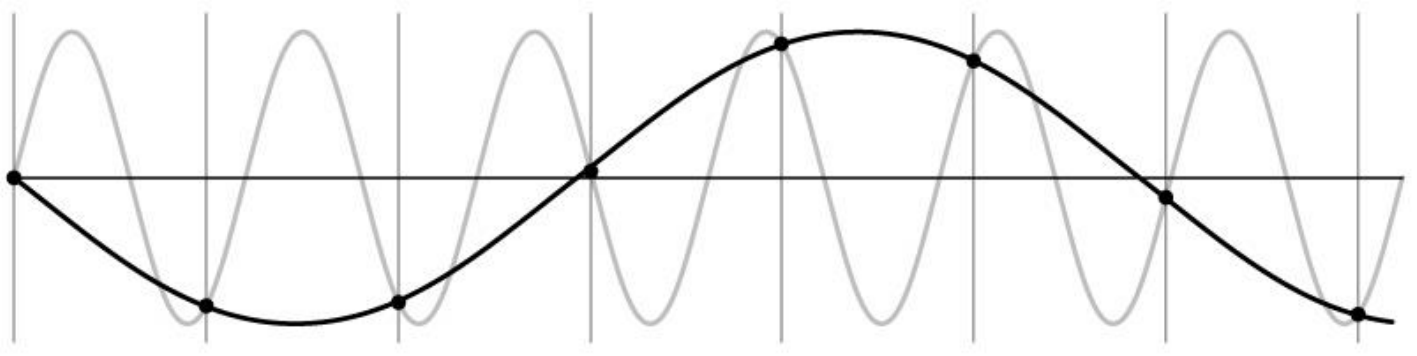
- Simple example: undersampling a sine wave
  - unsurprising result: information is lost



S. Marschner

Undersampling

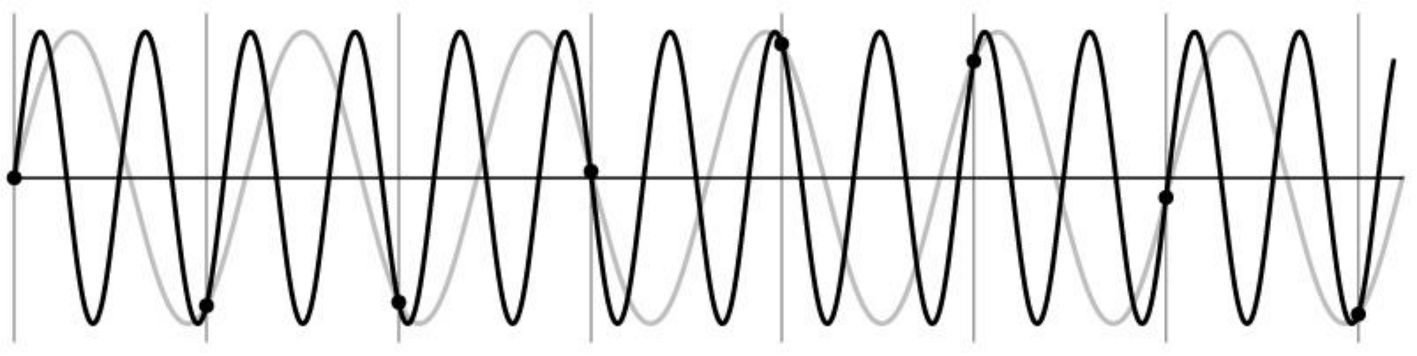
- Simple example: undersampling a sine wave
  - unsurprising result: information is lost
  - surprising result: indistinguishable from lower frequency



S. Marschner

Undersampling

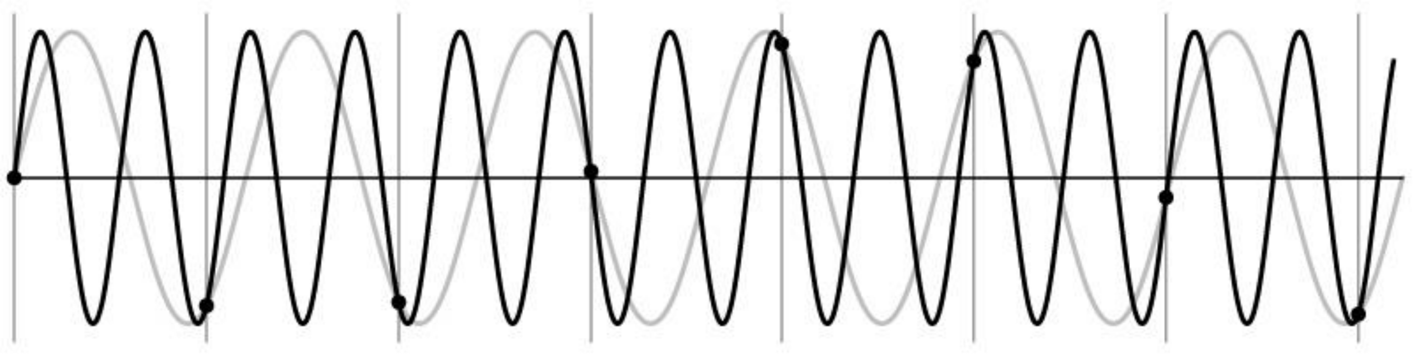
- Simple example: undersampling a sine wave
  - Low frequency also was always indistinguishable from higher frequencies



S. Marschner

Undersampling

- *<u>Aliasing</u>*<u>:</u> signals "traveling in disguise" as other frequencies

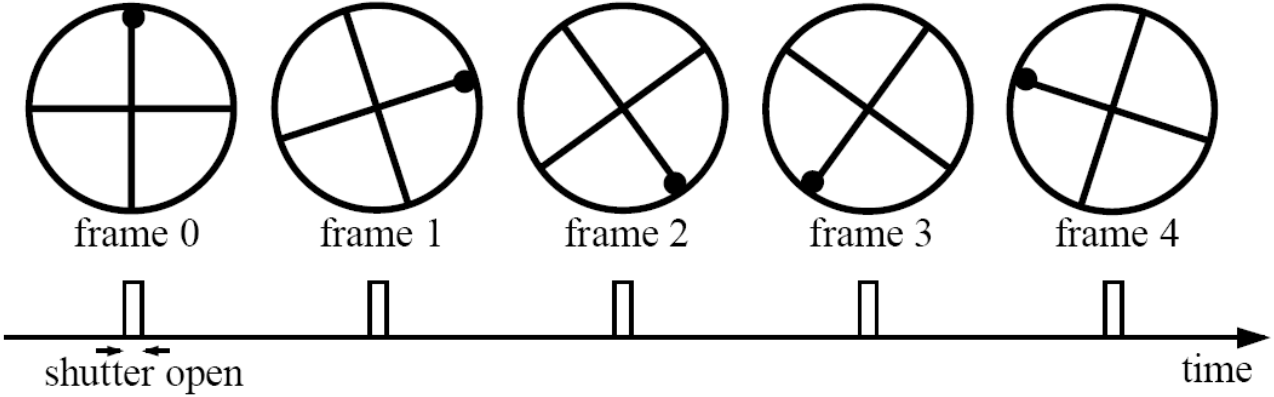


S. Marschner

Aliasing in video

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):



