COSC342 - Assignment 3 - Sean Moir

To implement passing parameters to the shader, I had to initialise uniform variables for the material details details, such as ambient color, specular color, opacity, light position, the texture and the specular exponent.

I did this by creating the missing getters in **Group.cpp** and passing the material data to the shader program through setting the data on the uniforms mentions earlier.

To implement the Blinn-Phong shading I copied MY lab solution to the OpenGL lighting lab and modified the Phong shading to use Blinn-Phong shading, by taking the half-way vector of the Eye and Reflection vector.

To implement the Transparency option in the render engine I created a uniform variable for the opacity of the material and then used that opacity for the alpha value for the result fragment in the fragment shader, I also had to enable OpenGL Blending for the transparency effect to work, I did this in the main **renderApp.cpp** file.

For the special effects I copied MY solution for the Render-To-Texture part of the OpenGL lab, I then implemented a listener in the render loop for keys 0, 1, 2 and 3 to changed to the specific special effect, these keys then there corresponding integer value to the PostEffects.cpp, which is then passed on through a uniform variable to the shader to use for deciding what effect to perform on the texture buffer, 0 is pass-through (no effect), 1 is a box-blur, 2 is a Sobel filter and 3 is a glass effect.

I tested this application real-time by making small incremental steps along the way and running the application to test whether the modifications I've made have improved or regressed the application, in the case of debug data, I used the debugger within my IDE and print statements to figure out what variables are set to.

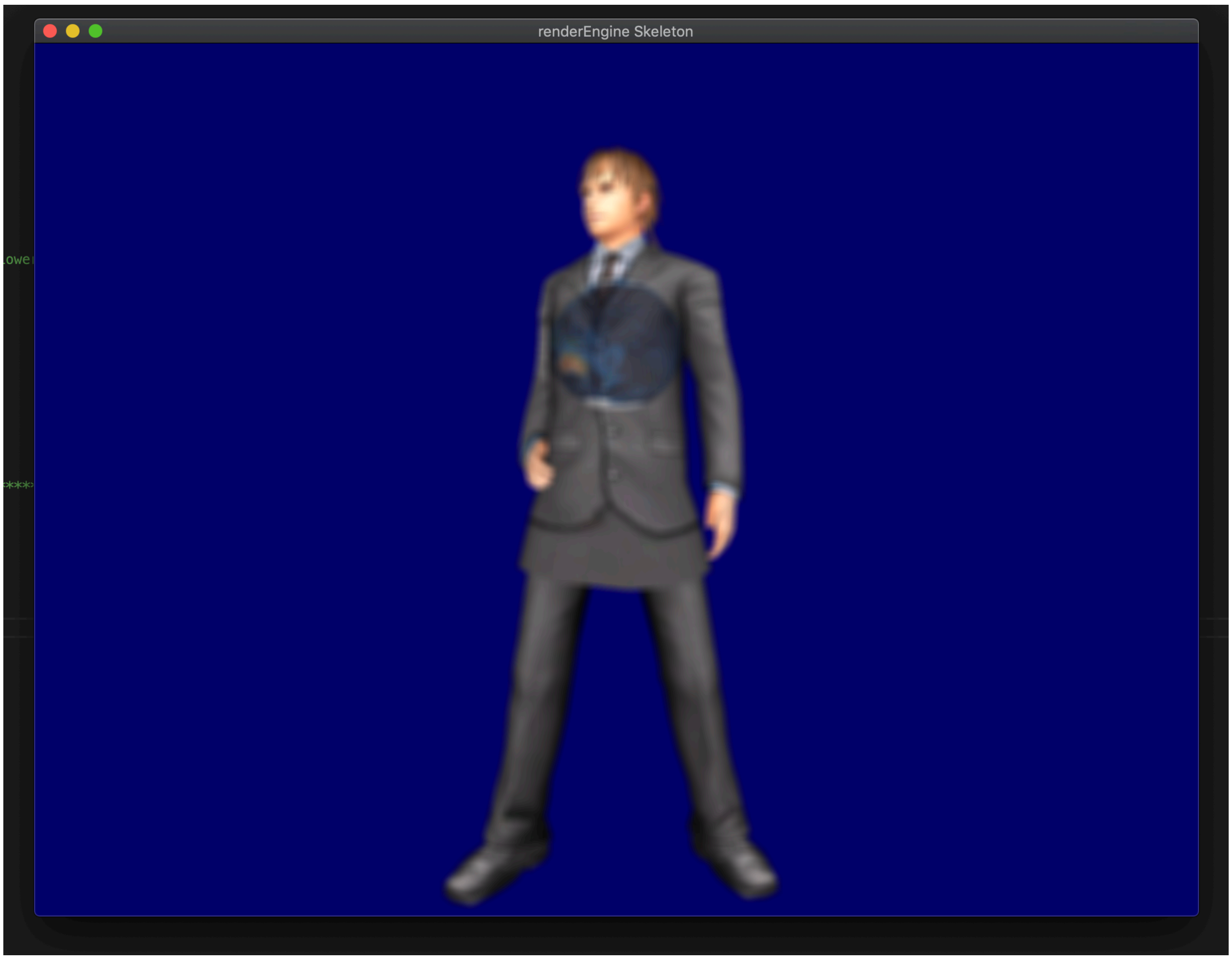
There doesn't seem to be a performance penalty between pass-through and the filters, there will be a performance penalty, but the application seems to be frame limited to 60FPS (16.67 ms/Frame) and thus, due to the light GPU load this program has, the special effects are efficient enough that they do not drop the program below ~60FPS.

