IMAT3905

Game Engine Development

Assignment: Group Game Programming

Module Leader: Peter Cooke

GitHub repository: <https://github.com/Hoodrn03/Game-Engine-Game>

Group Members:

Ryan

Michael

Dominykas Korklinevskis P16184502

Shafay

Table of Contents

[Introduction 3](#_Toc7614933)

[Outline 3](#_Toc7614934)

[Game Mechanics 3](#_Toc7614935)

[Coding Standards 3](#_Toc7614936)

[Work Allocation 3](#_Toc7614937)

[Ryan’s tasks 3](#_Toc7614938)

[Michael’s tasks 3](#_Toc7614939)

[Dominykas’s tasks 3](#_Toc7614940)

[Shafay’s tasks 4](#_Toc7614941)

[Mile Stones 4](#_Toc7614942)

[Meeting Minutes 4](#_Toc7614943)

[Conclusion 4](#_Toc7614944)

# Introduction

In this module, our task was to get together into groups and develop our game using previous game engine project from IMAT3904 Game Engine Architecture. We have created a group of 4 people and after discussing game ideas to implement, group decided to create a (Bow game).

# Outline

A (Bow game) is a 2D arcade style game developed in 3D game engine. 2 archers are positioned in a level terrain. Player plays as an archer on the left side, which has bow with an arrow. On the right side enemy archer is positioned with the same equipment. Players’ task is to kill the enemy by rotating bow and changing power to shoot the arrow in order to hit enemy. Enemy will lose if he will lose all of his hit points. On other hand, enemy has the same goal as player and will try to hit him and take all of his hit points.

# Game Mechanics

# Coding Standards

Variable consistency;

Tasks created as objects and what type of variables from other objects are used;

Doxygen;

# Work Allocation

After some discussion group decided to allocate tasks in this manner:

## Ryan’s tasks

Ryan was responsible for creating physics and collision detection for the game;

## Michael’s tasks

Michael was responsible for creating and managing levels with providing necessary assets for the game;

## Dominykas’s tasks

Dominykas was responsible for creating enemy NPC with AI behaviour, Player and NPC health, text to screen values in the game, checking player and NPC turns and sending arrow state to NPC.

The first task was creating an NPC character object class in the game, which constructor would take model, position and orientation. All of those values are being assigned in a scene from JSON files.

Next thing was to implement an AI component, which would have all calculations and algorithms in it. AI component is assigned in NPC constructor.

AI calculated the range of possible angle and power and tries to shrink its minimum and maximum range values as the arrow hit the ground, telling if its arrow was undershot or overshot the player. If undershot, next time range will switch so NPC would shoot arrow higher or stronger, if overshot, lower angle or weaker power.

NPC had all calculations from AI combined in game update and will run only if it is NPCs turn.

Implemented way to see Player health on top left corner and NPC health on top right corner. When either of them are hit, their health will be reduced. When Player health reduces down to 0, a message will be shown informing that player lost, if NPC health reduces to 0, message will be shown informing that player won.

Last thing was to implement a class which would be responsible for holding information which turns it is. Which class sends state if turn between NPC and player, notifying if it is their turn to act. When it is player’s turn, the player will be allowed to control the player and shoot the arrow while NPC will be idle. When NPC turn, the enemy will calculate the angle and power of the bow and shoot the arrow, disabling player’s access to control player character. Finally, awaiting turn between player and NPC turns is on when arrow still flies, disabling player and NPC from acting.

## Shafay’s tasks

Shafay was responsible for mouse input calculation for bow movement and camera switching.

# Mile Stones

# Meeting Minutes

# Conclusion