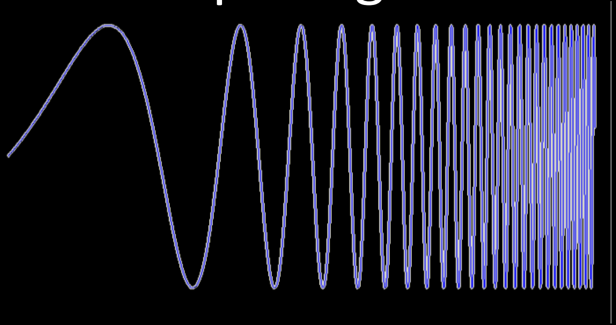
Without dot, wheel appears to be rotating slowly backwards! (counterclockwise)

S. Seitz

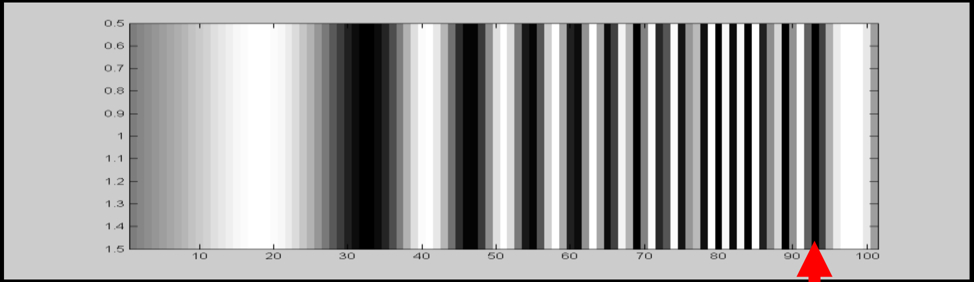
Aliasing in images

What's happening?

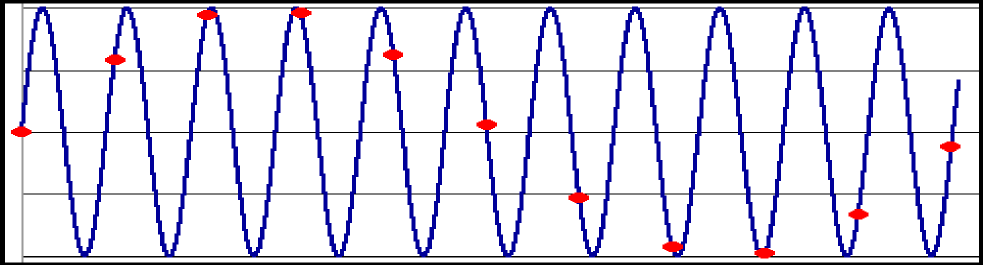
Input signal:



Plot as image:



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



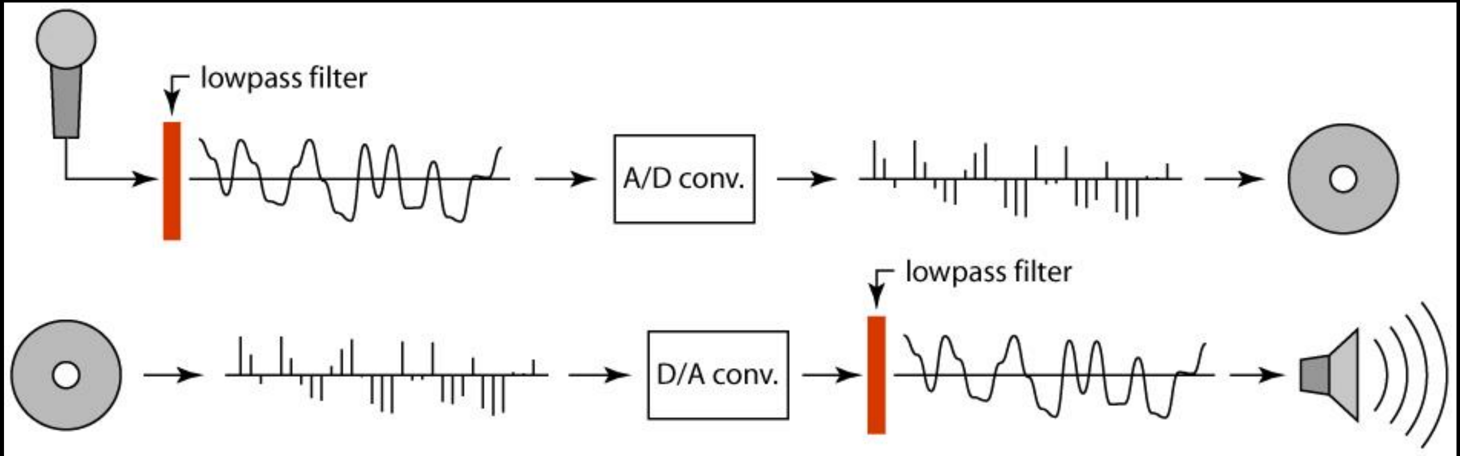
Alias!
Not enough samples

Antialiasing

- Sample more often
  - Join the Mega-Pixel craze of the photo industry
  - But this can’t go on forever
- Make the signal less “wiggly”
  - Get rid of some high frequencies
  - Will loose information
  - But it’s better than aliasing

Preventing aliasing

- Introduce *lowpass* filters:
  - remove high frequencies leaving only safe, low frequencies to be reconstructed



