Lecture 3:
Intro to Signal Processing: Sampling, Aliasing, Antialiasing

Computer Graphics and Imaging
UC Berkeley CS184/284A
Sampling is Ubiquitous in Computer Graphics and Imaging
Video = Sample Time

Harold Edgerton Archive, MIT
Photograph = Sample Image Sensor Plane
Rasterization = Sample 2D Positions
Ray Tracing = Sample Rays
Lighting Integrals: Sample Incident Angles
Sampling Artifacts in Graphics and Imaging
Wagon Wheel Illusion (False Motion)

Created by Jesse Mason, https://www.youtube.com/watch?v=QOwzkND_ooU
Moiré Patterns in Imaging

Read every sensor pixel  Skip odd rows and columns
Jaggies (Staircase Pattern)

This is also an example of “aliasing” – a sampling error
Jaggies (Staircase Pattern)

Retort by Don Mitchell
Sampling Artifacts in Computer Graphics

Artifacts due to sampling - “Aliasing”

• Jaggies – sampling in space
• Wagon wheel effect – sampling in time
• Moire – undersampling images (and texture maps)
• [Many more] ...

We notice this in fast-changing signals (high frequency), when we sample too slowly
Antialiasing Idea: Filter Out High Frequencies Before Sampling
Video: Point Sampling vs Antialiased Sampling in Time

Thin stream of water from kitchen tap

Point in Time
1/4000 sec exposure

Motion Blurred
1/60 sec exposure
Video: Point Sampling in Time

60 fps video. 1/4000 second exposure is sharp in time, causes time aliasing.
Video: Motion-Blurred (Antialiased) Sampling in Time

60 fps video. 1/60 second exposure is motion-blurred in time, no aliasing.

CS184/284A
Rasterization: Point Sampling in Space

Note jaggies in rasterized triangle where pixel values are pure red or white
Rasterization: Antialiased Sampling

Pre-Filter
(remove frequencies above Nyquist)

Sample

Note antialiased edges in rasterized triangle where pixel values take intermediate values
Point Sampling

One sample per pixel
Antialiasing
Point Sampling vs Antialiasing

Jaggies

Pre-Filtered
Antialiasing vs Blurred Aliasing

- Blurred Jaggies (Sample then filter)
- Pre-Filtered (Filter then sample)
This Lecture

Let’s dig into the fundamental reasons why this works
And look at how to implement antialiased rasterization
Frequency Space
Sines and Cosines

\[
\cos 2\pi x \\
\sin 2\pi x
\]
Frequencies \( \cos 2\pi f x \)

\[
f = \frac{1}{T}
\]

\( f = 1 \)

\( f = 2 \)

\( \cos 2\pi x \)

\( \cos 4\pi x \)
Fourier Transform

Represent a function as a weighted sum of sines and cosines

\[ f(x) = \frac{A}{2} + \frac{2A \cos(t\omega)}{\pi} - \frac{2A \cos(3t\omega)}{3\pi} + \frac{2A \cos(5t\omega)}{5\pi} - \frac{2A \cos(7t\omega)}{7\pi} + \ldots \]
Fourier Transform Decomposes A Signal Into Frequencies

\[ f(x) \quad F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i \omega x} \, dx \quad F(\omega) \]

**spatial domain** \[ \rightarrow \] **Fourier transform** \[ \rightarrow \] **frequency domain**

**Inverse transform**

\[ f(x) = \int_{-\infty}^{\infty} F(\omega) e^{2\pi i \omega x} \, d\omega \]

Recall \( e^{ix} = \cos x + i \sin x \)
Higher Frequencies Need Faster Sampling

Periodic sampling locations

$f_1(x)$

Low-frequency signal: sampled adequately for reasonable reconstruction

$f_2(x)$

$f_3(x)$

$f_4(x)$

$f_5(x)$

High-frequency signal is insufficiently sampled: reconstruction incorrectly appears to be from a low frequency signal
Undersampling Creates Frequency Aliases

High-frequency signal is insufficiently sampled: samples erroneously appear to be from a low-frequency signal.

