Design

In the Sceneview project, we had to construct a scene given a scenefile. I do the parsing of the scene file in the scene graph. Here, I split the scene construction into separate steps with respective helper methods. These methods include copying information about the lights, copying information about the scene primitives, and so on. The main task of this file is appropriately going through the scene graph object primitives and transformations. I treat this as a recursive function that accumulates a transformation matrix for each node. Each child node of the current node will receive the transformation matrix from its parent, and be able to "add" to it (in this case, adding to a transformation matrix means multiplying!).

In order to ensure that tessellation occurs only when necessary, I have a m_tesselated Boolean in SceneviewScene.cpp. After tessellation occurs, this Boolean is flipped, and we only draw the objects rather than re-tesselating them within renderGeometry().

One major design decision I made was to set the shape parameters to 10,10. As per the handout, I chose not to implement updating the scene basesd on GUI parameters. I chose 10,10 because it preserved much shape detail without compromising too much on runtime.

To test my project, I compared its results to the demo's results and found them to be the same (as far as I could see!). The only differences I spotted were when the scenefile used meshes or other extra-credit primitives that I didn't implement. I also ran the valgrind memory analyzer and didn't find any memory leaks.