**【Three.js入门】灯光与阴影、平行光阴影属性、聚光灯的属性和应用**

**Three.js 系列文章目录**

| **Three.js 专栏** | **参考链接** |
| --- | --- |
| Three.js 入门案例 | [程序人生——与足球共舞的火柴人](https://zahuopu.blog.csdn.net/article/details/128171267?spm=1001.2014.3001.5502) |
| Three.js 入门（一） | [创建第一个场景和物体（轨道控制器、坐标轴辅助器…）](https://blog.csdn.net/qq_45902692/article/details/128354520?spm=1001.2014.3001.5501) |
| Three.js 入门（二） | [处理动画、尺寸自适应、双击进入/退出全屏](https://blog.csdn.net/qq_45902692/article/details/128363780?spm=1001.2014.3001.5501) |
| Three.js 入门（三） | [图形用户界面GUI、BufferGeometry创建矩形、随机生成三角形](https://blog.csdn.net/qq_45902692/article/details/128380569?spm=1001.2014.3001.5501) |
| Three.js 入门（四） | [纹理及其常用属性、透明纹理、环境遮挡贴图与强度](https://blog.csdn.net/qq_45902692/article/details/128394703?spm=1001.2014.3001.5501) |

【使用 Three.js 实现的效果】

**一、灯光与阴影的关系与设置**

灯光阴影：

* 材质要满足能够对光照有反应
* 设置渲染器开启阴影计算 renderer.shadowMap.enabled = true
* 设置光照投射阴影 directionalLight.castShadow = true
* 设置物体投射阴影 sphere.castShadow = true
* 设置物体接收阴影 plane.receiveShadow = true

**下面的解释与上面的灯光阴影相对应：**

.shadowMap 它包含阴影贴图的引用

.castShadow 如果设置为 true 该平行光会产生动态阴影

.castShadow 对象是否被渲染到阴影贴图中

.receiveShadow 材质是否接收阴影

我们先创建一个球体和一个平面，为球体阴影到平面上做准备。之后以上的四步也是缺一不可的（渲染器阴影计算，光照投射阴影，球体投射阴影，平面接收阴影）

// 创建一个球

const sphereGeometry = new THREE.SphereBufferGeometry(1, 20, 20)

// 设置材质

const material = new THREE.MeshStandardMaterial()

// 结合实体和材质

const sphere = new THREE.Mesh(sphereGeometry, material)

// 打开球体的投射阴影

sphere.castShadow = true

// 添加到场景中

scene.add(sphere)

// 创建平面

const planeGeometry = new THREE.PlaneBufferGeometry(10, 10)

const plane = new THREE.Mesh(planeGeometry, material)

plane.position.set(0, -1, 0)

// 正面旋转 90°，调整平面的位置

plane.rotation.x = -Math.PI / 2

// 开启平面接收阴影

plane.receiveShadow = true

scene.add(plane)

// 环境光：均匀的照亮场景中的所有物体

const light = new THREE.AmbientLight(0xffffff, 0.9)

scene.add(light)

// 平行光：方向从一个平行光位置 position 到 target 位置

const directionLight = new THREE.DirectionalLight(0xffffff, 0.95)

// 设置光的位置

directionLight.position.set(10, 10, 10)

// 开启光照投射阴影

directionLight.castShadow = true

scene.add(directionLight)

// 初始化渲染器

const renderer = new THREE.WebGLRenderer()

// 设置渲染的尺寸大小

renderer.setSize(window.innerWidth, window.innerHeight)