S. Marschner

(Anti)Aliasing in the Frequency Domain

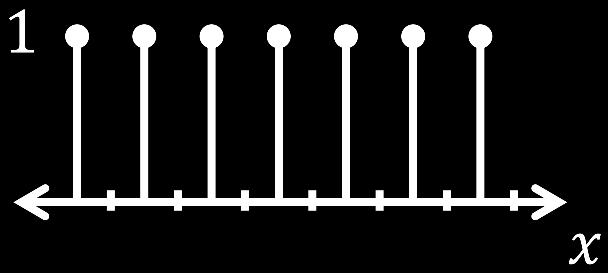
Impulse Train

Define a *comb* function (impulse train) in 1D as follows

$$comb_M[x] = \sum_{k=-\infty}^{\infty} \delta[x - kM]$$

where *M* is an integer

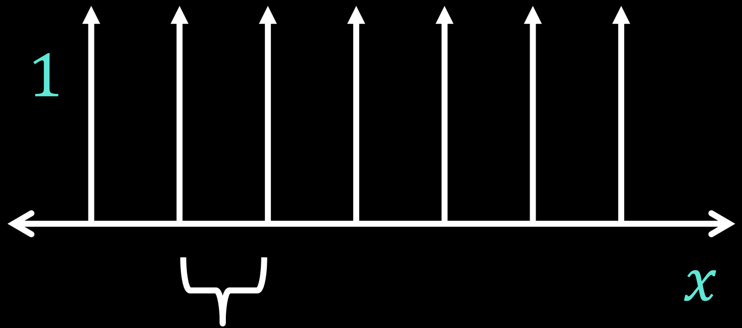
$comb_2[x]$



B.K. Gunturk

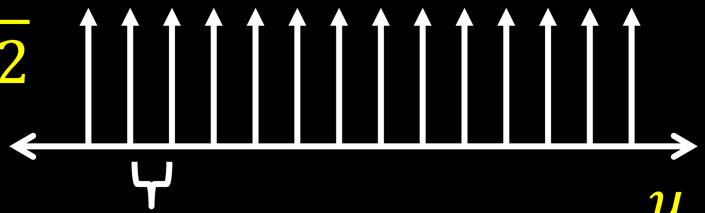
FT of Impulse Train in 1D

$$comb_2(x)$$



$$\longleftrightarrow$$

$$\frac{1}{2} comb_{\frac{1}{2}}(u)$$



Remember:

Scaling $f(ax)$ $\quad \frac{1}{|a|} F\left(\frac{u}{a}\right)$

B.K. Gunturk

Impulse Train in 2D *(bed of nails)*

$$comb_{M,N}(x,y) \equiv \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} \delta(x-kM, y-lN)$$

FT of Impulse Train in 2D (bed of nails)

- Fourier Transform of an impulse train is also an impulse train:

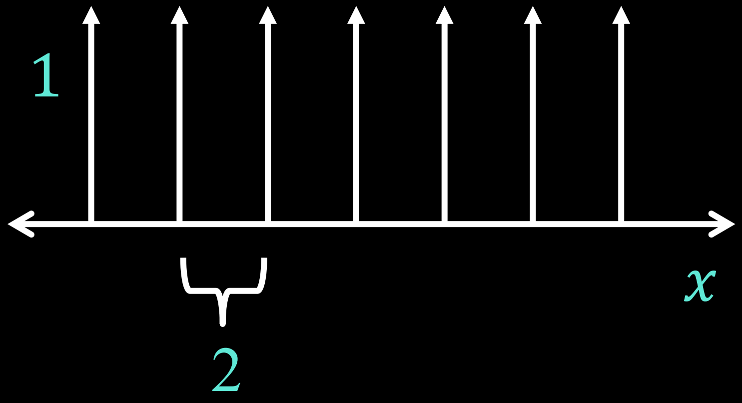
$$\underbrace{\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty}\delta(x-kM,\, y-lN)}_{comb_{M,N}(x,y)} \Leftrightarrow \underbrace{\frac{1}{MN}\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty}\delta\left(u-\frac{k}{M},\, v-\frac{l}{N}\right)}_{comb_{\frac{1}{M},\frac{1}{N}}(u,v)}$$

As the comb samples get further apart, the spectrum samples get closer together!

