410262327 資工三乙 黃予桐

題目:

作業一 墨西哥帽子

操作說明及討論:

大帽子:太陽，中帽子:地球，小帽子:月球。

上方左邊四個按鈕分別是加速地球公轉、減速地球公轉、加速月球公轉、減速月球公轉。後面三個顏色條是太陽顏色改變，地球顏色改變，月球顏色改變。

滑鼠拖曳左鍵轉動、右鍵平移、滾輪得以縮放。

由於此次時間太短，製作時間不完全因此參考了吳祐任學長的程式碼，發現其js整理在同一個檔案非常清楚，能夠好好利用。

之前不怎麼有web coding的經驗，但是以我來說unity最後輸出會有一個webgl的格式，所以有稍微認識webgl。

一開始coding不外乎就是先把html裡面的網頁架構都大略了解，依照id去對照javascript裡面的funtion等等，按鈕的建立，滑鼠操作如何影響網頁的呈現，顏色條的程式碼如何加上去，其實這些看似簡單但實際去查資料去操作卻是要花不少時間。

這個作業讓我對於webgl有更進一步的認識，的確在一開始起步時會覺得困惑，但是萬事起頭難，寫完回頭看後發現，其實沒有什麼，而且挺有趣的。

程式碼:

<html>

        <head>

                <title>星球旋轉</title>

                <style>canvas { width: 100%; height: 90% }</style>

        </head>

        <body>

            <button id= "increasemoonspeed">increase moon speed</button>

            <button id= "decreasemoonspeed">decrease moon speed</button>

            <button id= "increaseearthspeed">increase earth speed</button>

            <button id= "decreaseearthspeed">decrease earth speed</button>

            <input type="color" id="sunColorPicker" value="#ff0000">

            <button id="applyColorBtn">Change sun colour</button>

            <input type="color" id="earthColorPicker" value="#ff0000">

            <button id="applyEarthColorBtn">Change earth colour</button>

            <input type="color" id="moonColorPicker" value="#ff0000">

            <button id="applyMoonColorBtn">Change moon colour</button>

            <input type="range" min="1" max="300" value="100" id="sunScaleSlider">

            <div id="stats"></div>

            <script src="three.js"></script>

            <script src="TrackballControls.js"></script>

            <script src="plane.js"></script>

            <script>

                var stats = document.getElementById('stats');

                var camera,scene,renderer,controls;

                var moonspeed=0.05,earthspeed=0.01;

                var sun,earth,moon;

                var angle=0,angle2=0;

                var loadedTexture;

                var nRows = 1000;//越高越密

                var nColumns = 1000;

                var pointsArray = [];//點陣列

                var pointsArray1 = [];

                var pointsArray2 = [];

                var data = new Array(nRows);

                for(var i=0;i<nRows;i++)data[i]=new Array(nColumns);

                for( var i = 0; i < nRows; i++ ) {//依墨西哥帽子公式數入陣列

                        var x = Math.PI\*(4\*i/nRows-2.0);

                    for( var j = 0; j < nColumns; j++ ) {

                            var y = Math.PI\*(4\*j/nRows-2.0);

                            var r = Math.sqrt(x\*x+y\*y);

                            var colors=[];

                            var colors1=[];

                            var colors2=[];

                       if(r)data[i][j]= Math.sin(r)/r ;

                       else data[i][j]=1;

                       }

                }

                init();

                animate();

                function init(){

                    camera = new THREE.PerspectiveCamera(75, window.innerWidth/window.innerHeight, 1, 1000);

                    camera.position.z = 500;

                    controls = new THREE.TrackballControls( camera );

                    controls.addEventListener('change',render);

                    scene = new THREE.Scene();

                    //--------------------------------------

                    var sunn=100;  //太陽大小

                    for(var i=0; i<nRows-1; i++) {  //設定row線顏色

                             for(var j=0; j<nColumns-1;j++) {

                             pointsArray.push((2\*i/nRows-1)\*sunn, data[i][j]\*sunn, (2\*j/nColumns-1)\*sunn);

                             colors.push(1);

                             colors.push(1);

                             colors.push(0);

                              }

                        }

                    for(var j=0; j<nColumns-1; j++) {  //設定columns線顏色

                             for(var i=0; i<nRows-1;i++) {

                             pointsArray.push((2\*i/nRows-1)\*sunn, data[i][j]\*sunn, (2\*j/nColumns-1)\*sunn);

                             colors.push(1);

                             colors.push(0.5);

                             colors.push(0);

