Middle East Technical University Department of Computer Engineering

CENG 477

Introduction to Computer Graphics

Fall 2014-2015

Assignment 3 - Basics of OpenGL Shading Language

Due date: 15 December 2014, Monday, 23:55

1 Objectives

This assignment aims to get you familiar with OpenGL Shading Language(GLSL) by implementing texture mapping, height mapping and lighting for the Earth.

2 Specifications

In this assignment, you will render 3D model of the Earth by implementing texture mapping, height mapping and lighting using GLSL. The initial scene consisting of background stars, the Sun, keyboard interactions and the configurations for the orbital motions is given to you in the template code. In addition to these, you are expected to get some configuration parameters from the command line and render the 3D model of the Earth accordingly. Details of the command line parameters will be given in the following section.

2.1 Command Line Parameters

The Earth is modeled by number of vertices. The vertex count and the radius of the Earth is read from the command line. The respective order of the parameters and their explainations are as follows:

- **Number of Latitudes:** Number of vertical vertices that are used to draw the Earth. The vertical sample size.
- **Number of Longitudes:** Number of horizontal vertices that are used to draw the Earth. The horizontal sample size.
- **Radius:** The length of the line from the center of the Earth to any point on its surface.

The total number of vertices on the Earth is `[Number of Latitudes]*[Number of Longitudes]`. You will draw and texturize the 3D model of the Earth using those vertices.

A sample run of the program with the arguments is as follows:

```
$ ./hw3 180 180 1
```

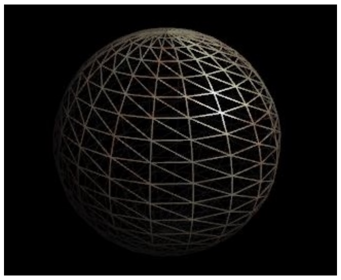
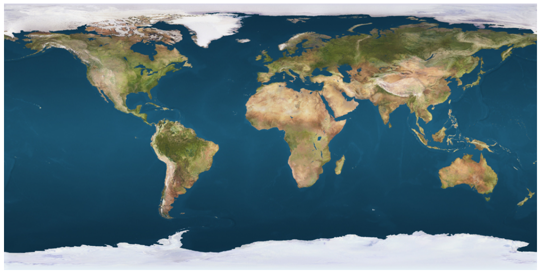


Figure 1: An example triangulation of a sphere. Number of vertices is 20 * 20 = 400.

2.2 Texture Mapping

Texture mapping is the process of mapping a texture (an image) to the surface of a polygon. Every vertex in the polygon is assigned to a texture coordinate. These texture coordinates are also known as u-v coordinates. To compute these coordinates, a method called unwrapping is used. You could think of this as converting a 3D model into a 2D image. Then a 2D image specifying the texture is matched with the 2D representation of the model and for each vertex, its u-v coordinates are computed. These u-v coordinates are between 0.0 and 1.0, where the lower left corner is specified as (0.0, 0.0) and upper left corner as (1.0, 1.0).

You are expected to map a texture to the 3D model of the Earth. You will first need to generate the texture map from the provided image and calculate the u-v coordinates for each vertex. Then you will use these coordinates in the Vertex and Fragment Shaders to map the texture to the Earth.



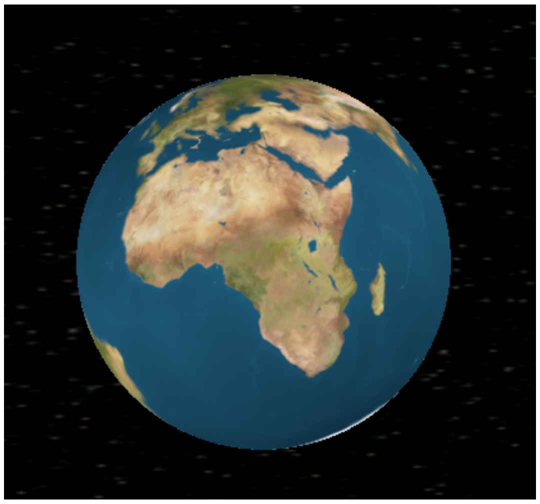


Figure 2: Texture map(left) and mapped 3D model(right) of the Earth

2.3 Height Mapping

Height map is simply a greyscale image used to generate the elevation data for the models. Each pixel's color is interpreted as a distance of displacement. The whiter the pixel color is, the higher the distance of displacement will be. In this assignment, you will displace the geometric position of the vertices to elevate the surface of the Earth using provided height map. In order to do that, you will calculate corresponding color values from the height map for each vertex as you did in the texture mapping. Then you will displace each pixel along the direction of its normal vector according to the color value. The calculation for the new position of a vertex **i** is:

$$P_i = p_i + n_i * d * c$$

where $\mathbf{p_i}$ is the actual geometric position of the vertex **i**, $\mathbf{n_i}$ is the normal vector of the vertex **i**, **c** is the color value that is obtained from the height map for the vertex **i** and **d** is the displacement level.