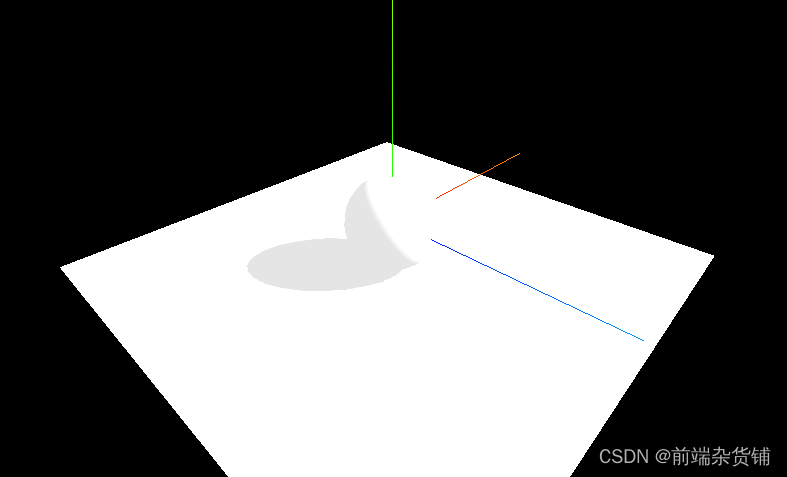
// 开启场景中的阴影贴图

renderer.shadowMap.enabled = true

// 将 webgl 渲染的 canvas 内容添加到 body

document.body.appendChild(renderer.domElement)

......



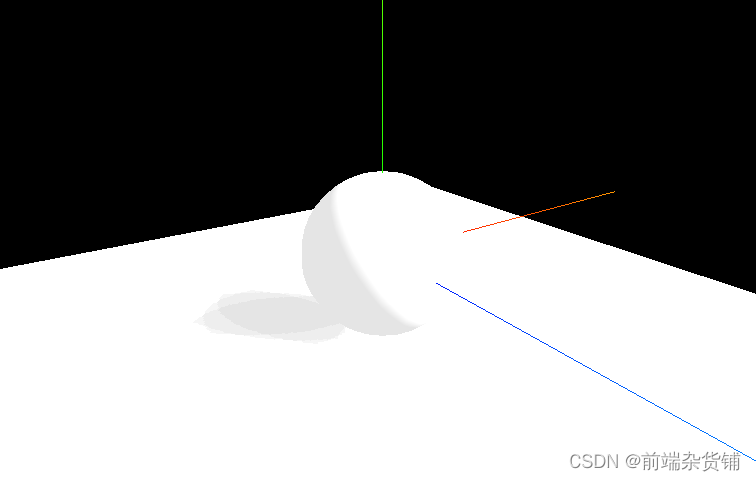
**二、平行光阴影属性**

**1、设置阴影模糊度**

.radius 将此值设置为大于 1 的值将模糊阴影的边缘。较高的值会在阴影中产生不必要的条带效果。

// 设置阴影贴图模糊度

directionLight.shadow.radius = 20

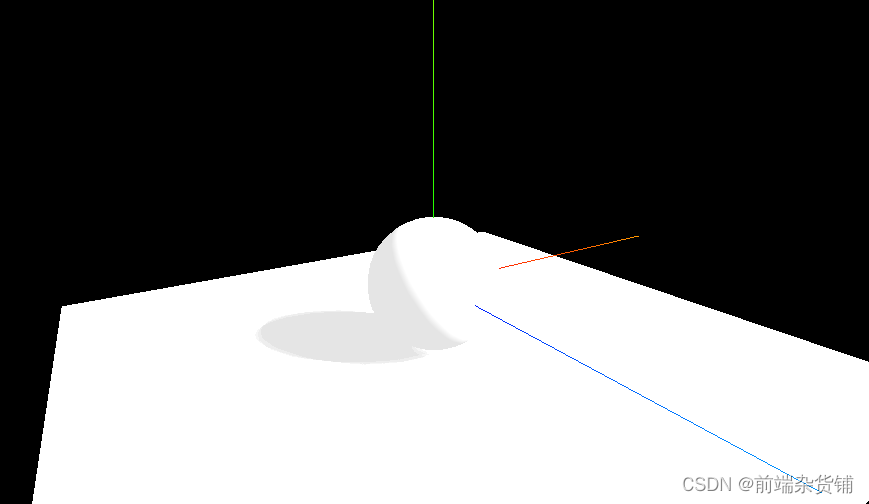


**2、阴影贴图**

.mapSize 一个 Vector2 定义阴影贴图的宽度和高度。较高的值会以计算时间为代价提供更好地阴影质量。但值必须是 2 的幂，默认值是（512, 512）

// 设置阴影贴图的分辨率

directionLight.shadow.mapSize.set(2048, 2048)



