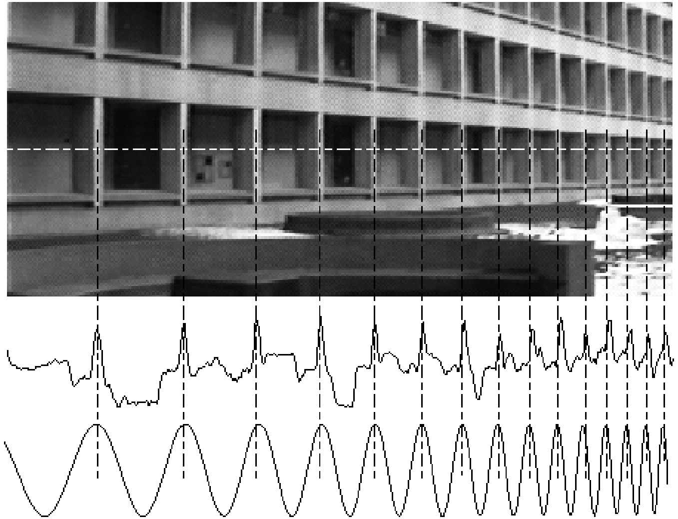
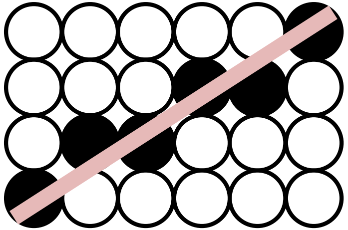


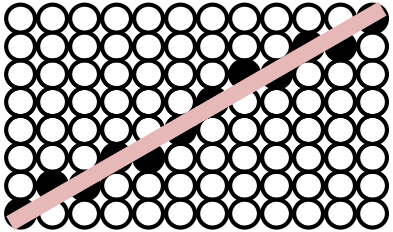
Lecture 2: Image Processing & Antialiasing

Representing lines: Point sampling, single pixel

- Midpoint algorithm: in each column, pick the pixel with the closest center to the line
  - A form of point sampling: sample the line at each of the integer X values
  - Pick a single pixel to represent the line's intensity, full on or full off
- Doubling resolution in x and y only lessens the problem, but costs 4 times memory, bandwidth, and scan conversion time!



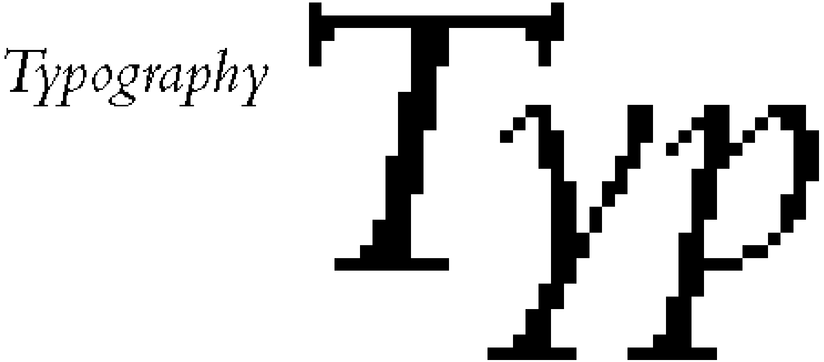
Line approximation using point sampling



Approximating same line at 2x the resolution

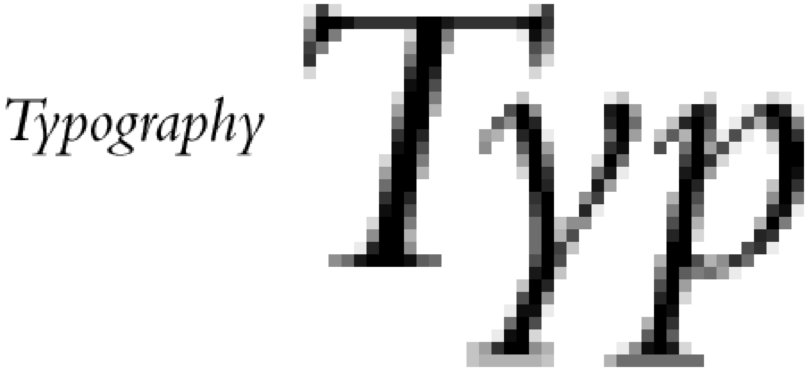
Jaggies & Aliasing

- "Jaggies" an informal name for artifacts from poorly representing continuous geometry by a discrete 2D grid of pixels
  - Jaggies are a manifestation of sampling error and loss of information (aliasing of high frequency components by low frequency ones)