Two frequencies that are indistinguishable at a given sampling rate are called “aliases”.
“Alias” = False Identity

“Batman” = Bruce Wayne’s alias to hide his true identity
Visualization of Frequency Space
2D Frequency Space

Spatial Domain

Frequency Domain

Note: Frequency domain also known as frequency space, Fourier domain, spectrum, ...
Constant

Spatial Domain

Frequency Domain

(0,0)
\[ \sin\left(\frac{2\pi}{32}\right)x \text{ — frequency } 1/32; \text{ 32 pixels per cycle} \]
\[
\sin\left(\frac{2\pi}{16}\right) x \quad \text{— frequency 1/16; 16 pixels per cycle}
\]
\[ \sin\left(\frac{2\pi}{16}\right)y \]
\[ \sin\left(\frac{2\pi}{32}\right)x \times \sin\left(\frac{2\pi}{16}\right)y \]
\[ \exp\left(-r^2/16^2\right) \]
$\exp(-r^2/32^2)$
\[
\exp\left(-\frac{x^2}{32^2}\right) \times \exp\left(-\frac{y^2}{16^2}\right)
\]
Rotate 45 $\exp\left(-\frac{x^2}{32^2}\right) \times \exp\left(-\frac{y^2}{16^2}\right)$
Filtering
Visualizing Image Frequency Content

Spatial Domain

Frequency Domain
Filter Out Low Frequencies Only (Edges)

Spatial Domain

Frequency Domain
Filter Out High Frequencies (Blur)

Spatial Domain

Frequency Domain
Filter Out Low and High Frequencies

Spatial Domain

Frequency Domain
Filter Out Low and High Frequencies

Spatial Domain  Frequency Domain
Filtering = Convolution
Convolution

Signal: 1 3 5 3 7 1 3 8 6 4

Filter: 1 2 1
Convolution

Signal

| 1 | 3 | 5 | 3 | 7 | 1 | 3 | 8 | 6 | 4 |

Filter

| 1 | 2 | 1 |

Result

| 12 |   |   |   |   |   |   |   |   |   |

$1 \times 1 + 3 \times 2 + 5 \times 1 = 12$
Convolution

Signal: 1 3 5 3 7 1 3 8 6 4

Filter: 1 2 1

Result: 12 16
Convolution

Signal: 1 3 5 3 7 1 3 8 6 4

Filter: 1 2 1

Result: 12 16 18

\[ 5 \times 1 + 3 \times 2 + 7 \times 1 = 18 \]
Convolution Theorem

Convolution in the spatial domain is equal to multiplication in the frequency domain, and vice versa.

Option 1:
- Filter by convolution in the spatial domain

Option 2:
- Transform to frequency domain (Fourier transform)
- Multiply by Fourier transform of convolution kernel
- Transform back to spatial domain (inverse Fourier)
Convolution Theorem

Spatial Domain

Fourier Transform

Frequency Domain

\[ * \] = \[ \times \] = \[ \text{Inv. Fourier Transform} \]
Box Filter

Example: 3x3 box filter
Box Function = “Low Pass” Filter

Spatial Domain

Frequency Domain
Wider Filter Kernel = Lower Frequencies

Spatial Domain  Frequency Domain
Wider Filter Kernel $= $ Lower Frequencies

As a filter is localized in the spatial domain, it spreads out in frequency domain.

Conversely, as a filter is localized in frequency domain, it spreads out in the spatial domain
Efficiency?

When is it faster to implement a filter by convolution in the spatial domain?

When is it faster to implement a filter by multiplication in the frequency domain?
Nyquist Frequency & Antialiasing
Nyquist Theorem

Theorem: We get no aliasing from frequencies in the signal that are less than the Nyquist frequency (which is defined as half the sampling frequency) *