B.K. Gunturk

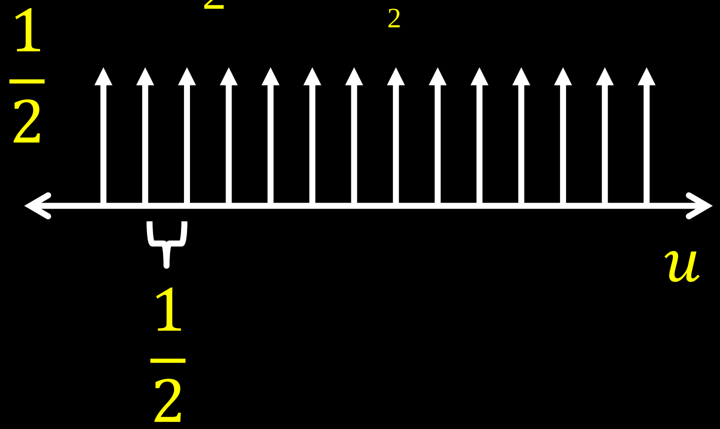
FT Impulse Train in 1D

$comb_2(x)$



$\leftrightarrow$

$\frac{1}{2} comb_{\frac{1}{2}}(u)$

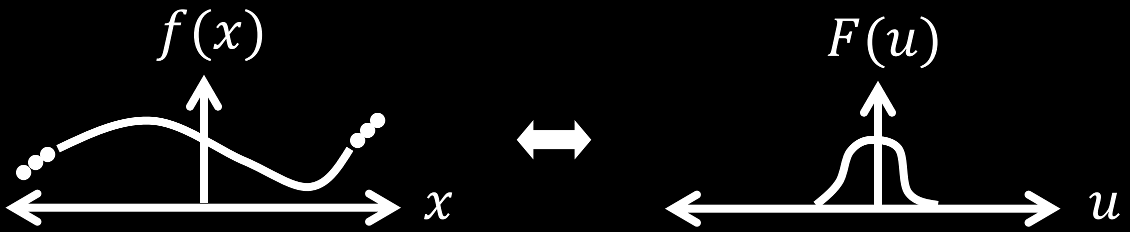


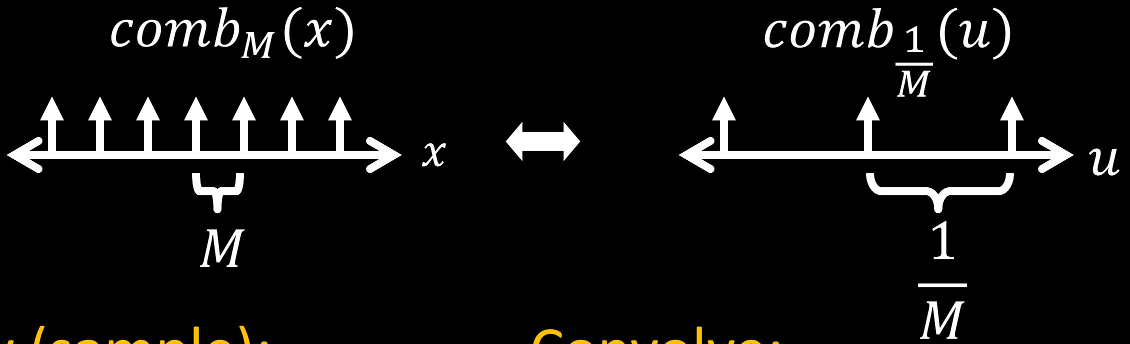
Remember:

Scaling $f(ax)$ $\qquad \frac{1}{|a|} F\left(\frac{u}{a}\right)$

B.K. Gunturk

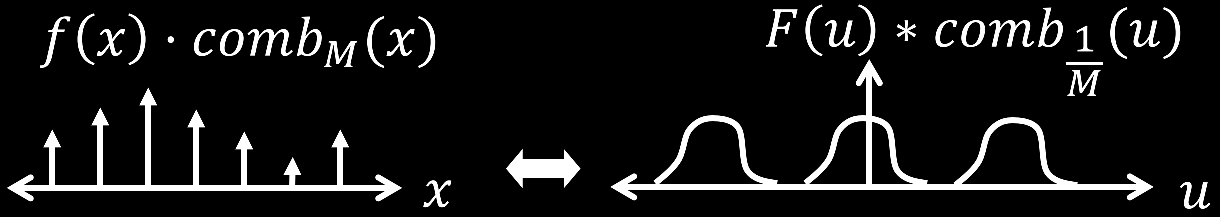
Sampling low frequency signal



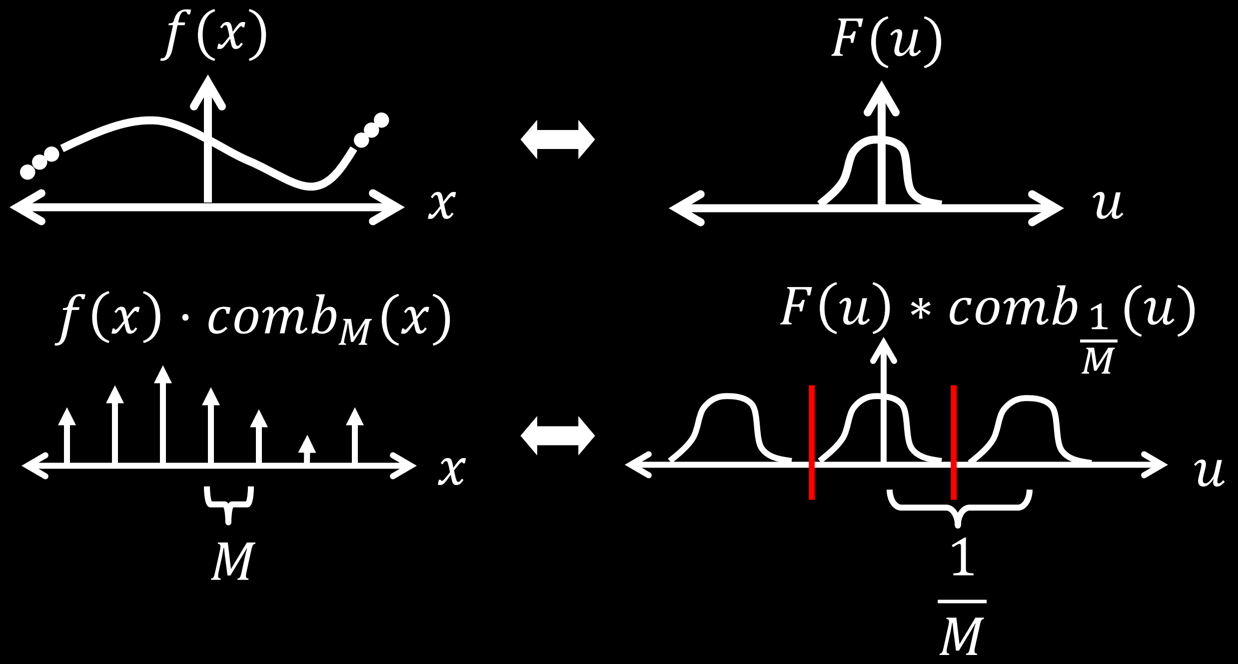


Multiply (sample):

Convolve:



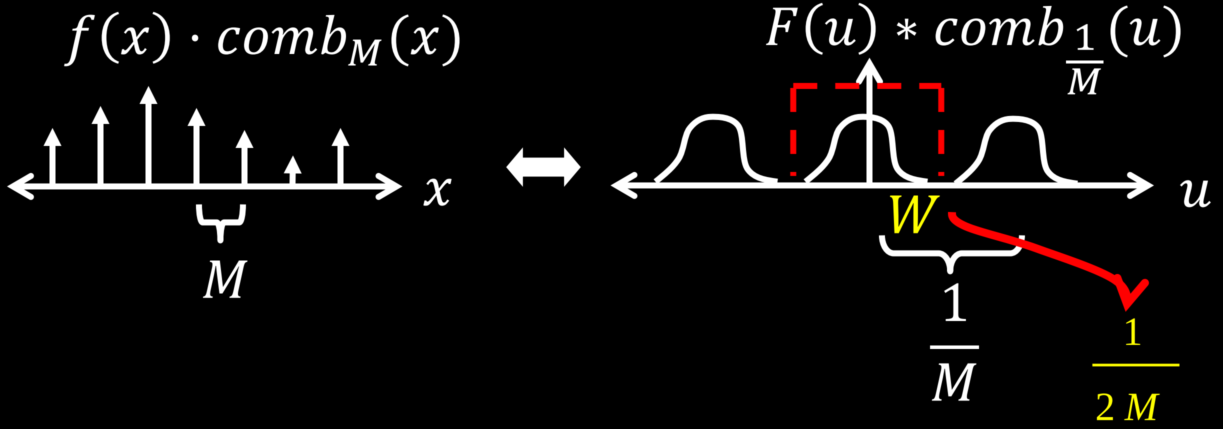
B.K. Gunturk



No "problem" if the maximum frequency of the signal is "small enough"