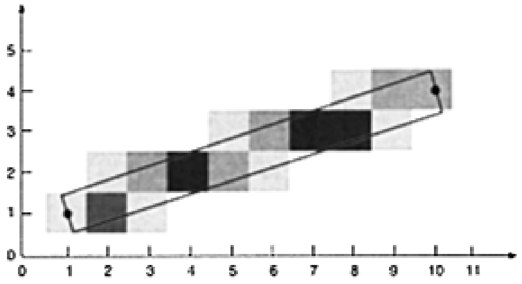
Jaggies & Aliasing

- Effect of jaggies can be reduced by anti-aliasing, which smoothes out the pixels around the jaggies by averaging
  - Shades of gray instead of sharp black/white transitions
  - Diminishes HVS' response to sharp transitions



Representing lines: Area sampling

- Represent the line as a unit width rectangle, use multiple pixels overlapping the rectangle (for now we think of pixels as squares)



- Instead of full on/off, calculate each pixel intensity proportional to the area covered by the unit rectangle

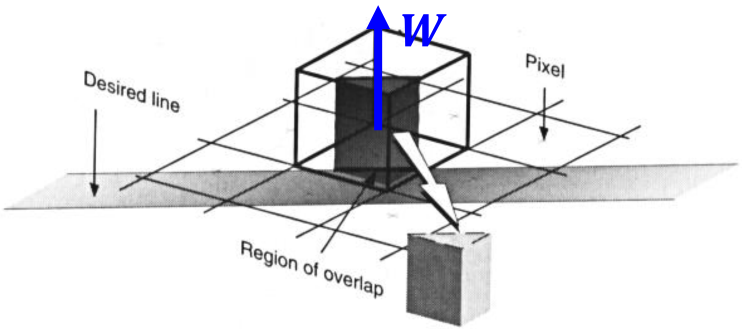
"Box Filter" Represents Unweighted Area Sampling

- For each pixel intersecting the line, intensity contributed by each sub-area of intersection dA is $\boldsymbol{W}(x, y)dA$
  - For box filer: $\boldsymbol{W}(x, y)$=1

- Then total intensity of the pixel (between 0 and 1) integrated over area of overlap is:

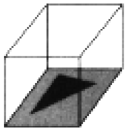
$$\int_A \boldsymbol{W}(x, y)dA$$

  - For box filter: total area of overlap



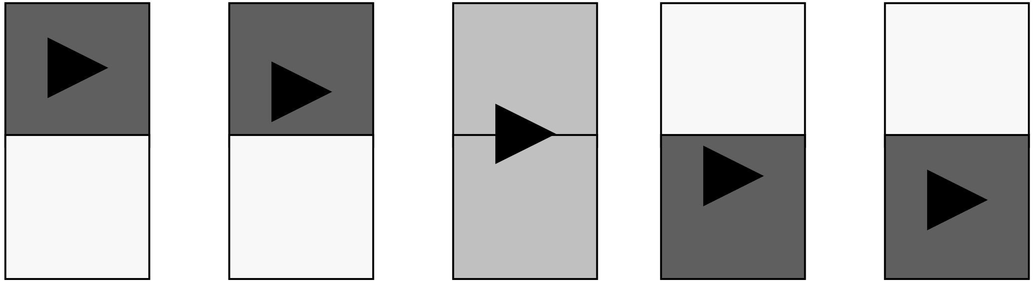
Unweighted Area Sampling

- Box filter
  - Local support: 1 pixel
    - No color unless the pixel overlaps with primitive
  - Unweighted integration
    - Intensity indifferent of the location of primitive in the pixel
    - Creating “winking” artifact when primitive moves across pixel boundaries



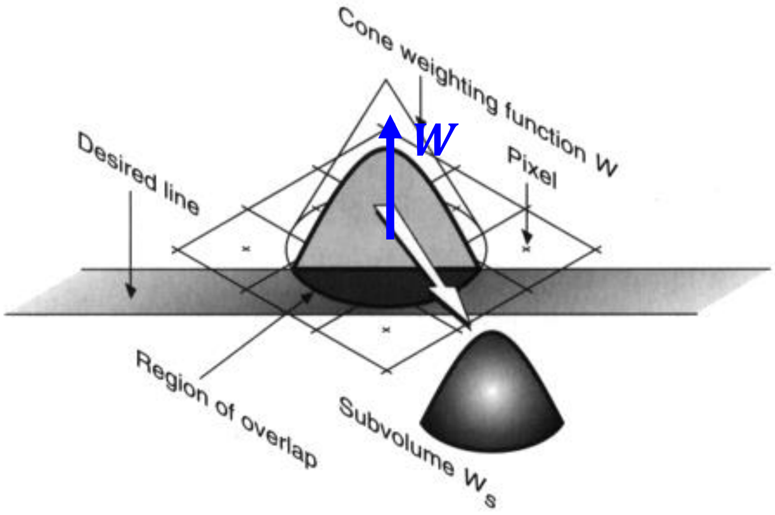
(a)

(b)



