CPSC 314
Assignment 1: A Broken Lightbulb. Introduction to Three.js, WebGL, and Shaders

Due 11:59PM, Sep 17, 2017

1 Introduction

The main goals of this assignment are to setup your graphics development environment, including checking your browser compatibility, enabling the use of local files, and an initial exploration of the uses of vertex and fragment shaders. For this exploration you will be using a template provided by the instructor, including shader code (`.glsl` files).
Your main work will be to develop a high level understanding of how the code works, to modify or write shaders, and to use rudimentary communication between the JavaScript program and the shaders. Some of the details of what is going on in the rest of the code will only become clear a bit later in the course. You are of course welcome to take a peek now, especially for the last part of the assignment. Some of the concepts are explained in Appendix A of your textbook, and in the web resources listed on the course web page.
To program a shader, you will use a programming language called GLSL (OpenGL ES Shading Language version 1.0). Note that there are several versions of GLSL, with more advanced features, available in regular OpenGL. Make sure that any code you find while trying to learn GLSL is the correct version.
This assignment uses a simple scene consisting of an "Armadillo" character and a broken "lightbulb". Your task will be to "fix" this lightbulb and give it new functionality. You can move the camera around the scene by dragging with a mouse, pan by holding down the right mouse button while dragging, and zoom by scrolling the mouse wheel.

1.1 Template

- The file `A1.html` is the launcher of the assignment. Open it in your preferred browser to run the assigment, to get started.
- The file `A1.js` contains the JavaScript code used to set up the scene and the rendering environment. You will need to make minor changes in it to answer the questions.

- The folder `glsl` contains the vertex and fragment shaders for the armadillo and lightbulb geometry. This is where you will do most of your coding.
- The folder `js` contains the required JavaScript libraries. You do not need to change anything here.
- The folder `obj` contains the geometric models loaded in the scene.
- The folder `images` contains the texture images used.

1.2 Execution

As mentioned above, the assignment can be run by opening the file `A1.html` in any modern browser. However, most browsers will prevent pages from accessing local files on your computer. If you simply open `A1.html`, you may get a black screen and an error message on the console similar to this:

```
XMLHttpRequest cannot load armadillo.vs.glsl. Cross origin
  requests are only supported for protocol schemes: http,
  data, https.
```

Please see this web page for options on how to run things locally:

`https://threejs.org/docs/#manual/introduction/How-to-run-thing-locally`

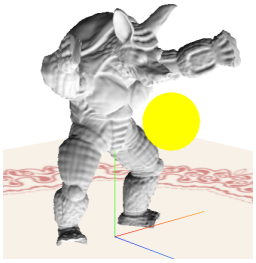
2 Work to be done (100 pts)

First, ensure that you can run the template code in your browser. See the instructions above. Study the template to get a sense of how it works.

1. (70 points)

   (a) **15 pts** Moving & Coloring the Lightbulb.
   The variable `lightPosition` (the position of the lightbulb center in world coordinates) declared in `A1.js` is changed using the keyboard, and passed to the lightbulb vertex shader (in `lightbulb.vs.glsl`) using `uniform` variables. First, modify the lightbulb vertex shader to move the lightbulb in response to keyboard input. Then, change the color of the lightbulb to yellow in the lightbulb fragment shader (in `lightbulb.fs.glsl`). Important: **do not** use Three.js functions; you must modify the shader for credit.

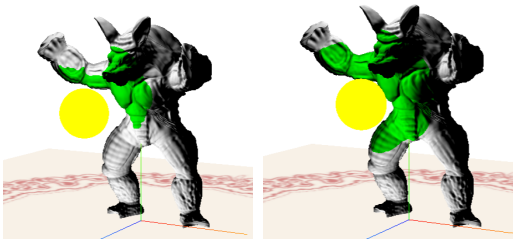
   (b) **30 pts** Fixing the Lightbulb.
   Modify `A1.js` and the armadillo shaders (`armadillo.vs.glsl` and `armadillo.fs.glsl`), to colour points on the surface of the armadillo based on the cosine of the angle between its normal and the direction vector to the center of the lightbulb. When correctly coded the lightbulb will be "fixed," illuminating different parts of the armadillo as it's moved around, as illustrated in the figure below.



Hint 1: See how uniforms are passed to the lightbulb shaders. *Hint 2:* You should pass the necessary information about the lightbulb to the armadillo shaders. *Hint 3:* See how varying variables are passed to the armadillo fragment shader.

(c) **25 pts** Radioactive Lightbulb.

For this part you will need to modify `armadillo.fs.glsl` to further color the armadillo fragments green when in close proximity to the lightbulb, as illustrated in the figures below. One simple way is to check if a fragment is within a specified distance to the lightbulb, and if it is, set its color to green.



Hint: You should use the appropriate uniform variable in the armadillo shader.

2. **Part 2:** (30 pts) Creative License

For this part we want to see what you can do. Your ideas should use at least one new shader, and should be of a similar complexity to the previous tasks. If you have any doubts, make sure to OK it with a prof or TA. Some possible suggestions might be:

- deform the vertices as a function of time.
- explode the model along face normals to view all the triangles that make it up.
- illuminate the armadillo with colored light.

Bonus marks may be given at the discretion of the marker for particularly noteworthy explorations.

2.1 Hand-in Instructions

You do not have to hand in any printed code. Create a README.txt file that includes your name, student number, and login ID, and any information you would like to pass

on to the marker. Create a folder called "a1" under your "cs314" directory. Within this directory have two subdirectories named "part1," and "part2", and put all the source files, your makefile, and your README.txt file for each part in the respective folder. Do not use further sub-directories. The assignment should be handed in with the exact command:

```
handin cs314 a1
```