CPSC 314
Assignment 3 — Coding

Due Fri Nov 28, 2014

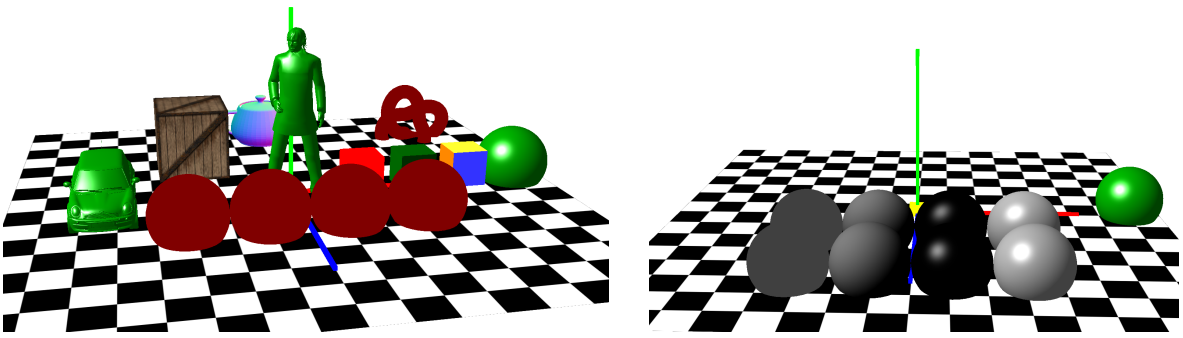
Illumination Models and Using three.js

The assignment is based on *three.js*, which is a javascript API that lets you write WebGL applications without having to worry about low-level details. It allows for the easy specification of basic material shaders, including texture maps, Phong illumination, environment maps, and more. At the same time, it also conveniently allows you to author your own vertex and fragment shaders.

You can choose to work alone or in pairs for this assignment.

In the first question, you will be experimenting with the Phong illumination model that is already provided for you in the `three.js` API, as well as implementing the Phong illumination model in your own fragement shader. The figure below on the left represents the starting template code. The figure below on the right represents an example solution, where the spheres in the front row are rendered using the `three.js` Phong shader, illustrating the ambient, diffuse, specular, and total components, and the spheres in the back row are the same but rendered using the shader you will write.

In the second question, you have the opportunity to do whatever you like, namely to build an interesting scene, a simple application, or to more deeply explore an aspect of computer graphics that you find to be of personal interest.



1. Shaders
   (a) (2 points) Create a copy of the A3 template code, available from the lectures web page. Ensure that it runs on your platform of choice (Chrome, Firefox, Safari). As with the previous assignment, you will want to ensure that you have local file access enabled. You should see a rotating version of the scene below. In order to focus on the relevant parts of the scene for the first question, begin by commenting out many of the objects that are instantiated in `a3.js`, from the green cube all the way to where the car and statue are loaded near the end of the file.

(b) (8 points) Now add a row of four spheres to the front part of the checkerboard, located along z=10 in the local coordinate frome. These spheres should be rendered using Phong illumination shader, i.e., `THREE.MeshPhongMaterial`, with the reflectances given below (and as also described in the comments in `a3.js`:

sphere 1: ambient reflectance of 0.25, with no other contribution;
sphere 2: diffuse reflectance of 0.5, with no other contribution;
sphere 3: specular reflectance of 0.125 and n=40, with no other contribution;
sphere 4: sum of the ambient, diffuse, and specular contributions.

The spheres should be lit by the existing light source that is already specified in the template code. Use the example sphere in the code as a useful template for creating your new spheres.

(c) (8 points) The second row of four spheres are rendered using the shader `myPhong` that is defined in `a3.html`, and which assigns a dark red constant colour to these spheres by default. Your goal is to create a fragment shaders that implements the Phong illumination model. The template code is already set up to provide you with all the input variables that you will need. Develop your solution in stages, i.e., first develop only the ambient component and convince yourself that it is producing a correctly-rendered result for the left-most sphere. Then develop the portion of the shader for the diffuse component, which if done correctly, should reproduce the `three.js` shader results for the second diffusely-rendered sphere. Then move on to developing the specular component. Your final shader can be completed by adding around 12 lines of code to the template.

Debugging shaders developed for the `three.js` environment takes a bit of practice. If you get a blank image, open the Developer Console and be sure to scroll all the way back to the top, where the early and most-informative error messages can be seen. Develop your shader in **small, incremental steps**. Note that the GLSL language provides you with functions such as `dot(L,N), pow(x,n), reflect(I,N)`. Google `GLSL functions` to read more about these. Also be sure to **normalize** the vectors in your lighting models!

Submit your assignment electronically using:`handin cs314 a3`

2. (16 points) Develop your own scene

This question is your opportunity to build an interesting scene, a simple application, or to more deeply explore an aspect of computer graphics that you find to be of personal interest. You can use the template code from any of the assignments as a starting point. Our expectations here are in line with the limited time that remains for the course. One simple choice would be to develop a scene that further explores some of the features of the `three.js` API.

Submit your assignment electronically using: `handin cs314 a4`