郭隆邦 2023-01-28

4. OrbitControls辅助设置相机参数

实际开发的时候，一方面可以通过OrbitControls旋转缩放预览3D模型，另一方面也可以辅助你选择合适的相机参数。

OrbitControls知识点回顾

相机控件 `OrbitControls` 旋转缩放平移本质上就是在改变相机 `Camera` 的参数。

- 旋转：拖动鼠标左键
- 缩放：滚动鼠标中键
- 平移：拖动鼠标右键

`OrbitControls` 改变相机位置 `.position`

通过 `OrbitControls` 旋转和缩放，本质上就是在改变透视投影相机 `PerspectiveCamera` 的位置 `.position` 。

渲染循环中不停地打印相机的位置属性，你可以通过相机控件旋转或缩放三维场景，同时通过浏览器控制台观察相机位置变化。

```js
function render() {
  requestAnimationFrame(render);
  // 浏览器控制台查看相机位置变化
  console.log('camera.position',camera.position);
}
render();
```

通过 `OrbitControls` 设置相机位置 `.position`

上节课关于相机整体预览三维场景代码设置的时候，第一步是根据渲染范围的数量级，大概设置相机的位置参数，其实第二部，相机位置具体参数，可以借助 `OrbitControls` 可视化旋转或缩放，然后选择一个合适的渲染效果，浏览器控制台记录下此时的相机位置。

```js
camera.position.set(200, 200, 200);//第1步：根据场景渲染范围尺寸设置
camera.position.set(-144, 95, 95); //第2步：通过相机控件辅助设置OrbitControls
```

`OrbitControls` 改变相机 `.lookAt` 观察目标

通过 `OrbitControls` 平移，OrbitControls的 `.target` 属性会发生变化， `.target` 属性对应的就是透视投影相机 `PerspectiveCamera` 的 `.lookAt` 观察目标`。

```js
function render() {
  requestAnimationFrame(render);
  // 浏览器控制台查看controls.target变化，辅助设置lookAt参数
  console.log('controls.target',controls.target);
}
render();
```

通过 `OrbitControls` 设置 `.lookAt()` 参数

参照 `OrbitControls` 设置相机位置 `.position` 的过程，你可以平移三维场景，然后选择一个合适的渲染效果，记录下此时相机控件目标属性 `controls.target` 的值，然后作为透视投影相机 `.lookAt()` 的参数。

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
// camera.lookAt(0, 0, 0);
const x = -1.2,y = -15,z = 10;//通过OrbitControls辅助设置
camera.lookAt(x, y, z);

// 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(x, y, z); //与lookAt参数保持一致
controls.update(); //update()函数内会执行camera.lookAt(controls.target)
```

← 3. 加载.gltf文件(模型加载全流程)

5. gltf不同文件形式(.glb)→