郭隆邦 2023-01-29

# 1. 生成圆弧顶点

通过代码算法生成圆弧线上的顶点坐标，并最后绘制一个圆弧效果。

学习本节课之前，确保你已经掌握章节2 中，关于几何体顶点数据的讲解。

```js
const geometry = new THREE.BufferGeometry(); //创建一个几何体对象
const R = 100; //圆弧半径
const N = 50; //分段数量
const sp = 2 * Math.PI / N;//两个相邻点间隔弧度
// 批量生成圆弧上的顶点数据
const arr = [];
for (let i = 0; i < N; i++) {
    const angle =  sp * i;//当前点弧度
    // 以坐标原点为中心，在XOY平面上生成圆弧上的顶点数据
    const x = R * Math.cos(angle);
    const y = R * Math.sin(angle);
    arr.push(x, y, 0);
}
//类型数组创建顶点数据
const vertices = new Float32Array(arr);
// 创建属性缓冲区对象
//3个为一组，表示一个顶点的xyz坐标
const attribue = new THREE.BufferAttribute(vertices, 3);
// 设置几何体attributes属性的位置属性
geometry.attributes.position = attribue;

// 线材质
const material = new THREE.LineBasicMaterial({
    color: 0xff0000 //线条颜色
});
// 创建线模型对象    构造函数：Line、LineLoop、LineSegments
// const line = new THREE.Line(geometry, material);
const line = new THREE.LineLoop(geometry, material);//线条模型对象
```

## 生成圆弧顶点数据

以坐标原点为中心，在XOY平面上生成圆弧上的顶点数据。

绘制圆弧线，本质就是绘制一个正n边形，n越大，圆弧细分数或者说精度越高。

通过for循环沿着圆弧线，通过三角函数计算顶点坐标，批量生成圆弧上顶点数据。

```js
const R = 100; //圆弧半径
const N = 50; //分段数量
const sp = 2 * Math.PI / N;//两个相邻点间隔弧度
// 批量生成圆弧上的顶点数据
const arr = [];
// N控制圆弧精度：就是创建多少个顶点
for (let i = 0; i < N; i++) {
    const angle =  sp * i;//当前点弧度
    // 以坐标原点为中心，在XOY平面上生成圆弧上的顶点数据
    const x = R * Math.cos(angle);
    const y = R * Math.sin(angle);
    arr.push(x, y, 0);//xyz坐标
}
```

## 线模型渲染圆弧线

使用`Line`渲染圆弧线会有一个缺口，不完全闭合，使用`LineLoop`可以封闭最后缺口。

```js
// 线材质
const material = new THREE.LineBasicMaterial({
    color: 0xff0000 //线条颜色
});
// 创建线模型对象    构造函数：Line、LineLoop、LineSegments
// const line = new THREE.Line(geometry, material);
const line = new THREE.LineLoop(geometry, material);//线条模型对象
```

使用Line渲染，也可以修改for循环条件多增加一个点绘制圆弧。

```js
for (let i = 0; i < N; i++) {

}
```

```js
// 多绘制一个点
for (let i = 0; i < N + 1; i++) {
```

```
}
```

## 绘制半圆弧

```js
const sp = 2 * Math.PI / N;//完整圆弧
const sp = 1 * Math.PI / N;//半圆弧
```

## 圆弧设置圆心坐标

```js
const R = 100; //圆弧半径
const N = 50; //分段数量
const sp = 2 * Math.PI / N;//两个相邻点间隔弧度
// 设置圆心坐标
const cx = 200;
const cy = 100;
for (let i = 0; i < N+1; i++) {
    const angle = sp * i;//当前点弧度
    const x = cx + R * Math.cos(angle);
    const y = cy + R * Math.sin(angle);
    arr.push(x, y, 0);
}
```

← 7. 模型加载进度条

2. 几何体方法.setFromPoints()→