SUMMARY OF FOREST CONCERT

森林演唱会总结

功能描述/代码清单/功能展示/分工自评

Function description / code list /
function display / division of labor self-assessment

目录

content

01 功能描述

02 代码清单

03 功能展示

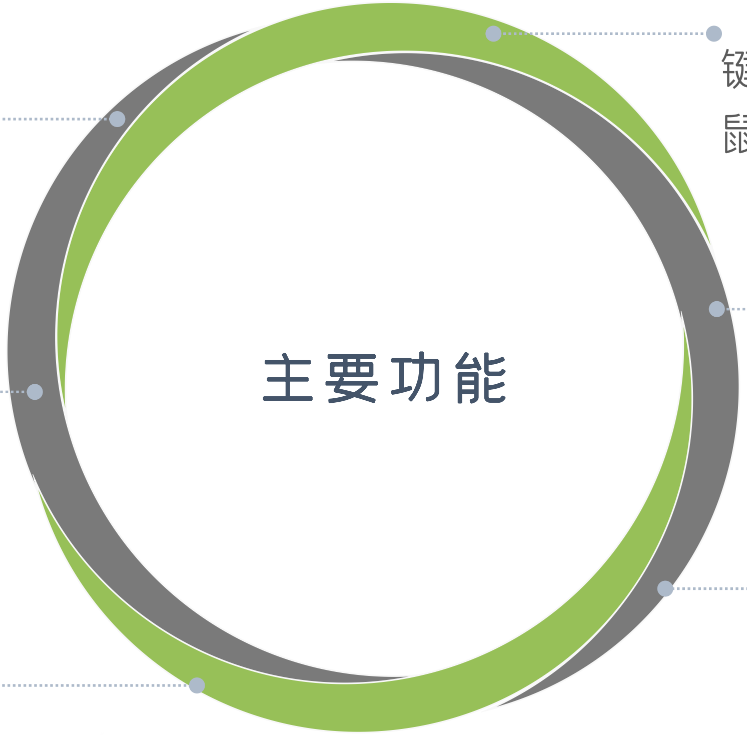
04 分工自评

PART 01

功能描述

Function description

主要功能描述



1.1缩放功能

通过鼠标齿轮的操控
可实现画面的缩放

1.2平移功能

键盘方向键（上下左右键）操作，
鼠标右键长按，可实现平移转换。

1.3视角转换

鼠标左键长按，视角转换。

1.4 音频播放

加载完成网页后音乐自动播放

1.5动画功能

实现伴舞角色，挥动的荧光棒，
不同方向转动的立方体显示屏

1.6舞台灯光

设置舞台灯光跟随任
务进行移动，给人物
添加舞台灯光的效果

PART 02

核心代码清单

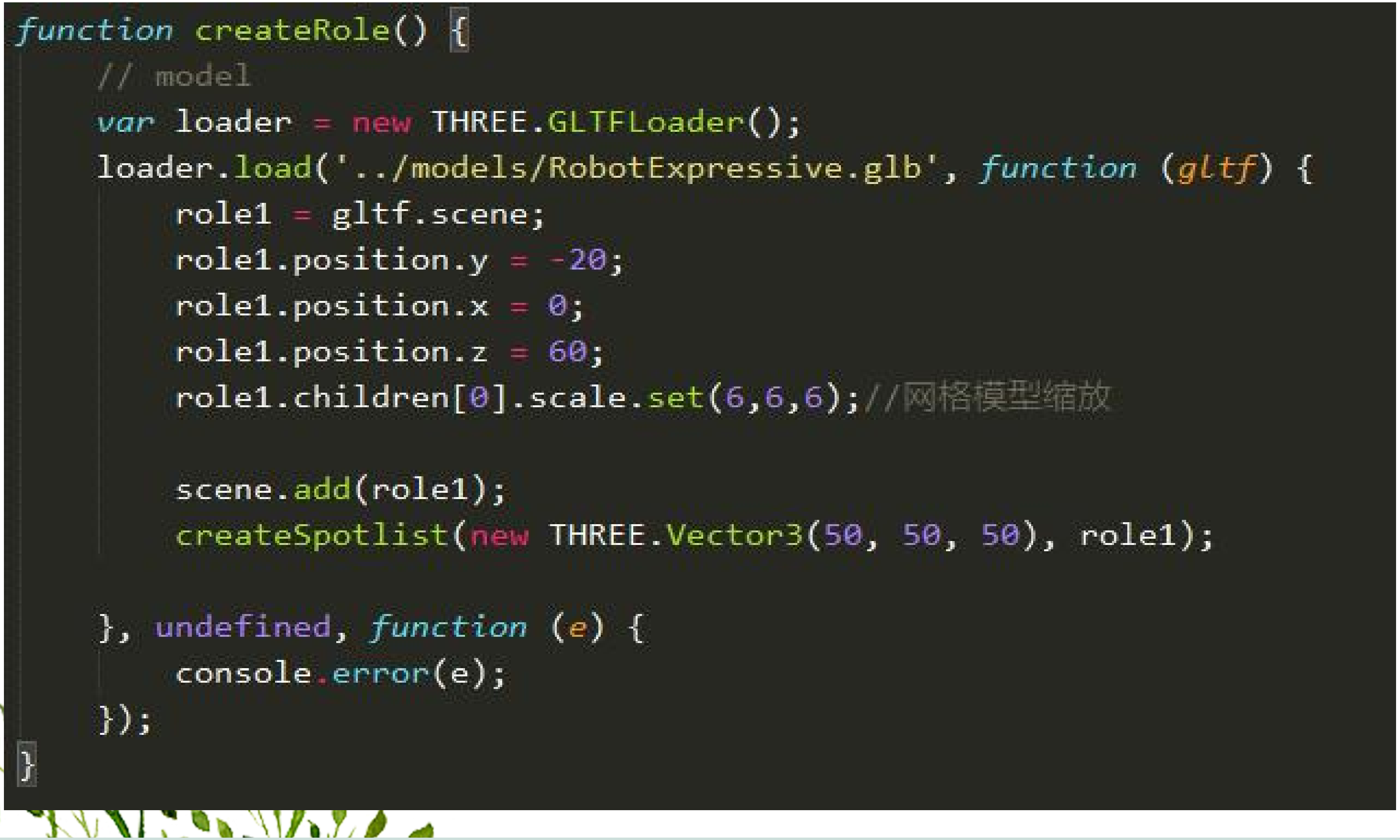
Core code list

核心代码清单

设计所采用的实现工具为IJ，开发环境为web及webgl，主要工具库：基于Three.js库，主要包括three.js、dat.gui.min.js、OrbitControls.js、util.js、GLTFLoader.js。

