Texturing

CS559 – Fall 2017
Lecture 21
November 13 2017
Before fragment shading

Color per triangle
Color per vertex

Interpolation

How to get complexity?
   Lots of Triangles?
Why not lots of triangles?

Hard to model / author / design

Hard to draw fast

Hard to sample

(triangles get smaller than a pixel)

Hard to maintain the models

Hard to store the models
The pipeline (1988)
The full fixed-function pipeline (1992)
Per-Pixel Coloring

Have bigger triangles
Change color per pixel (fragment)

Change other stuff
  normals
  positions
Paint Pictures on Triangles

Assuming you have hardware
  Easier (can use 2D tools, photographs)
  Less to store
  Less to model
  Faster to draw
  Easier to sample
Why not paint on triangles?

Things really aren’t flat

Parallax / self-shadowing / illumination effects

More advanced “texturing” to get these later
Texture Mapping

A function (map) of pixels to colors (later other properties)

Two big pieces:
1. Need to “name” the pixels
   Texture Coordinates
2. Need to describe the functions
What coordinates to use

Screen Coordinates?
World Coordinates?
Local Coordinates?
Triangle Coordinates?

User-defined, convenient coordinates?
Checkerboard example

Screen space checkers
World-space checkers
Triangle-space checkers

Texture Coordinates
Texture Coordinates

Assigned to a vertex
Independent of position

Can be 1D, 2D, 3D, 4D
  just a number/vector to use in function
For a triangle...

Specify texture coordinates at vertices
Interpolate within the triangle
Barycentric Coordinates

An alternate way of describing points in triangles
These can be used to interpolate texture coordinates

Gives the same result as previous slide
Method in textbook (Shirley)

\[
\begin{align*}
x &= x_1 + x_2 + x_3 \\
&= \frac{\text{Area}(x, x_2, x_3)}{\text{Area}(x_1, x_2, x_3)} \\
&= \frac{\text{Area}(x_1, x, x_3)}{\text{Area}(x_1, x_2, x_3)} \\
&= 1
\end{align*}
\]
The usual case

Triangles
2D texture coordinates

Draw the picture – 2D map
How to define the function

\[ f(u,v) \rightarrow r,g,b \]

Procedurally – write a program
   easy now that we have shaders
Image-based – use a picture
   look up the colors
Image-Based Textures

Texture Mapping often means this but it is a specific case

Very convenient – since we can paint, photograph, pre-compute, ...
Tricky Part

A (triangle) Pixel might map to...
  Much smaller than a (picture) pixel
  Much bigger than a (picture) pixel

(diagram)
Minification and Magnification
Sampling / Aliasing Issue
Point Sampling

Look up U,V in the grid

Nearest Neighbor
Interpolation

Minification Case
Magnification Case (diagrams)
Is this only a problem with images?

Aliasing in procedural textures

Checkerboard(u,v) =
(int(u) + int(v)) \% 2 ? Black : White

Zoom out? (400\% - every 4^{th} pixel)
Zoom in?
Anti-Aliasing Procedural Textures

It’s Hard.
Back to the image case...

Each pixel in the result (square? circle?)
comes from
Some shape in the source image
stretched by texture coordinates
distorted by perspective

Filtering – average over this region
MIP Map

- Repeatedly halve the image to make a “pyramid”
  - Until there’s 1 pixel (which is average of whole)
- Given a position and square size
  - Use square size to pick pyramid level
  - Use bilinear interpolation to get position

- But only have pyramid for 1, 2, 4, 8… pixel squares
  - Linear interpolate between levels!
  - E.g. 5 = ¼ way between 4 and 8, so compute 4 and 8 and interpolate
  - Tri-Linear Interpolation! - looks at 8 texels (4 per level)
Programming This in GL

A lot of steps, lots can go wrong

Need to load in an image and get to graphics card memory
Need to build the MIP MAP
Getting Your Object Ready

Need texture coordinates your shader can use anything

How does lighting combine with texture?
Accessing the texture map from a shader

Concept of a **Sampler**

Special variable type that encapsulates lookup into an image
Lots of parameters to set up

What Image
What kind of filter for minification
What kinds of filter for magnification
How to deal with texture boundaries
  Wrap? Mirror? Clamp?
Texture Unit (?)
  connect sampler to hardware