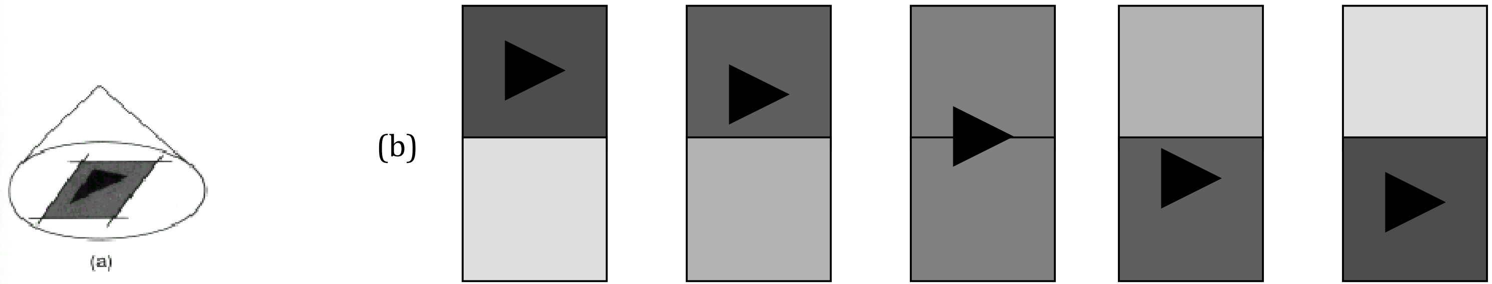
“Cone Filter” for Weighted Area Sampling

- Area sampling, but the overlap between filter and primitive is weighted so $\boldsymbol{W}(x, y)$ is greater when (x, y) gets closer to pixel center
- Cone has:
  - Linear falloff
  - Circular symmetry
  - Base width of 2
- Intensity of pixel is the “subvolume” inside the cone over the line (see picture)



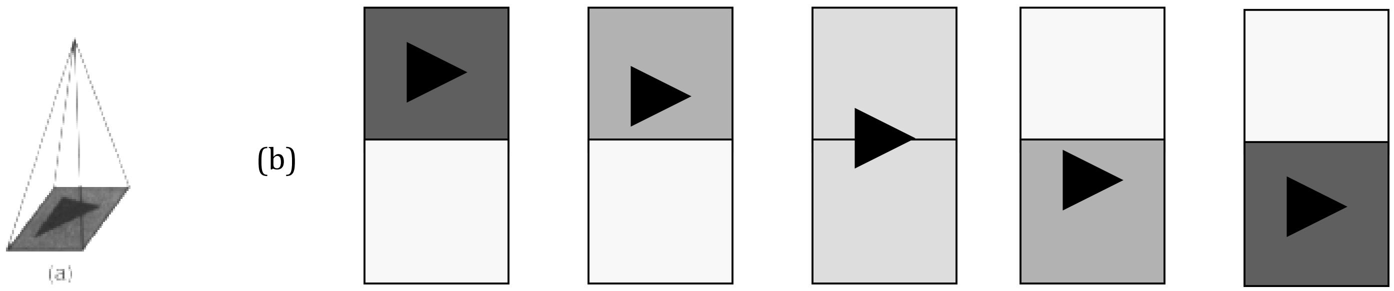
Weighted Area Sampling

- Cone filter
  - Greater support: 2 pixels
  - Greater smoothness in the changes of intensity



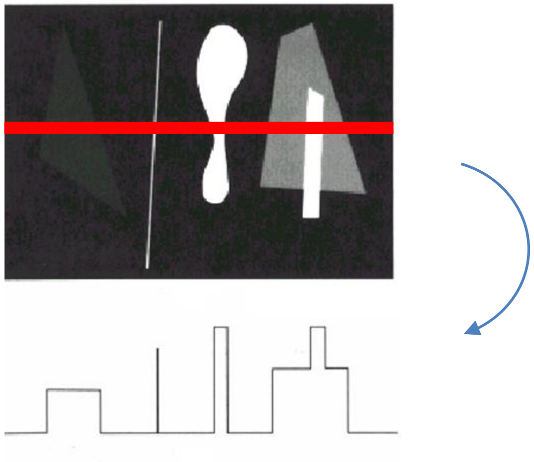
Weighted Area Sampling

- Pyramid filter
  - Support: 1 pixel
  - Approximates circular cone to emphasize area of overlap close to center of pixel

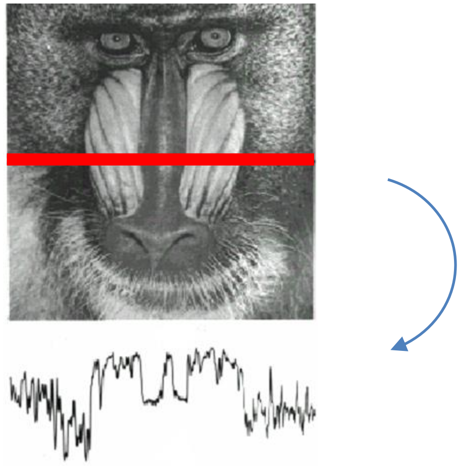


Sampling of Images

- Scan converting an image is digitizing (sampling) a series of continuous intensity functions, one per scan line
- We will use single scan lines for simplicity, but everything still applies to images

