CMSI 371-01
Computer Graphics
Spring 2014

Assignment 0415

This assignment seeks to wrap up scene rendering so that we can move on to dynamic/interactive behavior.

Outcomes

This assignment will affect your proficiency measures for outcomes *1c*, *2a–2c*, *3a*, *3d*, *3e*, and *4a–4f*.

Proficiencies of **+** can now be applied to all outcomes in this assignment.

Not for Submission

If you have been following along with the Angel textbook, at this point, with the exceptions of Sections 2.11, 2.12, 3.14, and 4.10, we have pretty much covered Chapters 1–4.

The lighting model that we have seen, plus a few more details, can be seen in greater detail in Angel Chapter 5 and the orange book Chapter 9.

And of course, the C or C++ code in those chapters must be superseded by or adapted into JavaScript and WebGL.

For Submission

For the following tasks, keep building on *homework/pipeline* on your `git` repository. Do rename files, however, to better reflect what you have going now (yes, that means no *hello-webgl.html*s etc. anymore—you're way past "hello" now!).

The New Normal

Add normal vectors to all of your shapes, particularly your sphere implementation. You may use any technique for generating them, including (correctly) using the functions given to you, writing code of your own, and manually specifying them (ouch, but if you insist on doing all that typing, then knock yourself out). You want to do this because you will then…

…Light It Up

Implement a lighting model for your scene. At a minimum, you should use the model shown in class and detailed in the reading. You can go beyond that if you wish (e.g., the UberLight model described in Chapter 12 of the orange book).

Complete Your Scene

The title says it all. Use transforms liberally to position, rotate, and scale objects. Use projection and camera/view matrices to get full flexibility in terms of framing and displaying your scene.

You are "The Architect"

In accomplishing the above, you will need to fill out your matrix library (see what I did there?) with the remaining useful transforms. Building the camera matrix will also require vector functions. Implement what you need.

Commit and push your work to your `git` repository under *homework/pipeline*.