CMSI 371-01

Computer Graphics

Spring 2013

Assignment 0404

This assignment seeks to wrap up the geometry and your basic scene content so that we can move on to dynamic behavior and the fragment shader.

Outcomes

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

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. And of course, any C or C++ code in those chapters is 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!).

Add Transforms to Your Objects

Give your objects one or more transform properties (typically the *instance transformation*: a composition of rotate, scale, and translate) and extend your scene drawing code to apply those transforms. Yes, you will need to touch the vertex shader. Yes, you will likely need something similar to the 2D `canvas`'s `save` and `restore` functions.

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*.

Up Next

Some of you might benefit from knowing what is next, and you may want to work on it even now: dynamic behavior. You will be asked to adapt a variety of approaches (as appropriate to your scene) to bring things to life: tweening, user interaction, and physics, to name a few. This stage becomes highly individualized, and so deserves this advance notice. Keep this next step in mind.