CMSI 371-01

Computer Graphics

Spring 2013

Assignment 0418

As advertised, this assignment (your second-to-last!) focuses on interaction and dynamic behavior. Specifics can't really be stipulated here because this will be very dependent on your scene. But just have *something* that isn't trivial. Definitely do more than what the sample code has!

Outcomes

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

Not for Submission

The previously mentioned readings from the Angel textbook, if you have that, remain relevant to this assignment (and may actually be more meaningful now that you have a bigger picture of things).

For Submission

Take the Red Pill

Build on a variety of sources to add interactive or dynamic elements to your scene. Some examples:

- Adapt the keyframe tweener library from Assignment 0212 so that it can be used with your scene (*Hints:* The easing functions don't change; your scene's objects should get a `keyframes` data structure; and finally the biggest changes you'll make code-wise will be in your periodically-called scene-updating function)
- Set up JavaScript event handlers (if these are not familiar to you, talk to me and/or read Chapter 6 of the JavaScript text) to enable interactive navigation, viewing, or any other user-driven manipulation of your scene. There are many ways to do this; Angel Sections 3.13 (page 180) and 4.9.3 (page 247) offer some ideas, but you don't have to restrict yourself to those. In the GitHub bazaar, the *sierpinski-webgl* sample from CMSI 370 illustrates how you might do interactive mouse rotation. "Hitting" specific objects in your scene will require *unprojection* (i.e., the inverse of projection—you are moving from 2D space to your 3D world space).
- Implement physics-based behavior like gravity, acceleration, or anything else. You may integrate a physics engine or roll your own simple one.

Remember that these approaches are largely orthogonal and therefore can be mixed and matched. Note also that although some changes to your model and view code may be necessary, they should be quite distinct and separate from the actual dynamic behavior. In fact, these changes may have no effect at all on the view code, with the model affected mainly through additional properties or data.

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