2.1 实现人物模型导入模块

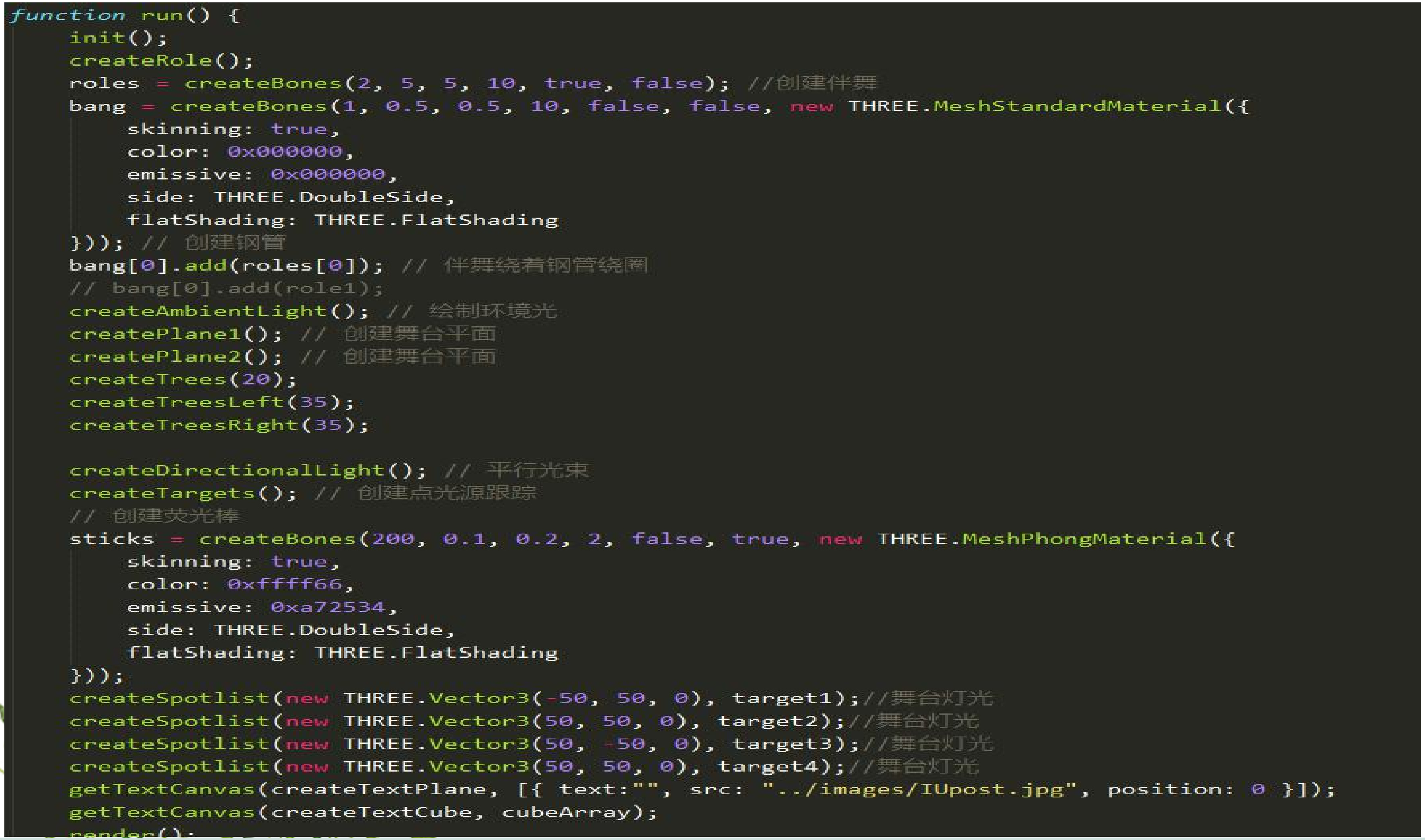
在createRole()函数中，实现对glb模型的加载，设置导入模型的位置，为其添加点光源。



```
function createRole() {
    // model
    var loader = new THREE.GLTFLoader();
    loader.load('../models/RobotExpressive.glb', function (gltf) {
        role1 = gltf.scene;
        role1.position.y = -20;
        role1.position.x = 0;
        role1.position.z = 60;
        role1.children[0].scale.set(6,6,6);//网格模型缩放

        scene.add(role1);
        createSpotlist(new THREE.Vector3(50, 50, 50), role1);

    }, undefined, function (e) {
        console.error(e);
    });
}
```

