CPSC 314
Assignment 3: Shades of Armadillo

Due 11:59pm, March 10, 2014

1 Introduction

The goal of this assignment is to explore lighting and shading of 3D models. You will be simulating three different light effects described in class: Phong shading with a single light source, extending the shader to use multiple lights, and Toon (cartoon-like) shading. In this assignment you will be *adding* new pairs of shaders (both vertex and fragment shaders) to carry out these tasks. You will be writing a bit more code, but you can modify the shaders from previous assignments for this purpose. As with the previous assignment, press 'm' when running the assignment to switch between the different shaders.

Template: The template code is found in the main assignment directory. It is very similar to the previous template. The main differences are:

- There is a new version of the Armadillo model that includes vertex normals. **Important:** make sure you use the new version, as the appearance with the old one will not be as good. If you are using a slow computer, you can use the lower resolution model (`Armadillo_lowres.obj`) also included in the Mesh directory.
- WorldState has been modified to support multiple lights.
- The `mode` variable has been modified slightly in obvious ways; now the modes are `MODE_PHONG`, `MODE_MULTI`, and `MODE_TOON`. Depending on which mode is used, the Armadillo should be shaded with a different pair of shaders.
- The gem has been turned into a light source.

You will need to edit the code and create some new shaders for this assignment. Hints are given in the template code about where to make the changes. Look for comments marked with "@@...".

2 Work to be done (100 pts)

(30 pts) Phong Reflection and Phong Shading. In the previous assignment templates we used a simplified lighting model to calculate the color of the armadillo at any given vertex. The full Phong *reflection* model expands on this by adding specular highlights. In addition, Phong *shading* computes the lighting per fragment, using the interpolated values of the fragment's position and normal. This will give a much nicer looking armadillo.

Use a single point light source, based at the gem's current position. So when you move the gem you will also move the light source.

The following image, taken from the Wikipedia article on the Phong reflection model, shows how the different components look individually and summed together:

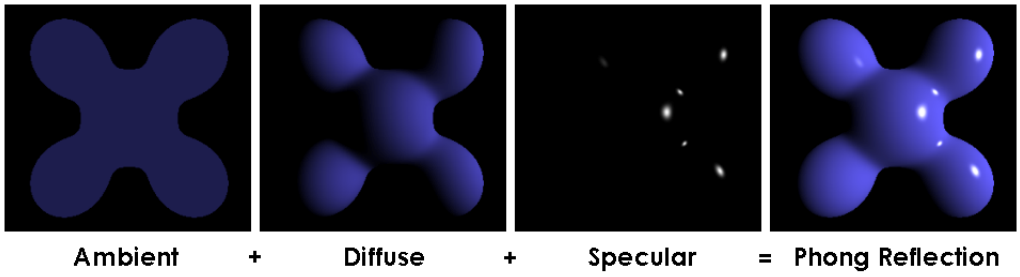

Figure 1: Phong Reflection Model

The main calculations should all go in the fragment shader (in file phong.fs.glsl). Note that you will still need a vertex shader (in phong.vs.glsl), to pass the appropriate information to your fragment shader.

(20 pts) Multiple Lights. In addition to the light coming from the gem, set up two more light sources in the world that also get passed through to the shaders. Your job is to evaluate the Phong lighting equation for each light source and sum the results to determine the final light intensity at each pixel.

(20 pts) Toon Shading. This is an example of a "non-photo realistic" (or NPR) shader. It emulates the way cartoons use very few colors for shading, and the color changes abruptly, while still providing a sense of 3D for the model. This can be implemented using a quantized version of the diffuse reflection term. Instead of making the intensity smoothly vary with $\cos\theta$ (where θ is the angle of incidence between the light vector and normal vector), you quantize this variation into a small number of steps.

Here is an example of what the armadillo might look like with a toon shader:

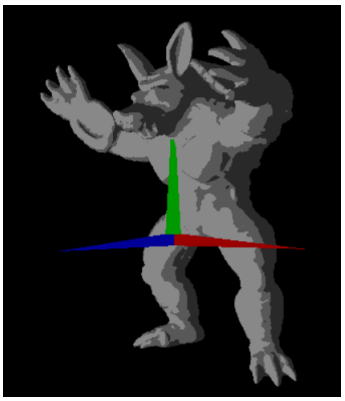

Figure 2: Example of Toon Shading

If you need some inspiration the following movies and video games were rendered with toon shading (also called cel shading) techniques:

`http://en.wikipedia.org/wiki/List_of_cel-shaded_video_games`.

(30 pts) Creative License. This part is up to you. You should create a new shader or modify one of the above shaders to implement some interesting lighting effects similar to the complexity of the ones above. Some recommendations are:

- Implement anisotropic material, such as brushed metal. See Section 14.4 of textbook.
- Add spot lights and directional lights, perhaps animate them.
- Explore other non-photo realistic shaders.
- Simulate fog.

You might come across a lot of really interesting shaders that require texture mapping or use of textures in some way (Shadow mapping, sky boxes, etc). We recommend you hold off on these until the next assignment since that will deal specifically with texture mapping and significant work would need to be done to the current template to read and write to texture files.

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

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 "assn3" under your "cs314" directory. Put all the source files, your makefile, and your README.txt file for each part in the folder. Do not use further sub-directories. The assignment should be handed in with the exact command:

```
handin cs314 assn3
```