# 本次场景规划

草坪

雨景

下雨声

河流

树林

远景

跳动的蓝火

# 草坪

*/\*\**

*\* 创建一个草地地面*

*\*/*

    var geometryGrass = new THREE.PlaneBufferGeometry(1000, 1000); *//矩形平面*

*// 加载草地纹理贴图*

    var textureGrass = new THREE.TextureLoader().load("grass.jpg");

*// 设置纹理的重复模式*

    textureGrass.wrapS = THREE.RepeatWrapping;

    textureGrass.wrapT = THREE.RepeatWrapping;

*// uv两个方向纹理重复数量*

    textureGrass.repeat.set(10, 10);

    var materialGrass = new THREE.MeshLambertMaterial({

      color: 0x777700,

      map: textureGrass,

*// depthTest: false,*

    });

    var meshGrass = new THREE.Mesh(geometryGrass, materialGrass); *//网格模型对象Mesh*

    scene.add(meshGrass); *//网格模型添加到场景中*

    meshGrass.rotateX(-Math.PI / 2);

# 雨景

精灵粒子系统

*// 创建一个组表示所有的雨滴*

    var groupRain = new THREE.Group();

*// 批量创建表示雨滴的精灵模型*

*for* (let i = 0; i < 2000; i++) {

      var spriteRainMaterial = new THREE.SpriteMaterial({

        map: textureRain, *//设置精灵纹理贴图*

      });

*// 创建精灵模型对象*

      var spriteDrop = new THREE.Sprite(spriteRainMaterial);

      groupRain.add(spriteDrop);

*// 控制精灵大小,*

      spriteDrop.scale.set(1, 5, 1); *//// 只需要设置x、y两个分量就可以*

      var k1 = Math.random() - 0.5;

      var k2 = Math.random() - 0.5;

*// 设置精灵模型位置，在空间中随机分布*

      spriteDrop.position.set(1000 \* k1, 300 \* Math.random(), 1000 \* k2)

    }

    scene.add(groupRain); *//雨滴群组插入场景中*

*//下雨效果函数*

    function rain() {

*// 每次渲染遍历雨滴群组，刷新频率30~60FPS，两帧时间间隔16.67ms~33.33ms*

*// 每次渲染都会更新雨滴的位置，进而产生动画效果*

      groupRain.children.forEach(sprite => {

*// 雨滴的y坐标每次减2*

        sprite.position.y -= 2;

*if* (sprite.position.y < 0) {

*// 如果雨滴落到地面，重置y，从新下落*

          sprite.position.y = 300;

        }

      });

    }

# 跳动的蓝火

精灵粒子系统 TWEEN.js

generateFireSprite 更新火焰粒子，同时可以控制渐变颜色

initSparkParticle 控制粒子的运动

Tween.js 控制整体动画效果

*//火焰粒子*

*//refer:https://www.jianshu.com/p/1d8666399b8f*

    function generateFireSprite() {

      var canvas = document.createElement('canvas');

      canvas.width = 16;

      canvas.height = 16;

      var context = canvas.getContext('2d');

      var gradient = context.createRadialGradient(canvas.width / 2, canvas.height / 2, 0, canvas.width / 2, canvas

        .height / 2, canvas.width / 2);

      gradient.addColorStop(0, 'rgba(255,255,255,1)');

      gradient.addColorStop(0.2, 'rgba(0,255,255,1)');

      gradient.addColorStop(0.4, 'rgba(0,0,64,1)');

      gradient.addColorStop(1, 'rgba(0,0,0,1)');

      context.fillStyle = gradient;

      context.fillRect(0, 0, canvas.width, canvas.height);

*return* canvas;

    }

    function initSparkParticle(particle, delay) {

      var particle = *this* instanceof THREE.Sprite ? *this* : particle;

      var delay = delay !== undefined ? delay : 0;

      particle.position.set(0, 0, 0);

      particle.scale.x = particle.scale.y = Math.random() \* 32 + 16;

      new TWEEN.Tween(particle)

        .delay(delay)

        .to({}, 400)

        .onComplete(initSparkParticle)

        .start();

      new TWEEN.Tween(particle.position)

        .delay(delay)

        .to({

*//控制火焰高度*

*//x: Math.random() \* 3000 - 500,*

          y: Math.random() \* 3000 - 500,

*//z: Math.random() \* 3000 - 500*

        }, 10000)