2.2.主要内容的创建

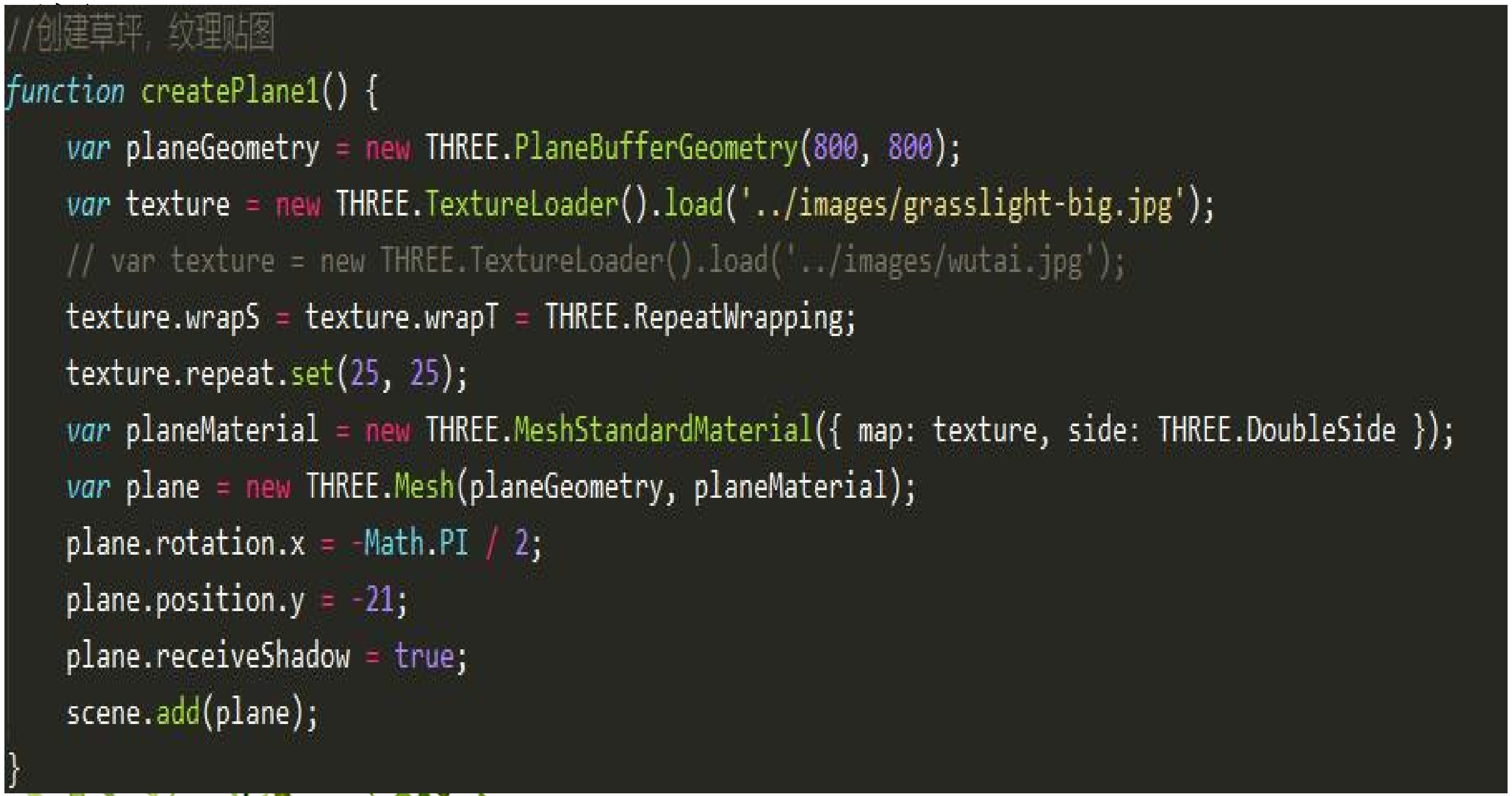
在run()函数中，进行初始化，创建主角、伴舞、钢管、舞台、舞台灯光及荧光棒。



```
function run() {
    init();
    createRole();
    roles = createBones(2, 5, 5, 10, true, false); //创建伴舞
    bang = createBones(1, 0.5, 0.5, 10, false, false, new THREE.MeshStandardMaterial({
        skinning: true,
        color: 0x000000,
        emissive: 0x000000,
        side: THREE.DoubleSide,
        flatShading: THREE.FlatShading
    })); // 创建钢管
    bang[0].add(roles[0]); // 伴舞绕着钢管绕圈
    // bang[0].add(role1);
    createAmbientLight(); // 绘制环境光
    createPlane1(); // 创建舞台平面
    createPlane2(); // 创建舞台平面
    createTrees(20);
    createTreesLeft(35);
    createTreesRight(35);

    createDirectionalLight(); // 平行光束
    createTargets(); // 创建点光源跟踪
    // 创建荧光棒
    sticks = createBones(200, 0.1, 0.2, 2, false, true, new THREE.MeshPhongMaterial({
        skinning: true,
        color: 0xffff66,
        emissive: 0xa72534,
        side: THREE.DoubleSide,
        flatShading: THREE.FlatShading
    }));
    createSpotlist(new THREE.Vector3(-50, 50, 0), target1);//舞台灯光
    createSpotlist(new THREE.Vector3(50, 50, 0), target2);//舞台灯光
    createSpotlist(new THREE.Vector3(50, -50, 0), target3);//舞台灯光
    createSpotlist(new THREE.Vector3(50, 50, 0), target4);//舞台灯光
    getTextCanvas(createTextPlane, [{ text:"", src: "../images/IUpost.jpg", position: 0 }]);
    getTextCanvas(createTextCube, cubeArray);
    render();
```

2.3.创建森林地面

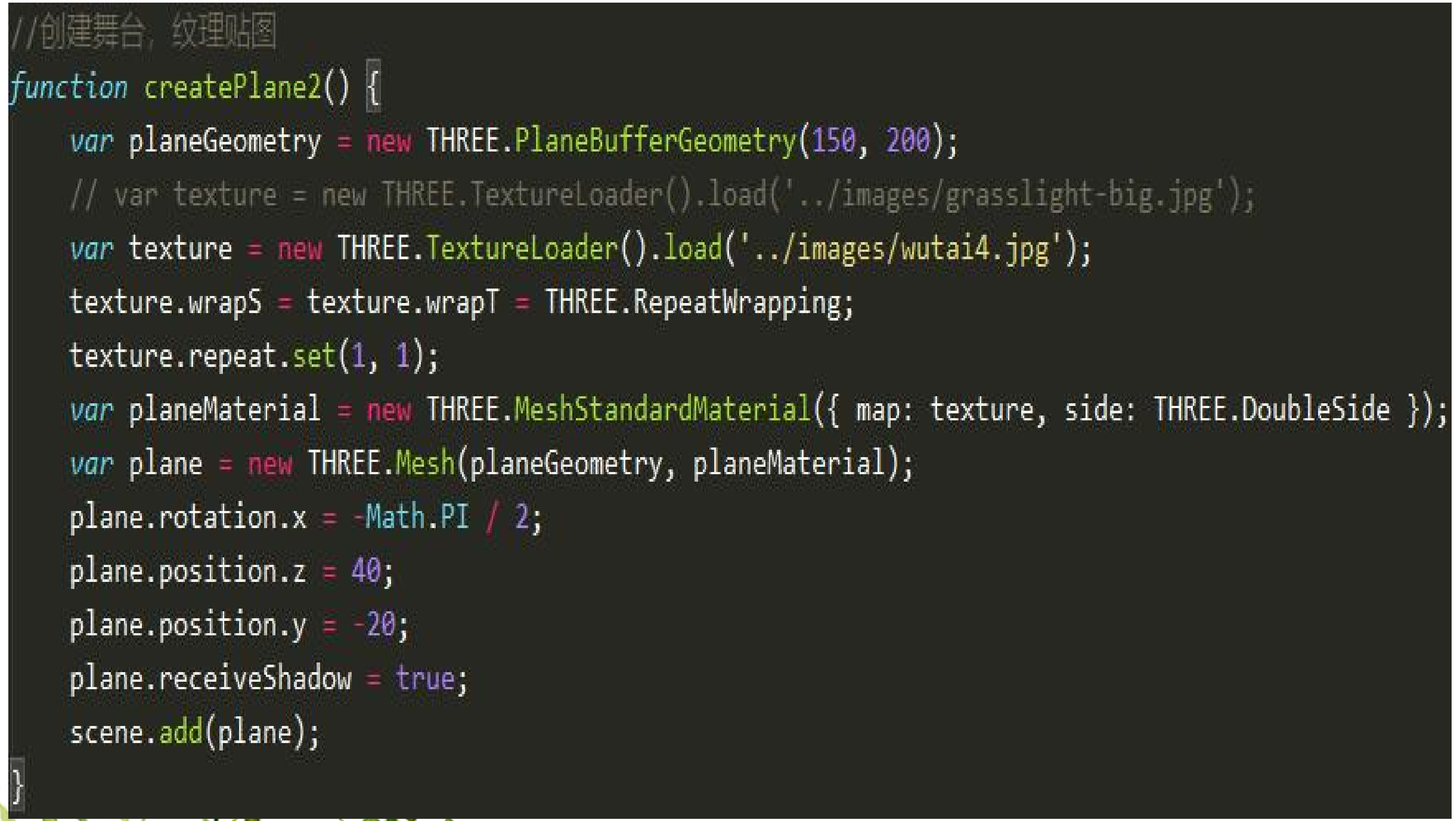
在createPlane1（）这个函数中，实现对森林的纹理贴图的加载及其相关设置（大小，纹理重复，位置，阴影



```
//创建草坪，纹理贴图
function createPlane1() {
    var planeGeometry = new THREE.PlaneBufferGeometry(800, 800);
    var texture = new THREE.TextureLoader().load('../images/grasslight-big.jpg');
    // var texture = new THREE.TextureLoader().load('../images/wutai.jpg');
    texture.wrapS = texture.wrapT = THREE.RepeatWrapping;
    texture.repeat.set(25, 25);
    var planeMaterial = new THREE.MeshStandardMaterial({ map: texture, side: THREE.DoubleSide });
    var plane = new THREE.Mesh(planeGeometry, planeMaterial);
    plane.rotation.x = -Math.PI / 2;
    plane.position.y = -21;
    plane.receiveShadow = true;
    scene.add(plane);
}
```

