郭隆邦 2023-01-20

14. 阵列立方体和相机适配体验

视频讲解

本节课通过阵列一片立方体，进一步体验下透视投影相机的投影规律。

for循环创建一列模型

```js
const geometry = new THREE.BoxGeometry(100, 100, 100);
//材质对象Material
const material = new THREE.MeshLambertMaterial({
    color: 0x00ffff, //设置材质颜色
    transparent: true,//开启透明
    opacity: 0.5,//设置透明度
});
for (let i = 0; i < 10; i++) {
    const mesh = new THREE.Mesh(geometry, material); //网格模型对象Mesh
    // 沿着x轴分布
    mesh.position.set(i*200,0,0);
    scene.add(mesh); //网格模型添加到场景中
}
```

双层for循环创建阵列模型

```js
//创建一个长方体几何对象Geometry
const geometry = new THREE.BoxGeometry(100, 100, 100);
//材质对象Material
const material = new THREE.MeshLambertMaterial({
    color: 0x00ffff, //设置材质颜色
    transparent: true,//开启透明
    opacity: 0.5,//设置透明度
});
for (let i = 0; i < 10; i++) {
    for (let j = 0; j < 10; j++) {
        const mesh = new THREE.Mesh(geometry, material); //网格模型对象Mesh
        // 在XOZ平面上分布
```

```
            mesh.position.set(i * 200, 0, j * 200);
            scene.add(mesh); //网格模型添加到场景中
        }
    }
```

相机位置拉远，可以看到更大的观察范围

```js
const camera = new THREE.PerspectiveCamera(30, width / height, 1, 3000);
// camera.position.set(292, 223, 185);
//在原来相机位置基础上拉远，可以观察到更大的范围
 camera.position.set(800, 800, 800);
 camera.lookAt(0, 0, 0);
```

超出视锥体远裁界面的范围的会被剪裁掉

```js
// const camera = new THREE.PerspectiveCamera(30, width / height, 1, 3000);
const camera = new THREE.PerspectiveCamera(30, width / height, 1, 8000);
// camera.position.set(292, 223, 185);
// 超出视锥体远裁界面的范围的会被剪裁掉，不渲染  可以调整far参数适配
camera.position.set(2000, 2000, 2000);
camera.lookAt(0, 0, 0);
```

改变相机观察目标

```js
// const camera = new THREE.PerspectiveCamera(30, width / height, 1, 3000);
const camera = new THREE.PerspectiveCamera(30, width / height, 1, 8000);
camera.position.set(2000, 2000, 2000);
// camera.lookAt(0, 0, 0);
// 改变相机观察目标点
camera.lookAt(1000, 0, 1000);
```

注意相机控件OrbitControls会影响lookAt设置，注意手动设置OrbitControls的目标参数

```js
// 设置相机控件轨道控制器OrbitControls
const controls = new OrbitControls(camera, renderer.domElement);
// 相机控件.target属性在OrbitControls.js内部表示相机目标观察点，默认0,0,0
// console.log('controls.target', controls.target);
controls.target.set(1000, 0, 1000);
```

```
controls.update();//update()函数内会执行camera.lookAt(controls.targe)
```

远小近大投影规律

透视投影相机的投影规律是远小近大，通过相机观察阵列立方体大小变化，可以看到距离相机越远，立方体的渲染视觉效果越小。

fov改变

增加相机视角fov，视锥体范围更大，意味着可以看到渲染范围更大，远小近大的视觉效果更明显。

← 13. stats查看threejs渲染帧率

15. Threejs常见几何体简介→