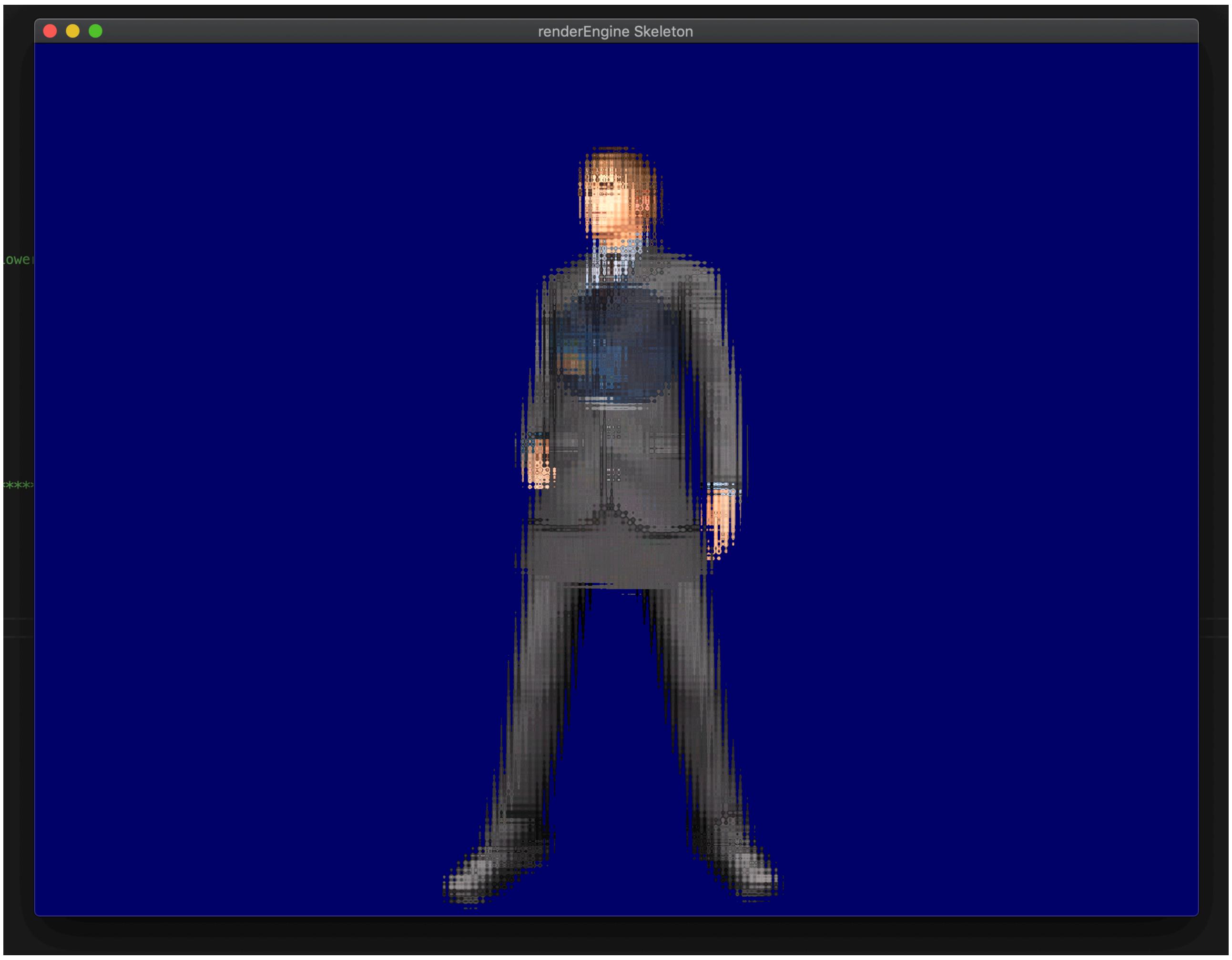
The only known (to me) flaw of this program is the specular exponent value (Ns) being passed through, **Objloader.cpp** seems to be passing a Ns value a lot higher than what is in the mtl files, this seems like it might have been an issue with the assimp library, and thus did not really have the time to fix.