B.K. Gunturk

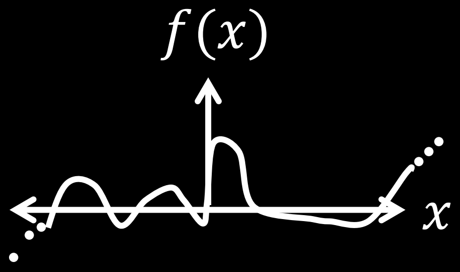
Sampling low frequency signal

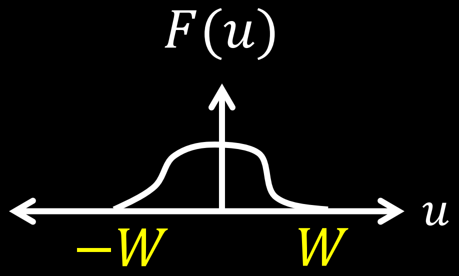


If there is no overlap, $W < \frac{1}{2M}$*, the original signal can be recovered from its samples by low-pass filtering.*

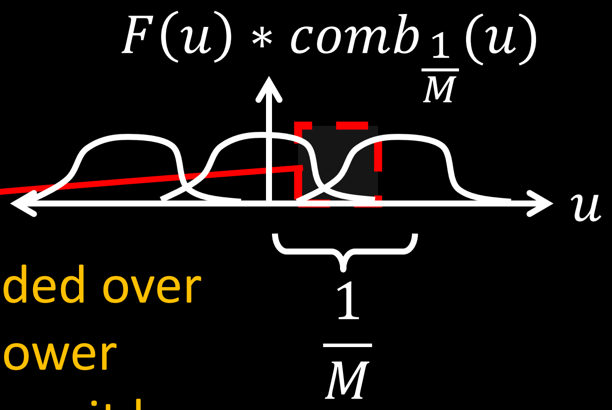
B.K. Gunturk

Sampling high frequency signal



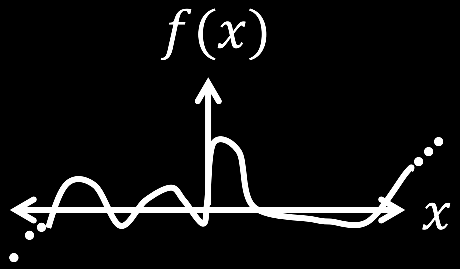


$$< f(x) \cdot comb_M(x) >$$

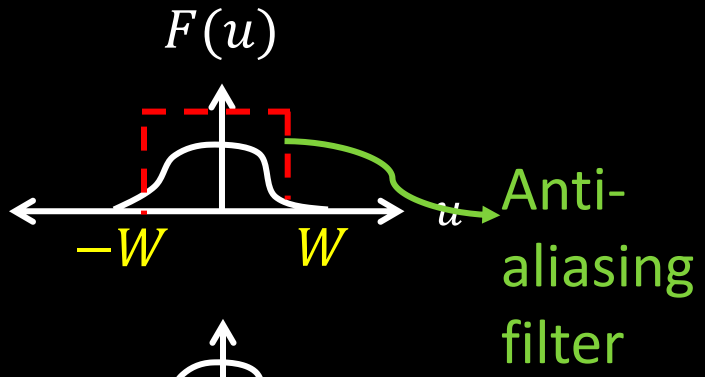


Overlap: The high frequency energy is folded over into low frequency. It is "aliasing" as lower frequency energy. And you cannot fix it once it has happened.

Sampling high frequency signal



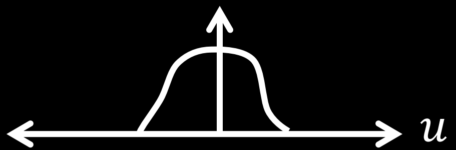
$\Leftrightarrow$



$$f(x) * h(x)$$

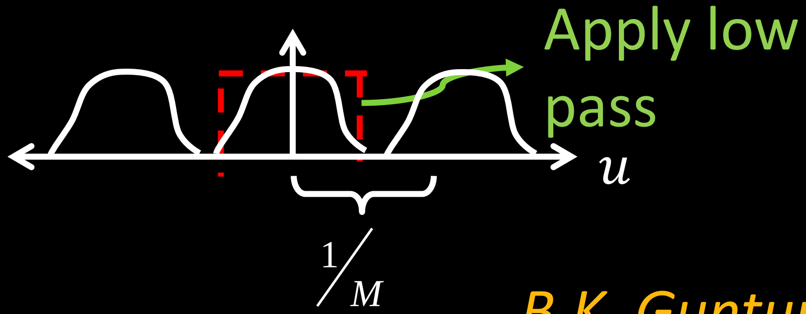
Anti-aliasing filter

$\Leftrightarrow$



$$[f(x) * h(x)]comb_M(x)$$

$\Leftrightarrow$

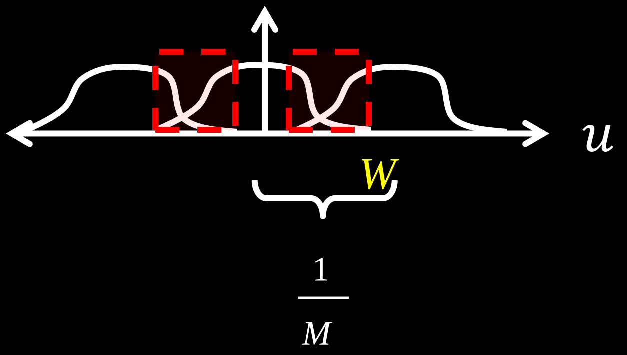


B.K. Gunturk

Sampling high frequency signal

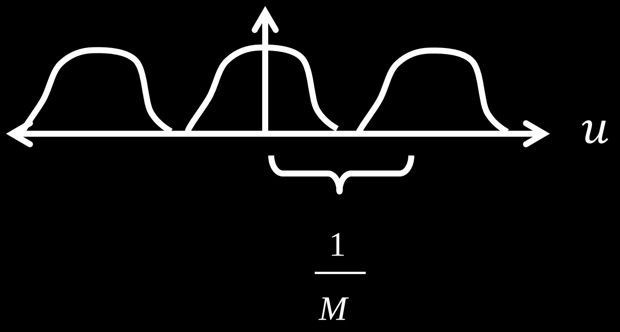
Without anti-aliasing filter:

