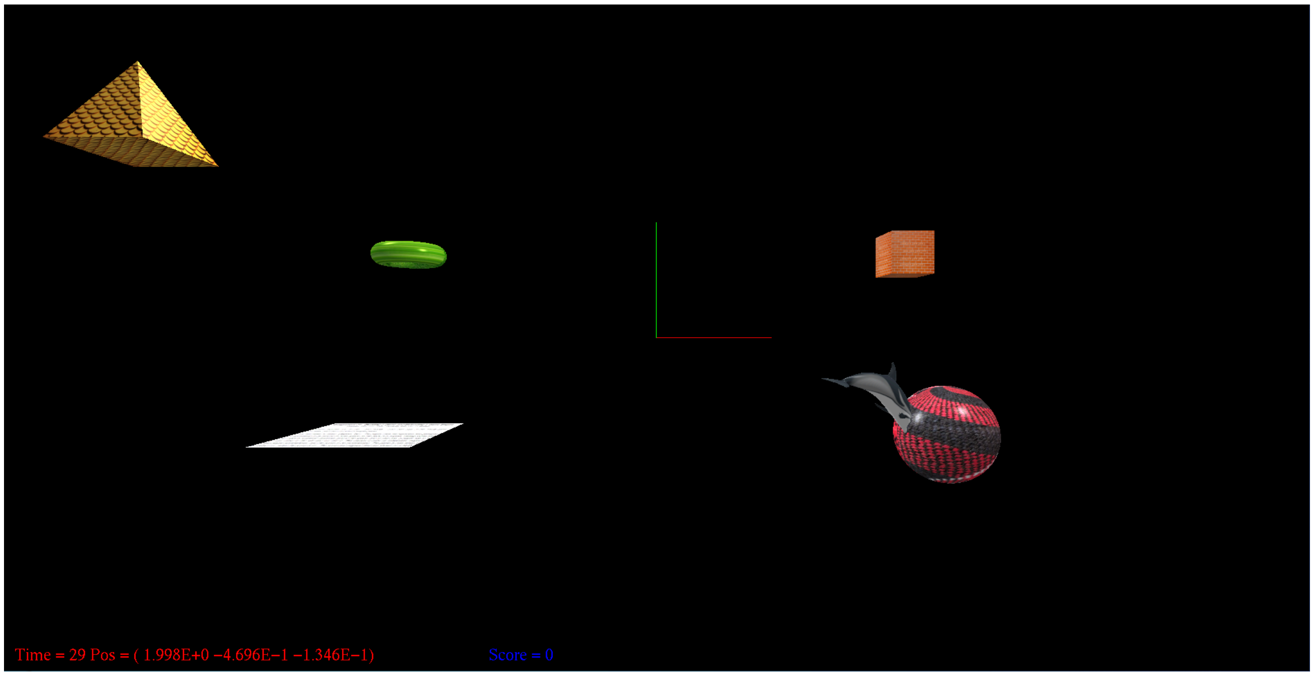
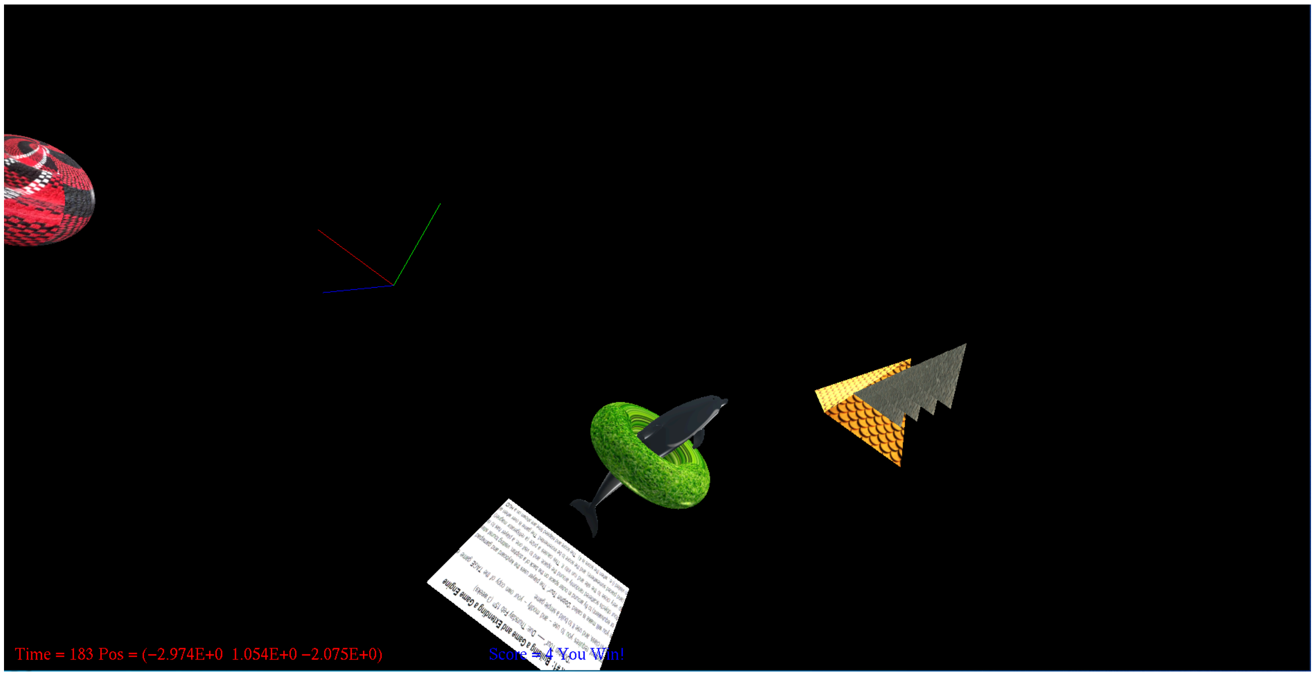
1. Gabriele Nicula, CSC165-01, A1 – Dolphin Tour
2. Screenshot(s):


