CSE378: Computer Graphics

Assignment 2 – Pong

Name: Ahmed Mohamed Ahmed Aly

Id: 18P9313, UEL

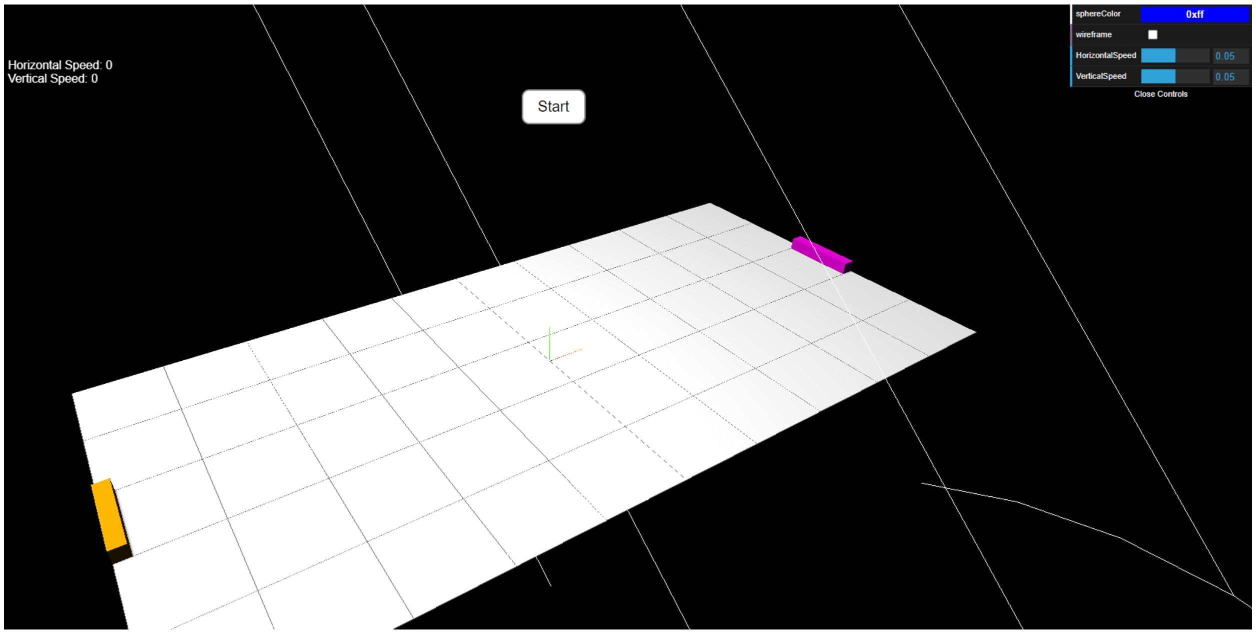
GitHub Link:

