Subdivision & Texture Mapping

15-462
Project 2

Outline

- Project Overview
- Subdivision
  - Loop Subdivision
  - Adjacency Data Structure
- Texture Mapping
  - OpenGL

Project Overview

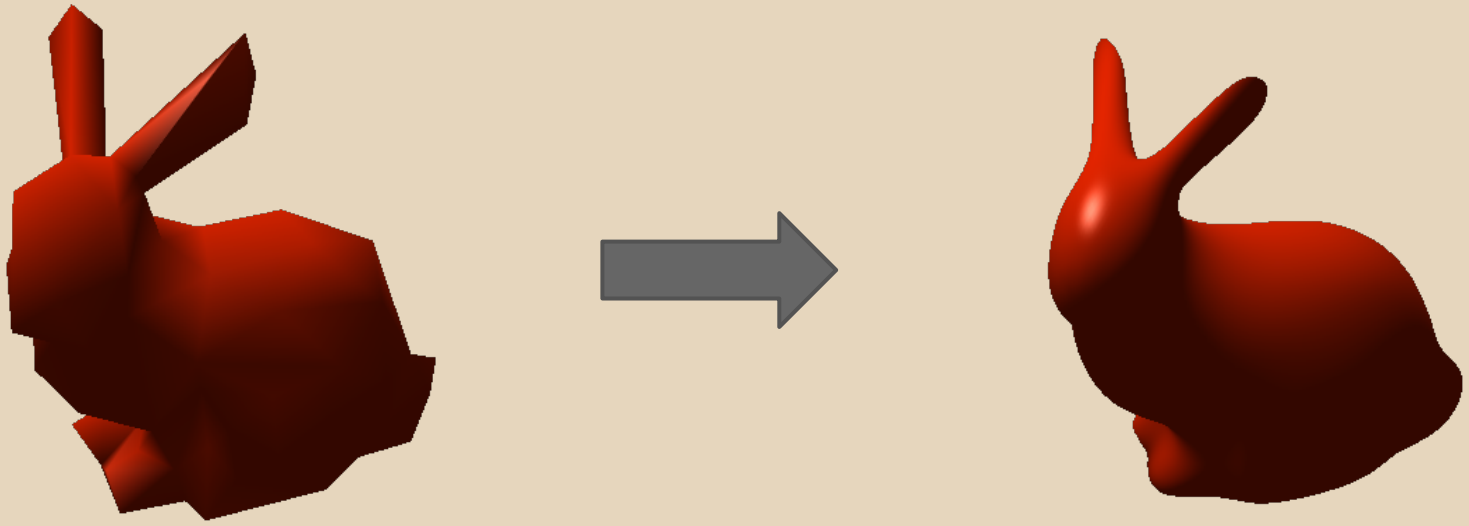
Surface Subdivision

- Start with Polygon Mesh
- Refine mesh by creating new faces and vertices
- Repeat

Project Overview

Surface Subdivision

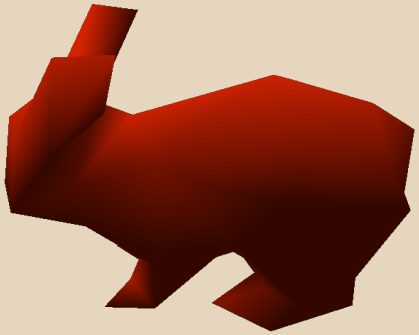
- Start with Polygon Mesh
- Refine mesh by creating new faces and vertices
- Repeat

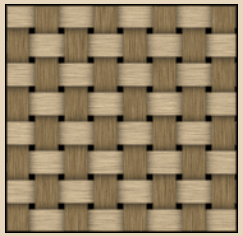


Project Overview

Texture Mapping

- Start with Polygon Mesh and Texture
- Map vertices of mesh into the texture
- OpenGL interpolates the colors