$$f(x)\,comb_M(x) \leftrightarrow$$



With anti-aliasing filter:

$$[f(x) * h(x)]\,comb_M(x) \leftrightarrow$$



B.K. Gunturk

Aliasing in Images

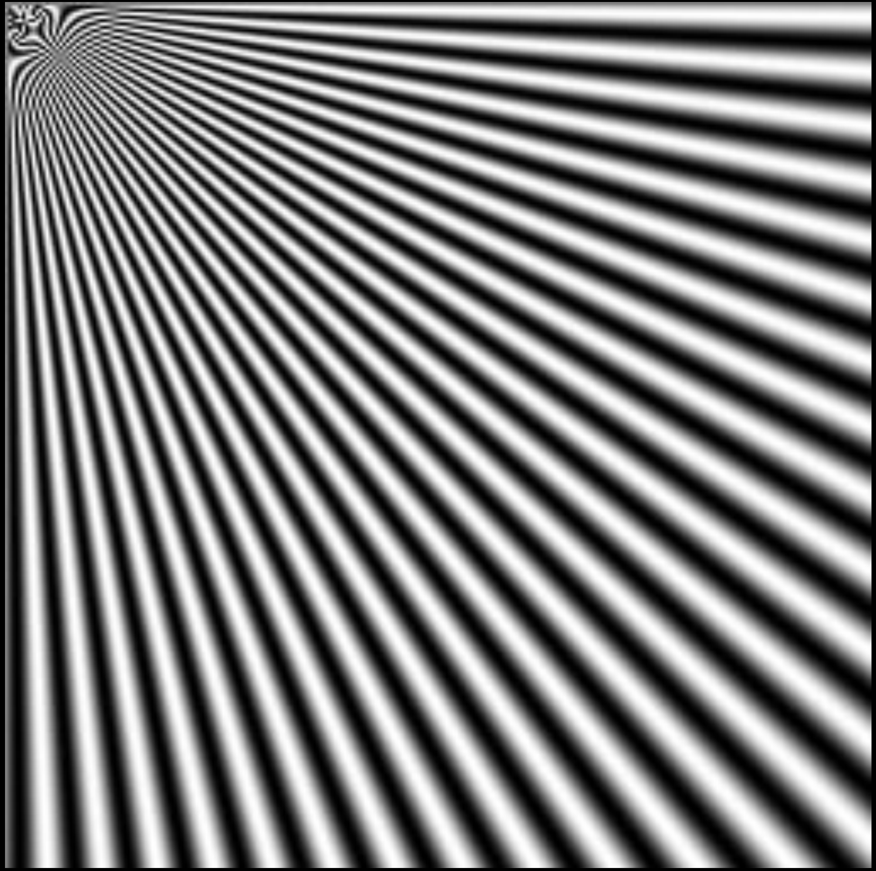
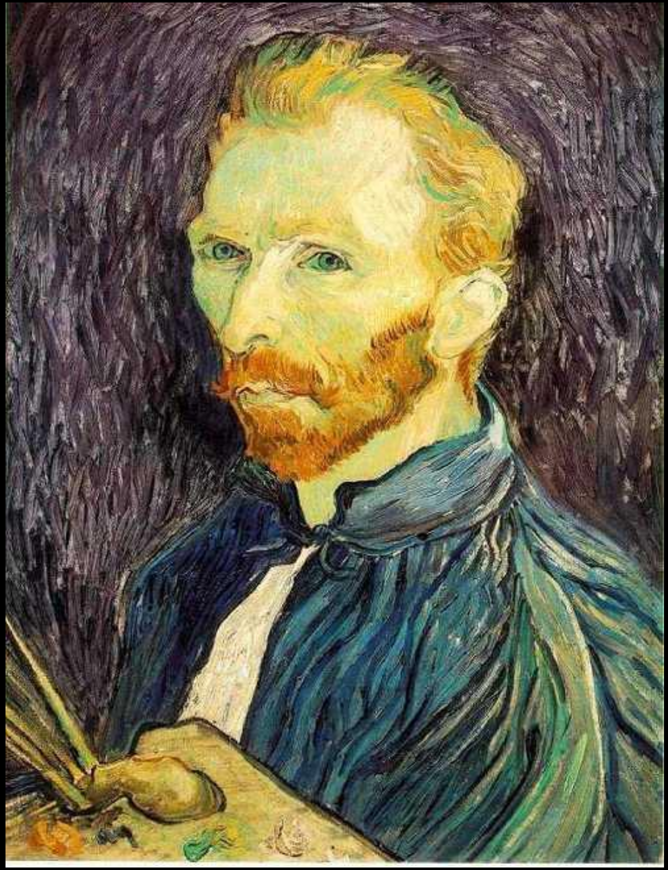


Image half-sizing

Suppose this image is too big to fit on the screen.

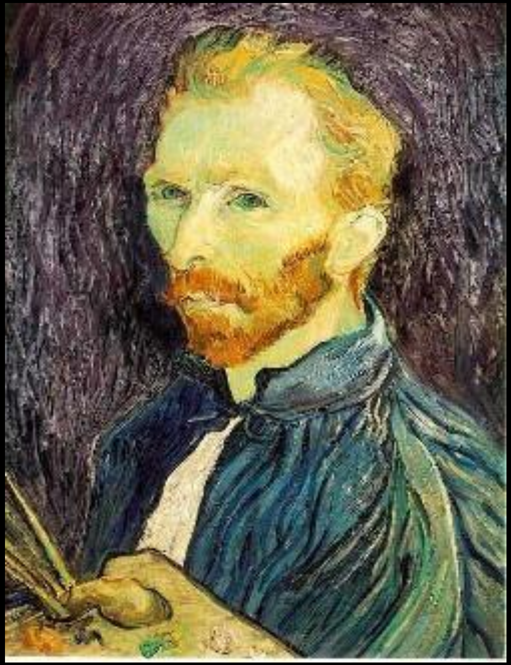
- How can we reduce it e.g. generate a half-sized version?