2.4.创建舞台纹理

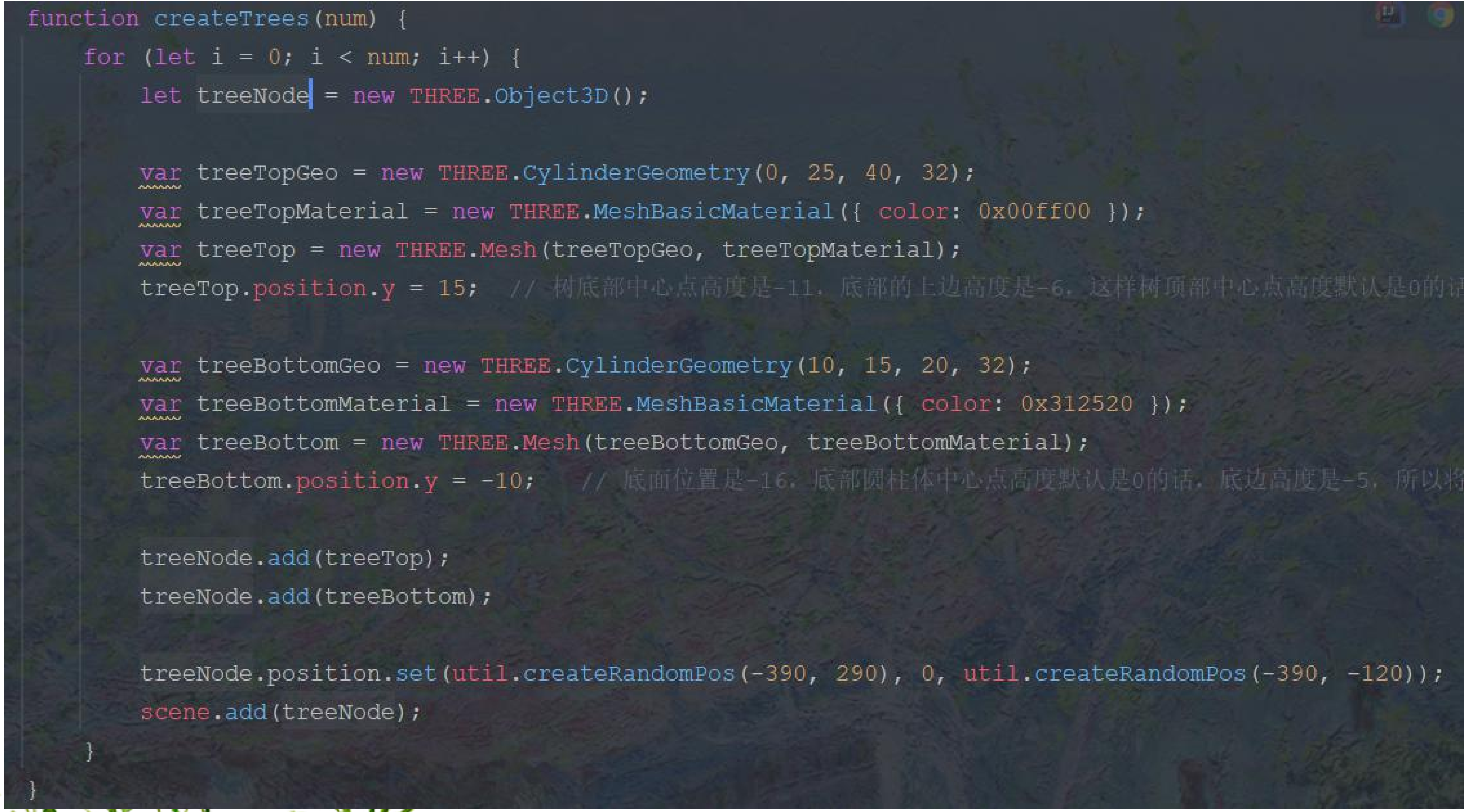
在createPlane2()这个函数中，加载作为舞台的纹理贴图，并对其进行相关设置（大小，纹理重复，位置，阴影等）。



```
//创建舞台，纹理贴图
function createPlane2() {
    var planeGeometry = new THREE.PlaneBufferGeometry(150, 200);
    // var texture = new THREE.TextureLoader().load('../images/grasslight-big.jpg');
    var texture = new THREE.TextureLoader().load('../images/wutai4.jpg');
    texture.wrapS = texture.wrapT = THREE.RepeatWrapping;
    texture.repeat.set(1, 1);
    var planeMaterial = new THREE.MeshStandardMaterial({ map: texture, side: THREE.DoubleSide });
    var plane = new THREE.Mesh(planeGeometry, planeMaterial);
    plane.rotation.x = -Math.PI / 2;
    plane.position.z = 40;
    plane.position.y = -20;
    plane.receiveShadow = true;
    scene.add(plane);
}
```