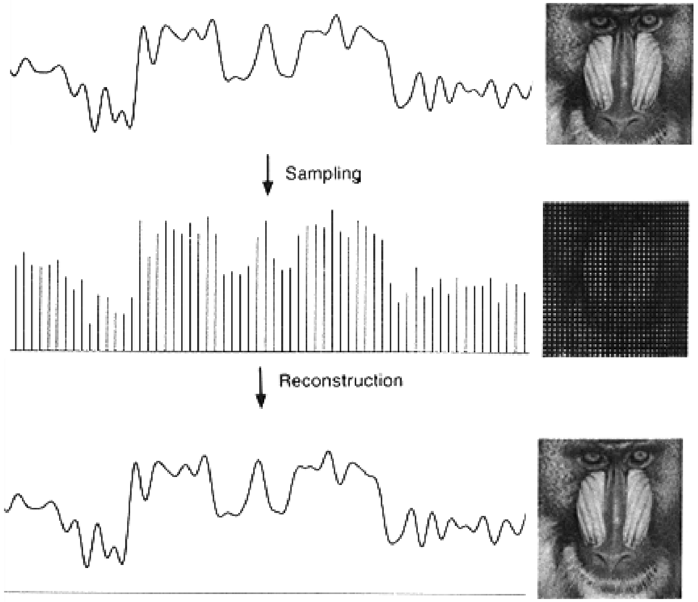


Scan line from synthetic scene



Scan line from natural scene

The Sampling/Reconstruction/Display Pipeline – overview

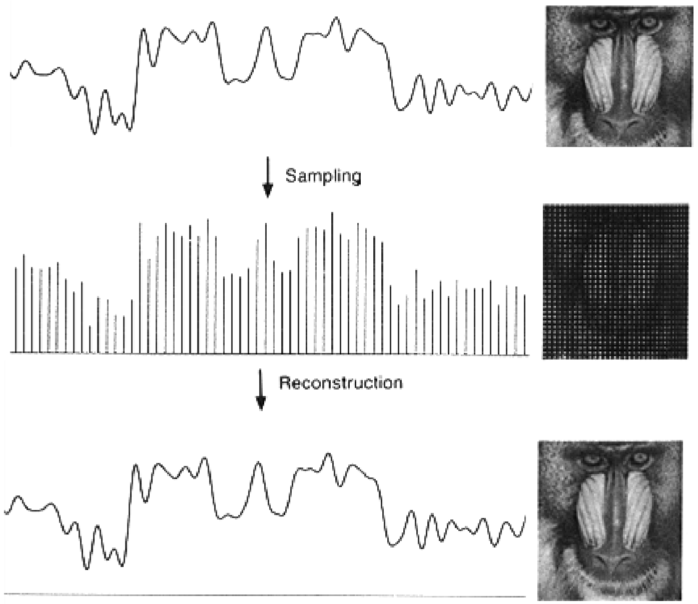


Original continuous signal:
$u : \mathbb{R} \rightarrow \mathbb{R}$

Sampled signal:
$S : \mathbb{Z} \rightarrow \mathbb{R} : n \longmapsto u(n)$

Reconstructed signal:
$\bar{S} : \mathbb{R} \rightarrow \mathbb{R}$
(many reconstruction methods)

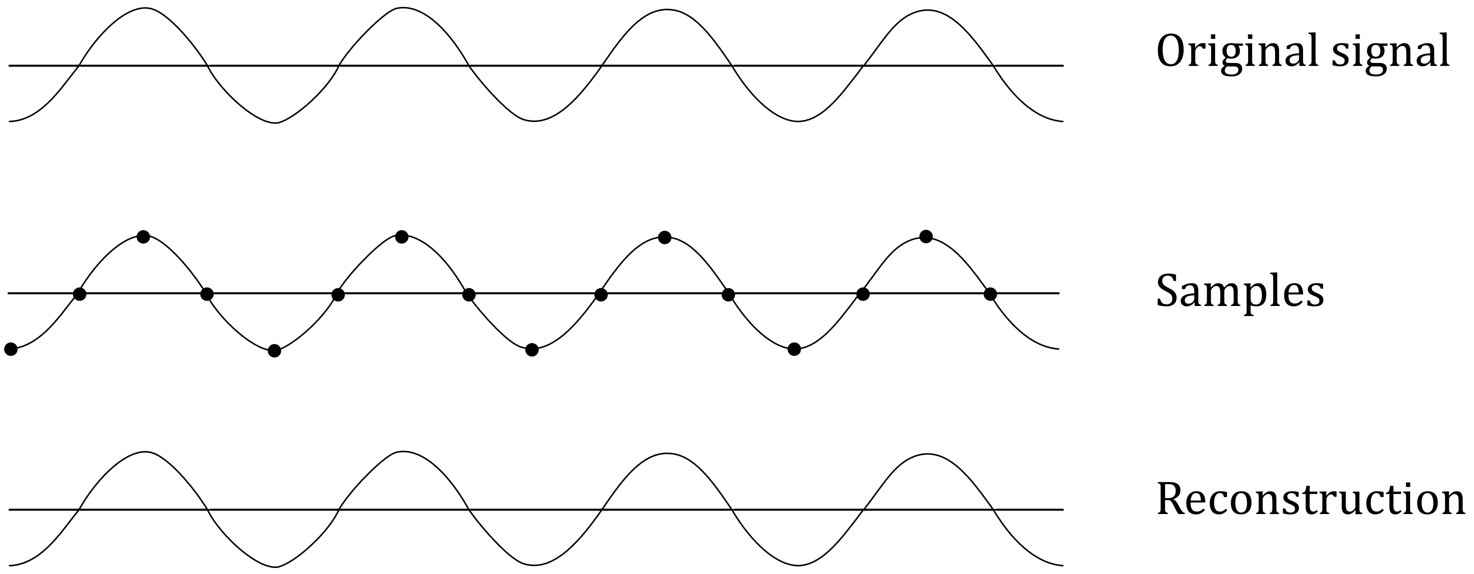
The Sampling/Reconstruction/Display Pipeline – overview