* Won’t cover proof in course, see Shannon sampling theorem
Signal vs Nyquist Frequency: Example

\[ \sin\left(\frac{2\pi}{32}\right)x \] — frequency 1/32; 32 pixels per cycle

Max signal freq = 1/32

Nyquist freq. of sampling = 1/2 * 1/16 = 1/32

No Aliasing!
Signal vs Nyquist Frequency: Example

\[ \sin\left(\frac{2\pi}{16}\right)x \] — frequency 1/16; 16 pixels per cycle

Max signal freq = 1/16

Nyquist freq. of sampling = 1/2 * 1/16 = 1/32

Sampling = every 16 pixels

Aliasing!
Signal vs Nyquist Frequency: Example

\[ \sin\left(\frac{2\pi}{16}\right)x \] — frequency 1/16; 16 pixels per cycle

Max signal freq = 1/16

Nyquist freq. of sampling = \(\frac{1}{2} \times \frac{1}{8} = \frac{1}{16}\)

No Aliasing!
Visual Example:
Image Frequencies &
Nyquist Frequency
Image Frequency: Visual Example

In the following image sequence:

- Image is 512x512 pixels
- We will progressively blur the image, see how the frequency spectrum shrinks, and see what the maximum frequency is
Image Frequency: Visual Example

Spatial Domain

Frequency Domain
Image Frequency: Visual Example

Spatial Domain

Frequency Domain

Max signal freq ≈ 1/2
(Why 1/2?)
Image Frequency: Visual Example

Spatial Domain

Frequency Domain

Max signal freq $\approx 1/4$
Image Frequency: Visual Example

Spatial Domain

Frequency Domain

Max signal freq ≈ 1/8
Image Frequency: Visual Example

Spatial Domain

Frequency Domain

Max signal freq ≈ 1/16
Image Frequency: Visual Example

Spatial Domain

Frequency Domain

Max signal freq ≈ 1/32
Image Frequency: Visual Example

Spatial Domain

Frequency Domain

Max signal freq ≈ 1/64

CS184/284A
Nyquist Frequency: Visual Example

In next sequence:

• Visualize sampling an image every 16 pixels

• Visualize when image is blurred enough that image frequencies match Nyquist frequency (no aliasing)
Nyquist Frequency: Visual Example

Spatial Domain

Frequency Domain

Max signal freq $\approx 1/2$
Nyquist Frequency: Visual Example

Spatial Domain

Sampling = every 16 pixels

Frequency Domain

Max signal freq ≈ 1/2

Nyq. freq = 1/32
Nyquist Frequency: Visual Example

Spatial Domain

Sampling = every 16 pixels

Frequency Domain

Max signal freq $\approx 1/4$

Nyq. freq = 1/32
Nyquist Frequency: Visual Example

Spatial Domain

Frequency Domain

Max signal freq ≈ 1/8

Nyq. freq = 1/32

sampling = every 16 pixels
Nyquist Frequency: Visual Example

Spatial Domain

Frequency Domain

Max signal freq ≈ 1/16

Nyq. freq = 1/32

sampling = every 16 pixels
Nyquist Frequency: Visual Example

Spatial Domain

Sampling = every 16 pixels

Frequency Domain

Max signal freq ≈ 1/32

Nyq. freq = 1/32
Nyquist Frequency: Visual Example

Spatial Domain

frequency = every 16 pixels

Frequency Domain

Max signal freq $\approx 1/64$

Nyq. freq = $1/32$
Nyquist Frequency: Visual Example

Recap:

- Filter (blur) original image to reduce maximum signal frequency
- Create low-resolution image by sampling only every 16 pixels
  - (Sampling frequency is 1/16, and Nyquist frequency is 1/32)

Aliasing         Which do you prefer?         Overblurring

1 2 3 4 5 6
Nyquist Frequency: Visual Example

Recap:

• Filter (blur) original image to reduce maximum signal frequency
• Create low-resolution image by sampling only every 16 pixels
  • (Sampling frequency is 1/16, and Nyquist frequency is 1/32)

Aliasing and overblurring can be objectionable even at small image sizes

Which do you prefer?
Antialiasing
Reminder: Nyquist Theorem

Theorem: We get no aliasing from frequencies in the signal that are less than the Nyquist frequency (which is defined as half the sampling frequency).

Consequence: sampling at twice the highest frequency in the signal will eliminate aliasing.
How Can We Reduce Aliasing Error?