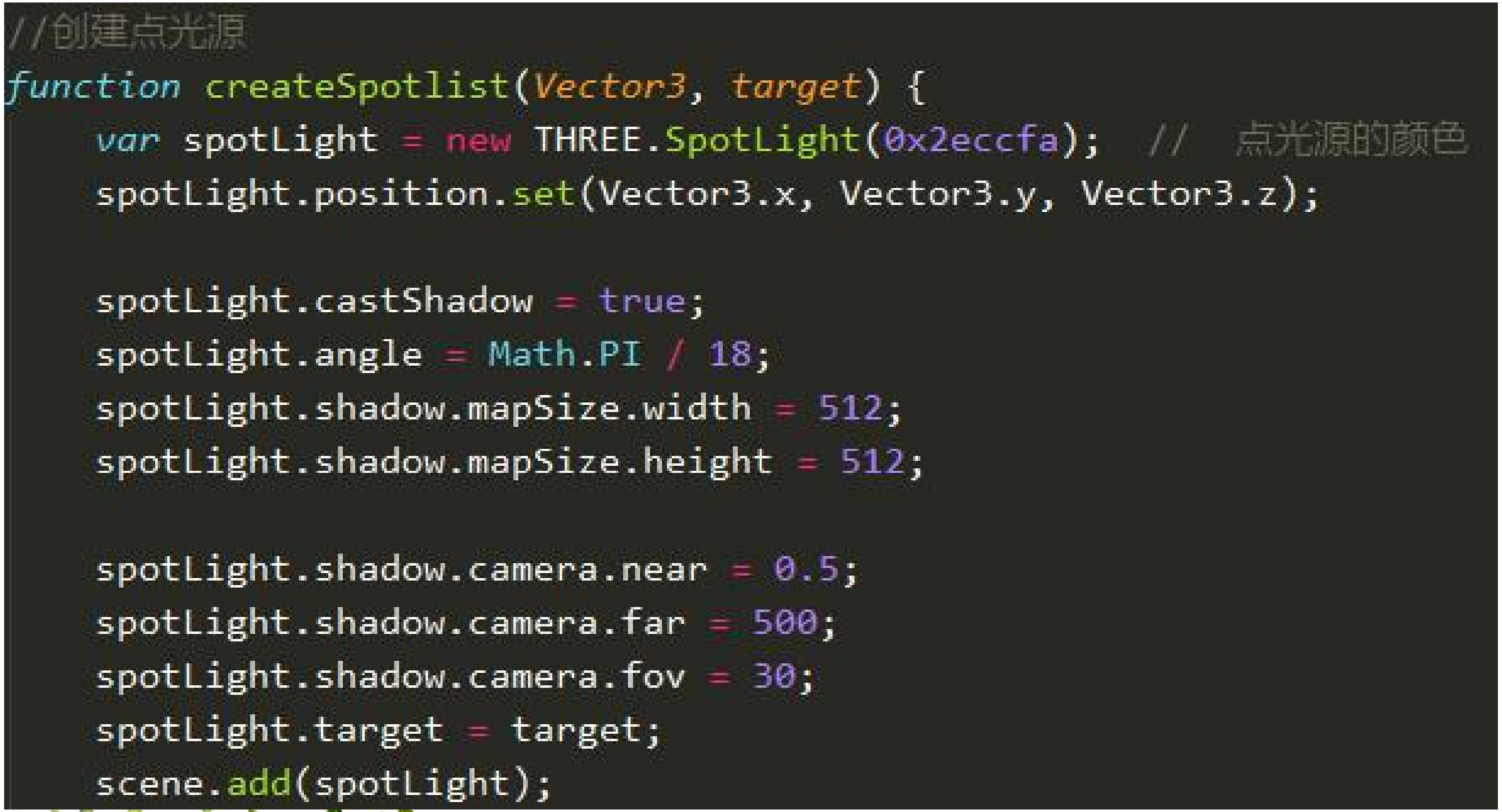
2.5.创建森林

在createTrees（num）这个函数中，创建3d树，随机生成树的位置。



```
function createTrees(num) {
    for (let i = 0; i < num; i++) {
        let treeNode = new THREE.Object3D();

        var treeTopGeo = new THREE.CylinderGeometry(0, 25, 40, 32);
        var treeTopMaterial = new THREE.MeshBasicMaterial({ color: 0x00ff00 });
        var treeTop = new THREE.Mesh(treeTopGeo, treeTopMaterial);
        treeTop.position.y = 15;  // 树底部中心点高度是-11，底部的上边高度是-6，这样树顶部中心点高度默认是0的话

        var treeBottomGeo = new THREE.CylinderGeometry(10, 15, 20, 32);
        var treeBottomMaterial = new THREE.MeshBasicMaterial({ color: 0x312520 });
        var treeBottom = new THREE.Mesh(treeBottomGeo, treeBottomMaterial);
        treeBottom.position.y = -10;    // 底面位置是-16，底部圆柱体中心点高度默认是0的话，底边高度是-5，所以将

        treeNode.add(treeTop);
        treeNode.add(treeBottom);

        treeNode.position.set(util.createRandomPos(-390, 290), 0, util.createRandomPos(-390, -120));
        scene.add(treeNode);
    }
}
```