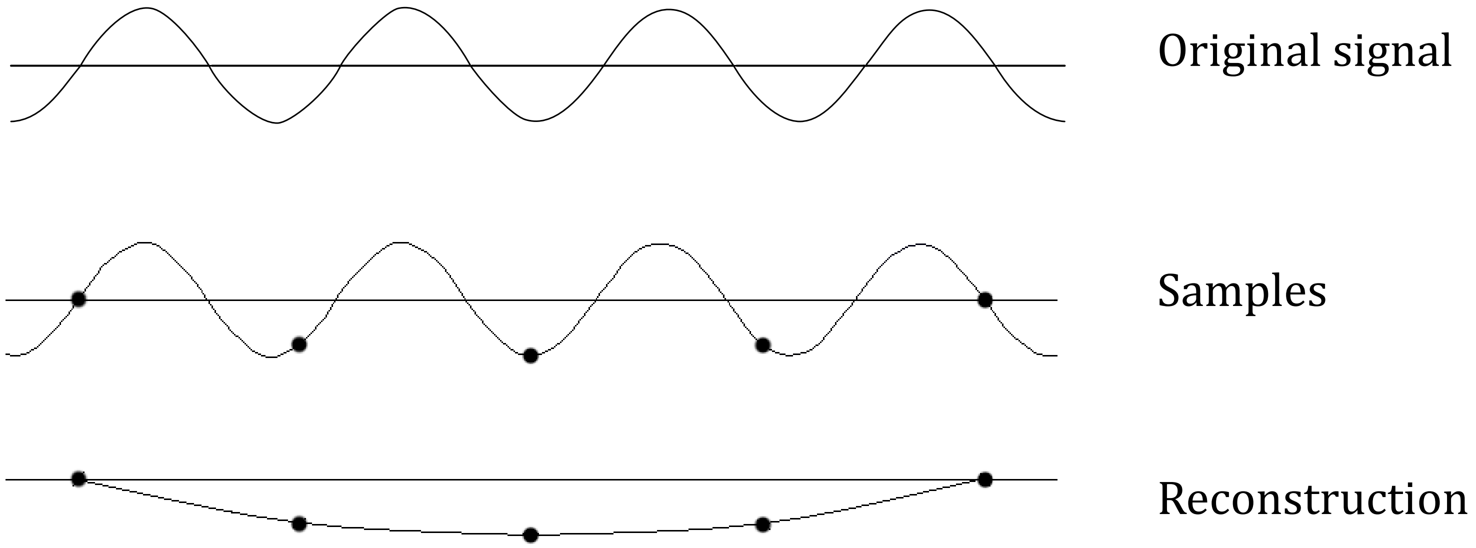
Intuitively, the samples we have, the more accurate is our reconstruction.

But how many samples are sufficient?

Sampling: The Nyquist Limit



Sampling: The Nyquist Limit



Sampling: The Nyquist Limit

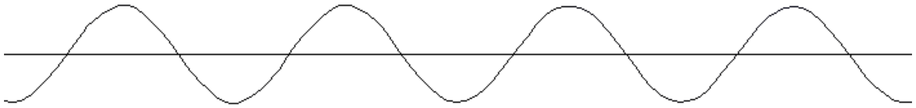
- Sampling frequency must be

  2 times more than

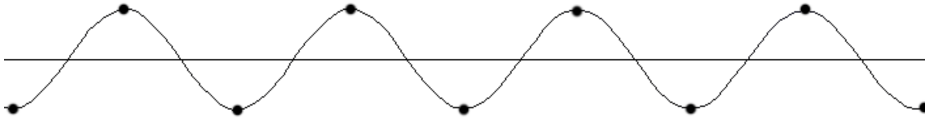
  the highest frequency in the signal (the Nyquist limit).

