**Assignment 3a**

**CS4610**

**Michael Rallo – msr5zb – 12358133**

**3/23/2017**

**Assignment 2b**

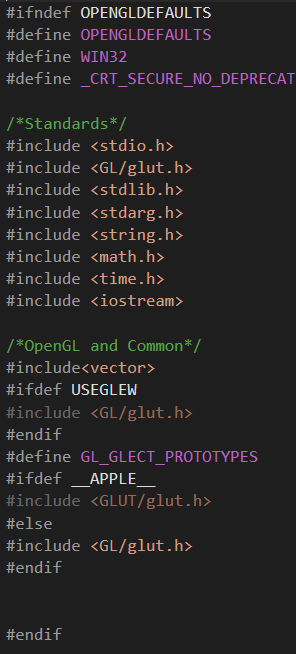
The *purpose* of this assignment was to familiarize ourselves with the OpenGL command of the hidden surface removal, illumination, shading and texture mapping.

Objectives:

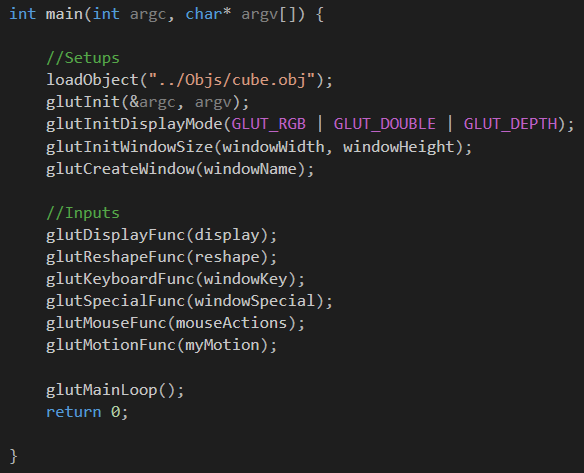
1. Interactively change the field of view and aspect ratio of the camera.
2. Interactively change the values of the near and far clipping plane.
3. Support for at least two light source.
4. Interactively turn light(s) on and off.
5. Support flat and Gouraud shading models. (Hint: In order to do Smooth/Gouraud shading, you need to provide the normal vector for each vertex using glNormal. The vertex normal can be computed as the average of all the faces/triangles adjacent to the current vertex as described in [here](http://faculty.missouri.edu/duanye/course/cs4610-spring-2017/assignment/ComputeVertexNormal.pdf)).
6. Interactive change the (RGBA) values associated with the global ambient light.
7. Interactive change the (RGBA) values associated with the ambient, diffuse and specular component of the light sources.
8. Interactive change the (RGBA) values associated with the ambient, diffuse and specular material properties of the objects.

Approach: Upon loading in a .OBJ file, I scale and calculate the normal vectors for each vertex. This was probably the most challenging part of the assignment. For each vertex, I had to go through all faces to check to see if it included the vertex in question. From there, I calculated and averaged the face normal vectors. Changing the field of view and aspect ratio was pretty easy, a simple variable was used which could be inputted via the keyboard. I added in two light sources (toggle-able) being default red and green. Shading modes can be toggled as well (decided by a simple variable). All RGBA values can be adjusted, though I kept the Green and Red lights to stay primarily those colors. View Keyboard input section for more.

**The Header File (OpenGLDefaults.h)**

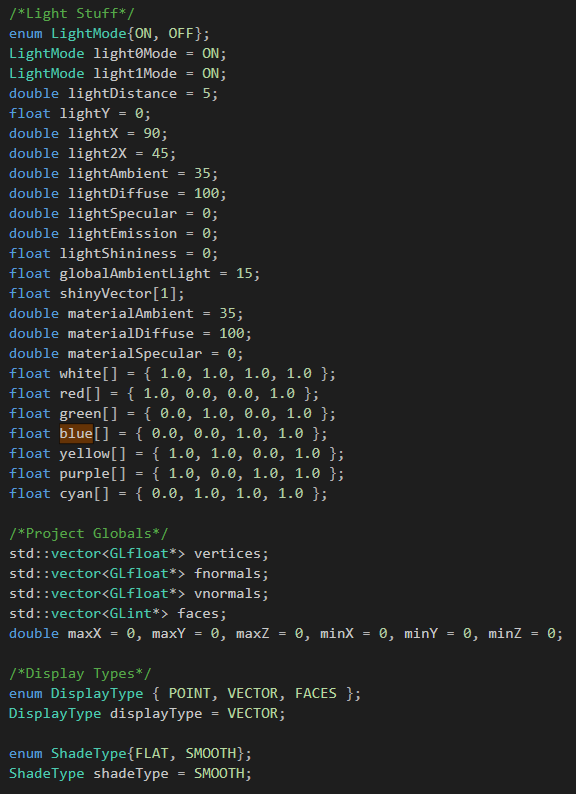
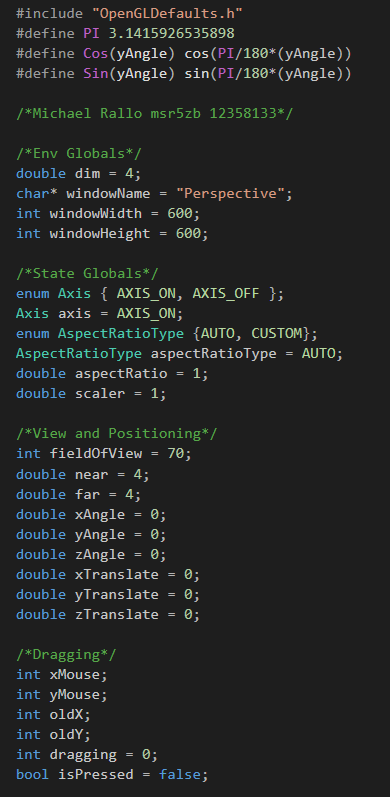
First and foremost, I have decided to include a header file to be used for this assignment’s, as well as future assignments’, libraries. Note this file include OpenGl basic libraries, as well as printing for debugging and math for easy/complex calculations.