Time = 29 Pos = (1.998E+0 −4.696E−1 −1.346E−1)   Score = 0


Time = 183 Pos = (−2.974E+0 1.054E+0 −2.075E+0)   Score = 4 You Win!

3. How the game is played:

   Dolphin starts in the center and there are 4 objects arranged towards the corners of the screen.

   The objective is to control the dolphin to visit all 4 objects to win.

   Note that the Dolphin is tethered to the camera when the camera is off the dolphin.

Keyboard Controls:

A/D: Yaw Left/Right

W/S: Move Forward/Backward

ArrowUp/ArrowDown: Pitch Up/Down

Spacebar: Alternate between connected camera, and disconnected camera

Q/E: Roll Left/Right

2, 3: Wireframe true/false on the dolphin

4: Reset camera to 0, 0, 0

ESC: Escape to abort

Note: SpaceBar and 2, 3, 4 keys work only after clicking the game window

Game controller

X-Axis Left Joystick: Yaw Left/Right

Y-Axis Left Joystick: Move Forward/Backward

X-Axis Right Joystick: Pitch Up/Down

Controller Button 3/4: Move Forward/Backward

4. Additional Game Activity:
   Added dolphin roll on its moving direction axis with keyboard keys ‘Q’/’E’
5. Additional Game Object
   The additional game object is a manual golden textured object pyramid inspired by the course example. It is defined through its 18 vertices exactly like in the course example ManualPyramid.java. I applied a texture and positioned the object in the left upper corner.
6. “Refrigerator magnet” object is a textured triangle-shaped magnet. It is defined as a manual indexed object with 3 vertices. The definition is in a1\MyMagnetObject.java.
   The magnets are placed as “badges” on/near the custom Pyramid whenever a “site” is visited for the first time by the Dolphin. The magnets are placed indexed on the Y coordinate to look like a vertical string of magnets.
7. List of changes to TAGE engine:
   a) tage\GameObject.java: Added methods for computing movements of the game object, specifically:
      public void moveForwardBack(float speed, Vector3f cameraLoc)
      public void pitch(float speed)
      public void yaw(float speed)

public void roll(float speed)

b) tage\input\action: Added classes that implement movements of the dolphin, specifically:

tage\input\action\ForwardBackActionJ.java

tage\input\action\ForwardBackActionK.java

tage\input\action\PitchActionJ.java

tage\input\action\PitchActionK.java

tage\input\action\RollActionK.java

tage\input\action\YawActionJ.java

tage\input\action\YawActionK.java

8. The Logitech Wingman Cordless game controller did not work on “pitch” when bound to Axis.RY as my Xbox controller did. So I changed “pitch” to bind to Axis.RZ which seems to be the left-right movement of the secondary joystick and it works. I tried many things, like binding to SLIDER as the device manager/controller device shows on Windows, or tried using other axes.

Also, I wish I had more time to properly comment all the classes and methods that I implemented.

9. - Display dolphin world coordinates on HUD1 - to see the movement limits

- Also having forward/backward dolphin movement on two Game controller buttons to check the controller buttons.

- Double tethering of the dolphin: on the camera when the camera is off the dolphin and on origin (0,0,0) in general to have “world boundaries”.

10. List of assets:

assets\textures\assign1.png - made by me

assets\textures\brick1.jpg - made by me

assets\textures\corvette1.jpg - Closeup of Red and Black Striped Woven Fabric · Free Stock Photo (pexels.com)

assets\textures\gold1.jpg - Photo of Cone Pattern · Free Stock Photo (pexels.com)

assets\textures\grass1.jpg - Green Grass · Free Stock Photo (pexels.com)

assets\textures\magnet1.jpg - Metal Sheet · Free Stock Photo (pexels.com)