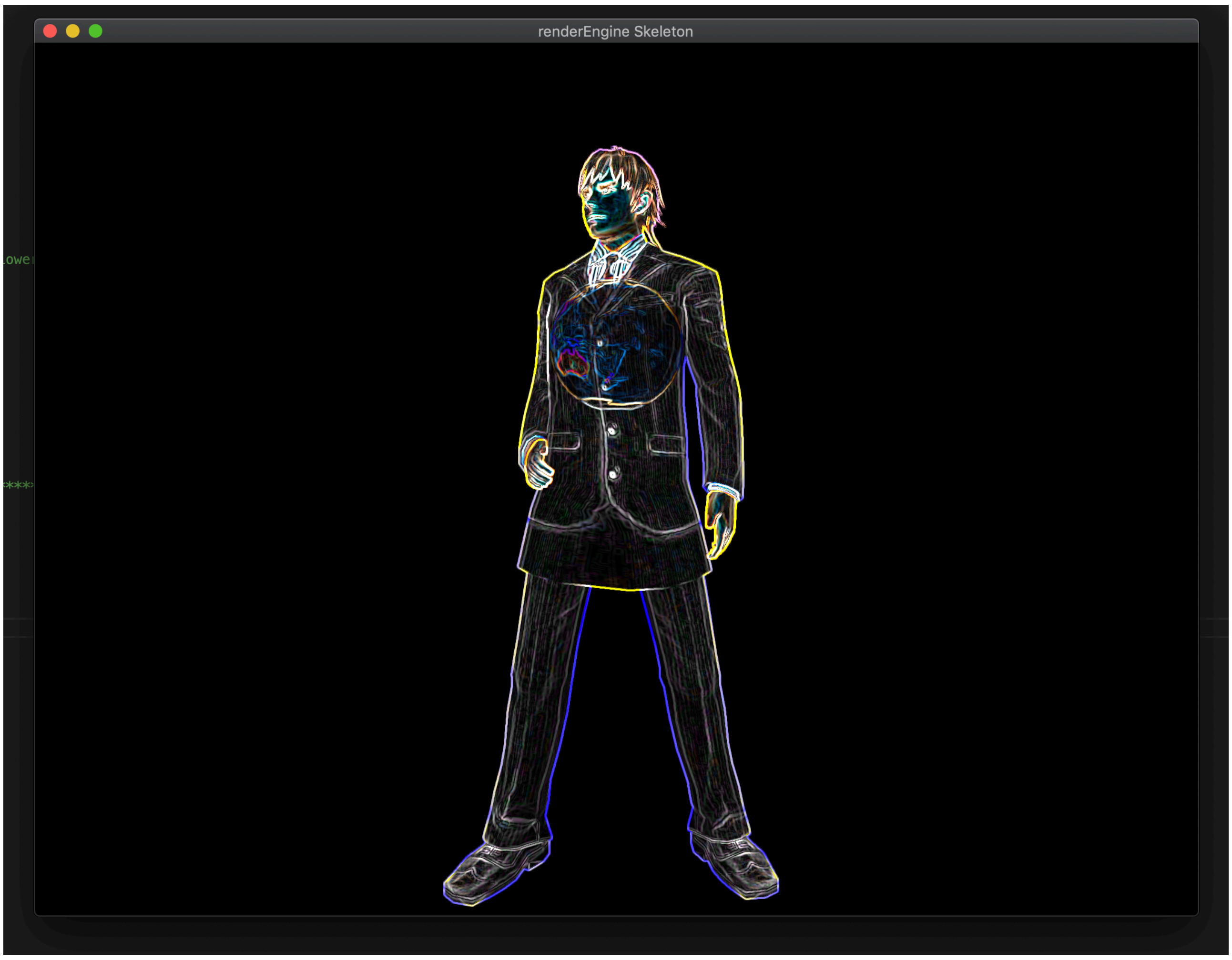
Hard to overcome bugs that I came across, mainly surrounded the transparency implementation, I was passing the opacity to the shader properly and setting the alpha channel of the fragment shader, but it did not seem to make a difference, I eventually figured out that I had forgot to enable GL_BLEND, this bug plagued me for a while.

