Assignment 7: Normal-mapped Objects

Introduction

In this assignment, you will do normal mapping to enhance objects' appearance without causing any deformations to them. For normal mappings, at first, calculate per-vertex tangents and the tangent space matrix. Next, tangent space matrix is used in all lighting and shading computations, i.e., lighting, and shading need to be done in the tangent space for normal mapping. In the tangent space, vertex normals work as one of the three orthonormal bases. Instead of attribute normals, normals encoded in the normal maps of respective textures are used for lighting and shading calculations. In the end, objects' enhanced appearance with normal mapping will be compared with the ordinary texture mapping.

Specification (Basic Part)

Below are the lists of things you need to do in this assignment:

1. **Calculate per vertex tangents for the sphere, add the vertex tangent as an attribute, and define the tangent space matrix** **4 points**

   You can start working on this assignment using the uploaded example **NormalMappedPlane** as a reference**.** Replace the plane with a sphere of radius 8. Add **updateVertexTangents** function to compute per vertex tangents for the sphere and add as an attribute. From the cross products of tangents and normals, you can compute bi-tangents You can then define the tangent space matrix. In **NormalMappedPlane**, tangent space matrix has been defined in the vertex shader after vertex tangent is passed as an attribute.

   Hints: Review Lectures **CSCD470 570 Lecture 29** and **CSCD470 570 Lecture 30** along with the uploaded example **NormalMappedPlane**.

2. **Normal Mapping** **5 points**

   In **NormalMappedPlan**e, two texture units are active simultaneously: one is the original texture and another one is generated from Adobe Photoshop for normal mapping. For this assignment, in the attached zipped file, there are image files for mapping earth and mars surface textures (i.e., earth.jpg and mars.jpg) onto the sphere and their corresponding normal map textures (i.e., earthNormal.png and marsNormal.png). .

   a. After you determine the tangent space matrix from step 1, compute light parameters, i.e., light direction, and viewer's directions in the tangent space.

b. Next, the original texture will be used for color computation and the normal mapped texture will be used for retrieving normals to do lighting and shading computations in the tangent space.
Hints: Shader files in the uploaded example **NormalMappedPlane** will be helpful to understand this part on the final computation of color for normal mapping.

3. Switching between Normal Mapping and Ordinary Texture Mapping 2.5 points

With the press of the keyboard button 'n'/'N', you will switch between the normal mapped and ordinary texture mapped sphere.

a. For normal mapping, normals for computing the lighting and shading are encoded in the tangent space and are retrieved from the normal map. So, you do all lighting computations in the tangent space. However, for ordinary texture mapping, normals, vertex positions, and eye positions are converted to the camera space and all lighting computations are done in the camera space. Objects' normals are used to calculate lighting and shading.
Hints: See the vertex shader of the uploaded example **NormalMappedPlane** to understand the computation better. Calculate camera-space normals, convert light and viewer's direction in the camera space, and do the necessary calculation to find the texture from the original texture color. Do that modification in your code. Next, add some Boolean to synchronize between the keyboard button press.

b. Below are some outputs after you complete the basic part of the assignment. You can see that the object at the left has been enhanced with normal mapping and provides richer appearance in comparison with the object at the right.

Figure 1: (Left) Normal mapped and (right) ordinary texture-mapped sphere.

4. Others: **1 point**

a. Rotation of the Sphere

- The sphere centered at the origin will continuously rotate around the Y-axis. Use **glutTimerFunc** with a suitable time interval for that.

b. Key-interaction

- Switch between the normal-mapped mode and the ordinary texture mode by pressing 'n'/'N'.
- 'm'/'M' toggles between the earth surface and mars surface.

c. Light Parameters and Matrices

(1) Here are lighting parameters:

```
vec4 light_position(20.0, 20.0, 20.0, 1.0);
vec3 light_ambient(0.3, 0.3, 0.3);
vec3 light_color(1.0, 1.0, 1.0);
```

(2) Here are the metrices:

```
mat4 projection_matrix = perspective(radians(60.0f), 1.0f, 1.0f, 40.0f);

mat4 view_matrix = lookAt(vec3(eye[0], eye[1], eye[2]), vec3(center[0], center[1], center[2]), vec3(0.0f, 1.0f, 0.0f));

                where,
                        GLfloat eye[3] = { 0.0f, 0.0f, 22.0f };
                        GLfloat center[3] = { 0.0f, 0.0f, 0.0f };
```

Bonus part: (mandatory for graduate students; undergraduate students will be given 2 bonus points)

In this part, whatever you have drawn in the basic part, you will draw onto a texture using a frame buffer object. Next, you will switch to the default frame buffer and map that texture onto a cube. Below is the output.

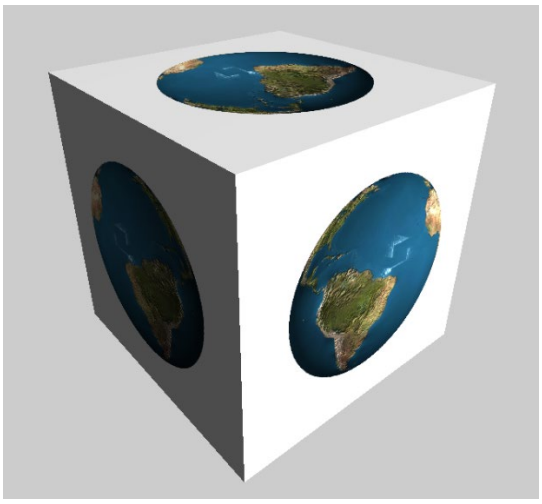

Figure 2: Normal mapping by rendering onto a texture.

The video demo demonstrates that you can still interact with keyboard buttons to switch between models as well as normal mapping vs. ordinary texture mapping by rendering onto a texture. Note, the texture is continuously getting updated.

For the bonus part, for rendering onto the texture using a frame buffer object, use the same matrix and lighting parameters used in the basic part. For rendering onto the default frame buffer, use the same lighting parameters. The demo uses the following projection matrix. The cube has a dimension of 1x1x1.

```
mat4 view = lookAt(vec3(0.0, 1.0f, 1.75f), vec3(0.0f, 0.0f, 0.0f), vec3(0.0f, 1.0f, 0.0f));
mat4 projection_matrix = glm::perspective(radians(70.0f), 1.0f, 0.3f, 20.0f);
```

Hints: This part will be discussed in the class.

Submission

Submit all necessary files in a zipped format. Name the file as **Firstname_ Lastname_7.zip** (your first name and last name). Submission deadline is Tuesday, June 7, 11:59 pm.

This assignment weights 12.5% of this course.