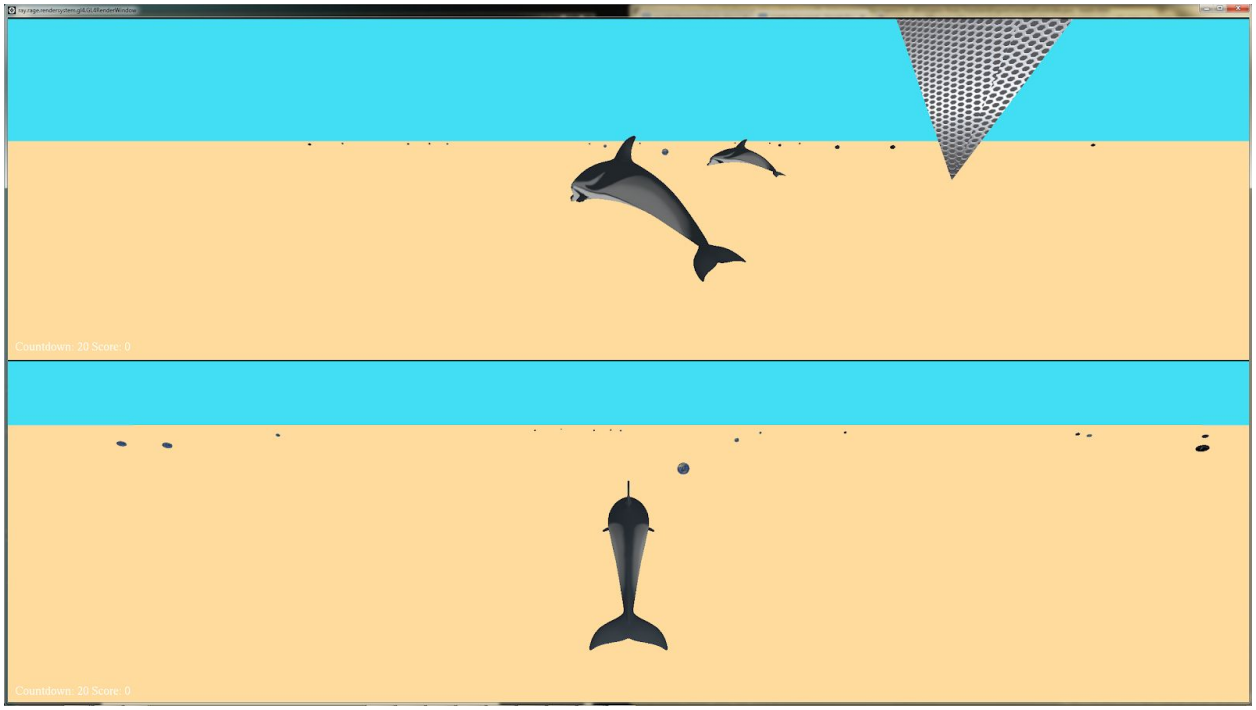
James Thornton

ScreenShot:



Compile from command window: navigate to “Dolphin Starship” folder and enter command:
javac a2*.java myGameEngine*.java

Run from command window: navigate to “Dolphin starship” folder and enter command: java
java a2.MyGame

Note: there is also a compile batch file

How Game is Played: The game has two players. There is a separate viewport and dolphin avatar assigned to each player. Player 1(The top viewport) is controlled with the keyboard and mouse. Player 2(The bottom viewport) is controlled with a controller. The object of the game is to visit as many planets as possible before the time runs out. Only one point can be obtained for visiting each planet. Once a planet has been visited by a player, the other player can no longer get points for visiting that planet. The planets are marked by a specific behavior (bouncing for player 1 and warping for player 2). When the time runs out, the player with the most points wins and the planets they visited will orbit around them. If the players tie, the planets orbit around the object at the origin of the game.

Keyboard Inputs: (Player 1)

A / D / W / S: moves avatar left / right / forward / backward.

Left-arrow/ Right arrow: yaw rotation around V-axis
Mouse: Orbits around avatar (elevation and azimuth).
+/-: Zooms in and on avatar ()

Controller Inputs: I used the same controller as the one in class
X-axis: Avatar yaw rotation
Y-axis: moves player forward and backward
RX-axis: Azimuth rotation around avatar
RY-axis: Elevation rotation around avatar
Right-trigger: moves avatar right
Left-trigger: moves avatar left
D-Pad: Forward zooms in, Back zooms out

Note about controls: zoom is limited to 3 units away from dolphin and 0.6 units away from dolphin, Elevation is limited to 50 degrees above the object down to 10 degrees to prevent full circle, going beneath the ground plane, and motion sickness;

Scoring: A player gets one point and only one point for visiting each planet. Once a planet is visited and marked by a player, The other player cannot obtain points for visiting that planet. The player with the most points at the end of the game wins. If the player's scores are equal, the result is a tie.

Node Controllers: I implemented two simple controllers. They are both used for when the players visit the planet. The first controller is a bounce controller that causes it's nodes to bounce up and down; it is used when player 1 visits the planet first. The second controller is a warp controller which causes the planets to scale up and down kind of like a pump; it is used when player 2 visits the planets first. I also used rages orbit controller that causes the visited planets to orbit the winning player, or in the case of a tie, the object at the origin.

Node Relationships: There are two node group for the planets. One is called "planetGroup1n", which has 20 child earth nodes, the other is called "planetGroup2n" which has 20 child moon nodes; these are used along with an arrayList called "freePlanets", which contains all of the earth and moon nodes, to add the bounce and warp controllers to the planets when they are visited

Camera Controller: I implemented an orbit controller with constrained elevation and 360 degree azimuth movement. See "note about controls" for elevation restriction.

Missed Requirements: When controlling player two with the gamepad. If you pull the RY-axis all the way down and release it quickly, there is a shutter in the camera. I believe this has

to do with the controller and not the code, as that is the only time it happens. If you release it smoothly, the shutter doesn't happen.

Added Features: NA

5029 Lab Machine: The first computer immediately on the left when entering the lab. The label on top of the case sad "DIGDUG"

List of Assets:

- Earth.obj: provided
- sphere.obj: provided
- Moon.jpeg: provided
- Blue.jpeg: provided
- Chain-fence.jpeg: provided
- Hexagons.jpeg: provided
- Bright-red.jpeg: provided
- Bright-blue.jpeg: provided
- Bright-green.jpeg: provided
- Cube.mtl: provided
- Default.mtl: provided
- dolphinHighPoly.obj: provided