郭隆邦 2023-02-25

1. 后处理(发光描边OutlinePass)

查看threejs文件包目录`examples/jsm/postprocessing/`，你可以看到Three.js提供了一个扩展库`EffectComposer.js`,通过EffectComposer可以实现一些后期处理效果。

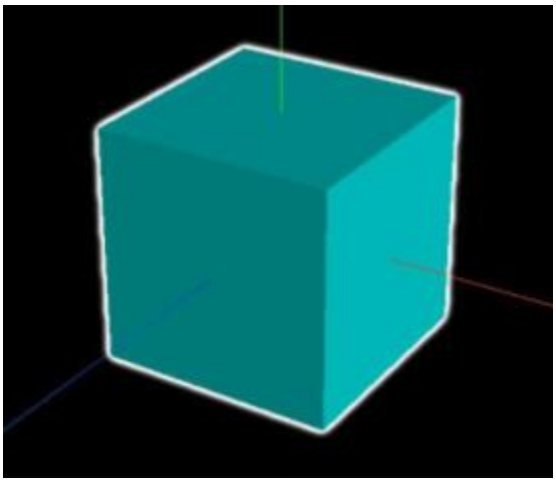
所谓threejs后期处理，就像ps一样，对threejs的渲染结果进行后期处理，比如添加发光效果。

不同功能后处理通道

查看threejs文件包目录`examples/jsm/postprocessing/`，你可以看到threejs提供了很多后处理通道，想实现什么样的后期处理效果，需要调用threejs对应的后处理通道扩展库。

- `OutlinePass.js`：高亮发光描边
- `UnrealBloomPass.js`：Bloom发光
- `GlitchPass.js`：画面抖动效果

比如`OutlinePass.js`扩展库提供的类`OutlinePass`就可以给一个模型添加一个高亮发光描边，下面就给大家演示下如何实现。



引入`EffectComposer.js`

你可以在threejs文件包目录`examples/jsm/postprocessing/`找到扩展库EffectComposer.js。

代码中引入后处理扩展库EffectComposer.js

```js
import { EffectComposer } from 'three/addons/postprocessing/EffectComposer.js';
```

大家都知道three.js WebGL渲染器执行渲染方法`.render()`会得到一张图像，如果你需要对一个webgl渲染器的渲染结果进行后期处理，就把它作为`EffectComposer`的参数。

```js
// 创建后处理对象EffectComposer，WebGL渲染器作为参数
const composer = new EffectComposer(renderer);
```

渲染器通道`RenderPass`

RenderPass.js扩展库目录：`examples/jsm/postprocessing/`

```js
// 引入渲染器通道RenderPass
import { RenderPass } from 'three/addons/postprocessing/RenderPass.js';
```

通过`EffectComposer(renderer)`指定了需要后处理的渲染器`WebGLRenderer`，渲染器通道`RenderPass`的作用是指定后处理对应的相机`camera`和场景`scene`。

```js
// 创建一个渲染器通道，场景和相机作为参数
const renderPass = new RenderPass(scene, camera);
```

给`EffectComposer`添加一个渲染器通道`RenderPass`。

```js
// 设置renderPass通道
composer.addPass(renderPass);
```

`OutlinePass`通道

`OutlinePass`可以给指定的某个模型对象添加一个高亮发光描边效果。

OutlinePass.js扩展库目录：`examples/jsm/postprocessing/`

```js
// 引入OutlinePass通道
import { OutlinePass } from 'three/addons/postprocessing/OutlinePass.js';
```

创建OutlinePass通道

```js
// OutlinePass第一个参数v2的尺寸和canvas画布保持一致
const v2 = new THREE.Vector2(window.innerWidth, window.innerHeight);
// const v2 = new THREE.Vector2(800, 600);
const outlinePass = new OutlinePass(v2, scene, camera);
```

OutlinePass属性 `.selectedObjects`

three.js场景中有多个模型的话，你希望给哪个模型对象设置发光描边效果，就可以通过OutlinePass的选择对象属性 `.selectedObjects` 设置。

```js
// 一个模型对象
outlinePass.selectedObjects = [mesh];
// 多个模型对象
outlinePass.selectedObjects = [mesh1,mesh2,group];
```

设置OutlinePass通道

最后把创建好的OutlinePass通道添加到后处理composer中。

```js
// 设置OutlinePass通道
composer.addPass(outlinePass);
```

渲染循环执行 `EffectComposer.render()`

渲染循环中后处理EffectComposer执行 `.render()` ，会调用webgl渲染器执行 `.render()` ，也就是说 `renderer.render(scene, camera)` 不用再执行。

```js
// 渲染循环
function render() {
    composer.render();
    // renderer.render(scene, camera);
    requestAnimationFrame(render);
}
render();
```

修改OutlinePass默认描边效果

参考下节课具体讲解

← 3. Sprite模拟下雨、下雪

2. OutlinePass描边样式→