GLSL, Lighting and Textures

This assignment will be 15 of the 55 points available for the programming assignments.

Notes for CSE 470/598
Instructor: Ross Maciejewski

1 Project Guidelines

- Have fun, learn, be creative, and create something you are proud to show-off!
- No teamwork allowed!
- Comment your code - you will be glad you did. You will be reusing code throughout the semester.
- You will upload your projects on myASU.
- If you have **good reasons** that you cannot complete a project on time **and** you have written documentation, then we can make adjustments to due dates. However, you must notify us before the due date that you would like to discuss such an arrangement. Good reasons would be illness, family emergency, visiting a conference to present a paper, …

2 Upload your projects

- Check the due date of the project on myASU
- Use myASU (digital dropbox) to upload your project. (Do not send the program as email attachment to the instructor or TA!)
- Your project should be packed in a zip file and named with the following naming convention:
  PX.Lastname.V1.zip
  (if you submit some updated files, use a different version)
  So my second project would be named
  P2.Maciejewski.V1.zip
  If I then found a last minute mistake and want to upload again I would use the name P2.Maciejewski.V2.zip
- If there is someone with the same last name in the course you can add your first name after the last name in the filename.
- The zip file should include all the source code and should be ready to compile. That means it should include your Visual C++ project files.
- The zip file should include an executable and the glut32.dll so that it can be started directly by clicking on it. (from wherever the file is downloaded to).
- A short text file called “instructions.txt” – this text file should include the following:
  - a) instructions about how to navigate in your program, including the function of special keys (e.g. x exits the program)
  - b) special functions that you implemented
  - c) changes from the original specification that you negotiated with the instructor or TA to the original specification
- You can use rar instead of zip to compress your files

3 Project 3

3.1 Purpose

This project has been designed for you to learn GLSL, lighting and textures.

3.2 Overview

You should take the sample code provided on blackboard and compile the code.

3.3 Required Functionality

Here are the elements that your program must have. Despite this "laundry list" of requirements, there is some room for creativity! Note: If the text suggests a menu item "Sound - Off" that would mean that "Off" is an entry in the submenu "Sound".

- **(5p) Create a ground plane** – You will create a square ground plane for the objects in your scene to sit on.

- **(10p) Create two snowmen** – You will write a program that creates two snowmen modeled by calls to gluSphere and glutSolidCone. Each snowman must be a hierarchical data structure as done in assignment 2. The 1st snowman (henceforth referred to as Snowman 1) will consist of a base sphere (for the body) and a head sphere with a cone on top of it. The base sphere will have three button sphere, the head sphere will have two button eye spheres and a cone nose. The second snowman (Snowman 2) will be the same except it will NOT have buttons, eyes or a nose.

  ```
  quadratic = gluNewQuadric(); //Create a pointer to the quadric object
  gluQuadricNormals(quadratic, GLU_SMOOTH); //Create smooth normal
  gluSphere(quadratic, 100.0, 20, 20); // or whatever parameters you want
  ```

- **(15p) Render Snowman 1 –** You will do fragment shader lighting. As we discussed in class, use the GLSL tracking of the OpenGL state for material properties and the light position in your shader code and you won't need uniforms for that. The head, body, buttons and caps should all be made of different material. You can find fun material values in the material.txt file.

- **(15p) Render Snowman 2** – You will take a picture of your face and wrap it around the head of the snowman such that it is your face on both sides of the snowman head.

- **(20p) Render the ground plane and Snowman 2 body and hat** – You will create procedural textures for the ground plane, the Snowman 2 body and hat. You will have horizontal red and white stripes on the snowman's body and hat, and a checkerboard on the ground. A procedural texture is create programmatically, not input as an image. Guess what, the Red Book has lots of examples for this!

- **(5p) Set up a perspective projection camera for this assignment**

- **(10p) Navigation**: The camera is controlled as follows:
  You can go forward/backward with the cursor up/down keys or rotate around the y-axis using the left and right keys.

- **(10p) Graduate Students –** You will set up a menu option in the program Textures – Default and Textures – Noise (a menu item texture with two options under it). In default, you do the undergrad portion. In the noise portion, the Snowman 1 body is rendered to look like marble using Perlin noise functions.

- **(10p) Graduate Students –** You will set up a menu option in the program Lighting – Default and Lighting – Bump (a menu item lighting with two options under it). In default, you do the undergrad portion. In the bump portion, the snowman 1 hat is rendered using a bump map lighting shader to look bumpy. Use spherical looking bumps.

Total points are out of 100. Undergraduate points scale out of 80 (so 80/80 = 100/100 for undergraduates). Undergraduates may earn an additional 10% maximum for completing the graduate portion (thus earning 110/100).

3.4 Grading

These points represent an initial weighting of the functionality. **Important:** You get full points for a *good* implementation of the required functionality. In principle you will get full points if you implement the requirements correctly, but there are many ways to do things according to specification that we might consider *unreasonable* or *undesired*. Especially we will deduct points if the design choices make it difficult to judge the program (E.g. you choose materials that are so dark that it hard to see anything, the objects move too slowly or too fast, the control of the program is extremely difficult.)

Note: we might change the weights slightly if a requirement is misunderstood by a majority of the class or some other special circumstance. I do not expect that to happen though.