COMP 371 Nicolas Boyer, 40263939
Tiberiu Popa

COMP 371 Computer Graphics Final Report

INTRODUCTION:

The proposal, my project was described as a scene of a sidewalk or road with rain falling onto it, forming a puddle and reflection of a light source. The pavement would have a texture and simulate a wet surface due to being covered in water. The raindrops would also cause a ripple upon hitting the ground. The scene would also include other objects in order to make it look like a proper art piece or scene in a video game or animated movie.

The user can move the scene's camera to move around to view the scene at different angles. They can also use the enter key to disable rain, or the spacebar to disable the splashes the water leaves after the ground (this was not in the original proposal, but was an extra feature that I added for the fun of it).

OBJECTIVES REACHED/NOT REACHED:

The first objective of the scene was to render raindrops as transparent objects with reflective properties. This objective was partially met, as the rain simulation of the program does work, and the raindrops do have some reflective properties when looked at closely, as they will shine when exposed to light. This objective wasn't met fully due to a lack of time and inability to get the raindrops to render transparently without seeming completely invisible or hurting the program's performance too much.

The second objective was also partially met. The road in the scene does have a realistic texture, though it doesn't appear wet. This is due to the task of modifying a texture to seem wet within the program being too difficult of a task for me to achieve. I did find some wet concrete textures online that I tried using, though they did not repeat properly on a plane and thus I decided to use a nicer-looking non-wet road.

The third objective was met. There is a light source in the middle which is placed to simulate the street lamp object present in the scene, and the raindrops can sometimes be seen reflecting some of the light.

The fourth objective was also met. When a raindrop hits the ground, it makes a splash before disappearing. This can be turned on and off with the spacebar.

The fifth objective wasn't entirely met. There is a puddle present in the scene, and even though the raindrops do dissipate even when hitting the puddle, it doesn't have an effect on the puddle's geometry itself.

There were also some things that were mentioned in the project description that weren't listed in the five objectives on the proposal. The scene features textured 3D models, which is an aspect that I mentioned implementing in the proposal, but didn't list within my objectives, even though it is an integral part of making a scene with rain in it interesting.

CHALLENGES:

Whilst mentioning the proposal, one of the challenges I encountered and that was made obvious to me was following the set of objectives I put for myself. Having completed the project, I can say now that I should've described my objectives more broadly to not only make things easier for myself but also leave more room for me to include more features. I should've included the rain being able to be turned on and off in my objectives as well as the models present in the scene, for example. The objectives I set for myself were way too difficult for my skill set when it comes to OpenGL, and I should've kept the scope smaller and focused on objectives that were easier to achieve.

Another big challenge I faced was getting the rainfall to actually work. I attempted many different solutions, which involved very complex particle systems and geometry shaders, and the solution I came up with was one I could understand better than all the previous ones I had attempted, though I do believe that the method that I ended up using (rendering all the raindrops as separate models) wasn't the most optimal for the program's performance.

Finally, the biggest challenge that I faced was getting model imports to work. As Assimp wasn't working and the professor ended up letting the students use Tinyobjloader for our projects instead to work around the linker errors that me and others were facing on docker. Tinyobjloader did function better, but it was difficult to use due to the lack of instructions on how to parse obj files through it present online compared to Assimp. Loading models was one of the features that took me the most time throughout the whole project to complete.