**Main**



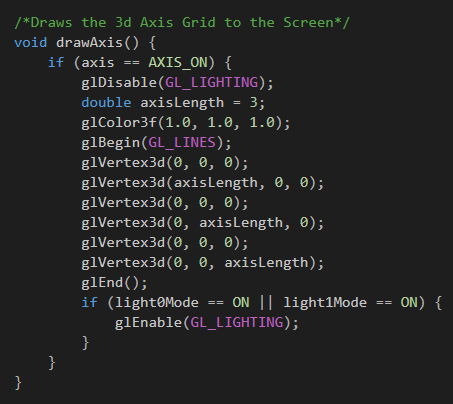
Our main function is similar to that from Assignment 2a. Note by default we load in the cube object file.

**Global Variables**



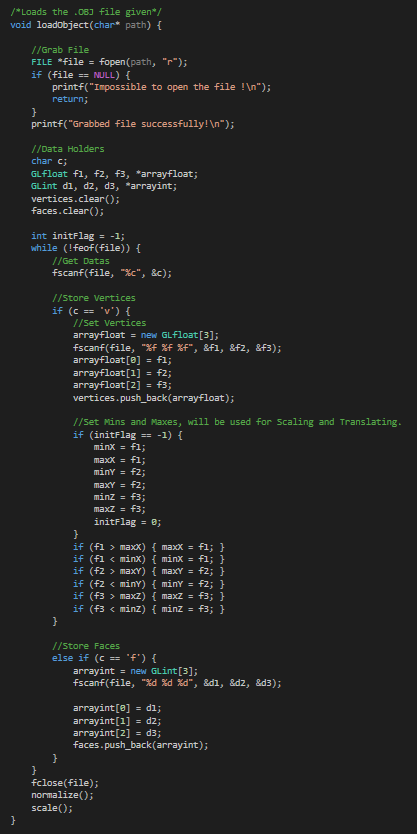
For this assignment, I will be using global variables in order to change various settings (named conventionally). Note the fieldOfView, near, and far to handle clipping and camera view. Lighting variables for the lights, environment, and objects were also created. Positions for lights will also be manipulated so we can see the full effects of lighting in our program.

**Drawing the XYZ Grid (Extras)**

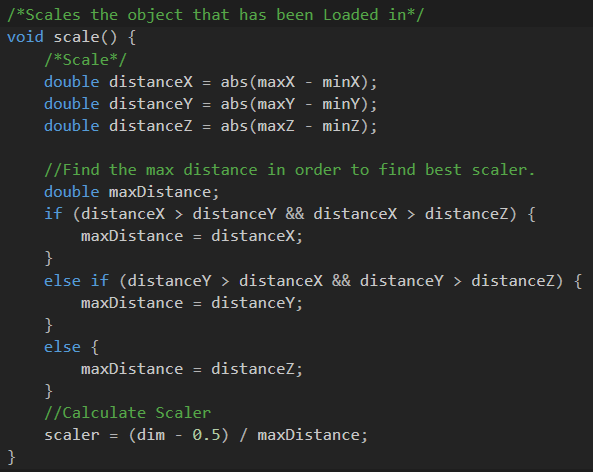


This is a simple function to draw a grid at the origin of our view in order for us to see the object more clearly. This can be toggled on and off with the “i” key. By default, it is on.

