Ramneek Chhatwal

Dr. Zordan

CPSC 4160 –2-D Game Engine Construction

Final Report

Moontian – Escape From The Planet Final Report

In this game, the character being played as is an astronaut that lives on the Moon. Much like how someone would live on Mars be a Martian, the playable character is the Moontian. Since the Moontian is the first person to live on the Moon, it goes over to different planets to try to figure out how to better progress its life and create a civilization on the Moon. Unfortunately, during one of these expeditions, there is a crashed landing on one of these foreign planets and the Moontian's rocket ship has broken into many pieces. The new mission is for the Moontian to collect all of the rocket ship pieces that have broken off in order to repair the ship and return back to the Moon. All this needs to happen while dodging incoming asteroids, which will kill the Moontian on impact. For the current iteration of the game, there are 5 rocket ship pieces that are needed to be collected in order for the game to finish with the Moontian getting safely off of the planet and back to its home. One wrong move, however, and the Moontian gets struck with an asteroid and dies with the game being over in a loss.

For my implementation, I made a class for most of the features of the game. I felt that there were enough differences within the different objects that a class would be a good idea. In most of the classes, I have different implementations of the constructor, update, render, and destroy functions. I originally had intended on using inheritance with at least the game objects in the form of an overall GameObject class. I would've had a rudimentary update and render function and I would override them in the regular object classes if I saw fit. When I was trying

to do this for the first assignment, I ran into a lot of linker errors and I chose to go the route I did instead, so I didn't have to spend as much time trying to figure out the issues that I had caused. Two of the classes that I have are actually managers that deal with more resource consuming features of the game. Those would be the particle emitter manager class and the text manager class. For the particle emitter class, I implemented it so that there was a class for just the individual particle and class that holds all of the desired particles and can render them. I chose to implement the text manager class as Dr. Zordan said in class that rendering text using ttf would be pretty resource intensive. The way I put together this was for the class to hold the text desired and only change the text when there was a need to, instead of having to re-render the text each game loop iteration.

Within my game engine class, I have an instantiation of each of the other helper classes that I can use to call their functions. I have all of these instantiations in a separate "init" function in the game engine class whereas the constructor only defines the game window, game renderer, and calls SDL, IMG, and TTF init. The main structure of the functions of the game engine go toward handling inputs, updating, and rendering. Within all of these, the engine will call the appropriate functions of the objects that need to have their variables changed. For example, only the game objects (astronaut, asteroid, and rocket ship pieces) need to have their x and y position updated, so their update functions get called in the game engine's update mechanic's function.

Throughout this semester, most of my challenges came down to how I was putting together all of the pieces of the code. I do feel like my overall organization of the project and implementation was good, but that did take most my time with this project. I spent many hours writing out what I wanted each class to do and how I wanted them to interact with each other in the most efficient way that I knew of. This is the first project that I had where I wasn't given

either starter code or very, very strict guidelines of how to do things and that challenged me a lot. Another part that I really was challenged with was that I'm not very artistically inclined, or overall creative for that matter. This project really stretched my abilities in both of those categories. I'm sure there are a lot more ways that my game could look better with different assets, color choices, font type, etc., but I am happy with what I was able to accomplish.

Going back to my challenges, one of the biggest things that I learned was code organization. I had a project over my internship over the summer where I was doing R&D by myself and I really struggled on how to come up with a finished project from scratch. I feel like if I had this experience before (also with this class being my first technical elective in my curriculum), I would've had more success with that project before. But now I am able to use this for future projects. I was also able to learn how to use more of the full extent of a library that doesn't necessarily have the most resources with it. The SDL documentation is good in its own right, but it did take some digging to find some examples of how to use certain functions within SDL, which is how I learn best. I know now that I am able to take on an unfamiliar library or framework and get what I want out of it with the right research and trying different ways of doing what I'm trying to accomplish.

In conclusion, this was probably one of the most fun and educational classes that I have had at Clemson, and I really do thank you for that, Dr. Zordan and Ethan. I've always had an eye on what it was like to make a game from scratch and since I know that I'm not necessarily the most artistic person, I feel like working with game engines could be a good spot for me to look to. I have applied to a couple of game development companies as a potential for a summer internship. For the future of this project, I plan to make it the best that I can for personal use. I don't plan on offloading it to Steam or another platform, as I did take most of the assets from the

internet with personal use licenses. There are more things that I would like to add to this game that I haven't been able to implement as of now, such as audio and different levels. I would even want to try to stretch the SDL library to its max and add video too it for the start and end screens. I currently have this project in my GitHub portfolio and I do plan on working on it during winter break to keep adding a more and more complete project to my portfolio so I can show what I was able to learn from this class.