**3、平行光投射相机的属性**

近端，远端，上下左右。当我们改变近端的值时，阴影的大小会相应的发生改变

// 设置平行光投射相机的属性

directionLight.shadow.camera.near = 18.2

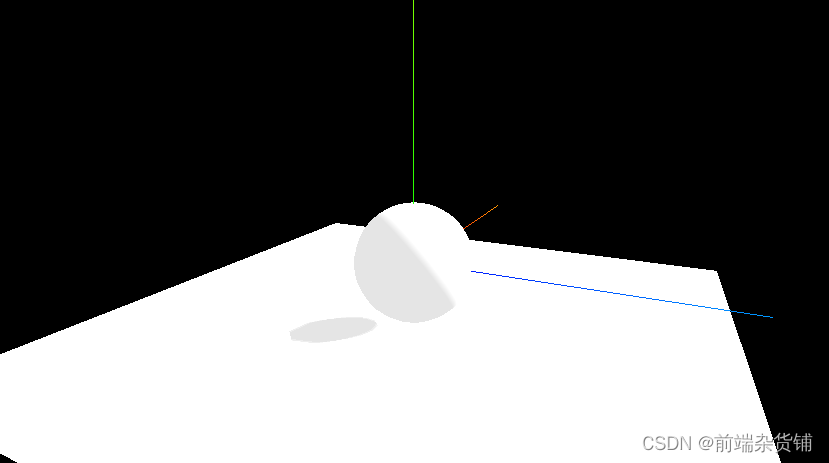
directionLight.shadow.camera.far = 500

directionLight.shadow.camera.top = 5

directionLight.shadow.camera.bottom = -5

directionLight.shadow.camera.left = -5

directionLight.shadow.camera.right = 5



**三、聚光灯的属性和应用**

聚光灯（SpotLight）：光线从一个点沿一个方向射出，随着光线照射的变远，光线圆锥体的尺寸也逐渐增大。

相关属性：

* color 十六进制光照颜色，缺省值 0xffffff（白色）
* intensity （可选参数）光照强度，缺省值 1
* distance 从光源发出光的最大距离，其强度根据光源的距离线性衰减
* angle 光线散射角度，最大值为 Math.PI/2
* penumbra 聚光锥的半影衰减百分比。在0和1之间的值。默认为 0
* decay 沿着光照距离的衰减量

使用聚光灯，改变 target 的位置，通过 GUI 来查看不同距离的显示效果

......

// 聚光灯

const spotLight = new THREE.SpotLight(0xffffff, 0.5)

// 设置光的位置

spotLight.position.set(5, 5, 5)

// 设置阴影贴图模糊度

spotLight.shadow.radius = 20

// 设置阴影贴图的分辨率

spotLight.shadow.mapSize.set(4096, 4096)

// 设置目标

spotLight.target = sphere

// 设置光照投射阴影

spotLight.castShadow = true

scene.add(spotLight)

