Texture Examples

Lecture 29

Robb T. Koether

Hampden-Sydney College

Fri, Nov 15, 2019

Outline

1. Finding Textures
2. Converting Textures to the DDS Format
3. Texture Coordinates
4. Textures and Meshes
5. Assignment

Outline

1. Finding Textures
2. Converting Textures to the DDS Format
3. Texture Coordinates
4. Textures and Meshes
5. Assignment

Finding Textures

- There are many websites that offer free textures.
- One such website is `http://texturelib.com/`
- If a texture is seamless, then it can be repeated horizontally and vertically without showing the seams.
- Google “seamless textures” to find more.

Outline

1 Finding Textures

2 Converting Textures to the DDS Format

3 Texture Coordinates

4 Textures and Meshes

5 Assignment

Converting Textures to the DDS Format

- Textures come in many formats: `.jpg`, `.bmp`, `.png`, `.gif`, etc.
- The `vglLoadTexture()` function requires that the format be `.dds`.
- The website `http://online-converting.com/image/convert2dds/` will convert graphics files from any standard format to the `.dds` format.

Outline

1. Finding Textures
2. Converting Textures to the DDS Format
3. Texture Coordinates
4. Textures and Meshes
5. Assignment

Texture Coordinates

- The coordinate system for textures has two coordinates s and t (equivalent to x and y).
- The corners of the texture are located at $(0, 0)$, $(1, 0)$, $(1, 1)$, and $(0, 1)$ (the unit square).
- However, in texture space, the basic texture is tiled throughout the plane.
- For example, the texture coordinates $(5.2, 9.6)$ will reference the same texel as $(0.2, 0.6)$.

Texture Coordinates

- The coordinate system for textures has two coordinates *s* and *t* (equivalent to *x* and *y*).
- The corners of the texture are located at $(0, 0)$, $(1, 0)$, $(1, 1)$, and $(0, 1)$ (the unit square).
- However, in texture space, the basic texture is tiled throughout the plane.
- For example, the texture coordinates $(5.2, 9.6)$ will reference the same texel as $(0.2, 0.6)$.
- This fact is quite handy.

Texture Coordinates

- Suppose we want to paste a brickwall texture onto a rectangle representing the side of a building.
- Suppose further that the dimensions of the wall are 20 feet wide by 8 feet high, i.e., vertices at $(0, 0)$, $(20, 0)$, $(20, 8)$, and $(0, 8)$.
- Suppose further that the texture represents a 1 foot by 1 foot unit.
- Then we may use the coordinates of the vertices as the texture coordinates.

Texture Coordinates

- Suppose, on the other hand, that the texture represents a 6 inch by 6 inch unit.
- Then we need to scale the vertex coordinates to produce the appropriate texture coordinates.
- What should the scale factor be?

Scaling Textures

A Textured Cylinder

```
uniform float tex_scale;
layout (location = 3) in vec2 vTexture;
out vec2 tex_coord;
tex_coord = tex_scale*vTexture;
```

- I suggest that you create a uniform variable `tex_scale`.
- In the application program, assign to it the appropriate scale factor.
- In the vertex shader, apply it to the value of `vTexture` before assigning to `tex_coord`.

Outline

1. Finding Textures
2. Converting Textures to the DDS Format
3. Texture Coordinates
4. Textures and Meshes
5. Assignment

Texture and Meshes

- I have written the `TexMesh` class, which is like the `Mesh` class except that it includes textures.
- I will provide you with copies of `texmesh.h` and `texmesh.cpp`.
- The three `create()` functions have five additional parameters.
    - `start_tex_s` – initial value of *s*.
    - `end_tex_s` – final value of *s*.
    - `start_tex_t` – initial value of *t*.
    - `end_tex_t` – final value of *t*.
    - `GLuint tx` – texture object.

Texture and Meshes

A Textured Cylinder

```
TexMesh pole;
pole.create(TEXMESH_CYLINDER,
    80, 0.0f, 2.0f*PI,    // 0 <= s <= 2 pi
    1, 0.0f, 10.0f,       // 10 units tall
    start_tex_s, end_tex_s,
    start_tex_t, end_tex_t,
    tx,                   // Texture object
    0.5f);
```

- Suppose that you want to create a pole 10 feet high with a diameter of 12 inches.
- What should be the ranges of the texture coordinates?

Texture and Meshes

A Textured Cylinder

```
TexMesh pole;
pole.create(TEXMESH_CYLINDER,
    80, 0.0f, 2.0f*PI,   // 0 <= s <= 2 pi
    1, 0.0f, 10.0f,      // 10 units tall
    0.0f, 2.0f,          // 2 copies around
    0.0f, 10.0f,         // 10 copies up
    Texture[woodTex],    // Wooden pole
    0.5f);
```

- The range of the *s* parameter should equal the circumference of the cylinder.
- The range of *t* could be 0 to 1, or 0 to 2, or 0 to 3, and so on, depending on the number of times you want the texture repeated.

Outline

1. Finding Textures

2. Converting Textures to the DDS Format

3. Texture Coordinates

4. Textures and Meshes

5. Assignment

Homework

Homework

- Read pages 259 - 263: Texture Mapping & Basic Texture Types
- Read pages 270 - 277: Texture Formats