**实验四、实现对Three.js的装机并设计第一个三维动画场景**

1. **实验目的和要求。**

根据视频讲解（链接在本页最后）的Three.js装机教程实现对Three.js的装机操作，并基于该库，根据《THREE.JS开发指南》第一章所学内容，设计第一个三维动画场景

**基本要求：**Three.js装机教程中任选一种方法完成装机，并设计一个简单基本的三维动画场景进行测试；

1. **实验步骤**

①导入three库中three.js三维引擎和OrbitControls.js鼠标操作三维场景。

<script src="M:\图形学\three.js-dev\build\three.js"></script>

<script src="M:\图形学\three.js-dev\examples\js\controls\OrbitControls.js">

②创建场景对象Scene：

var scene = new THREE.Scene();

③创建网格模型对象Mesh，其几何对象为Geometry，设定材质为material，通过add函数添加到场景scene：

var geometry = new THREE.BoxGeometry(25, 50, 50);

var material = new THREE.MeshLambertMaterial;

var mesh = new THREE.Mesh(geometry, material);

scene.add(mesh);

本实验中创建了两个模型mesh1和mesh2。

④设置光源。

var point = new THREE.PointLight(0xfffffff);

point.position.set(100, 100, 100);

scene.add(point);

var ambient = new THREE.AmbientLight(0x444444);

scene.add(ambient);

⑤设置相机

var camera = new THREE.PerspectiveCamera( 45, width / height, 1, 1000 );

camera.position.set(200, 0, 0);

camera.lookAt(scene.position);

⑥设置渲染器并渲染：

var renderer = new THREE.WebGLRenderer();

renderer.setSize(width, height);

renderer.setClearColor(0xb9d3ff, 1);

document.body.appendChild(renderer.domElement);

function render() {

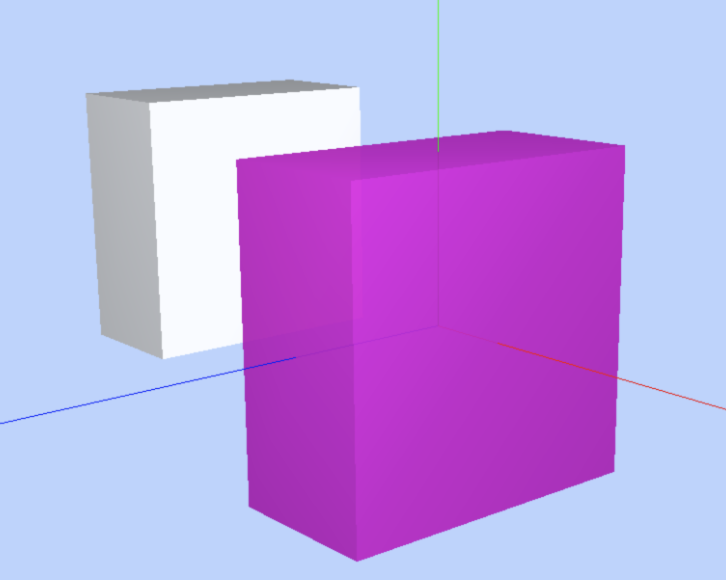
console.log(camera.position)

renderer.render(scene,camera);

}

render();

1. **实验结果**



**4、实验结果分析和结论。**

①在装机教程的基础上采用js文件方法完成Three.js的引入。

②完成了渲染操作。

③通过代码行：var controls = new THREE.OrbitControls(camera,renderer.domElement);创建控件对象，controls.addEventListener('change', render);实现了监听鼠标、键盘事件，使模型跟随鼠标动作。