CMPM163 HW3 Part 2

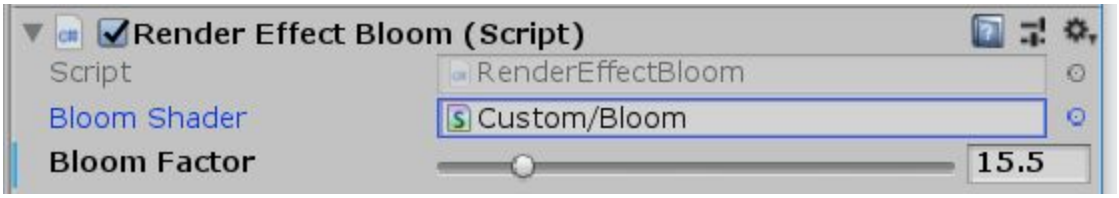
I am on a team with the following people: Jacob Daniels-Flechtner, Jason Chen, Oskar Alfaro, Junhao Su. It is a team of five people so I assume it will be expected of us to put forward more work than other teams. We all have some different ideas of what we would like to implement but we are all currently on the same page about what we want.

I would like to work with some of the post processing effects in Unity and implement them in code form. This is due to the fact that many of the post processing effects in Unity are very simplified. I want to create more realistic versions of these effects. Of the effects available, I would like to look into lighting the most, with an emphasis on ambient occlusion due to my other members being somewhat interested in the topic. Below, I have an image of my attempt to put ambient occlusion in an old scene I have made:

I used ambient occlusion to level out the lighting in this scene, to make the area feel a bit more even in terms of where the light goes. I tried to make the scene look as if it is sunset. However, I think this image could look much better. I do not believe that I can make a truly convincing sunset scene with the basic Unity ambient occlusion we have (this is the best I can do). Not to mention, learning how to calculate ambient light will be helpful in engines outside of Unity. So I will have to move to Shaders to make the scene better. For my final project, I will try to make a scene like this but with better ambient occlusion in order to really emphasize a sunset environment.

During HW2, we rendered blurriness in an interesting way. Basically, we were given a bloom shader we would use by mixing the image on screen with a blurry texture,

then we would have the camera reference this shader in the inspector like this:



Then, a script references the image of the shader and allows for user input on the intensity of the effect on the in-game image. This is how I would like to structure an ambient occlusion effect. First, I would like to look into the ways that Shaders calculate the diffusion of light. Ambient occlusion focuses on how exposed every area of the world is to ambient light, and I believe I can replicate this effect by increasing the diffused light from vertices in the shader. I would specify a diffuse color for the objects in the scene and add to it color corresponding to the ambient sources of light. Then, I would multiply the normal color of the frag component with this diffuse color. I would then link this shader to the camera with a script to edit the intensity using the method in the image above. I would play around the intensity until I find a compromise between overbearing diffusion and overly-subdued diffusion. If this process turns out to be too easy, I will experiment with other factors in lighting like how to make a pseudo-indirect lighting effect that does not cause Unity to run with a low frame rate (Unity's ability to do realtime indirect lighting could be much better as it causes considerable lag when it works) but this is just a stretch goal.