Project 2

Due: 20 April 2015

This is the basic assignment: model a scene with multiple objects including a sphere, a cylinder, and either another sphere or another cylinder, with multiple light sources and controls to adjust the locations of the objects, the light sources, and the shading model; plus some additional element of your own.

Now let's break down what that means and clarify the constraints.

Elements

The scene should include at least one sphere and at least one cylinder. The sphere will probably be based on the one we've been designing; the cylinder you'll have to write additional code for, but you can use the concept of recursive subdivision to make this relatively easy. The scene should also have either a second sphere or a second cylinder. Don't make a whole new buffer for the duplicate object; you should be able to update the uniforms and call `drawArray` again without re-binding the buffer, to get different visual results.

Movement of the objects in the scene should be controllable by the user. There should be some movement controls that move the scene objects en masse, and some movement controls that move only one of the scene objects, or moves them apart, or otherwise affects the scene objects differently.

Lighting should have at least two light sources and be user-controllable.

Shading should follow the Phong lighting model, including ambient, diffuse, and specular components, with different objects having different shading properties; user controls should be able to switch between flat (constant) and smooth shading, and between per-vertex (Gouraud) and per-fragment (Phong) shading.

The multiple objects represent a bit of a departure from our running example (and the book's), which you'll accomplish by a mixture of re-binding buffers and sending different uniform values to the shaders, between calls to drawTriangles. Many of the rubric items are available even without getting this working; the ones that require it as a prerequisite are marked.

The additional element (Marmorstein would call this "glitter") should be something that uses some aspect of WebGL lighting/shading or buffers or attributes and variables that we haven't really talked about. Some options include attenuating light with distance from the light source, implementing different kinds of light sources, or reading a mesh from a data file; but feel free to be more creative. Your addition shouldn't substantially interfere with the core functionalities described in the other paragraphs unless you talk to me about it first.

Rubric

The first twenty points (of 100) are for the very basics: do you include documentation that tells me where to go and what it does, and does your program successfully display a canvas that has something on it.

The rest of the points are for the following items, at 5 points each:

Scene: includes sphere

Scene: includes cylinder

Scene: displays three or more objects simultaneously, including at least one sphere, at least one cylinder

Scene: includes at least one repeated object type, without using a whole separate buffer of points

Scene: "duplicate" objects differ visibly in some shading-related property (such as diffuse color or shininess), (requires multiple objects)

Movement: some controls move the object(s) as a system of objects

Movement: some controls move objects with respect to each other (requires multiple objects)

Lighting: controls can change the location of (one of) the light source(s)

Lighting: scene contains multiple light sources at different locations with different properties (brightness and/or hue)

Shading: can show a flat/constant-shaded view of the model (with at least diffuse shading implemented)

Shading: can show a smooth-shaded view of the model (with at least diffuse shading implemented) (either per-vertex or per-fragment)

Shading: button or key switches between flat and smooth shading (either per-vertex or per-fragment)

Shading: when smooth shading is activated, different button or key switches between per-vertex and per-fragment shading

Shading: one or more objects displays visible specular reflection

Shading: different objects have different specular properties (requires multiple objects)

Additional: you add something and it works

Note that a full-point submission will generally be a single set of files, but partial-credit versions might be spread over multiple implementations: for instance if you didn't get multiple objects working, you might have one set of files displaying a sphere and another displaying a cylinder. If you do this you *must* clearly indicate in your readme what I'm supposed to be looking at for the different rubric items.

The "controls" described in several rubric items are up to you: they can be keyboard-based or can be buttons or some other HTML control, but however they work you should make sure they are clearly documented.