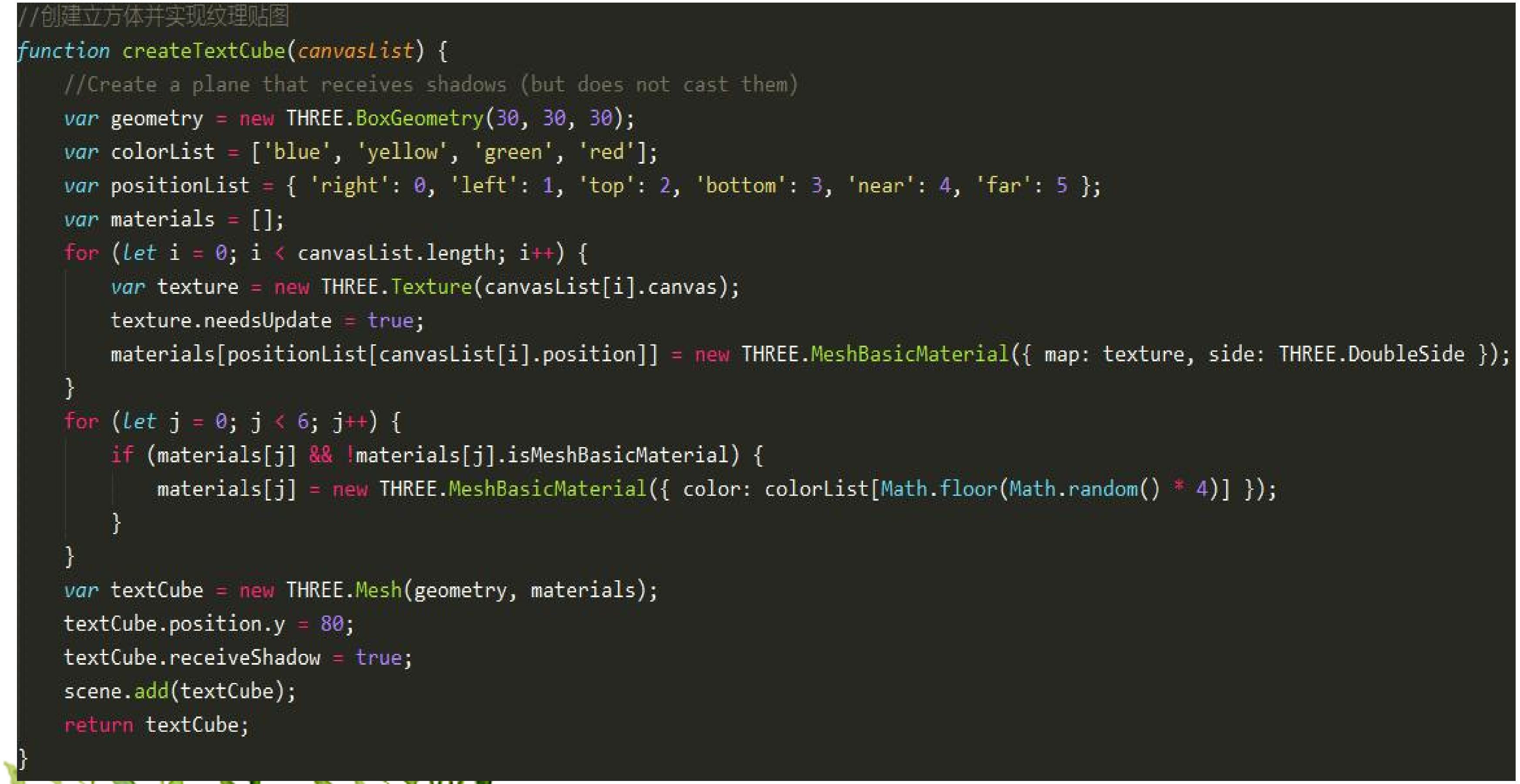
2.6.创建点光源



```
//创建点光源
function createSpotlist(Vector3, target) {
    var spotLight = new THREE.SpotLight(0x2eccfa);  //  点光源的颜色
    spotLight.position.set(Vector3.x, Vector3.y, Vector3.z);

    spotLight.castShadow = true;
    spotLight.angle = Math.PI / 18;
    spotLight.shadow.mapSize.width = 512;
    spotLight.shadow.mapSize.height = 512;

    spotLight.shadow.camera.near = 0.5;
    spotLight.shadow.camera.far = 500;
    spotLight.shadow.camera.fov = 30;
    spotLight.target = target;
    scene.add(spotLight);
```

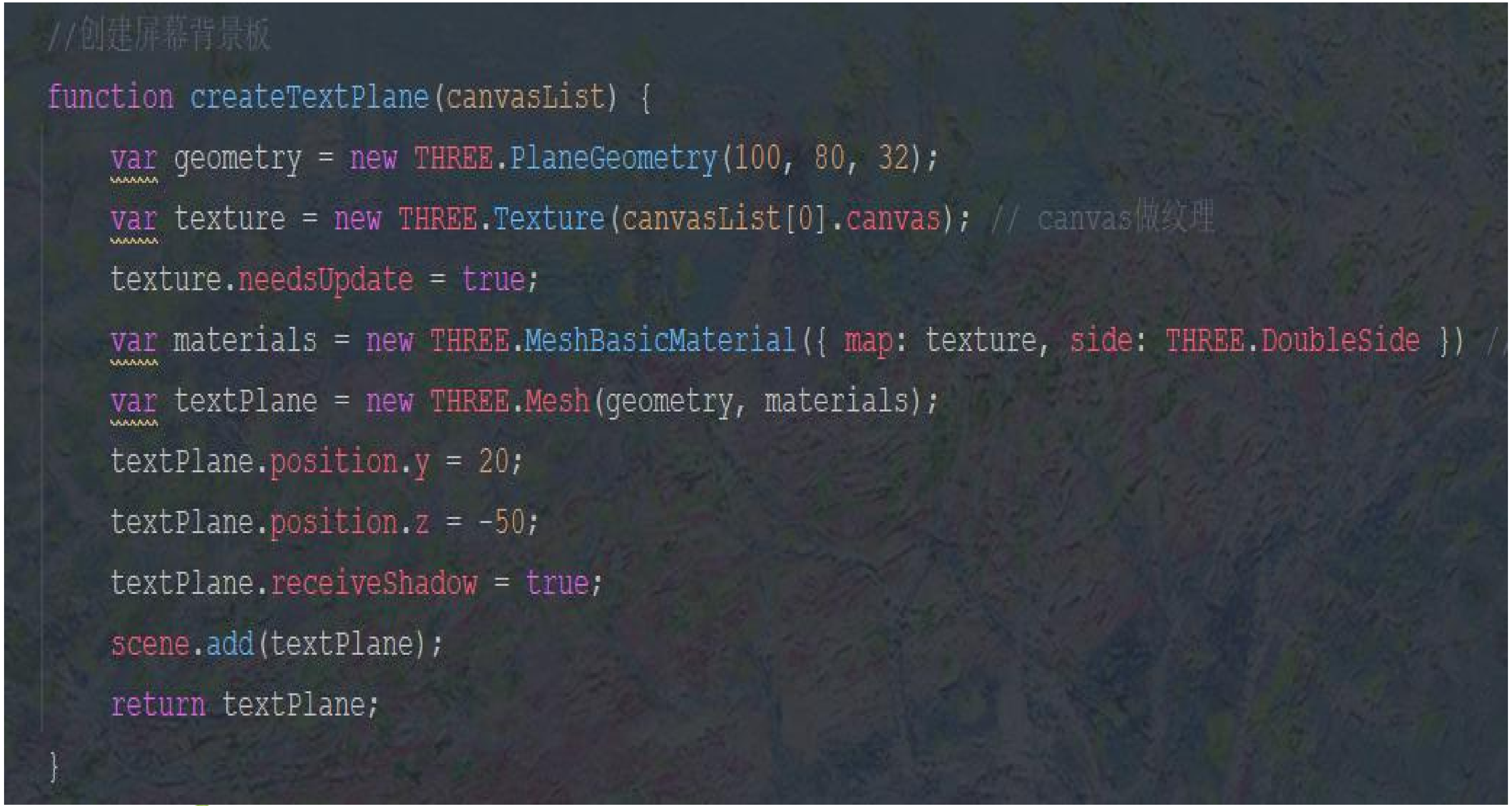
2.7. 创建转动的立方体屏幕

在createTextCube（）函数中，实现舞台中央的立方体屏幕的创建，并为其设置相应的参数，同时加载纹理贴图。



```
//创建立方体并实现纹理贴图
function createTextCube(canvasList) {
    //Create a plane that receives shadows (but does not cast them)
    var geometry = new THREE.BoxGeometry(30, 30, 30);
    var colorList = ['blue', 'yellow', 'green', 'red'];
    var positionList = { 'right': 0, 'left': 1, 'top': 2, 'bottom': 3, 'near': 4, 'far': 5 };
    var materials = [];
    for (let i = 0; i < canvasList.length; i++) {
        var texture = new THREE.Texture(canvasList[i].canvas);
        texture.needsUpdate = true;
        materials[positionList[canvasList[i].position]] = new THREE.MeshBasicMaterial({ map: texture, side: THREE.DoubleSide });
    }
    for (let j = 0; j < 6; j++) {
        if (materials[j] && !materials[j].isMeshBasicMaterial) {
            materials[j] = new THREE.MeshBasicMaterial({ color: colorList[Math.floor(Math.random() * 4)] });
        }
    }
    var textCube = new THREE.Mesh(geometry, materials);
    textCube.position.y = 80;
    textCube.receiveShadow = true;
    scene.add(textCube);
    return textCube;
}
```