Sampling: The Nyquist Limit

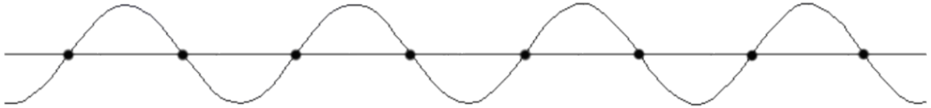
- Sampling right at the Nyquist limit can also be problematic:



Samples 1:



Samples 2:



Temporal Aliasing: Another Sampling Error

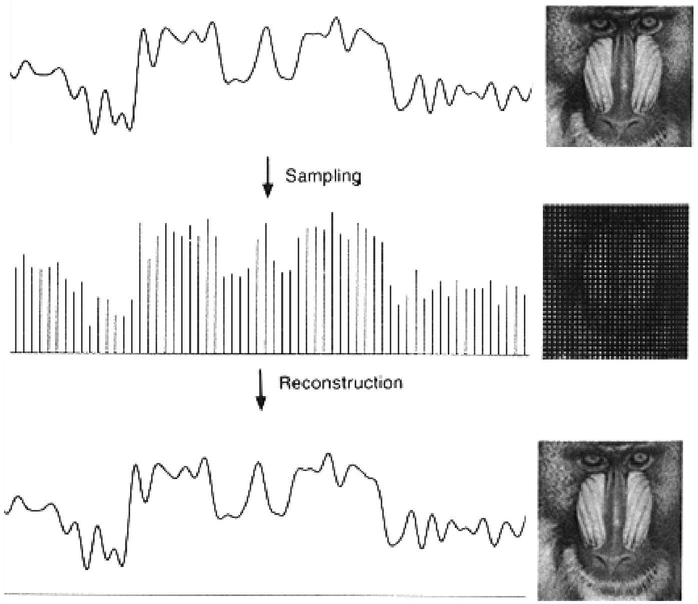
- Ever seen tires spin in a movie? Have you ever noticed that sometimes, they seem to be spinning backwards?

Temporal Aliasing: Another Sampling Error

- Ever seen tires spin in a movie? Have you ever noticed that sometimes, they seem to be spinning backwards?

- Its because the video frame-rate is lower than twice the frequency at which the wheels spin. This is temporal aliasing!

The Sampling/Reconstruction/Display Pipeline – overview



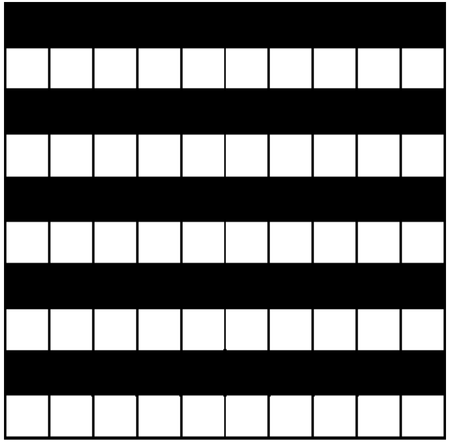
Intuitively, the samples we have, the more accurate is our reconstruction.

But how many samples are sufficient?

- 2x highest frequency

But what if the signal frequency is too high and we have a tight budget of samples?

Example Task: Down-sampling



Original Image

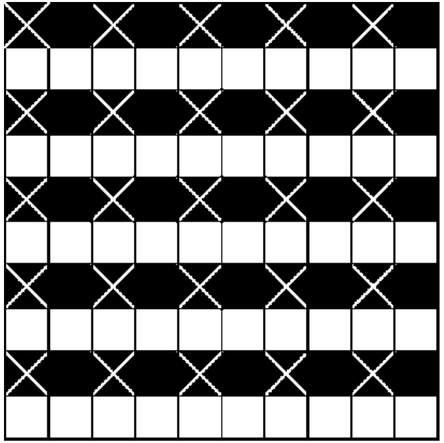


Image with sample points marked

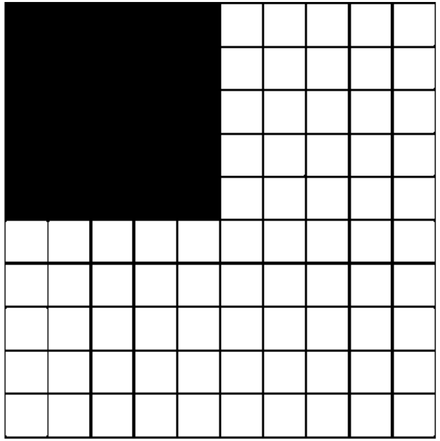
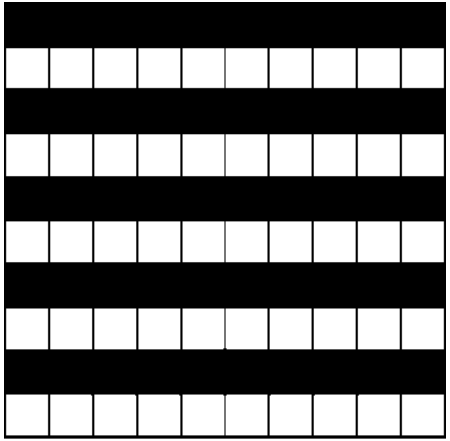