// 使用 GUI，改变球体在 x 轴的位置，查看投影效果

gui

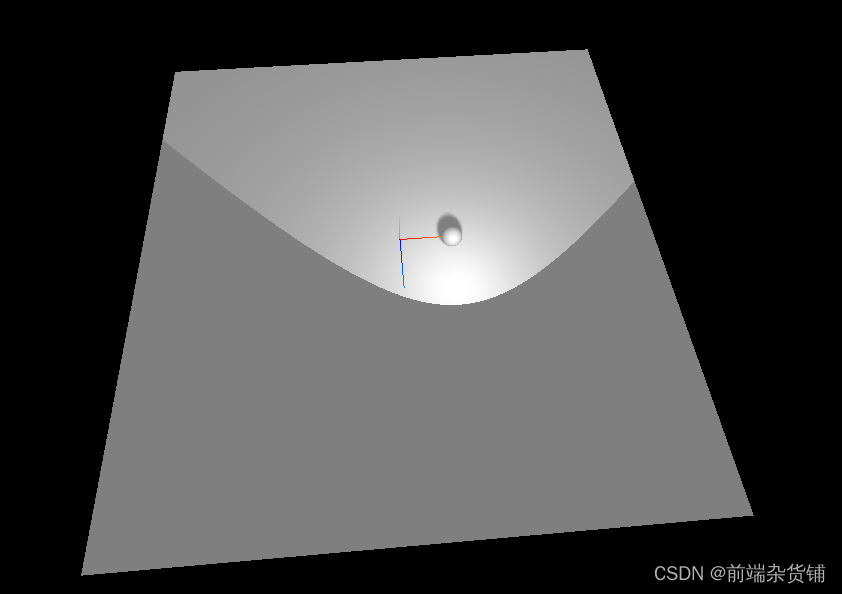
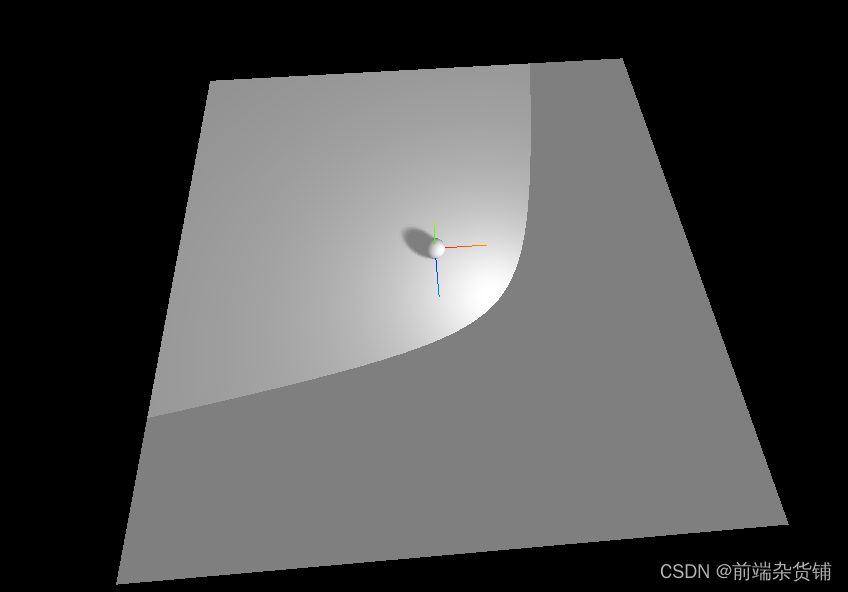
.add(spotLight.position, 'x')

.min(-5)

.max(5)

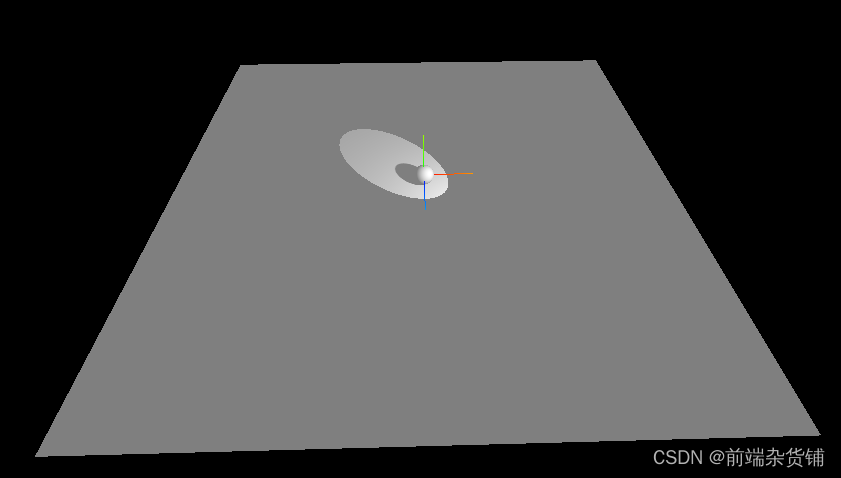
.step(0.1)

