



CSE378: Computer Graphics

Assignment 3 – Bricks Breaker

Name: Ahmed Mohamed Ahmed Aly

Id: 18P9313, UEL

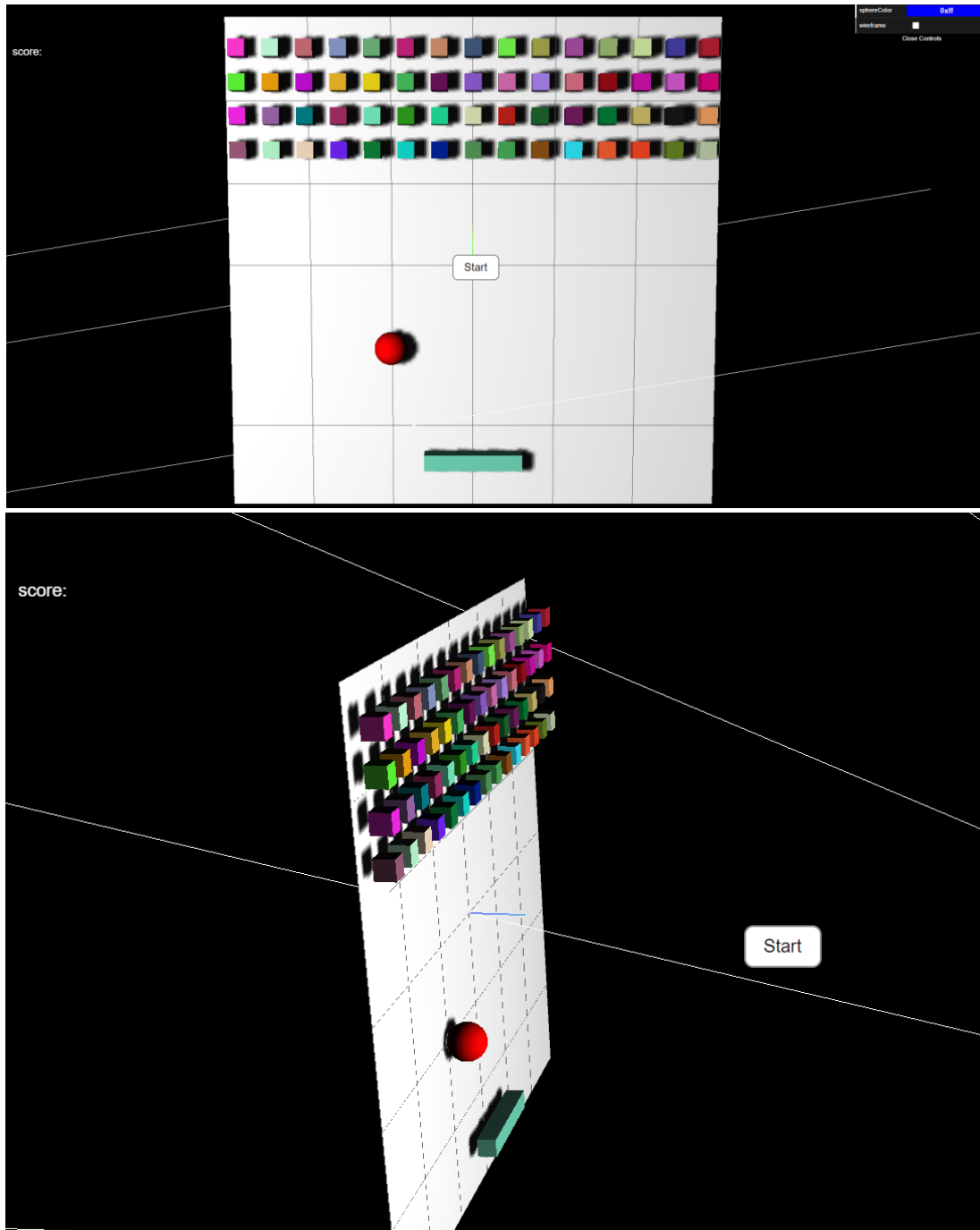
GitHub Link:

[github/AhmedElshobaky/Brick-Breakers-Threejs](https://github.com/AhmedElshobaky/Brick-Breakers-Threejs)

Demo:

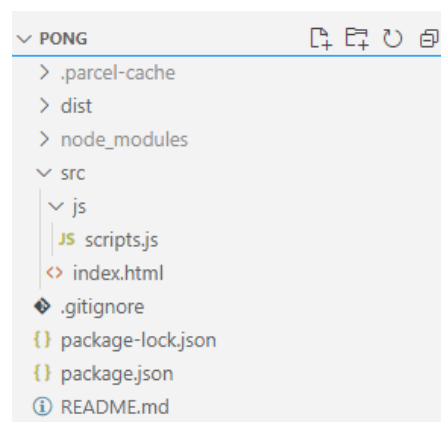
[Video](#)

Snippets:





Files hierarchy



Breaking down the code

Bricks Breaker project code consists of:

- A 3D scene using the three.js library and renders it using a WebGL renderer. The scene includes boxes, a sphere that represents the ball and the padel and bricks.