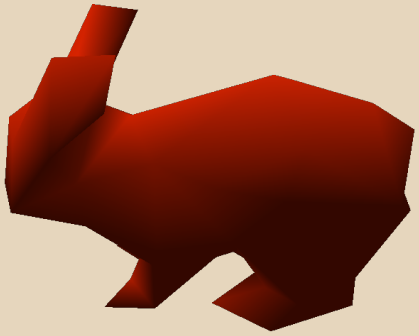


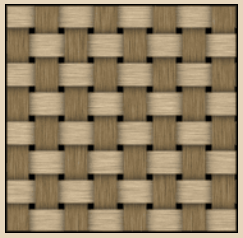


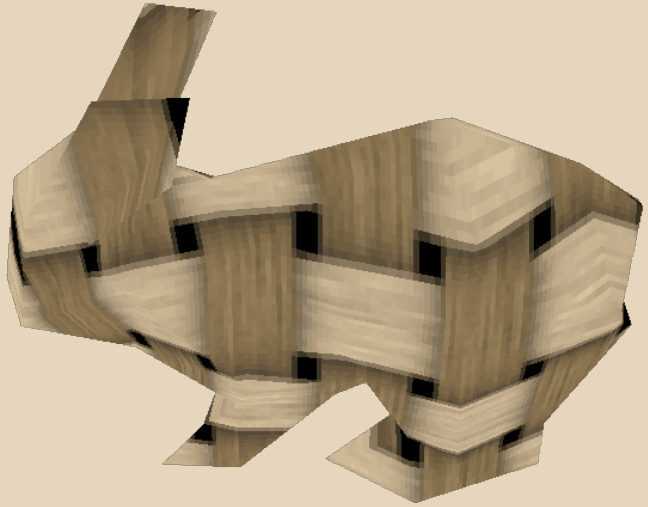
Project Overview

Texture Mapping

- Start with Polygon Mesh and Texture
- Map vertices of mesh into the texture
- OpenGL interpolates the colors





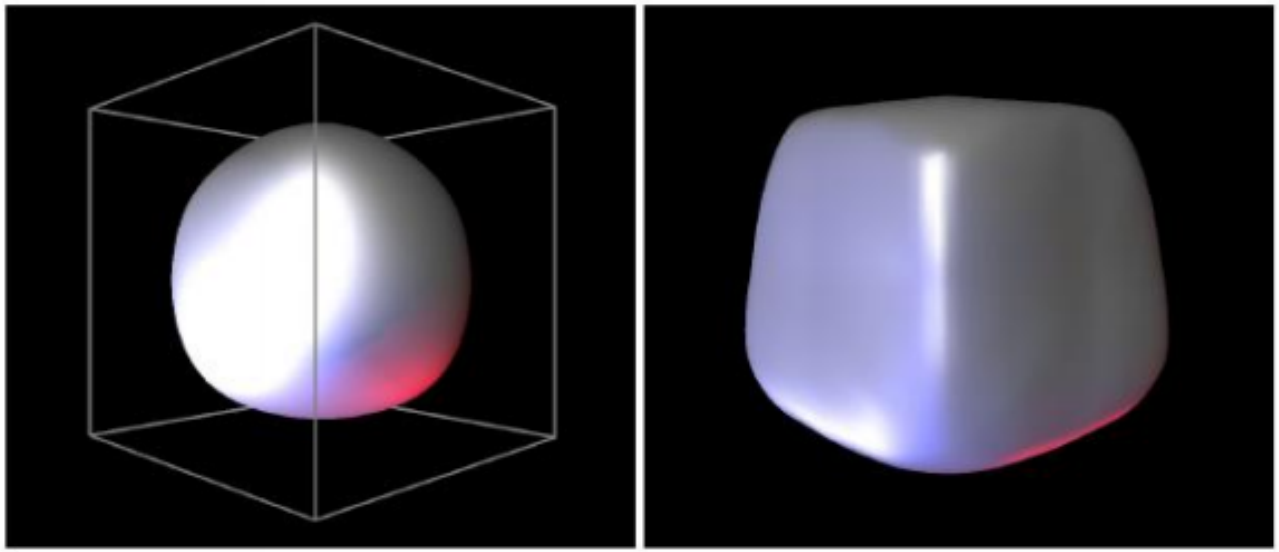


Outline

- Project Overview
- Subdivision
  - Loop Subdivision
  - Adjacency Data Structure
- Texture Mapping
  - OpenGL

