**Assignment Self Evaluation Sheet**

**Programming for Graphics and Games Assignment 2**

**Student’s Name**: Jake Walder

*This self-evaluation sheet is marked only on completeness (i.e. please be honest!). The purpose is to help you reflect on your performance and to help identify features of your work.*

|  |  |  |
| --- | --- | --- |
|  | **Yes** | **No** |
| Did I complete the minimum requirements for the assignment? | ✔ |  |
| Did I add any extensions to the assignment? |  | ✔ |
| Did I read up on the subject beyond lecture / lab contact? | ✔ |  |
| Did I spend enough time on the assignment? | ✔ |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Very happy | Satisfied | Disappointed | Ashamed |
| How happy am I with what I submitted? | ✔ |  |  |  |

|  |  |
| --- | --- |
| What sort of mark do I expect? | **60%** |

|  |  |
| --- | --- |
| References for sources and tutorials I used: | Free 3D model for the spaceship: https://free3d.com/3d-model/low-poly-spaceship-37605.html |
| Main features of my project: | * Intuitive and responsive gameplay * Multiple collision detections * Hierarchical objects with different states * Game object spawning randomised * Changes to shader code * Use of dynamic vectors |
| The best part of my performance was: | I am happiest with the fact that I have completed a full working game with no known bugs and an increasing difficulty. |
| The worst part of my performance was: | My management of gameplay code could be better, I would reorganise the placement of a lot of it (e.g. the movement commands). |
| One way in which I could improve my submission is: | I wish I could have added a scoring system but by the time I finished the game I struggled to find a simple way to implement text using OpenGL. |
| One thing I will do to improve my next submission is: | Plan everything that I want to include in the game and make sure I read up on how to perform the tasks in the most efficient way possible, rather than just adjusting on the fly. |

**Report**

**Introduction**

My game is a simple top down space shooter, where the aim is to survive as long as possible by dodging and destroying the asteroids that appear on the screen. The player is a ship that can move up, down, left and right and that can shoot the asteroids that move towards the ship. Over time more and more asteroids appear on the screen and some of them won’t be destructible meaning they have to be dodged.

**Previous Work**

I have previously worked in Unity and during my learning process I completed one of their tutorials called Space Shooter. This was a fairly simple game that when this assignment came around I decided to replicate from scratch in OpenGL, except slightly more advanced. Unity provides lots of features especially in particle effects and physics, so trying to replicate this would prove a lot of work in OpenGL. This game includes a concept the same as mine, except it lacks any increasing difficulty nor do the asteroids have varying states. I also wanted the user to have more control over the ship, so I gave the option to move the ship at half speed. Unity has an advantage where a lot of the physics and functions are already created, cutting down time required to make a game object, which I certainly struggled with as all four object types required different variables that I had to test by running the game. The game is similar in the movement, both ships perform rotations when moved left and right and move on a 2D plane within a set boundary. My ship does not have any inertia or acceleration which I wish I could have included to give it a more realistic feel. The asteroids also only move downwards towards the player and can be destroyed by shooting a laser. One thing I made sure to include in my project was two states of asteroid, one of which could not be destroyed. This forced the player to rely on the movement much more than the Unity tutorial. However, the rocket engine animation was not something I was able to include as I struggled with including particle effects in my game. On the other hand, I could compare my game to Space Invaders, the classic shoot ‘em up game. The biggest differentiation between the two is that Space Invaders includes set levels and waves whereas mine is purely randomisation. That means Space Invaders could lack ‘replayability’ and surprise, yet it does mean that it can be rigorously tested and designed meticulously. Space Invaders also includes a scoring system which I wish I could have implemented, as this gives the player a way to compare how they previously played and against others. My game is much more advanced in appearance as I have included 3D models to give a feel of visual depth and realism. This alongside the movement of the ship creates a more advanced game, but that’s not surprising as Space Invaders was originally released in 40 years ago!

**Design and Implementation**

The game is a simple design, using a standard game loop and utilising the SDL and OpenGL libraries. All the objects in the game inherit from one game object class, making it easy to control all the objects from one class. The game loop has three sections, managing inputs, updating the game and drawing to the screen. Using this it is simple to manipulate the game in ways such as manipulating the player, calculating collisions and moving and randomising the objects. I also use vectors to store any type of object that there will be multiple of on screen to organise and make them easy to calculate and draw.

**Analysis**

I’m overall happy with the work I have created, however it’s not without it’s weaknesses. Its main strength is within how well the game plays. As it is a very simple game, it was easy to polish everything I wanted in the game. The increasing difficulty (gained by the two types and increasing number of asteroids) gives a sense of reward to the player which the unity tutorial didn’t have. It also didn’t have the immersion gained by my moving star background, so I am pleased to have got that in. I am also proud of the maths I used for my collisions, however they struggle to compare with Unity’s tutorial as that uses collision boxes for better accuracy. The weaknesses in my game come mainly from lacking features. One feature that I fell short of getting in was scoring or any text in the game. OpenGL doesn’t have a simple way of inputting text which caused me to put other features of the game above it. I also never managed to get multiple textures on one object and instead I just used Blender to morph the texture of my object into one bitmap instead. Due to the nature of my game being 2D, the camera is fixed into position and therefore doesn’t have any complex functions, which could have been improved upon if I decided to make a 2.5D game (a game using 3D assets and a camera that pans around the world, but the game moves on a 2D plane). The Unity tutorial also uses particle effects which my game has none of; an explosion upon destruction of the asteroid would have been ideal but time simply limited this. Finally, I would have loved to put some more advanced physics and animations in the game (i.e. barrel rolls, acceleration) but I had to keep it simple for the sake of polishing the game.

**References**

chrisb@gamespy.com, C. (2018). *GameSpy.com - Hall of Fame*. [online] Web.archive.org. Available at: https://web.archive.org/web/20080408152913/http://archive.gamespy.com/legacy/halloffame/spaceinvaders.shtm [Accessed 11 Jan. 2018].

Cplusplus.com. (2018). *cplusplus.com - The C++ Resources Network*. [online] Available at: http://www.cplusplus.com/ [Accessed 11 Jan. 2018].

En.wikipedia.org. (2018). *Space Invaders*. [online] Available at: https://en.wikipedia.org/wiki/Space\_Invaders [Accessed 11 Jan. 2018].

Free3d.com. (2018). *Low Poly Spaceship Free 3D Model - .obj .dae .blend .fbx - Free3D*. [online] Available at: https://free3d.com/3d-model/low-poly-spaceship-37605.html [Accessed 11 Jan. 2018].

Opengl-tutorial.org. (2018). *OpenGL Tutorial*. [online] Available at: http://www.opengl-tutorial.org/ [Accessed 11 Jan. 2018].

Player's Guide To Programmable Videogames. (1981). Electronic Games.

Unity. (2018). *Unity - Space Shooter tutorial*. [online] Available at: https://unity3d.com/learn/tutorials/s/space-shooter-tutorial [Accessed 11 Jan. 2018].