**Loading the File (loadObject)**

****

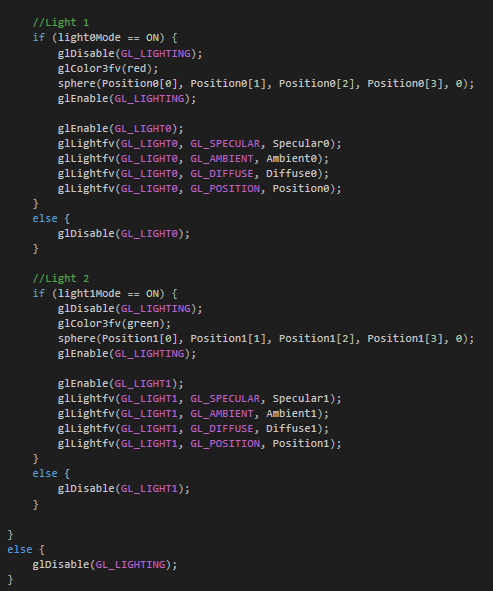
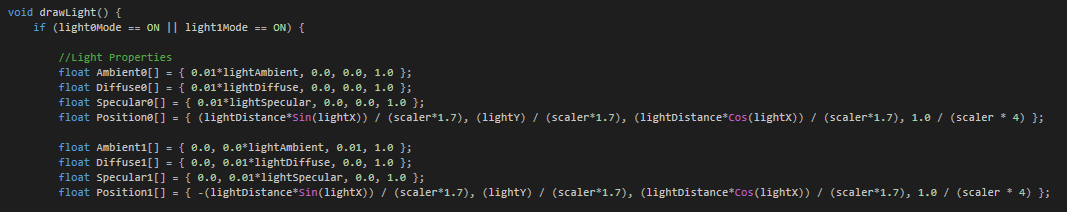
This function take the object the as a parameter and sets our global vertices and faces variables with the data the OBJ file contains. This function also sets the min/max values that will be later used for scaling/transitioning our object.

**Scaling (Scale)**

This scale method finds the greatest distance between the X, Y, and Z axis and uses that as a scaler for this Object. The reason we use the longest distance is so that we can scale everything equally whilst still being in our view.

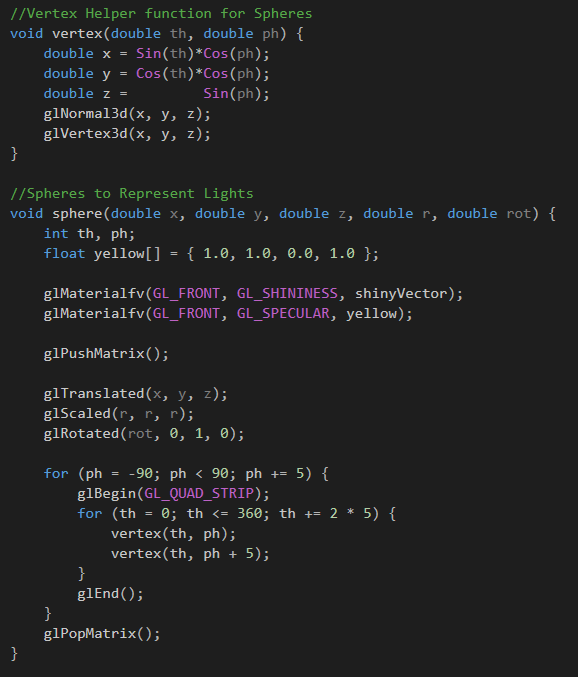
**LightSources**

This is our function to create our light sources. Note to adjustable RGBA values for each light. Also note how we use the scalar on the light source to keep a constant size throughout objects. Light Sources are toggle-able as well.



