郭隆邦 2023-03-03

1. 射线Ray

射线 `Ray` 和三维向量 `Vector3` 一样属于数学几何计算相关的API,可以进行射线交叉计算。

射线 `Ray`

学习Three.js中的射线 `Ray` 概念，你可以类比数学几何中提到的射线，在三维空间中，一条线把一个点作为**起点**，然后沿着某个**方向**无限延伸。

```
// 创建射线对象Ray
const ray = new THREE.Ray()
```

射线起点 `.origin`

射线 `Ray` 的起点 `.origin` 在3D空间中的坐标，可以用一个三维向量 `Vector3` 的x、y、z分量表示。

```
// 设置射线起点
ray.origin = new THREE.Vector3(1,0,3);
```

起点 `.origin` 属性值是三维向量 `Vector3` ，也可以用 `.set()` 方法设置。

```
// 设置射线起点
ray.origin.set(1, 0, 3);
```

射线方向 `.direction`

射线 `Ray` 的方向 `.direction` 通用用一个三维向量 `Vector3` 表示,向量长度保证为1，也就是单位向量。

```
// 表示射线沿着x轴正方向
ray.direction = new THREE.Vector3(1,0,0);
// 表示射线沿着x轴负方向
ray.direction = new THREE.Vector3(-1,0,0);
```

```
// 表示射线沿着y方向
ray.direction = new THREE.Vector3(0,1,0);
```

注意`.direction`的值需要是单位向量，不是的话可以执行`.normalize()`归一化或者说标准化。

```
ray.direction = new THREE.Vector3(5,0,0).normalize();
```

```
// 表示射线沿着xy坐标轴的中间线
ray.direction = new THREE.Vector3(1,1,0).normalize();
```

`.intersectTriangle()`方法

射线`Ray`有很多关于数学计算的方法，下面就先介绍一个与三角形交叉计算相关的方法`.intersectTriangle()`,简单说，就是计算一个射线和一个三角形在3D空间中是否交叉。

执行`.intersectTriangle()`方法，如果相交返回交点坐标，不相交返回空值`null`。

```
// 三角形三个点坐标
const p1 = new THREE.Vector3(100, 25, 0);
const p2 = new THREE.Vector3(100, -25, 25);
const p3 = new THREE.Vector3(100, -25, -25);
const point = new THREE.Vector3();//用来记录射线和三角形的交叉点
// `.intersectTriangle()`计算射线和三角形是否相交叉，相交返回交点，不相交返回null
const result = ray.intersectTriangle(p1,p2,p3,false,point);
console.log('交叉点坐标', point);
console.log('查看是否相交', result);
```

`.intersectTriangle()`参数4表示是否进行背面剔除，p1,p2,p3可以理解为一个三角形，有正反两面，一面是正面，一面是反面，关于三角形正反面可以参考2.3节讲解。

在一面观察p1,p2,p3，如果沿着三个点的顺序转圈是逆时针方向，表示正面，另一面观察，p1,p2,p3就是顺时针方向，表示背面。

`.intersectTriangle()` 参数4设为true，表示进行背面剔除，虽然从几何空间上讲，该案例源码射线和三角形虽然交叉，但在threejs中，三角形背面对着射线，视为交叉无效，进行背面剔除，返回值 `r` 是 `null` 。

```
const r = ray.intersectTriangle(p1,p2,p3,true,point);
console.log('查看是否相交', r);
```

← 7. 抗锯齿后处理

2. Raycaster(射线拾取模型)→