Term Project

CS480

12/20/2023

By Seth Lopez and Dustin Hurtz

Member Names	Code Implemented	Total Contribution
Dustin Hurtz	Planetary Hierarchy and Solar System Skybox and Observation Mode	50%
Seth Lopez	Ship Camera and Movement Skybox and Observation Mode	50%

Implementation:

UML Diagram:
Included inside our github project, the picture was too big to fit into the document.

Mesh Generation and Loading:

- Starship -

Our Starship uses the default model provided to us within the project, and we implemented it similarly to the previous projects. We instantiate the object within the graphics class as a mesh object, with both its .obj and .png file passed in to the constructor. The files are then read using loadModelFromFile() by using the AIScene object to create its vertices and indices. The png is then wrapped around the object to create the appearance of our starship. Our starship is able to move using the W/A/S/D keys, which allow you to fly throughout space.

- Sun, Planets and moons -

Our Sun, planets, and moons are all created and rendered using our sphere class. We define the number of slices as well as the png we would like to wrap around the object, and then initialize the sphere by calculating its vertices and indices. Afterwards, we take those vertices and indices and set them to our vertex and indice vector. We first render our sun in the center of our galaxy with a slight spin, and then render all of our planets outwards from the sun with an orbit. Each of our planets are situated with different sizes, rotational spin, and orbit speed. We decided on these attributes based off of their real life counterparts in order to make the scene seem more realistic. Our moon's orbit around our planets using the model stack, and orbit in many different ways depending on what planet they are a part of. Our earth's moon follows a normal orbit, and our planet Saturn, which has many moons, have them orbiting in many different unique ways.

- Comets -

We have one comet named Haley, Haley was created and rendered exactly like our planets and moons were, but has a much different orbit than the planets do. Firstly, it's orbital speed is much slower than most the planets and secondly, it has an orbit that's elliptical.

Texturing:

All of our objects that are textured use the base texturing class that we have been using in our previous programming assignments. This class is able to load a texture using a file, this is done by calling the SOIL_load_ogl_texture function, which gives us back an ID. And then we initialize the texture using glGenerateMinMap which generates a mapping for us, then glTexParameteri which lets us set a texture parameter. In our graphics class, when we start rendering the objects we call GLActiveTexture which allows us to display the texture that we wanted assigned to that object.

User Interaction:

Our Starship allows the user to use W/A/S/D in order to move around space as well as move their mouse which rotates the ship around the scene. The user can then fly to different planets to be able to see their rotation around the solar system. To interact with planets move as

close as possible as it is triggered when the player gets close to the planet and is looking at it then it is automatically triggered. To get out of this state press “e”.