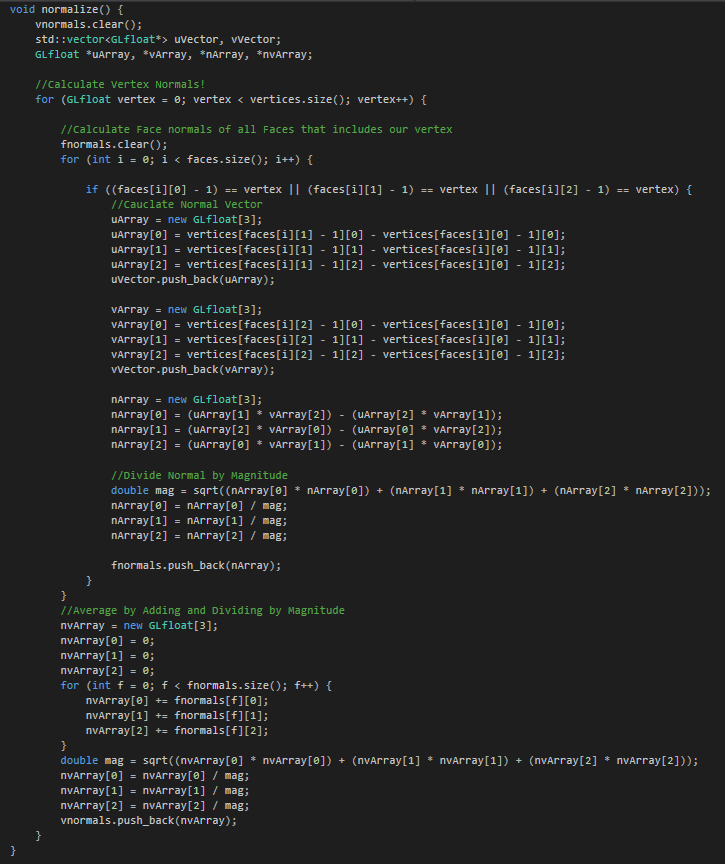
**LightSource Spheres**

**T**his function simply puts a sphere where the light source radiates from.



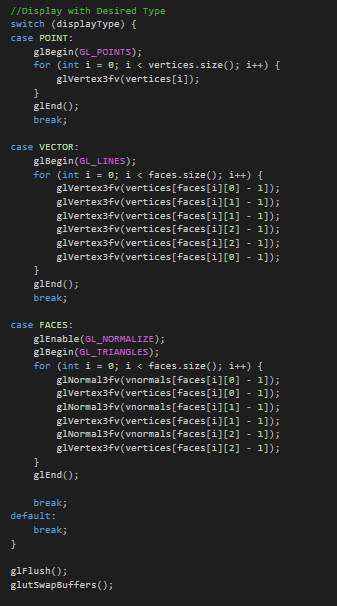
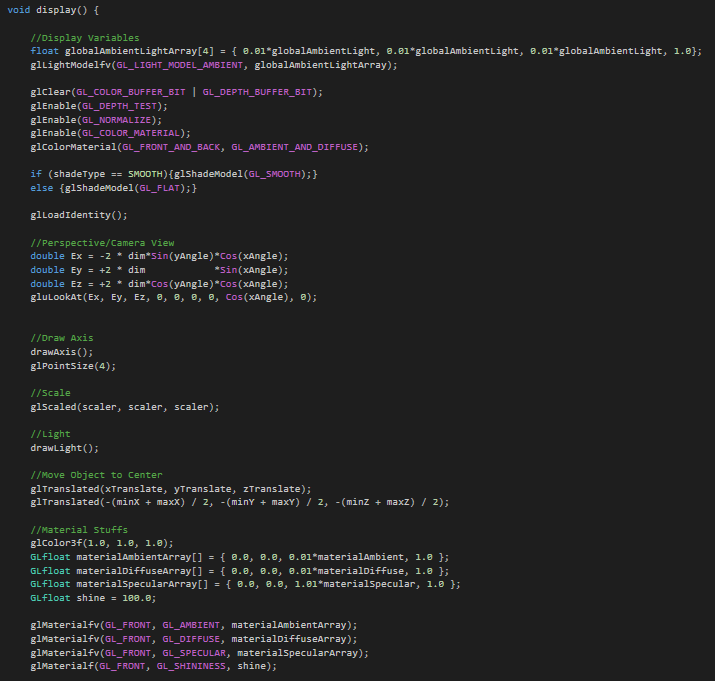
**Normal Vectors**

As described above, this is our ‘normalize’ function that calculates the normal vectors for all vertexes.

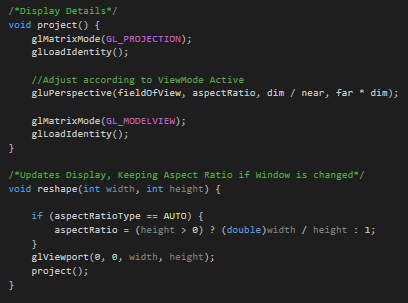


**Display**

Our display function has been updated to only handle perspective camera view. Important variables are also set here (depending on flags set). One of which is the Shadetype(GL\_SMOOTH/GL\_FLAT). Also notice the adjusted material values put in place for the objects to be created.



**Reshape**



Our reshape function is still very basic and has been redesigned to only handle projection/perspective mode rather than offering an orthographic option. Also note the Aspect Ratio Adjustability.

**Keyboard Input**

Key: Esc exits the program.

Keys: 1,2,3 loads in different Objects.

Keys: a,s,d changes display type.

Key: i toggles the XYZ grid.

Keys: +,- Zooms In/Out (field of view).

Keys: b,B Scales Objects.

Keys: t,T Changes the Aspect Ratio.

Keys: g,G controls the clipping range for the near value.

Keys: f,F controls the clipping range for the far value.

Keys: l,L toggles lights 1 and 2.

Keys: <,> rotates the lights.

Keys: [,],{,} moves lights.

Keys: o,O adjusts the Global Ambience.

Keys: h,H adjusts lightAmbient.

Keys: j,J adjusts the lightDiffuse.

Keys: k,K adjusts the lightSpecular.

Keys: n,N adjusts the materialAmbient.

Keys: c,C adjusts the materialDiffuse.

Keys: v,V adjusts the materiaSpecular.

Key: q toggles the ShadeType.

Keys: x,X,y,Y,z,Z Translates objects.