Subdivision

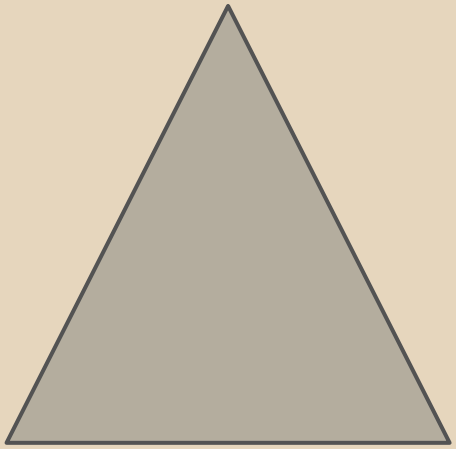
- Many different algorithms
  - Approximating v. Interpolating
  - Face Splitting v. Vertex Splitting
  - Continuity properties of final surface



Loop *Butterfly*

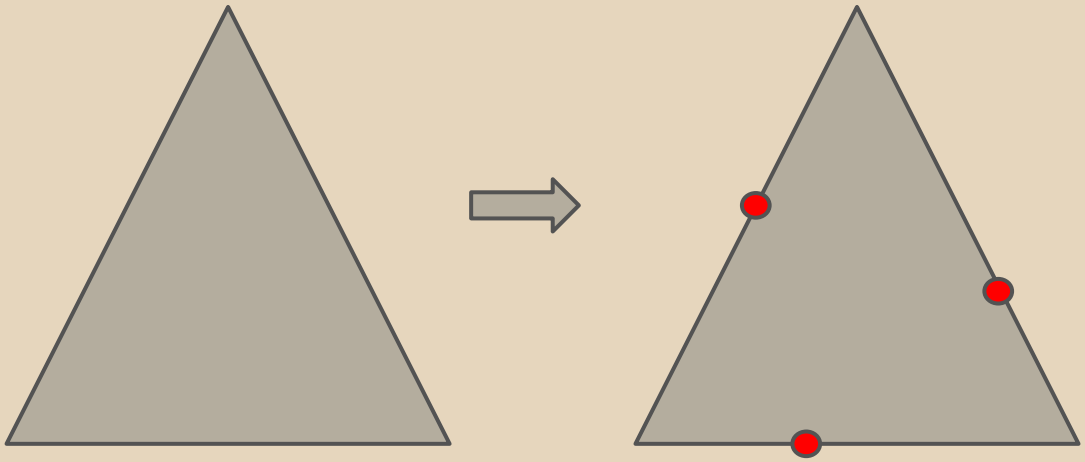
Loop Subdivision

- Approximating
- Face Splitting
- C2 continuity on regular meshes



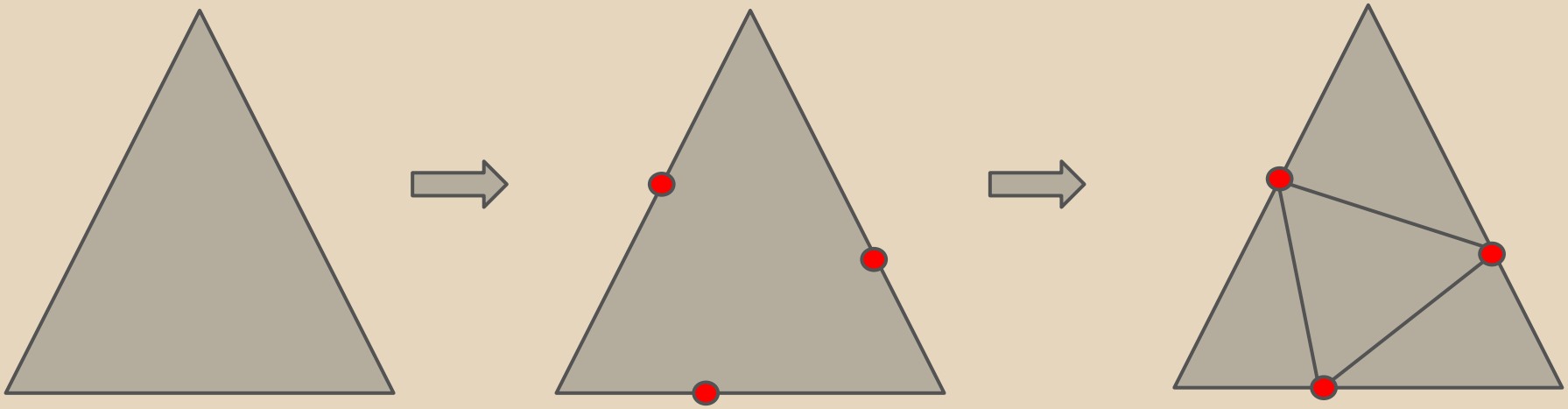
Loop Subdivision

- Approximating
- Face Splitting
- C2 continuity on regular meshes



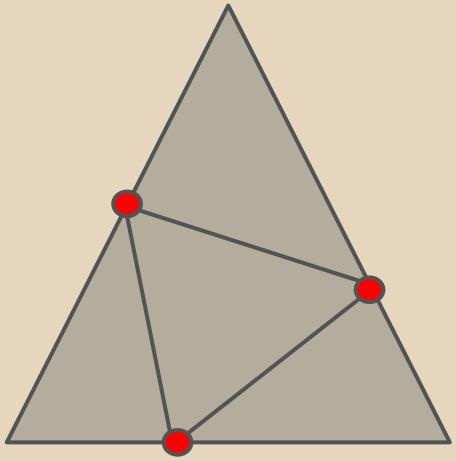
Loop Subdivision

- Approximating
- Face Splitting
