Thomas Service S1911936

Games Programming 3

Coursework

*I confirm that the code contained in this file (other than that provided or authorised) is all my own work and has not been submitted elsewhere in fulfilment of this or any other award*.

Thomas Service

Contents

1: Cover page

2: Contents

3-4: Extra Cameras

5 : Advanced Movement

1 –Extra Cameras

The base camera is used to keep a top down view of the player, and follows them as they move. In this project, two more cameras were added – One camera simply gives a different perspective, showing off the 3D aspect of the game. This view also follows the player, but in a different way, instead always looking at the player without the camera itself moving through the world space.

*Figure 1 – Straight Top Down View*

A group of jellyfish

Description automatically generated with medium confidence

*Figure 2 – View From A 45 Degree Angle*

A picture containing invertebrate, blue

Description automatically generated

This was done by creating another Camera object, creating the code to initialise and place it in the scene, then creating a placeholder Camera object called currentCamera, to allow easier switching between what camera is being used at a given moment. The first camera was bound to the “1” key, and the second is bound to “2”.

*Figure 3 – Second Camera Initialization*

myCamera2.initCamera(glm::vec3(25, 0, -25), 70.0f, (float)\_gameDisplay.getWidth() / \_gameDisplay.getHeight(), 0.01f, 1000.0f);

*Figure 4 – Code To Keep Second Camera Looking At The Player*

myCamera2.setLook(glm::vec3(ship.getTM().GetPos()->x, ship.getTM().GetPos()->y, ship.getTM().GetPos()->z));

*Figure 5 – Camera Swapping Code* *Text

Description automatically generated*

A third camera was also made, to add a closer perspective to follow the player from a third person perspective

*Figure 6 – Code For Third Camera*

myCamera3.setPos(glm::vec3(ship.getTM().GetPos()->x, ship.getTM().GetPos()->y - 1.0f, ship.getTM().GetPos()->z - 2.0f));

myCamera3.setLook(glm::vec3(ship.getTM().GetPos()->x, ship.getTM().GetPos()->y, ship.getTM().GetPos()->z));

2 – Advanced Movement

Given the 3D aspect of the game, movement along the Z axis was implemented. Left shift and Left Control moves the player forward and backward along the Z axis respectively, to make sure the player can line up with each asteroid.

*Figure 7 – Code For Movement Forward And BackwardA screenshot of a computer

Description automatically generated with medium confidence*