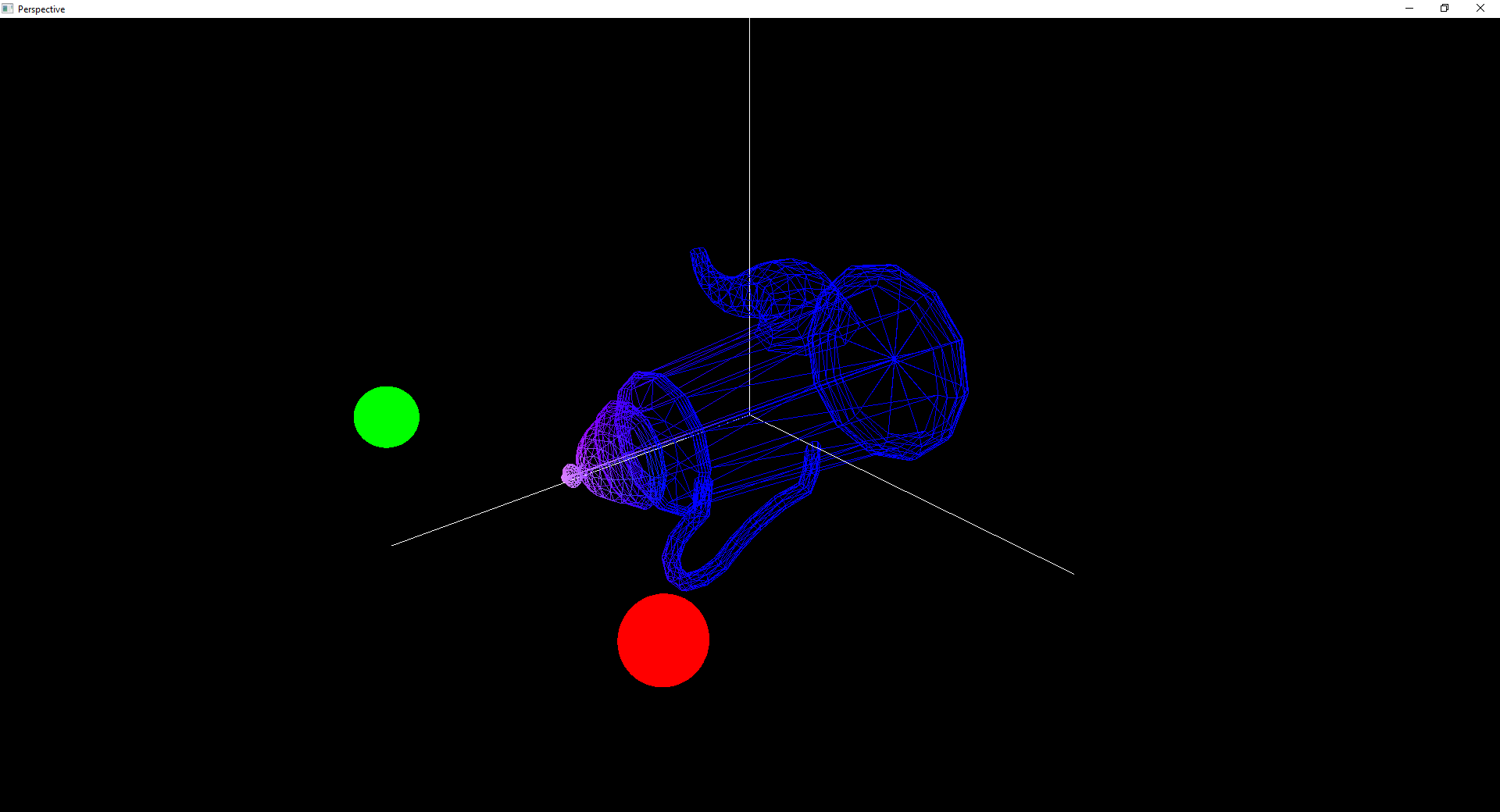
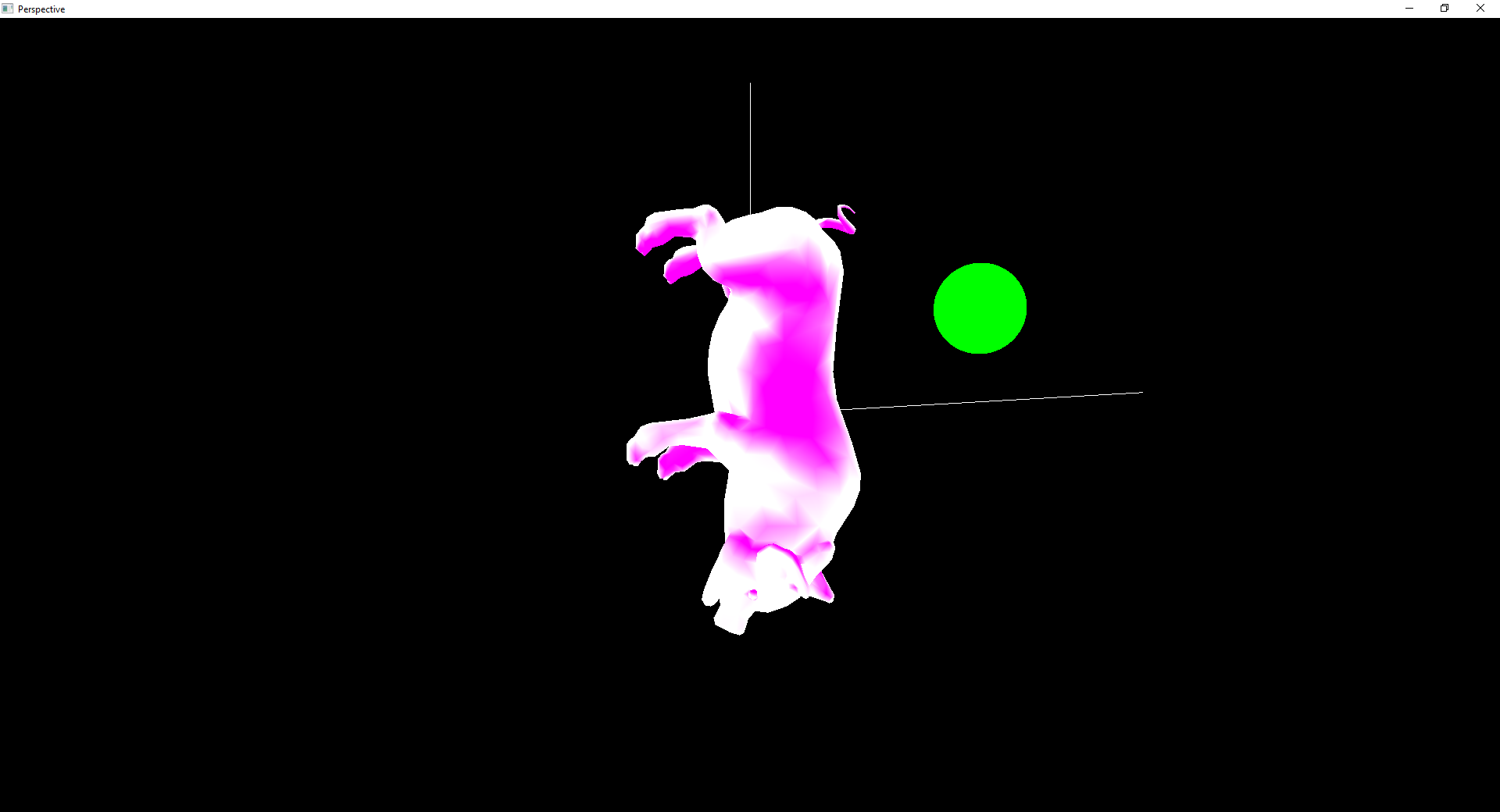
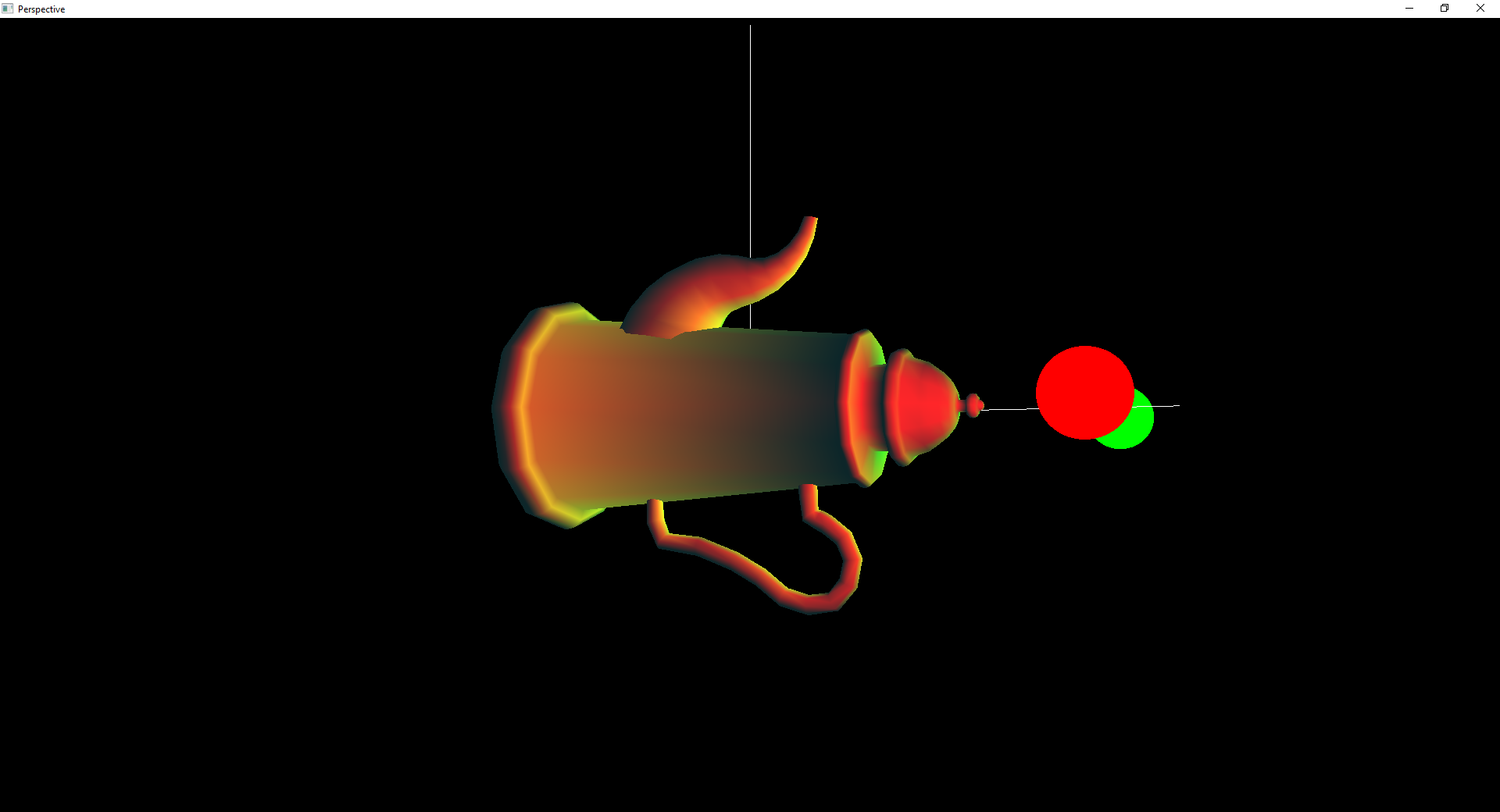
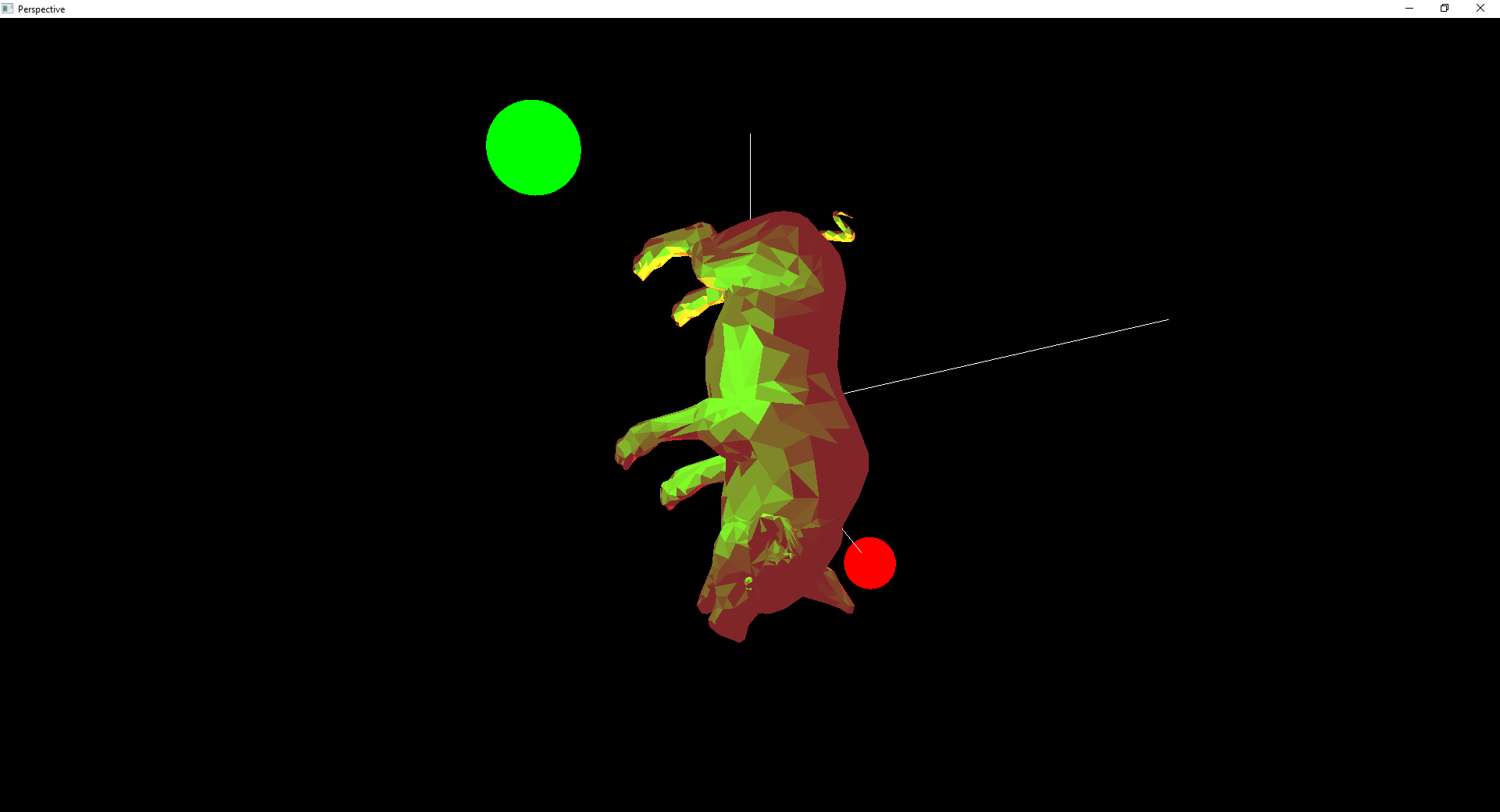
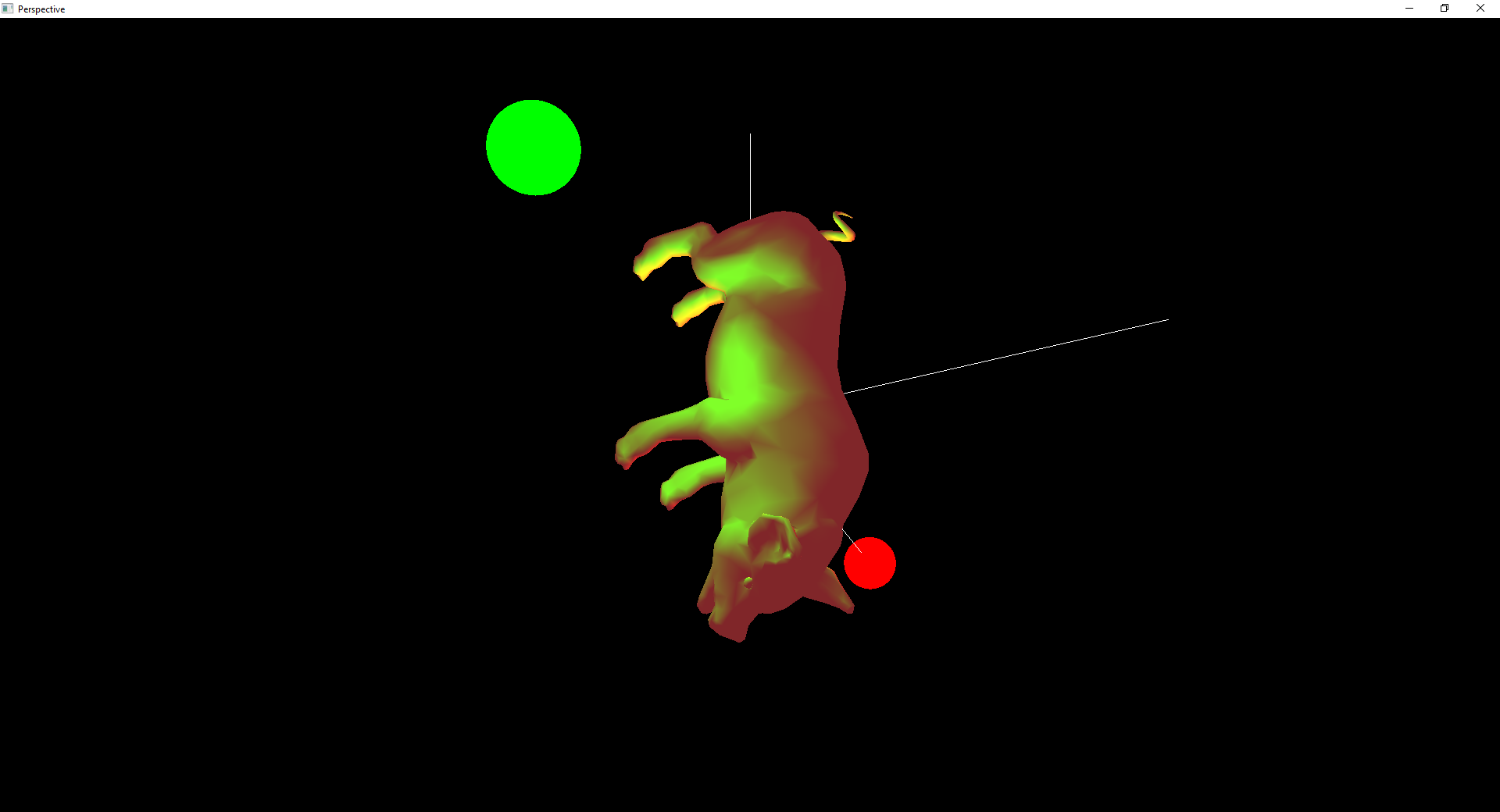
Key: r resets the object.

Arrow Keys adjusts/rotates object/view incrementally.

Click and Drag to Rotate Object, further you drag quicker it rotates.

**The Output**

The following are samples of outputs. Note how the Spheres represent the lightsources. Also note how the smoothing technique drastically affects how our objects looks. Adjusting the ambience, diffuse, and specular yields us very creative and interesting results as well!



The only issue I ran into is the loadtime when calculating the normal vectors for each vertex, though this may not be able to be helped.