- C2 continuity on regular meshes



Loop Subdivision

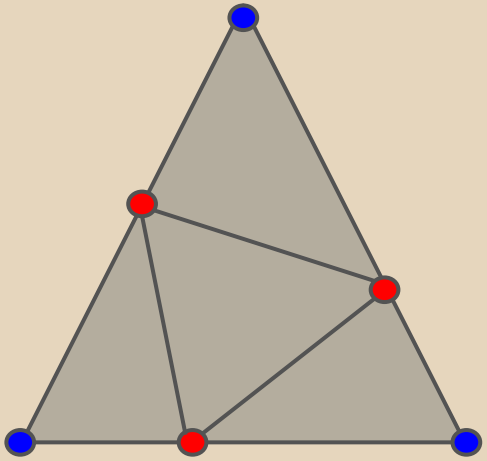
- Newly created vertices are called **odd vertices**



- Odd Vertices

Loop Subdivision

- Newly created vertices are called **odd vertices**
- Original vertices are called **even vertices**



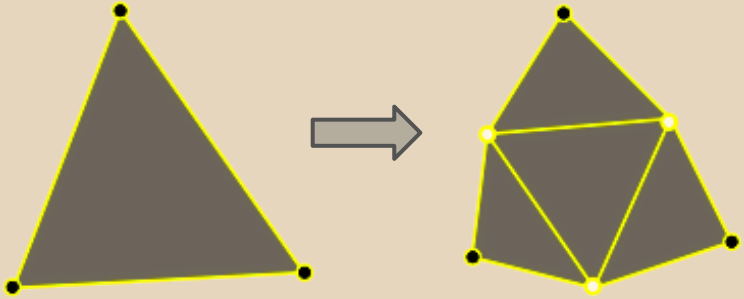
- Odd Vertices
- Even Vertices

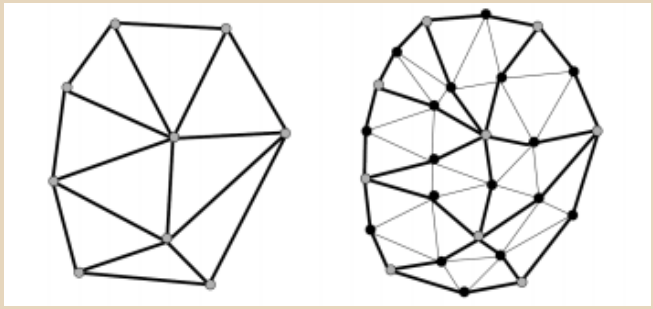
Loop Subdivision

- But "Approximating" means we recompute positions of all vertices (even and odd)

Loop Subdivision

- But "Approximating" means we recompute positions of all vertices (even and odd)

