

CS770/870 Spring 2017

OpenGL Textures

Notes originally created by Matt Plumlee

Also see: Angel 6e: ch 7.6

Preview

Opportunity: Polygon fill w/ pattern

Problem: How to specify, apply textures

OpenGL's solution

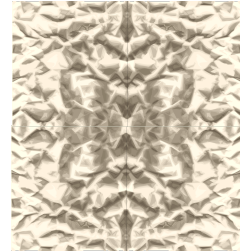
02/22/2017

CS 770/870 Spring 2017 Bergeron

1

Opportunity

- With lighting
 - Start with material color, properties
 - Modify color according to lighting model
 - Normals
 - Light sources
 - View direction
- What if material isn't single color?
 - Patterns: checkerboard, stripes, brick
 - Images: photos, renderings (reflections)



CS 770/870 Spring 2017 Bergeron

2

Problem

- Many ways to specify, apply texture
 - Is it from a file? Created in-program?
 - How many dimensions? 1D? 2D? 3D?
 - Is it monochrome? Multicolored?
 - Should it repeat? If so, how?
 - Should it respect lighting/shading?
 - Should it modify lighting?
 - Should it modify depth?
 - ...

CS 770/870 Spring 2017 Bergeron

3

OpenGL's Solution

- Texture ID (int) → internal texture spec.
- Parameters set on that spec.
- Fill texture buffer with values (color, etc.)
- When time to draw with texture
 - “Bind” texture—one to use during fill
 - Map texture to geometry (tie points in texture to each vertex)



CS 770/870 Spring 2017 Bergeron

4

OpenGL Texture Initialization

At texture-creation time

- 1. Ask GL for a texture ID


```
glGenTextures( 1, textureId ); // GLuint* textureId
```
- 2. Bind the texture ID to proper type


```
glBindTexture( GL_TEXTURE_2D, textureId );
```
- 3. Specify texture parameters


```
glTexParameteri( GL_TEXTURE_2D, GL_..., ... );
```
- 4. Specify data


```
glTexImage2D( GL_TEXTURE_2D, ... ); or
glGenerateMipmap( GL_TEXTURE_2D, ... );
```
- 5. Build texture coordinate buffer

2D version of coord buffer, normal buffer, etc

OpenGL Texture Usage

In order to draw...

- 6. “Bind” your texture of that type


```
glBindTexture( GL_TEXTURE_2D, textureID );
```
- 7. Load texture coord buffer


```
glBufferData( . . . );
```
- 8. Draw, unbind texture


```
glDrawArrays( . . . );
glBindTexture( GL_TEXTURE_2D, 0 );
```

Key glTexParameteri() Paramerters

Parameter	Values
GL_TEXTURE_MAG_FILTER magnification policy when texel covers many pixels	GL_NEAREST or GL_LINEAR
GL_TEXTURE_MIN_FILTER minification policy when pixel covers multiple texels	GL_NEAREST, GL_LINEAR, GL_NEAREST_MIPMAP_NEAREST, GL_NEAREST_MIPMAP_LINEAR, GL_LINEAR_MIPMAP_NEAREST, or GL_LINEAR_MIPMAP_LINEAR

Table 9-1 : Filtering Methods for Magnification and Minification, from <http://www.glprogramming.com/red/chapter09.html>

Key glTexParameteri() Paramerters

Parameter	Values
GL_TEXTURE_WRAP_S GL_TEXTURE_WRAP_T If texture space smaller than area	GL_CLAMP, GL_REPEAT
GL_TEXTURE_BORDER_COLOR	any four values in [0.0, 1.0]
GL_TEXTURE_PRIORITY	[0.0, 1.0] for the current texture object

Table 9-4 : glTexParameter*() Parameters , from <http://www.glprogramming.com/red/chapter09.html>

Code from Texture.java

```
int tex = glGenTextures();
glBindTexture( GL_TEXTURE_2D, tex );
glTexParameteri( GL_TEXTURE_2D,
                  GL_TEXTURE_MAG_FILTER,
                  GL_LINEAR );
glTexParameteri( GL_TEXTURE_2D,
                  GL_TEXTURE_MIN_FILTER,
                  GL_LINEAR );
glTexImage2D( GL_TEXTURE_2D, 0, GL_RGBA,
              w.get(0), h.get(0), 0,
              GL_RGB, GL_UNSIGNED_BYTE,
              image );
```

CS 770/870 Spring 2017 Bergeron

Texture code from Shape3D.redraw()

```
// has a loadTexCoordBuffer method to
// load the appropriate buffer if needed
// redraw code
if ( hasTexture )
    texture.bind(); // our Texture object
// Everything drawn will have this texture
glDrawArrays( GL_TRIANGLES, 0, nVertices );
if ( hasTexture )
    texture.unbind();
```

CS 770/870 Spring 2017 Bergeron

glGenerateMipmap

```
int tex = glGenTextures();
glBindTexture( GL_TEXTURE_2D, tex );
glTexImage2D( GL_TEXTURE_2D, 0, GL_RGBA,
              w.get(0), h.get(0), 0,
              GL_RGB, GL_UNSIGNED_BYTE,
              image );
glGenerateMipmap( GL_TEXTURE_2D );
glTexParameteri( GL_TEXTURE_2D,
                  GL_TEXTURE_MAG_FILTER,
                  GL_LINEAR );
glTexParameteri( GL_TEXTURE_2D,
                  GL_TEXTURE_MIN_FILTER,
                  GL_LINEAR_MIPMAP_LINEAR );
```

CS 770/870 Spring 2017 Bergeron

Lots To Explore with Textures

- Rendering to textures (requires pbuffers)
- 1D, 3D, and cube-mapping textures
- Blending multiple textures
- ...leads toward vertex shaders

CS 770/870 Spring 2017 Bergeron

Resources

- Some OpenGL texture resources
 - <http://www.opengl-tutorial.org/beginners-tutorials>
 - *general approach is good, but based on C-only image tools*
 - <https://learnopengl.com>
 - *also seems informative but based on SOIL C-based image tools*
 - <https://www.khronos.org/opengl/wiki/Texture>
 - *more than you want to know right now*

Review

- Initializing a texture

```
glGenTextures(1, textureID);
glBindTexture(GL_TEXTURE_2D, textureID);
glTexParameteri(GL_TEXTURE_2D, GL_..., ...);
glTexImage2D(GL_TEXTURE_2D, ...); or
glGenerateMipmap( id );
```
- Using a texture

```
glBindTexture(GL_TEXTURE_2D, textureID);
Need buffer for texture coordinate data
```