https://csdnimg.cn/release/blogv2/dist/pc/img/newCodeMoreBlack.png



// 设置聚光灯的角度

spotLight.angle = Math.PI / 10



// 设置从光源发出光的最大距离

spotLight.distance = 0

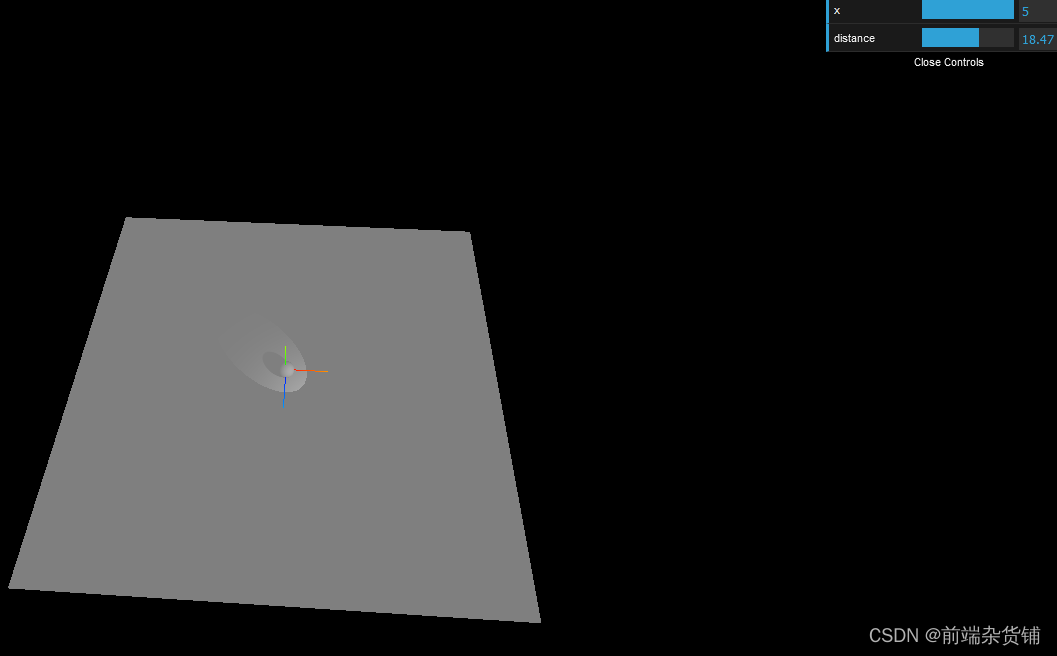
gui

.add(spotLight, 'distance')

.min(0)

.max(30)

.step(0.01)



// 聚光锥的半影衰减百分比

spotLight.penumbra = 0

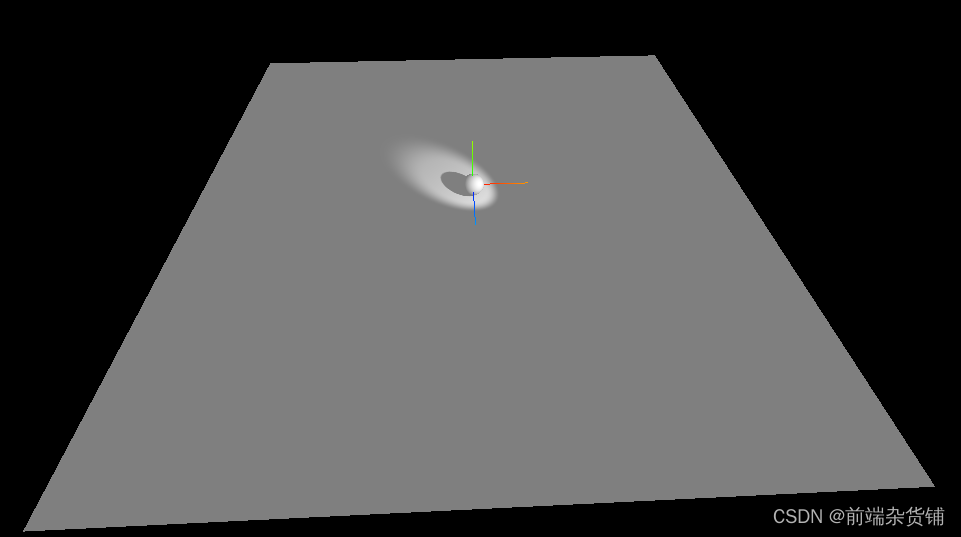
gui

.add(spotLight, 'penumbra')

.min(0)

.max(1)

.step(0.01)



// 设置沿着光照距离的衰减量

spotLight.decay = 0

......

gui

.add(spotLight, 'decay')

.min(0)

.max(5)

.step(0.01)

......

renderer.physicallyCorrectLights = true