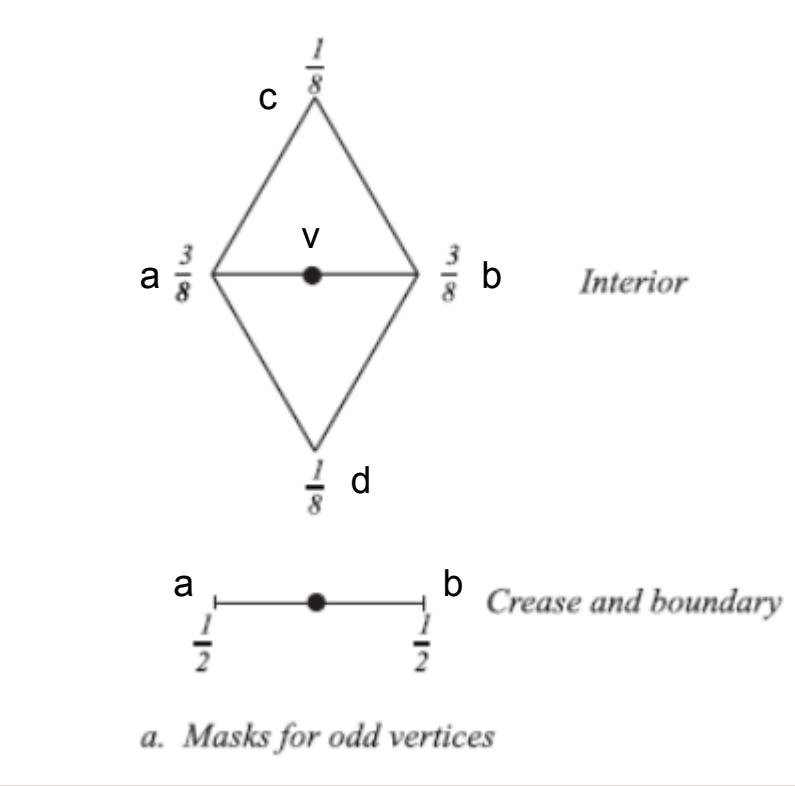




http://ezekiel.vancouver.wsu.edu/~cs442/lectures/winged-edge/subdivision/subdivision.html
http://www.cs.dartmouth.edu/~fabio/teaching/graphics08/lectures/10_SubdivisionSurfaces_Web.pdf

Loop Subdivision

- Computing odd vertices



a. Masks for odd vertices

Interior:

```
v = 3.0/8.0*(a + b)+
    1.0/8.0*(c + d)
```

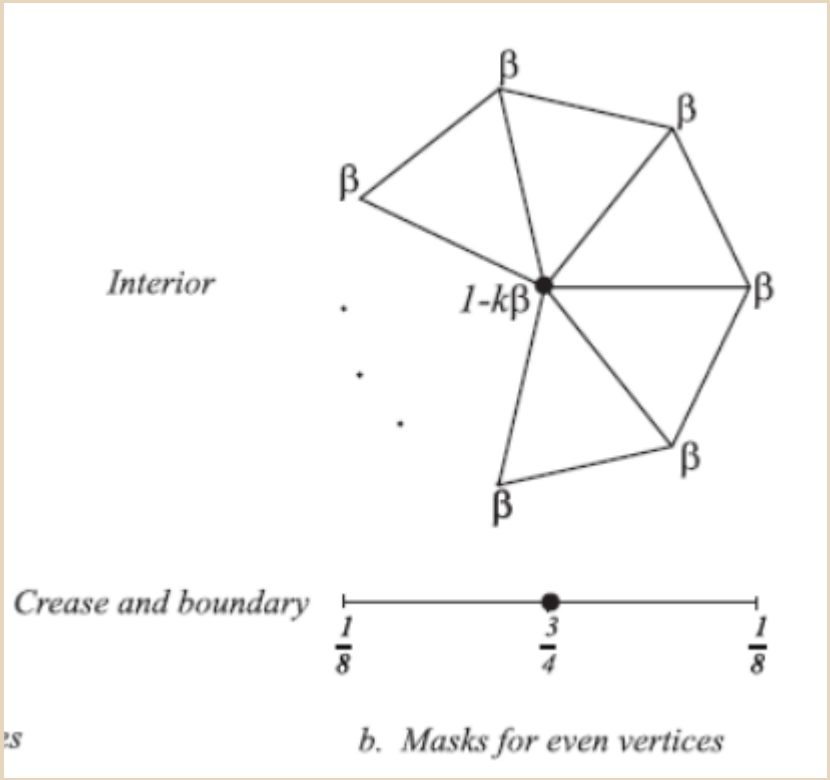
Boundary:

```
v = 1.0/2.0*(a + b)
```

Notice that to compute v we need some to know the nearby vertices.