2.8创建舞台大屏幕



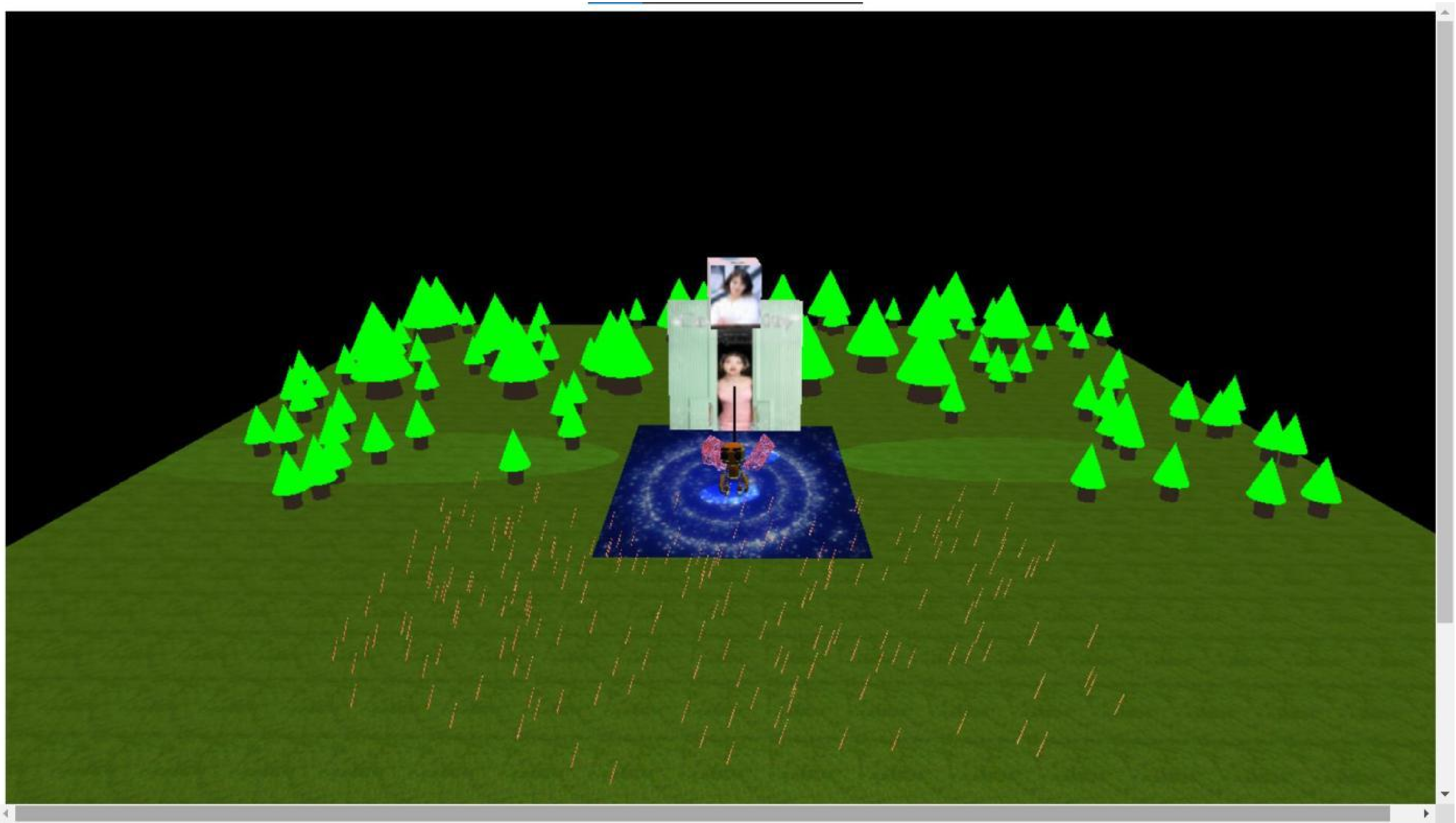
```
//创建屏幕背景板
function createTextPlane(canvasList) {
    var geometry = new THREE.PlaneGeometry(100, 80, 32);
    var texture = new THREE.Texture(canvasList[0].canvas); // canvas做纹理
    texture.needsUpdate = true;
    var materials = new THREE.MeshBasicMaterial({ map: texture, side: THREE.DoubleSide }) /
    var textPlane = new THREE.Mesh(geometry, materials);
    textPlane.position.y = 20;
    textPlane.position.z = -50;
    textPlane.receiveShadow = true;
    scene.add(textPlane);
    return textPlane;
}
```