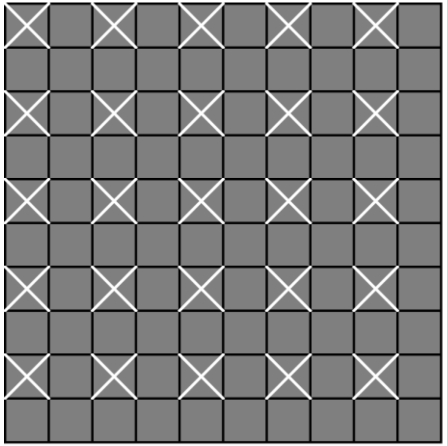
Image scaled using point samples

This doesn't look right at all. There are no stripes and the image now has a blacker average intensity

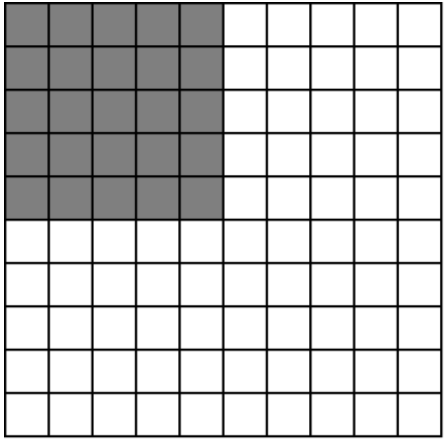
Pre-filtering (blurring), then down-sampling



Original Image



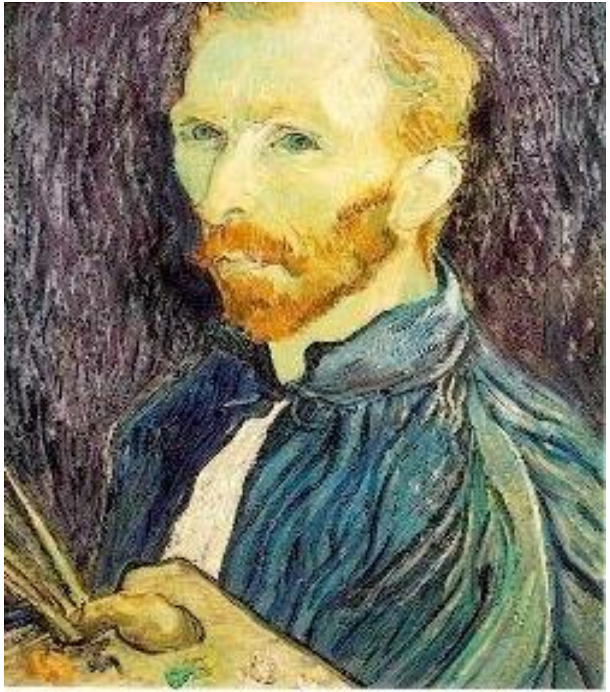
Prefiltered image with samples marked



Prefiltered image scaled

Remove the high frequency components, then sample

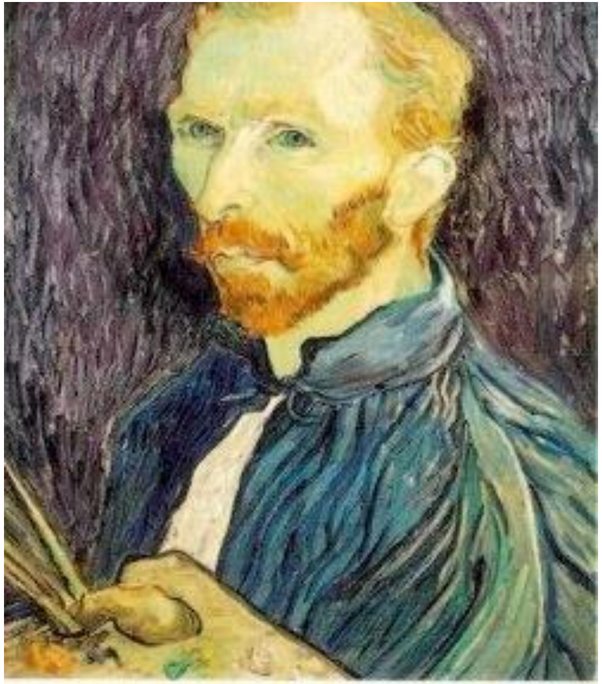
Without pre-filtering

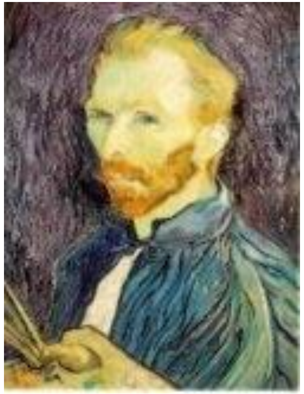


1/4

1/8

With pre-filtering





1/4

1/8

Low-Pass Filtering to Eliminate High Frequencies (shown for one scan line in Spatial Domain)