Loop Subdivision

- Computing even vertices



Interior:

```
v = v*(1-k*BETA(k)) +
(sum of all k neighbor
vertices)*BETA(k)
```

Boundary:

```
v = 1.0/8.0*(a + b) +
    3.0/4.0*(v)
```

Notice that to compute v we need know all neighboring vertices

BETA(n) = $\frac{1}{n}(5/8 - (\frac{3}{8} + \frac{1}{4}\cos\frac{2\pi}{n})^2)$

Loop Subdivision

- Computing odd vertices
- Computing even vertices

Important:

1. We need to be able to query adjacency information about the mesh.
2. We need to be able to tell if a vertex is a boundary or interior vertex.

Loop Subdivision

Algorithm (one iteration)

1. Build adjacency data structure

   Tricky

2. Compute odd vertices

   Straightforward once you finish step 1.

3. Compute even vertices

   Straightforward once you finish step 1.

4. Rebuild mesh / Connect vertices to create new faces

   Similar to Project 1 (when you created a mesh from a heightmap)

Adjacency Data Structure

What properties do you want?

- Efficient traversal and lookup
  - `get_adjacent_faces(&mesh, &triangle)`
  - `get_neighbor_vertices(&mesh, &vertex)`
- Efficient memory usage
- Efficient creation and update

Adjacency Data Structure

What data do you need in the structure?

Mesh Data

- Some combination of Vertices, Faces, Edges
- Adjacency information

Loop Subdivision Metadata

- implicit
  - all edges of index < i have been subdivided
- explicit
  - `if (!mesh.vertex[i].is_subdivided) ...`

Adjacency Data Structure

Useful Mesh Attributes

- Every triangle has 3 vertices
- Every triangle is adjacent to up to 3 other triangles

Adjacency Data Structure

Useful Mesh Attributes

- Every triangle has 3 vertices
- Every triangle is adjacent to up to 3 other triangles
- A given vertex has N neighbor vertices
- The same vertex is part of either N-1 or N triangles
  - *Why?*
  - *There is a useful implication of this for Loop Subdivision*

Adjacency Data Structure

Useful Adjacency Attributes

- Triangle -> Vertex
- Triangle -> Triangle

- Vertex -> Vertex
- Vertex -> Triangle

This is a simple representation that can handle the queries you need.

Adjacency Data Structure

Implementation

- How you implement (storing and building) the adjacency data structure can be more important than what you represent.
- Try not to use `std::map`
- Stick to C data structures (arrays and structs) for the best speed
- Be mindful that `malloc/new` and `free/delete` are slow

Outline

- Project Overview
- Subdivision
  - Loop Subdivision
  - Adjacency Data Structure
- Texture Mapping
  - OpenGL

Texture Mapping

- Texture: 2D bitmap image
- (Project 2) Stored in a 2d array:
  - `texture[height][width][4]`
- Pixels in a texture are texels
- texels coordinates are in the range [0,1]

- Want to map 2D image onto a 3D geometry (e.g. triangle mesh)

Texture Mapping

Texture Mapping Procedure:

1. For each vertex in the mesh
   a. Manually assign texel coordinates (u,v) in [0,1]
   b. (u,v) are used to "index" into texture array
2. Colors of faces are interpolated from colors of vertices

Texture Mapping in OpenGL

- Initialization
  - Enable GL texture mapping
  - Specify texture
  - Read image from file into array in memory or generate image using the program (procedural generation)
  - Specify any parameters
  - Define and activate the texture
- Draw
  - Draw objects and assign texture coordinates to vertices

Texture Mapping in OpenGL

Useful Resources / Tips

- http://www.glprogramming.com/red/chapter09.html
  - OpenGL Red Book!
- http://www.opengl.org/sdk/docs/man2/xhtml/glTexCoordPointer.xml
  - Like Vertex Arrays but for Texture Coordinates

- `GL_REPEAT` - Repeat the texture over the mesh
- `GL_REPLACE` - Replace mesh color with texture color
- `GL_BLEND` - Blend mesh color with texture color

Project 2

Questions?