S. Seitz

Image sub-sampling

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



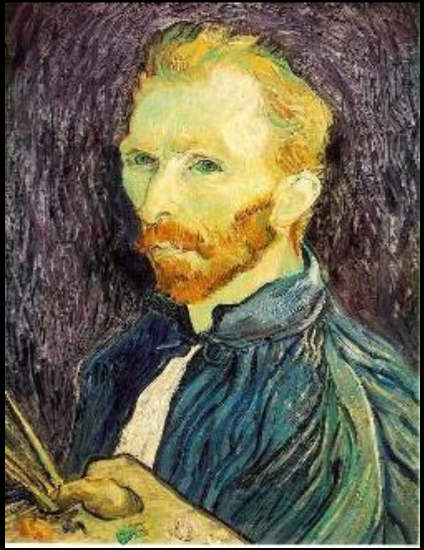
1/2

1/4

1/8

S. Seitz

Image sub-sampling



1/2

1/4 (2x zoom)

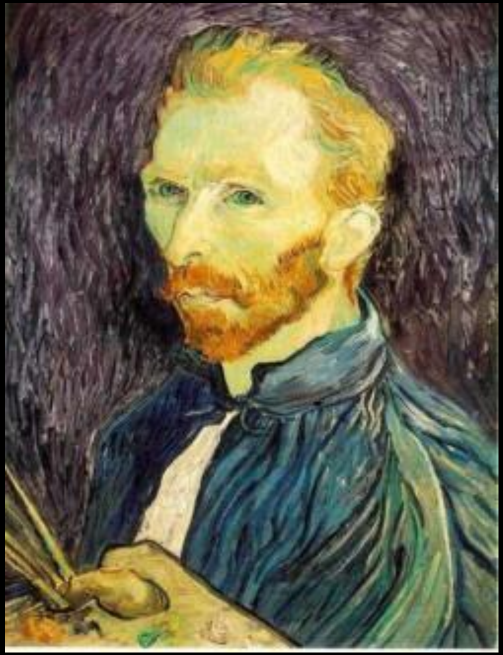
1/8 (4x zoom)

Aliasing! What do we do?

S. Seitz

Gaussian (lowpass) pre-filtering

Solution: *filter* the image, *then* subsample



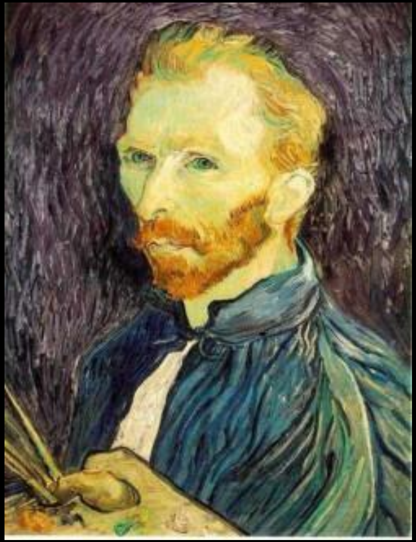
Gaussian 1/2

G 1/4

G 1/8

S. Seitz

Subsampling with Gaussian pre-filtering



Gaussian 1/2

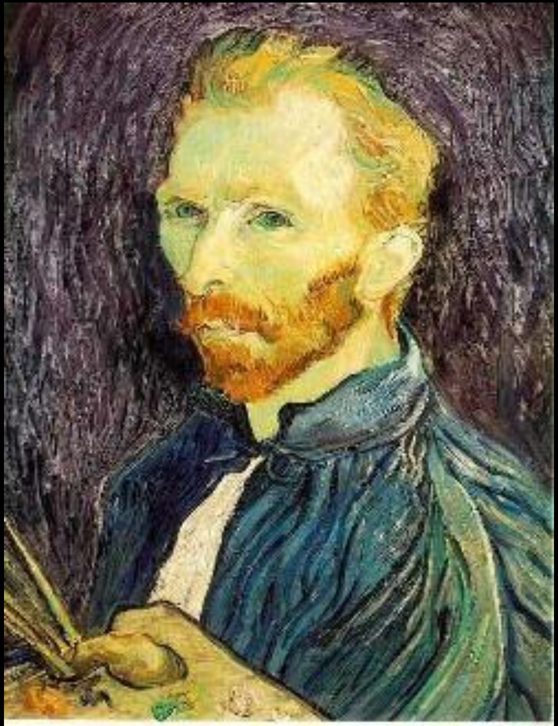
G 1/4

G 1/8

S. Seitz

Compare with...

Original

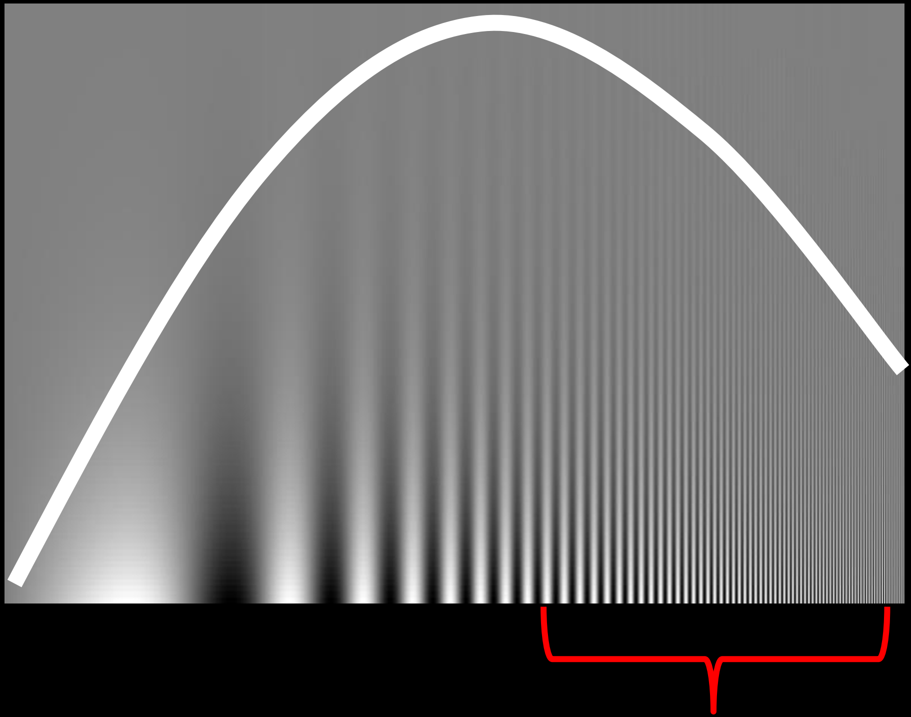


G 1/8 (4x zoom)

