

Computer Games Development CW208

GDD and Project Report

Open Ocean

|  |  |
| --- | --- |
| Ruslan Gavrilov  c00273521 |  |
| [06/03/2023] | |

Contents

Acknowledgements 4

Project Introduction 4

Game Overview 5

Feature Set 6

General Features 6

Gameplay 6

The Game World 7

Overview 7

Key Locations 7

Travel 7

Scale 7

Objects 7

Rendering System 9

Overview 9

Camera 9

Overview 9

Game Engine 10

Overview 10

Game Engine Detail #1 10

Collision Detection 10

The World Layout 11

Overview 11

World Layout Detail #1 11

World Layout Detail #2 11

Game Characters 12

Overview 12

Creating a Character 12

Enemies and Monsters 12

User Interface 12

Overview 12

User Interface Detail #1 12

User Interface Detail #2 12

Musical Scores and Sound Effects 13

Overview 13

Sound Design 13

Story 14

Hours of Gameplay 14

Victory Conditions 14

Character Rendering 16

Overview 16

Project Milestones 17

Project Review and Conclusions 17

References 17

**Acknowledgements**

I would like to thank the following people who assisted in completing this project including;

Philip Bourke of SETU for guidance and help in the creation of this project

SETU for the opportunity to show my project and represent the college

**Project Introduction**

This is an endless runner project set in the ocean. It was chosen to utilise swimming physics and motion sensing

**Game Overview**

Common Questions

What is the game?

The game is an endless runner focused on swimming as far as you can and avoiding obstacles and fish and collecting shells and pearls.

Why create this game?

I'm creating this game to differ from common endless runners and learn and utilise motion controlling so the player feels more immersed.

Where does the game take place?

This game takes place in an underwater cave.

What do I control?

You will be in charge of controlling the free diving swimmer.

How many characters do I control?

You control one main character in this game, The swimmer.

What is the main focus?

The player should swim as far as they possibly can while collecting shells and pearls and avoiding the obstacles and enemies along the way. Swim as far as you can!

What’s different?

This game differs from other endless runners by changing the scene to have more float like physics rather than the standard run, jump and kill mechanics. This game immerses you in the ocean environment and focuses on the players ability to dodge rather than attack.

This game also utilises motion sensing rather that keyboard or controller input, giving it a niche.

**Feature Set**

***General Features***

Underwater Scene

Underwater thematic

Utilisation of motion sensing

2D graphics

Lighting

32-bit colour

***Gameplay***

The player must swim through the world in order to keep playing the game and stay alive.

The player must avoid obstacles such as fish and mines in order to preserve their oxygen levels.

The player must collect bubbles that come off of plants in order to keep their oxygen level topped up.

The player should collect shells and pearls that appear from chests in order to obtain points, do this while avoiding obstacles.

The player must keep an eye on the squid approaching in the background in order to be prepared for its arrival.

The player must avoid the giant squid when it appears on the screen in order to not immediately drown.

Once the player drown his distance and score is updated and displayed before immediately restarting. The player is encouraged to beat their high score.

**The Game World**

***Overview***

This game follows an underwater scene.

The following describes the key components of the physical world.

***Key Locations***

Under water cavern that the player swims through.

***Travel***

The player swims through the world and moves forward.

***Scale***

The underwater scene is large and the player and different fish and enemies are scaled up to a proportion that feels realistic

***Objects***

Mines are large and in your face so that the player knows to avoid them.

The fish are scaled to be large enough but not larger than the player.

The Giant Squid is scaled to be large and imposing.

**Rendering System**

***Overview***

The game is rendered using SFML 2.5.1

***Camera***

***Overview***

The camera is ultimately stationary but comes off as moving as the background and game entities are moving.

**Game Engine**

***Overview***

This game runs off of the SFML 2.5.1 Library, Thor Library and Custom lighting libraries

***Game Engine Detail #1***

There are many headers and cpp files for each object and world environment that keeps track of the world and the event that occur in it.

***Collision Detection***

This game handles collision using square and rectangular hitboxes. It uses a custom version of squares colliding. Each object that can be interacted with has a hitbox that can be drawn in debug mode but ultimately not visible. Game actions are determined based on a collision event that is read in the main game file.

**The World Layout**

***Overview***

The world is a deep ocean scene that moves as the player swims along.

***World Layout Detail #1***

The world parallaxes to give it depth. there are ocean rocks and a variety of plants that cling to those rocks. This gives the game a more underwater feel.