- 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 of the padel.

```
function animate() {
  // if all bricks broked, congrate the user and end the game
  if (counter == 60){
    congratulationMsg();
  }
  updatePadelPosition();
  if(isPlaying){
    updateSpherePosition();
  }
  renderer.render( scene, camera );
}
```

- Update padel positioning function checks on global step 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 step = 0;
function onKeyDown(event) {
  // move left
  if (event.keyCode == 37) {
    console.log("left");
    step = -0.2;
  }
  if (event.keyCode == 39) {
    console.log("right");
    step = 0.2;
  }
}

// onkeyreleast event listener
function onKeyUp(event) {
  if (event.keyCode == 37) {
    step = 0;
  }
}
```

```

    if (event.keyCode == 39) {
        step = 0;
    }
}

var currentPosition = padel.position.x;
function updatePadelPosition() {
    currentPosition += step;

    // bound the padel to the borders
    if ( currentPosition < -12) {
        currentPosition = -12;
    } else if (currentPosition > 12) {
        currentPosition = 12;
    }
    padel.position.x = currentPosition;
}

```

- Bricks are made of boxes that are created inside nested for loop.

```

// create fruit basket geometry
var bricks = [];
var bricksRow = [];
const spacing = 2;
// create 2d array of bricks
for(let i = 0; i < 15; i++){
    for (let j = 0; j < 4; j++){
        const brickGeometry = new THREE.BoxGeometry( 1, 1, 1 );
        const brickMaterial = new THREE.MeshStandardMaterial( { color:
getRandomColor(), wireframe: false } );
        const brick = new THREE.Mesh( brickGeometry, brickMaterial );
        brick.position.set(14- (i*spacing) ,7 + (j*(spacing)), 1 );
        console.log(brick.position);
        bricksRow.push(brick);
        scene.add( brick );
        brick.castShadow = true;
    }
    bricks.push(bricksRow);
}

```

- 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 ambientLight = new THREE.AmbientLight( 0x333333, 1);
```

```

scene.add( ambientLight );

const spotLight = new THREE.SpotLight( 0xffffff, 1);
scene.add( spotLight );
spotLight.position.set( -30, 0, 60 );
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: 0.15,
  VerticalSpeed: 0.15,
};
gui.addColor( options, 'sphereColor' ).onChange(function(e){
  sphere.material.color.set(e);
})
gui.add( options, 'wireframe' ).onChange(function(e){
  sphere.material.wireframe = e;
})

```



- Finally, updating the sphere function, it updates the sphere with its new coordinates and checks if collision occurred the padel or any brick. If the y coordinate of the ball went lower than the padel then game is over.

```

var counter = 0;
var sphereHorizontalDirection = 1;
var sphereVerticalDirection = -1;
function updateSpherePosition() {
  sphere.position.x = sphere.position.x + options.HorizontalSpeed *
sphereHorizontalDirection;
  sphere.position.y = sphere.position.y + options.VerticalSpeed *
sphereVerticalDirection;
  //if sphere hit the edge of the field or players then change direction
  if (sphere.position.x < -14 || sphere.position.x >14) {
    sphereHorizontalDirection *= -1;

```

```
}
if (sphere.position.y > 14) {
    sphereVerticalDirection *= -1;
}
if (sphere.position.y < -14) {
    gameOver();
}

// if sphere hits a padel it changes direction
if (sphere.position.y < -11 && sphere.position.x > padel.position.x - 3
&& sphere.position.x < padel.position.x + 3) {
    sphereVerticalDirection *= -1;
    sphereHorizontalDirection = Math.random() * 2 - 1;
}

// if sphere hits a brick it changes direction and brick is removed
for (let i = 0; i < bricks.length; i++){
    for (let j = 0; j < bricks[i].length; j++){
        if (sphere.position.y > bricks[i][j].position.y - 1 &&
sphere.position.y < bricks[i][j].position.y + 1 && sphere.position.x >
bricks[i][j].position.x - 1 && sphere.position.x < bricks[i][j].position.x
+ 1){
            sphereVerticalDirection *= -1;
            sphereHorizontalDirection = Math.random() * 2 - 1;
            scene.remove(bricks[i][j]);
            bricks[i].splice(j, 1);
            counter++;
            updateScoreHTML();
        }
    }
}
}
```

References

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