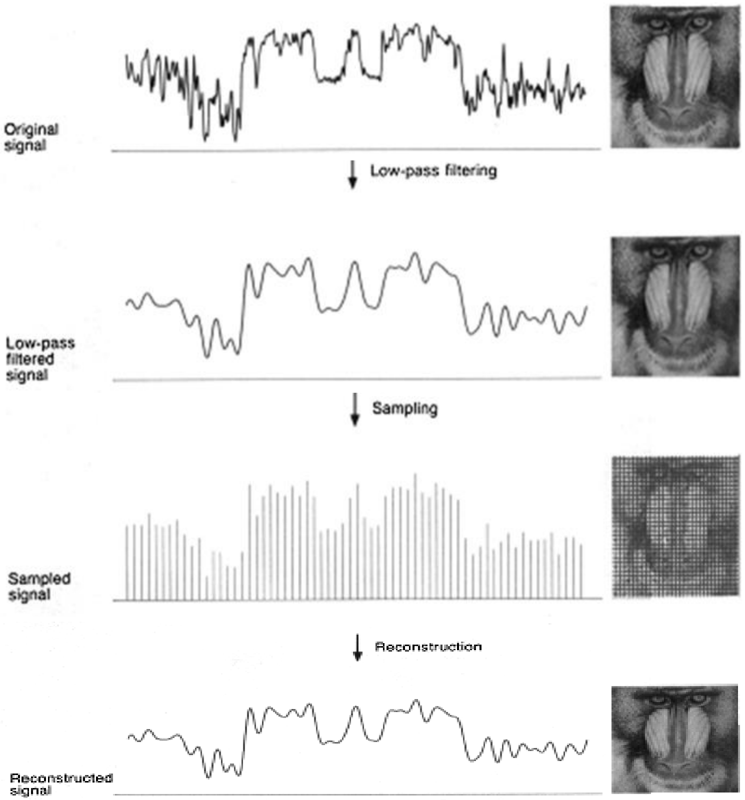


Fig. 14.20 The sampling pipeline with filtering. (Courtesy of George Wolberg, Columbia University.)

Discrete Convolution -- Review

- Think of an array as a function
- We take two arrays and generate a third
- We "slide" the filter along the other array and at each element, calculate a value by multiplying the pairs and summing the products to do the (weighted) averaging

Blurring by Convolution with Gaussian

0.05	0.1	0.05
0.1	0.4	0.1
0.05	0.1	0.05

Image

30	80	70	40	50	50	60
40	40	80	120	200	180	130
50	50	70	80	90	20	20
230	200	180	30	40	50	180
20	30	40	50	10	80	80
160	150	130	180	200	190	150
30	80	90	100	80	20	10

Blurred output

	62					

Blurring by Convolution with Gaussian

Image

30	80	70	40	50	50	60
40	40	80	120	200	180	130
50	50	70	80	90	20	20
230	200	180	30	40	50	180
20	30	40	50	10	80	80
160	150	130	180	200	190	150
30	80	90	100	80	20	10

0.05	0.1	0.05
0.1	0.4	0.1
0.05	0.1	0.05

Blurred output

	62	71				

Blurring by Convolution with Gaussian

Image

30	80	70	40	50	50	60
40	40	80	120	200	180	130
50	50	70	80	90	20	20
230	200	180	30	40	50	180
20	30	40	50	10	80	80
160	150	130	180	200	190	150
30	80	90	100	80	20	10

0.05	0.1	0.05
0.1	0.4	0.1
0.05	0.1	0.05

Blurred output

	62	71	102			

Blurring by Convolution with Gaussian

Image

30	80	70	40	50	50	60
40	40	80	120	200	180	130
50	50	70	80	90	20	20
230	200	180	30	40	50	180
20	30	40	50	10	80	80
160	150	130	180	200	190	150
30	80	90	100	80	20	10

0.05	0.1	0.05
0.1	0.4	0.1
0.05	0.1	0.05

Blurred output

	62	71	102	98		

Blurring by Convolution with Gaussian

(Values in the output are fake.)

Image

30	80	70	40	50	50	60
40	40	80	120	200	180	130
50	50	70	80	90	20	20
230	200	180	30	40	50	180
20	30	40	50	10	80	80
160	150	130	180	200	190	150
30	80	90	100	80	20	10

Blurred output

	62	71	102	98	50	
	20	10	10	70	0	
	20	-150	10	10	130	
	10	10	40	70	0	
	20	50	20	10	40	

Convolution for edge detection

Gradient along X

Gradient along Y

Convolution with
$[-0.5, 0, 0.5]$

Convolution with
$$\begin{bmatrix} -0.5 \\ 0 \\ 0.5 \end{bmatrix}$$

Gradient along X

Gradient along Y