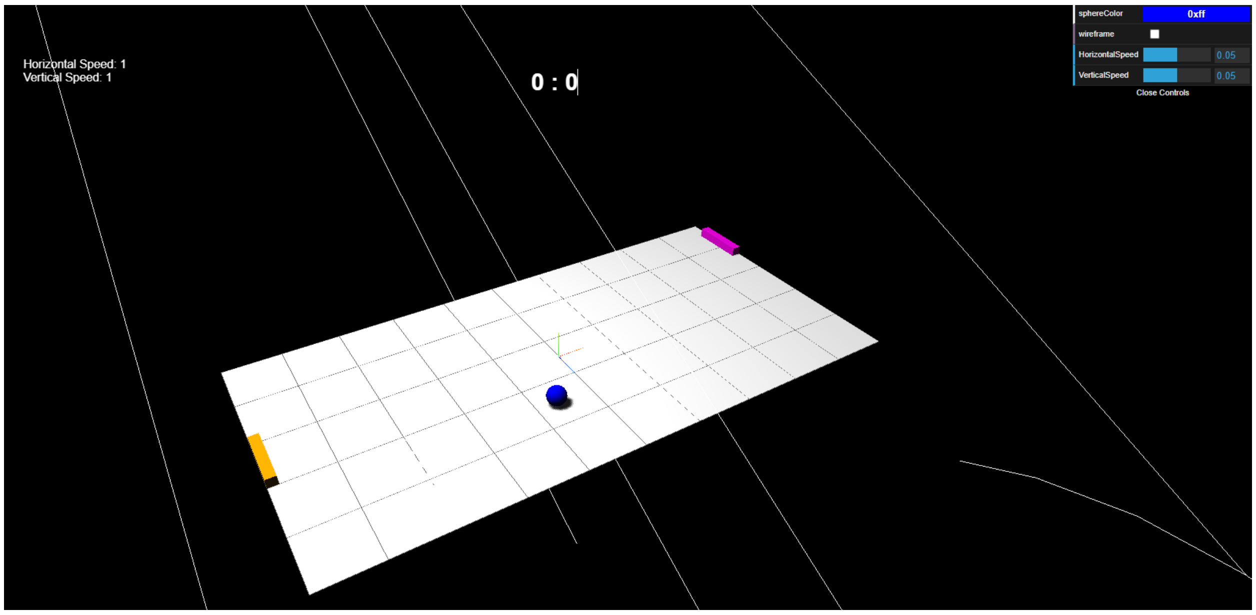
github/AhmedElshobaky/Pong-Threejs

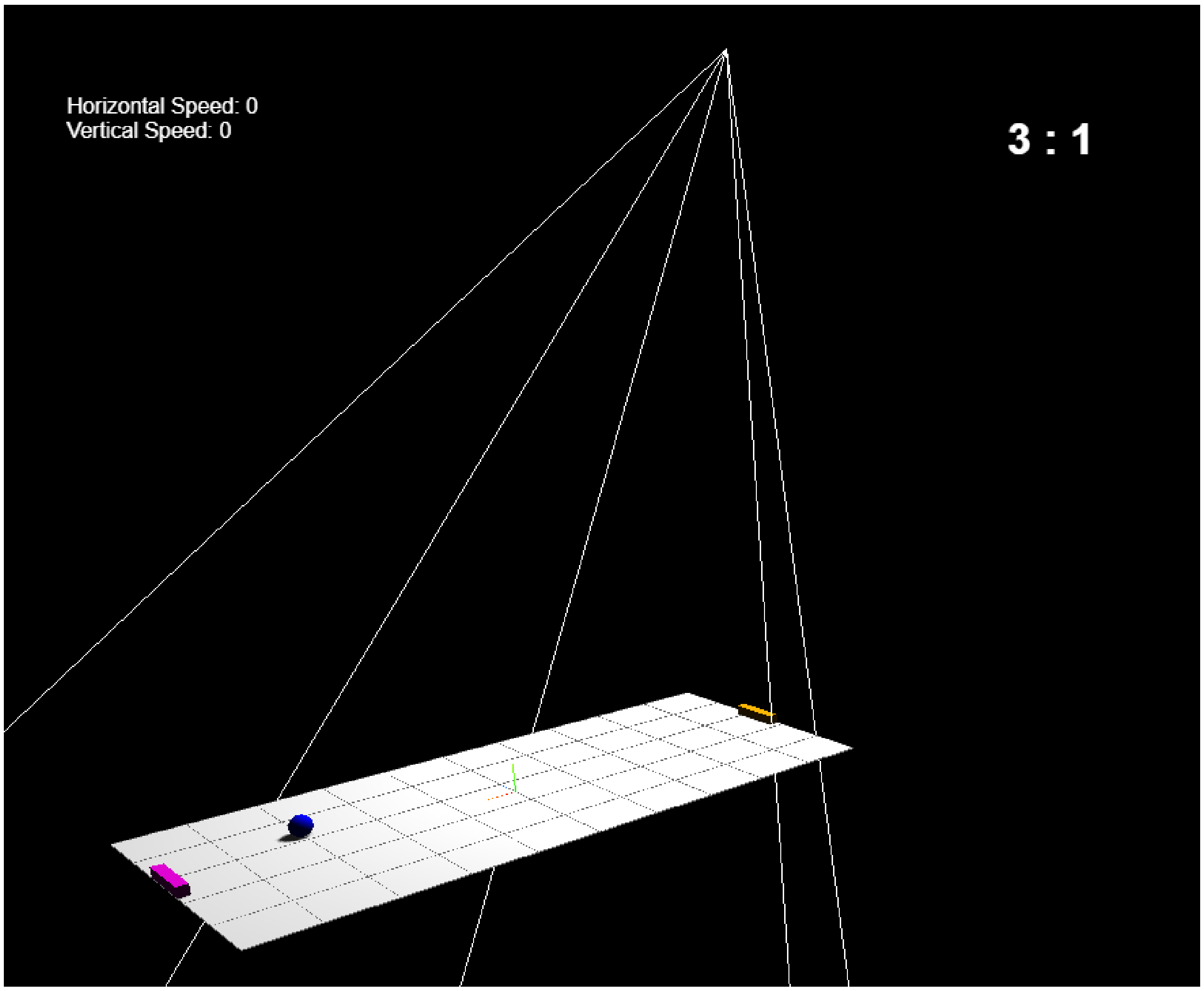
Demo:

Video

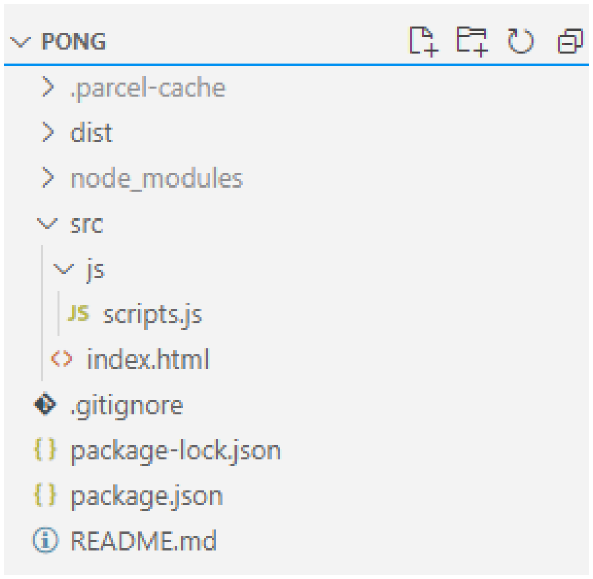
Snippets:







Files hierarchy



```
PONG
  > .parcel-cache
  > dist
  > node_modules
  v src
    v js
      scripts.js
    index.html
  .gitignore
  package-lock.json
  package.json
  README.md
```

Breaking down the code

Pong project code consists of:

- A 3D scene using the three.js library and renders it using a WebGL renderer. The scene includes a boxes, a sphere that represents a the ball and the players.
- An animation function that gets added to the rendering using

```
renderer.setAnimationLoop( animate );
```

- The animation function updates the positioning of the sphere and the positioning both players.

```
function animate() {
    updatePlayersPosition();
    if(isPlaying){
        updateSpherePosition();
    }
    renderer.render( scene, camera );
}
```

- Update players positioning function checks on global move1 and move2 which have values of 0.02 and -0.02 (step amount) only if the button is held on, otherwise both are zeros

```
// onkeydown event listener
var move1  = 0;
var move2 = 0;
function onKeyDown(event) {
  if (event.keyCode == 87) {
    move1 = -0.2;
  }
  if (event.keyCode == 83) {
    move1 = 0.2;
  }
  if (event.keyCode == 38) {
    move2 = -0.2;
  }
    if (event.keyCode == 40) {
    move2 = 0.2;
    }
}
```

```
// onkeyreleast event listener
function onKeyUp(event) {
  if (event.keyCode == 87) {
    move1 = 0;
  }
  if (event.keyCode == 83) {
    move1 = 0;
  }
    if (event.keyCode == 38) {
    move2 = 0;
    }
    if (event.keyCode == 40) {
    move2 = 0;
    }
}
```

```
function updatePlayersPosition() {
  position1 += move1;
  position2 += move2;
  //constraint the players to the plane coordinates
  if (position1 < -12) {
    position1 = -12;
  }else if(position1 > 12) {
    position1 = 12;
  }

  if (position2 < -12) {
    position2 = -12;
  }else if(position2 > 12) {
    position2 = 12;
  }

  p1.position.z = position1;
  p2.position.z = position2;
}
```

- Players are made of box geometries and MeshStandardMaterials to be able to cast shadows on the plane.

```
// create fruit basket geometry
const p1Geometry = new THREE.BoxGeometry(1, 1, 6);
const p1Material = new THREE.MeshStandardMaterial( { color: 0xffa400 } );
const p1 = new THREE.Mesh( p1Geometry, p1Material )
p1.position.set( -29.5, 0.5, 0 );
scene.add(p1);
p1.castShadow = true;
```

```
const p2Geometry = new THREE.BoxGeometry(1, 1, 6);
const p2Material = new THREE.MeshStandardMaterial( { color: 0xff00f0 } );
const p2 = new THREE.Mesh( p2Geometry, p2Material )
p2.position.set( 29.5, 0.5, 0 );
scene.add(p2);
p2.castShadow = true;
```