***World Layout Detail #2***

Light penetrates the ocean from the cracks in the rocks above. This gives the world a biot more life. There is also deep oceanic light that come off of plants, chest, and overall environment.

**Game Characters**

***Overview***

The character is a free diver in a blue suit. The player will control the swimming of this player.

***Enemies and Monsters***

This game offers many different kinds of enemies, the first being Fish:

The black fish : Slow moving, bigger than other fish, has a chase mechanic when it goes off the screen, if hit while chasing the player is immediately killed, otherwise you lose 5 oxygen.

The Red Fish: the base fish enemy, average movement, if hit the player takes 5 oxygen damage.

The Yellow fish: A more narrow fish that moves the fastest of all enemy fish, 5 oxygen is taken if hit.

Next is the Mine:

The mine is slower moving and floats slightly up and down. Being a mine, getting hit by one it causes an explosion and the player loses 50 total oxygen. The player should prioritise avoiding the mines at all cost as hitting a total of 2 without picking up oxygen will cause the player to drown.

Finally there is the Squid:

The squid comes to the forefront over time. The player may notice a shadow getting closer in the background. Once appearing properly to the screen the squid will begin to follow the player. Upon a random moment the Squid will chose to attack the player, he will move in an up and down pattern at a fast pace to try and hit the player. He will do this dash attack twice before returning to swimming in the background and repeat. If the squid hits the player then it's game over as he is seen as a kind of boss enemy.

**User Interface**

***Overview***

This game uses a very simplistic UI in order to not cloud the screen while the player is playing.

***User Interface Detail #1***

There is an oxygen meter in the top left which displays the amount of oxygen the player has. The amount is represented by a green bar that diminishes overtime, once the bar in not visible the player has drowned.

***User Interface Detail #2***

To the right of the screen you see the distance the player has swam in order to give the player some information for how long they have been playing. Under that is the amount of pearls the player has collected as they are the most valuable item in the game.

**Musical Scores and Sound Effects**

***Overview***

The music and sounds are focussed on underwater audio and a calming serene experience.

***Sound Design***

There is a constant under water bubbling sound and water movement in order to feel like the player is actually under water.

The breath sound utilises oxygen taken from a tank and uses scuba diver sounds to realise that goal.

The chest uses a wooden creaking sound and the bubbles use a bubble sound in order to use a realistic depiction of these sounds.

The mines use an underwater explosion that is slightly muted as sound travels differently under water.

The pick up sound is a simple pop, doesn't need to be more complicate as it just tells the player that you picked up and object.

The background music is a calming song so that the player doesn't feel stressed to swim faster but take their time and use their head to perform the right movement.

***Story***

You are a free diver, at the bottom of the ocean, your goal is to swim forward and collect as many shells and pearls as possible.

***Hours of Gameplay***

The game can last as long as the player is interested in beating their high score.

***Victory Conditions***

This is an endless runner as per the theme, the player is encouraged to keep going as far as they can, there is no win condition per say.

**Character Rendering**

***Overview***

Characters and environment are rendered using SFML 2.5.1 and displayer through it's library

**Project Milestones**

Movement of player

Movement of enemies

Addition of squid boss

Collisions

Pick ups

Parallax Background

Lighting

Score and distance system

Music and sounds

**Project Review and Conclusions**

Movement of the player was difficult to figure out as I was working with a new medium of a motion controller. I had to figure out the way the controller read the players hands and choose a suitable way for the player to move their hands in order to feel realistic. I settled on a swimming motions with closing a fist to go down.

After setting up a Parent class for enemies, it was easy to implement the different enemy functions and their behaviours, The most difficult was the squid.

Pick-Ups were the same, both the plant and chest have similar functionality so it was easy to replace certain variables and differentiate them. Collisions follows both of these milestones for which I utilised a library that I had previously written.

The most difficult to implement was lighting and adding a feeling of depth. Parralaxing of the background was the first step to add depth, but when adding a pre built library for light, the project wouldn’t run. I had to create my own light library in order to get the desired effect.

Sounds systems were implemented last and easy to find and use sounds that added to the feeling of the game.

This was a long project to do for one person, but it forced me to think outside of the box and create my own solutions rather than relying on previously build libraries.

**References**

**Web-site**

<https://www.swim-teach.com/types-of-swimming-strokes.html> : types of swimming techniques for game

<https://www.wwf.org.uk/what-we-do/planting-hope-how-seagrass-can-tackle-climate-change> : sea grass producing oxygen

<https://en.wikipedia.org/wiki/Deep-sea_fish> : deep sea fish