                              }

                        }

                    var sunbuffer = new THREE.BufferGeometry();  //buffer輸出

                    var sunmaterial = new THREE.LineBasicMaterial( { vertexColors: THREE.VertexColors } );  //材質輸出

                    sunbuffer.addAttribute( 'position', new THREE.Float32BufferAttribute( pointsArray, 3 ) );  //增加屬性

                    sunbuffer.addAttribute( 'color', new THREE.Float32BufferAttribute( colors, 3 ) );

                    sunbuffer.computeBoundingSphere();

                    mesh = new THREE.Line( sunbuffer, sunmaterial );

                    mesh.position.set(0,0,0); //將太陽設在中央

                    sun = mesh;

                    scene.add(mesh);

                    //-------------------------------------------

                    var earthh=30;

                    for(var i=0; i<nRows-1; i++) {

                             for(var j=0; j<nColumns-1;j++) {

                             pointsArray1.push((2\*i/nRows-1)\*earthh, data[i][j]\*earthh, (2\*j/nColumns-1)\*earthh);

                             colors1.push(0);

                             colors1.push(1);

                             colors1.push(0);

                              }

                        }

                    for(var j=0; j<nColumns-1; j++) {

                             for(var i=0; i<nRows-1;i++) {

                             pointsArray1.push((2\*i/nRows-1)\*earthh, data[i][j]\*earthh, (2\*j/nColumns-1)\*earthh);

                             colors1.push(0);

                             colors1.push(1);

                             colors1.push(1);

                              }

                        }

                    var earthbuffer = new THREE.BufferGeometry();

                    var earthmaterial = new THREE.LineBasicMaterial( { vertexColors: THREE.VertexColors } );

                    earthbuffer.addAttribute( 'position', new THREE.Float32BufferAttribute( pointsArray1, 3 ) );

                    earthbuffer.addAttribute( 'color', new THREE.Float32BufferAttribute( colors1, 3 ) );

                    earthbuffer.computeBoundingSphere();

                    meshearth = new THREE.Line(earthbuffer, earthmaterial );

                    earth = meshearth;

                    angle=0;

                    scene.add(meshearth);

                    //-----------------------------------------

                    var moonn=5;

                    for(var i=0; i<nRows-1; i++) {

                             for(var j=0; j<nColumns-1;j++) {

                             pointsArray2.push((2\*i/nRows-1)\*moonn, data[i][j]\*moonn, (2\*j/nColumns-1)\*moonn);

                             colors2.push(1);

                             colors2.push(1);

                             colors2.push(1);

                              }

                        }

                    for(var j=0; j<nColumns-1; j++) {

                             for(var i=0; i<nRows-1;i++) {

                             pointsArray2.push((2\*i/nRows-1)\*moonn, data[i][j]\*moonn, (2\*j/nColumns-1)\*moonn);

                             colors2.push(0);

                             colors2.push(0);

                             colors2.push(0);

                              }

                        }

                    var moonbuffer = new THREE.BufferGeometry();

                    var moonmaterial = new THREE.LineBasicMaterial( { vertexColors: THREE.VertexColors } );

                    moonbuffer.addAttribute( 'position', new THREE.Float32BufferAttribute( pointsArray2, 3 ) );

                    moonbuffer.addAttribute( 'color', new THREE.Float32BufferAttribute( colors2, 3 ) );

                    moonbuffer.computeBoundingSphere();

                    meshmoon = new THREE.Line(moonbuffer, moonmaterial );

                    moon = meshmoon;

                    angle2=0;

                    scene.add(meshmoon);

                    //---------------------------------------

                    renderer = new THREE.WebGLRenderer();

                    renderer.setSize(window.innerWidth, window.innerHeight);

                    renderer.setClearColor( 0x000000);  //背景設為黑色

                    document.body.appendChild(renderer.domElement);

                }

                function animate(){

                    stats.innerHTML = "Moon Speed : "+moonspeed+"  Earth Speed : "+earthspeed;

                    requestAnimationFrame(animate);

                    controls.update();

                    render();

                }

                function render(){

                    angle+=earthspeed;

                    angle2+=moonspeed;

                    sun.rotation.y += 0.01;

                    earth.rotation.y += 0.01;

                    earth.position.set(200\*Math.cos(angle), 200\*Math.sin(angle), 0);

                    moon.rotation.y+=0.01;

                    moon.position.set(200\*Math.cos(angle)+50\*Math.cos(angle2), 200\*Math.sin(angle)+50\*Math.sin(angle2), 0);

                    renderer.render(scene, camera);

                }

                </script>

        </body>

</html>

執行畫面:

由於月球過小可能看不太清楚

實際操作確定是有變色的

放大看後有變黃

變更顏色前:

一張含有 螢幕擷取畫面 的圖片

自動產生的描述

變更顏色後:

一張含有 螢幕擷取畫面 的圖片

自動產生的描述