Increase sampling rate (increase Nyquist frequency)

- Higher resolution displays, sensors, framebuffers…
- But: costly & may need very high resolution

Antialiasing

- Simple idea: remove (or reduce) signal frequencies above the Nyquist frequency before sampling
- How? Filter out high frequencies before sampling.
Regular Sampling

Note jaggies in rasterized triangle where pixel values are pure red or white
Antialiased Sampling

Pre-Filter
(remove frequencies above Nyquist)

Sample

Note antialiased edges in rasterized triangle where pixel values take intermediate values
A Practical Pre-Filter

A 1 pixel-width box filter will attenuate frequencies whose period is less than or equal to 1 pixel-width.

This is practical to implement — why?
Antialiasing By Averaging Values in Pixel Area

Convince yourself the following are the same:

Option 1:
- Convolve \( f(x,y) \) by a 1-pixel box-blur
- Then sample at every pixel

Option 2:
- Compute the average value of \( f(x,y) \) in the pixel
Antialiasing by Computing Average Pixel Value

In rasterizing one triangle, the average value inside a pixel area of $f(x,y) = \text{inside(triangle,x,y)}$ is equal to the area of the pixel covered by the triangle.
Antialiasing By Supersampling
Supersampling

We can approximate the effect of the 1-pixel box filter by sampling multiple locations within a pixel and averaging their values:

4x4 supersampling
Point Sampling: One Sample Per Pixel
Supersampling: Step 1

Take $N \times N$ samples in each pixel.

2x2 supersampling
Supersampling: Step 2

Average the N\times N samples “inside” each pixel.
Supersampling: Step 2

Average the NxN samples “inside” each pixel.
Supersampling: Step 2

Average the N\times N samples “inside” each pixel.
Supersampling: Result

This is the corresponding signal emitted by the display
Point Sampling

One sample per pixel
4x4 Supersampling + Downsampling

Pixel value is average of 4x4 samples per pixel
Antialiasing By Supersampling - Summary

• Antialiasing = remove frequencies above Nyquist before sampling
• We can attenuate these frequencies quite well with a 1-pixel box filter (convolution)
• We approximated the 1-pixel box sampling by supersampling and averaging
• Simple, good idea - high image quality, but costly
• May feel “right”, but can get even higher quality!
Supersampling Implementation Tips
Tip 1: Sample Locations

Regular sampling: sample location for pixel \((i,j)\)

\[(i+1/2, j+1/2)\]
Tip 1: Sample Locations

2x2 supersampling: locations for pixel (i,j)
Tip 1: Sample Locations

Sample locations for NxN supersampling?
Tip 2: Supersampling Multiple Triangles

So far, we rasterized only a single triangle:

- Supersample
- Then average down

How should this change when we rasterize $N$ triangles in the same image?

- Supersample and average down each triangle, one by one?
- Or supersample all $N$ triangles onto a high-res grid, then average down?

What are the algorithmic implications?

- E.g. what is the minimum memory needed?
Note: There is Much, Much More To Sampling Theory & Practice!
Things to Remember

Signal processing key concepts:

- Frequency domain vs spatial domain
- Filters in the frequency domain scale frequencies
- Filters in the sampling domain = convolution

Sampling and aliasing

- Image generation involves sampling
- Nyquist frequency is half the sampling rate
- Frequencies above Nyquist appear as aliasing artifacts
- Antialiasing = filter out high frequencies before sampling
- Interpret supersampling as (approx) box pre-filter antialiasing
Acknowledgments

Thanks to Kayvon Fatahalian, Pat Hanrahan, Mark Pauly and Steve Marschner for slide resources.
Sampling Food for Thought
Off-Grid Sampling?
Random Sampling?
Use Samples “Outside” Pixel?
Non-Uniform Sample Weighting?

What weighting function $k(x,y)$?
What frequency spectrum would a desirable $k(x,y)$ have?
Sampling Stress Test: Zone Plate

\[ f(x,y) = \sin(x^2+y^2) \]

What should this look like?

Real signal
(low frequency oscillation)

Aliasing from undersampling
increasingly high frequencies

Figure credit: Pat Hanrahan and Bryce Summers