Announcements

- HW 3 out
  - Barycentric coordinates for Problem 1

Texture mapping from 0 to infinity

- When you go close...

When viewed from a distance

- Aliasing!

How does area map over distance?

- At optimal viewing distance:
  - One-to-one mapping between pixel area and texel area

- When closer
  - Each pixel is a small part of the texel

- When farther
  - Each pixel could include many texels

Minification: Theoretical Solution

- Find the area of pixel in texture space
- “Filter” the area to compute “average” texture color
  - Filtering eliminates high frequency artifacts
  - How to filter?
    - Analytically compute area
    - Super-sample
    - But too expensive
MIP Maps

- MIP Maps
  - Multum in Parvo: Much in little
  - Proposed by Lance Williams
  - Stores pre-filtered/averaged versions of texture
  - Supports very fast lookup

- Assumptions:
  - Can’t really precompute every possible required area
  - But can precompute some areas

Image Pyramid

Filtering by Averaging

- Each pixel in a level corresponds to 4 pixels in lower level
- Average
- Gaussian filtering (more on this next lecture)

Using the MIP Map

- Find the MIP Map level where the pixel has a 1-to-1 mapping
- How?
  - Find largest side of pixel footprint in texture space
  - Pick level where that side corresponds to a texel
  - Compute derivatives to find pixel footprint

Given derivatives: what is level?

\[ \text{level} = \log[\max(\frac{du}{dx}, \frac{dv}{dx}, \frac{du}{dy}, \frac{dv}{dy})] \]

\[ \text{level} = \log(\sqrt{\left(\frac{du}{dx}\right)^2 + \left(\frac{dv}{dx}\right)^2 + \left(\frac{du}{dy}\right)^2 + \left(\frac{dv}{dy}\right)^2}) \]

- Gradients
  - Available in pixel shader

Using the MIP Map

- In level, find texel and
- Return the texture value: point sampling
- Bilinear interpolation
- Trilinear interpolation
Memory Usage

- What happens to size of texture?

MIPMAP

- Multi-resolution image pyramid
  - Pre-sampled computation of MIPMAP
  - 1/3 more memory

- Bilinear or Trilinear interpolation

Filtered Texturing

- Point sampled
- Mipmapped

Filtered Texturing

- Point sampled
- Mipmapped
- Summed area tables

Some basic assumptions

- Assume that the pixel only maps to squares in texture space
- In fact, assume it maps to squares at particular locations

Sampling and Antialiasing
Sampled representations

- How to store and compute with continuous functions?
- Common scheme for representation: samples
  - write down the function's values at many points

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

Filtering

- Processing done on a function
  - can be executed in continuous form (e.g. analog circuit)
  - but can also be executed using sampled representation
- Simple example: smoothing by averaging

Roots of sampling

- Nyquist 1928; Shannon 1949
  - famous results in information theory
- 1940s: first practical uses in telecommunications
- 1960s: first digital audio systems
- 1970s: commercialization of digital audio
- 1982: introduction of the Compact Disc
  - the first high-profile consumer application
- This is why all the terminology has a communications or audio "flavor"
  - early applications are 1D; for us 2D (images) is important

Sampling in digital audio

- Recording: sound to analog to samples to disc
- Playback: disc to samples to analog to sound again
  - how can we be sure we are filling in the gaps correctly?

Undersampling

- What if we “missed” things between the samples?
- Simple example: undersampling a sine wave
  - unsurprising result: information is lost
  - surprising result: indistinguishable from lower frequency
  - also was always indistinguishable from higher frequencies
  - aliasing: signals "traveling in disguise" as other frequencies
Preventing aliasing

- Introduce lowpass filters:
  - remove high frequencies leaving only safe, low frequencies
  - choose lowest frequency in reconstruction (disambiguate)