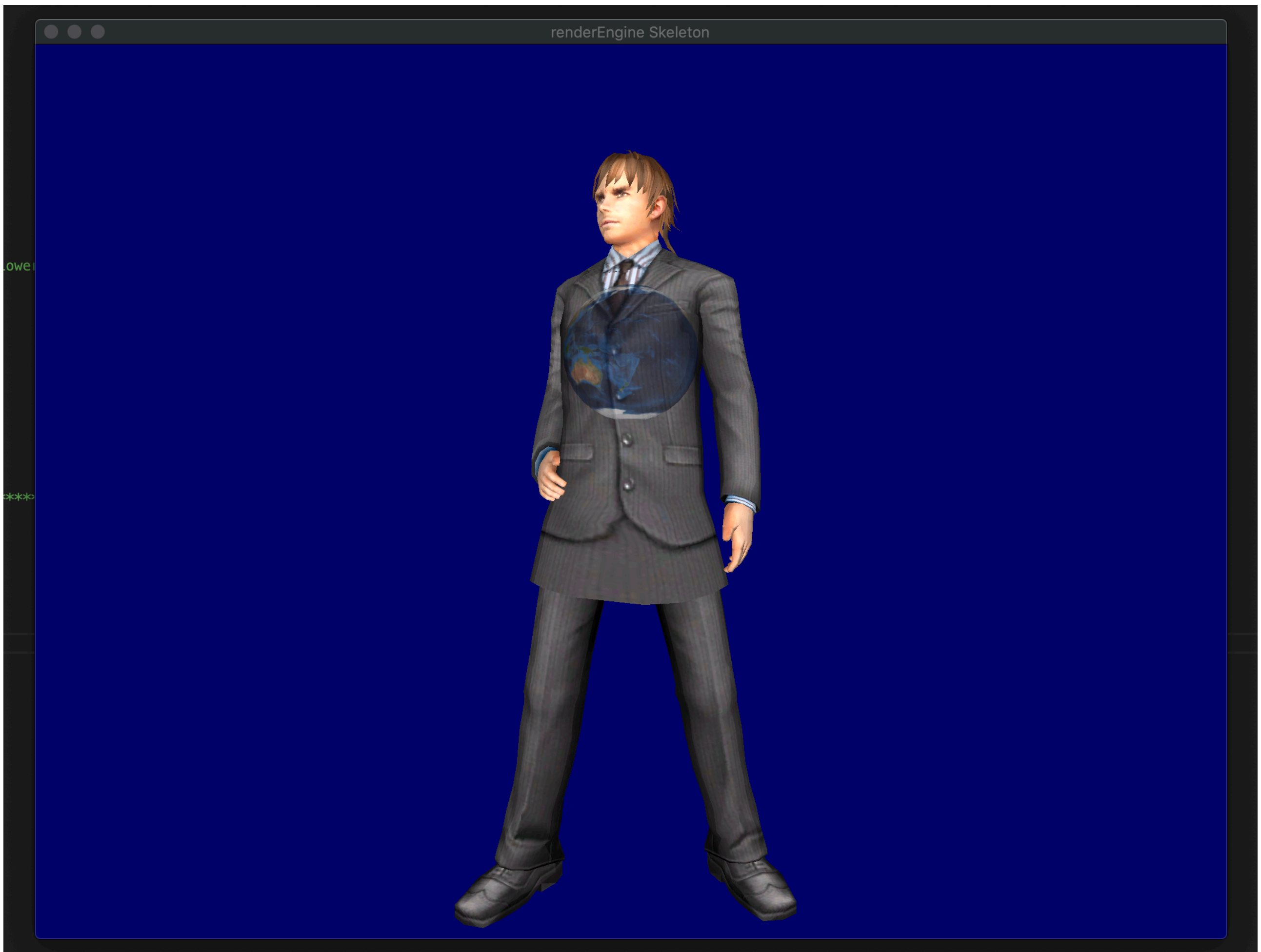
A lot of my errors resulted in a blue window with a frozen cursor, thus making it hard to get any screenshots of them.

Below are the end result of my special effects and a pass-through, original image.


renderEngine Skeleton


renderEngine Skeleton


renderEngine Skeleton


renderEngine Skeleton