Subsample 1/8 (4x zoom)

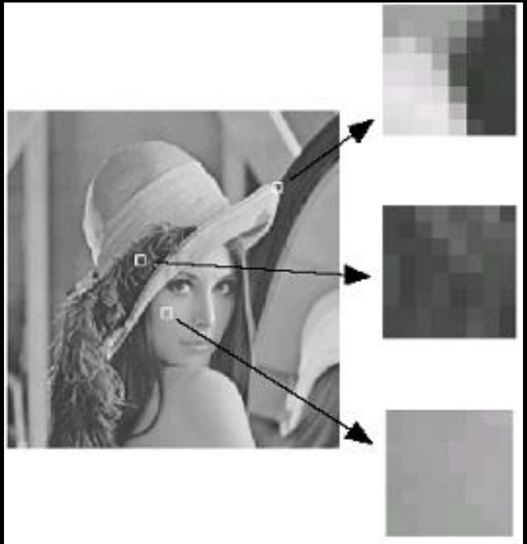
S. Seitz

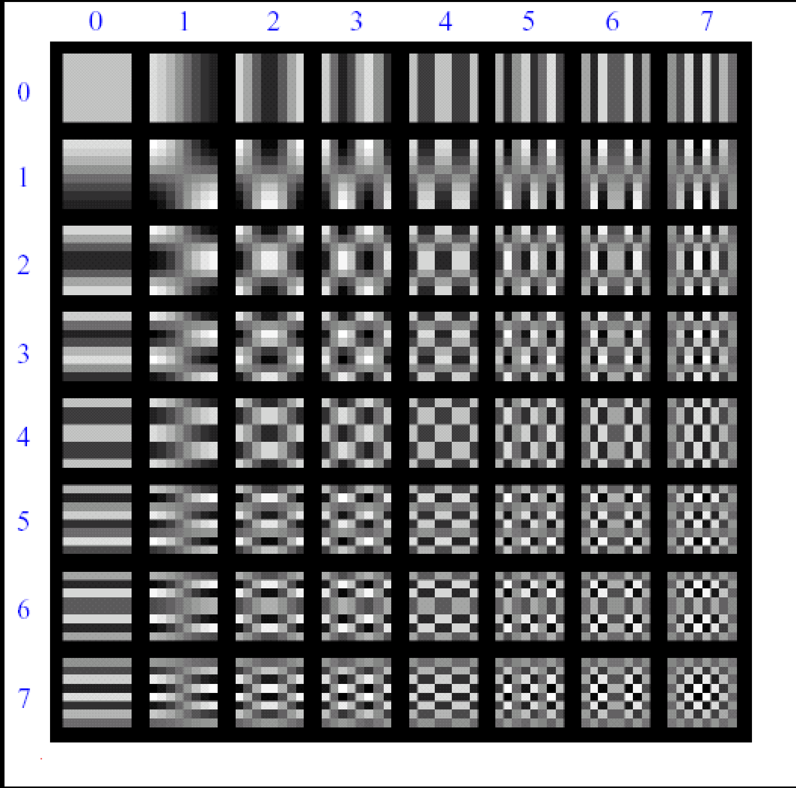
Campbell-Robson contrast sensitivity curve



The higher the frequency the less sensitive human visual system is…

Lossy Image Compression (JPEG)





Block-based Discrete Cosine Transform (DCT) on 8x8

Using DCT in JPEG

- The first coefficient $B(0,0)$ is the DC component, the average intensity
- The top-left coeffs represent low frequencies, the bottom right – high frequencies

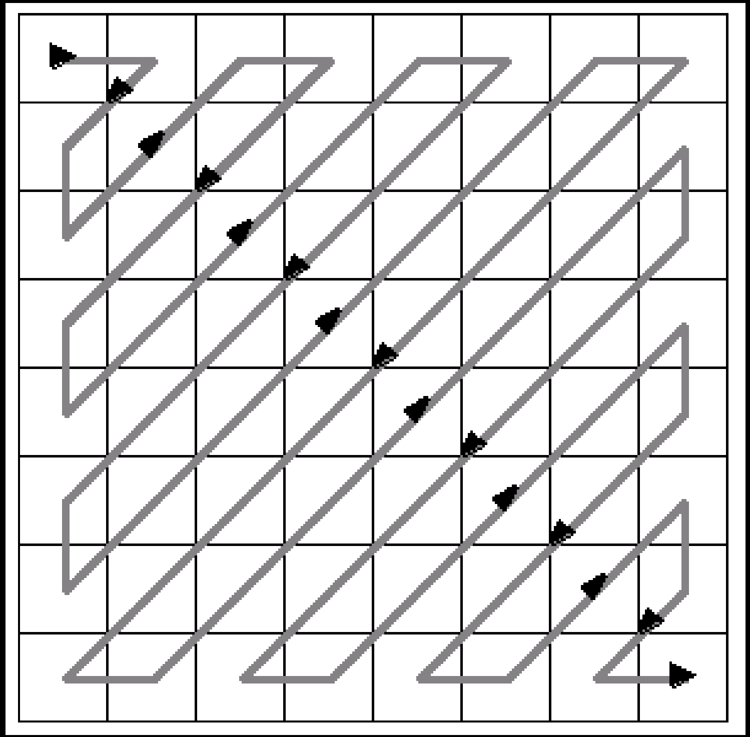


Image compression using DCT

- DCT enables image compression by concentrating most image information in the low frequencies

Quantization Table

3	5	7	9	11	13	15	17
5	7	9	11	13	15	17	19
7	9	11	13	15	17	19	21
9	11	13	15	17	19	21	23
11	13	15	17	19	21	23	25
13	15	17	19	21	23	25	27
15	17	19	21	23	25	27	29
17	19	21	23	25	27	29	31

Image compression using DCT

- Lose unimportant image info (high frequencies) by cutting $B(u,v)$ at bottom right
- The decoder computes the inverse DCT – IDCT

Quantization Table

3	5	7	9	11	13	15	17
5	7	9	11	13	15	17	19
7	9	11	13	15	17	19	21
9	11	13	15	17	19	21	23
11	13	15	17	19	21	23	25
13	15	17	19	21	23	25	27
15	17	19	21	23	25	27	29
17	19	21	23	25	27	29	31

JPEG compression comparison

89k

12k