PART 03

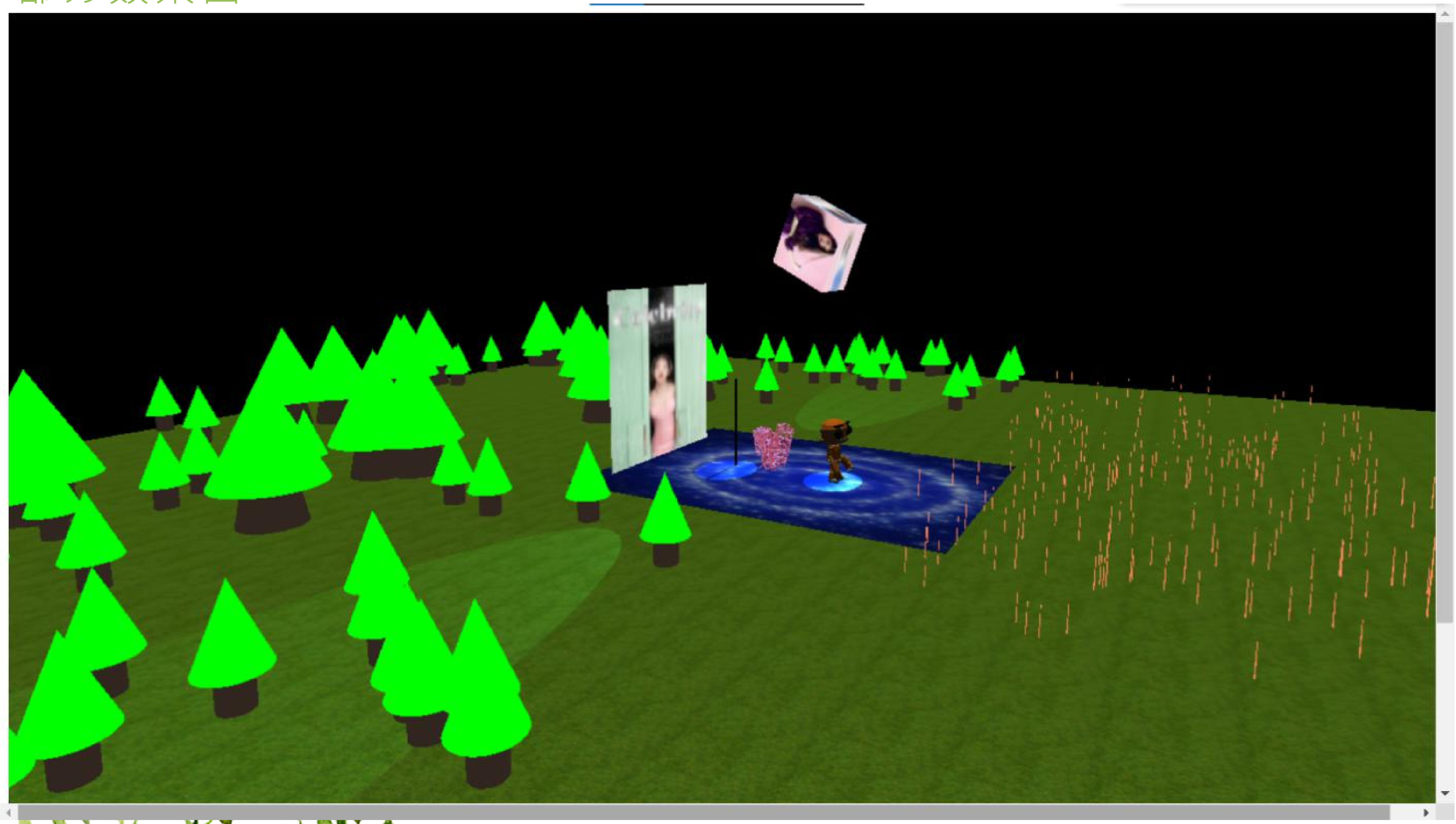
功能展示

Function display

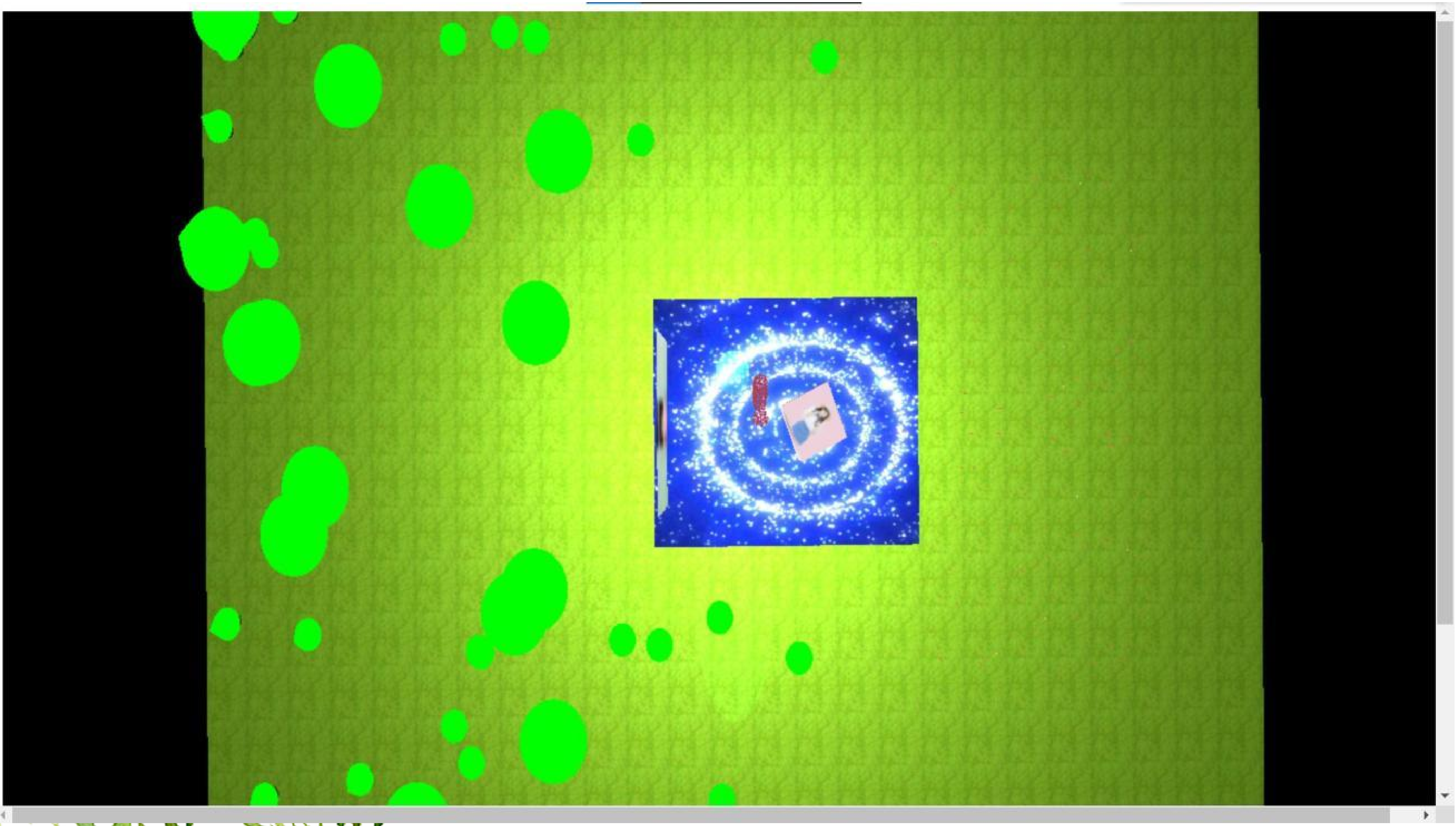
部分效果图



部分效果图



部分效果图



PART 04

分工及自评

Division of labor and self-evaluation

添加您的副标题

姓名	学号	分工	自评
容姿	201912213501009	代码编写，文档撰写，ppt编辑	90
史一郜	201912273501035	代码编写，文档撰写，ppt编辑	90

谢谢观看