- An Orbit control is added to the scene to allow the user to rotate and zoom the camera.

```
const orbit = new OrbitControls( camera, renderer.domElement );
```

- A lighting is added to the scene, including an ambient light and a spot light, the spot light is set to cast shadows.

```
const ambiantLight = new THREE.AmbientLight( 0x333333, 1);
scene.add( ambiantLight );

const spotLight = new THREE.SpotLight( 0xffffff, 1);
scene.add( spotLight );
spotLight.position.set( -30, 70, 0 );
spotLight.castShadow = true;
spotLight.angle = 0.50;
```

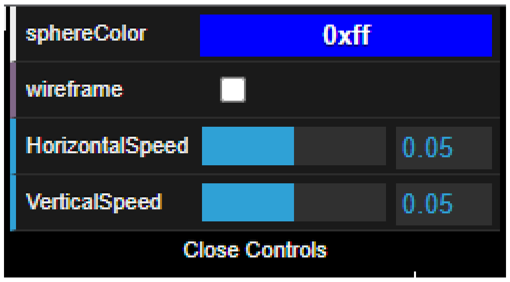
- dat.gui is added to give the user the ability control the sphere color, speed, wirefram using gui

```
const gui = new dat.GUI();
const options = {
    sphereColor: 0x0000ff,
    wireframe: false,
    HorizontalSpeed: 0.05,
    VerticalSpeed: 0.05
};

gui.addColor( options, 'sphereColor' ).onChange(function(e){
    sphere.material.color.set(e);
})

gui.add( options, 'wireframe' ).onChange(function(e){
    sphere.material.wireframe = e;
})

gui.add( options, 'HorizontalSpeed', 0, 0.1, 0.005 );
gui.add( options, 'VerticalSpeed', 0, 0.1, 0.005 );
```



sphereColor	0xff
wireframe	☐
HorizontalSpeed	0.05
VerticalSpeed	0.05

- Finally, updating the sphere function, it updates the sphere with its new coordinates and checks if collision occurred with the any of the players. If the x coordinate of the ball went lower than p1 or more than p2 then a point is counted to the other player.

```
function updateSpherePosition() {
    sphere.position.x = sphere.position.x + options.HorizontalSpeed *
sphereHorizontalDirection;
    sphere.position.z = sphere.position.z + options.VerticalSpeed *
sphereVerticalDirection;
    //if sphere hit the edge of the field or players then change direction
    if (sphere.position.z < -14 || sphere.position.z >14) {
     sphereVerticalDirection *= -1;
    }
    //p2 scored
    if (sphere.position.x < -31) {
      p2Counter++;
      resetSphere()
      updateScoreHTML()
    }
    //p1 scored
    if (sphere.position.x > 31) {
      p1Counter++;
      resetSphere()
      updateScoreHTML()
    }
    // if sphere hits a p1 it changes direction
    if (sphere.position.x < -28 && sphere.position.z > p1.position.z - 3 &&
sphere.position.z < p1.position.z + 3) {
      sphereHorizontalDirection *= -1;
      // increase speed by small random amount if it is not max already
      if (options.HorizontalSpeed > 0.2 || options.VerticalSpeed > 0.2) {
        options.HorizontalSpeed = 0.2;
        options.VerticalSpeed = 0.2;
      }else{
      options.HorizontalSpeed += Math.random() * 0.025;
      options.VerticalSpeed += Math.random() * 0.025;
      }
      updateSpeedHTML()
    }
    // if sphere hits a p2 it changes direction
    if (sphere.position.x > 28 && sphere.position.z > p2.position.z - 3 &&
sphere.position.z < p2.position.z + 3) {
      sphereHorizontalDirection *= -1;
      // increase speed by small random amount if it is not max already
      if (options.HorizontalSpeed > 0.2 || options.VerticalSpeed > 0.2) {
        options.HorizontalSpeed = 0.2;
        options.VerticalSpeed = 0.2;
      }else{
      options.HorizontalSpeed += Math.random() * 0.025;
      options.VerticalSpeed += Math.random() * 0.025;
      }
      updateSpeedHTML()
    }
}
```

References

Three.js. three.js docs. (n.d.). Retrieved January 6, 2023, from https://threejs.org/docs/