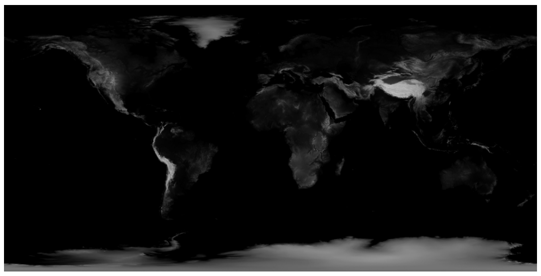
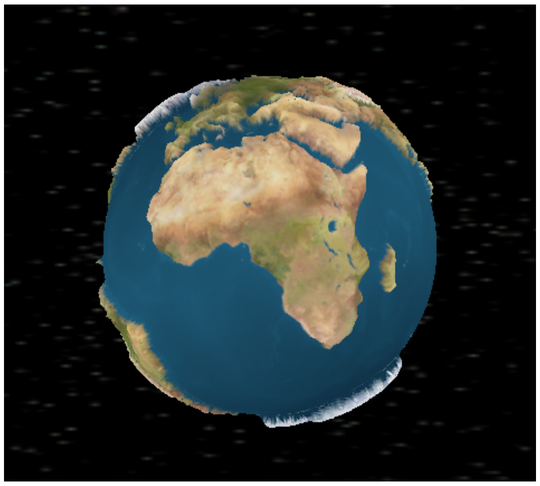



Figure 3: Height map(left) and mapped 3D model(right) of the Earth. Displacement level is 0.1.

The displacement level will be parameterized using '-' and '+' keys from the keyboard. While pressing the '-' key will decrease the level of displacement by `0.01`, pressing the '+' key will increase it by the same value.

Unlike the texture map of the Earth, the given height map is grayscale(1 channel) image. Therefore, instead of `GL_RGB`, `GL_LUMINANCE` should be used as a type of the image.

2.4 Lighting

For the lighting, the Sun always located at (0, 0, 0) is used as a point light source. While writing your shaders, you should darken or lighten the color of every pixel by considering the position of the Sun.

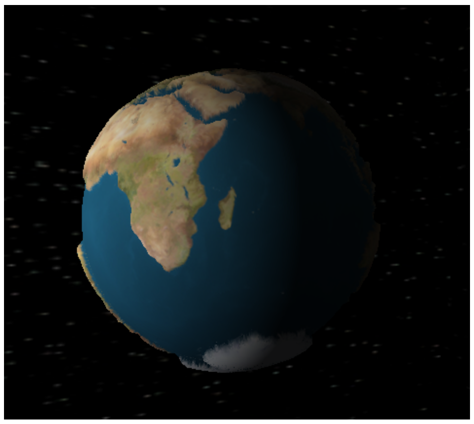


Figure 4: An example scene with lighting.

To implement lighting, Phong Illumination Model is used with the parameters given as below:

$$k_a = (0.25, 0.25, 0.25), I_a = (0.3, 0.3, 0.3),$$

$$k_d = (1, 1, 1), I_d = (1, 1, 1),$$

$$k_s = (1, 1, 1), I_s = (1, 1, 1), \alpha = 100.$$

The color calculation for a pixel **p** is given by:

$$c_p = c_{tex} * [k_a * I_a + k_d * (\hat{L} \cdot \hat{N}) * I_d + k_s * (\hat{R} \cdot \hat{V})^{\alpha} * I_s]$$

where $\mathrm{c}_{\mathrm{tex}}$ is the texture color of the pixel **p** that is obtained from the texture map.

2.5 Bonuses

There might be various extentions that can be included to this assignment such as adding other planets, implementing new shaders for the Sun and those planets, enhancing the shaders of the Earth etc. There will be bonus points(max. 10%) for such an extra effort.

3 Hints & Notes

- Try to do your works on the inek machines.
- First, read and understand the template code. There are comments in the code about what is done and what you are expected to do.
- When you first compile and run the provided template code, you will see the Sun and stars at the background. Camera, initially, looks at a position where the Earth will be displayed and it infinitely rotates around the Sun.
- You can use 'a' and 'd' keys to rotate the camera around the Earth and mouse wheel to zoom in/out.
- Height map of the Earth is provided to you at `res/heightmap.jpg`.
- Texture map of the Earth is provided to you at `res/texturemap.jpg`.
- Simple OpenGL Image Library(SOIL) is used to load images in the templete code. In case of having problem with it, you can download the source code from `http://www.lonesock.net/soil.html` and use it by compiling. You are also free to use any other image library to load images as long as it compiles with your code without having an error.

4 Regulations

1. **Programming Language:** C/C++
2. **Late Submission:** You can submit your codes up to 3 days late. Each late day will incur a penalty of 10 points. After 3 days, you will get 0.
3. **Cheating: We have zero tolerance policy for cheating**. People involved in cheating will be punished according to the university regulations.
4. **Newsgroup:** You must follow the newsgroup (news.ceng.metu.edu.tr) for discussions and possible updates on a daily basis. For any problems about the assignment, sending e-mail to TA's **should not** be the first choice. Instead, please ask your questions publicly via COW to make others know the issue.

5 Submission

Submission will be done via COW. Create a tar.gz file named `hw3.tar.gz` that contains all your source code, shader and image files.

Note: In the provided `Makefile`, there is a target called `dist`. To generate submission file, you may use this target by executing `make dist` command. This generates `hw3.tar.gz` file automatically. If you will use this command to submit this assignment, please make sure that all your files and folders are included in the created achive.

The following command sequence is expected to run your program on ineks:

```
$ tar -xzf hw3.tar.gz
$ make
$ ./hw3 180 180 1
```