        .start();

      new TWEEN.Tween(particle.scale)

        .delay(delay)

        .to({

          x: 0.01,

          y: 0.01

        }, 1000)

        .start();

    }

    function Fire() {

      TWEEN.update(); *//更新火苗*

    }

# 流动河流

Uv动画

*//水面流动函数*

    function flow(time) {

      textureWater.offset.y += time / 6000;

    }

关闭深度检测     depthTest: false,从而避免草坪和河流因深度信息导致的自遮挡

*/\**

*创建一条河流*

*\*/*

    var geometryWater = new THREE.PlaneBufferGeometry(100, 1000); *//矩形平面*

*//加载水的纹理贴图*

    var textureLoader = new THREE.TextureLoader();

    var textureWater = textureLoader.load('water.jpg'); *// 加载纹理贴图*

    var textureWater\_norm = textureLoader.load('water\_norm1.jpeg'); *// 加载法线贴图*

*// 设置纹理的重复模式*

    textureWater.wrapS = THREE.RepeatWrapping;

    textureWater.wrapT = THREE.RepeatWrapping;

    textureWater\_norm.wrapS = THREE.RepeatWrapping;

    textureWater\_norm.wrapT = THREE.RepeatWrapping;

    textureWater.repeat.set(1, 10);

    textureWater\_norm.repeat.set(1, 10);

    var materialWater = new THREE.MeshPhongMaterial({

*// color: 0x0000ff,*

      side: THREE.DoubleSide, *//两面可见*

      specular: 0x4488ee,

      shininess: 12,

      map: textureWater, *// 设置纹理贴图*

      normalMap: textureWater\_norm,

      depthTest: false,

      normalScale: new THREE.Vector2(1.5, 1.5),

      transparent: true,

*// 设置材质透明度*

      opacity: 0.5,

*//wireframe: true, //线框模式渲染*

    }); *//材质对象Material*

    var meshWater = new THREE.Mesh(geometryWater, materialWater); *//网格模型对象Mesh;*

    geometryWater.rotateX(-Math.PI / 2);

    scene.add(meshWater);

# 树林

精灵模型，使树面片【贴图】永远正对相机（用户视角）达到三维效果

*/\*\**

*\* 精灵创建树林效果*

*\*/*

*// 加载树纹理贴图*

    var textureRain = new THREE.TextureLoader().load("tree.png");

*// 批量创建表示一个树的精灵模型*

*for* (let i = 0; i < 20; i++) {

      var spriteMaterialTree = new THREE.SpriteMaterial({

        map: textureRain, *//设置精灵纹理贴图*

*//depthTest:false,*

      });

*// 创建精灵模型对象*

      var spriteTree = new THREE.Sprite(spriteMaterialTree);

      scene.add(spriteTree);

*// 控制精灵大小,*

      spriteTree.scale.set(100, 100, 1); *//// 只需要设置x、y两个分量就可以*

      var k1 = Math.random() - 0.5;

      var k2 = Math.random() - 0.5;

*// 设置精灵模型位置，在xoz平面上随机分布*

      spriteTree.position.set(1000 \* k1, 50, 1000 \* k2)

    }

# 远景

天空盒材质为双向

*/\**

*添加天空盒*

*\*/*

    var Skybox = new THREE.SphereGeometry(600, 50, 50);

    var materialSkybox = new THREE.MeshBasicMaterial({

      color: 0xffffff,

      side: THREE.DoubleSide,

    });

    var meshSkybox = new THREE.Mesh(Skybox, materialSkybox);

    scene.add(meshSkybox);

    var textureLoader = new THREE.TextureLoader();

    var texture = textureLoader.load('satara\_night\_1k.jpg');

    meshSkybox.material.map = texture;

    scene.background = texture;

# 下雨声

音频源循环播放

*/\**

*添加音频源*

*\*/*

    var listener = new THREE.AudioListener();

    var audio = new THREE.Audio(listener);

    var audioLoader = new THREE.AudioLoader();

    audioLoader.load('Rain.mp3', function (AudioBuffer) {

      audio.setBuffer(AudioBuffer);

      audio.setLoop(true);

      audio.setVolume(0.3);

      audio.play()

    });