三维可视化
控制器 & 贴图

李春芳

2020.6.24

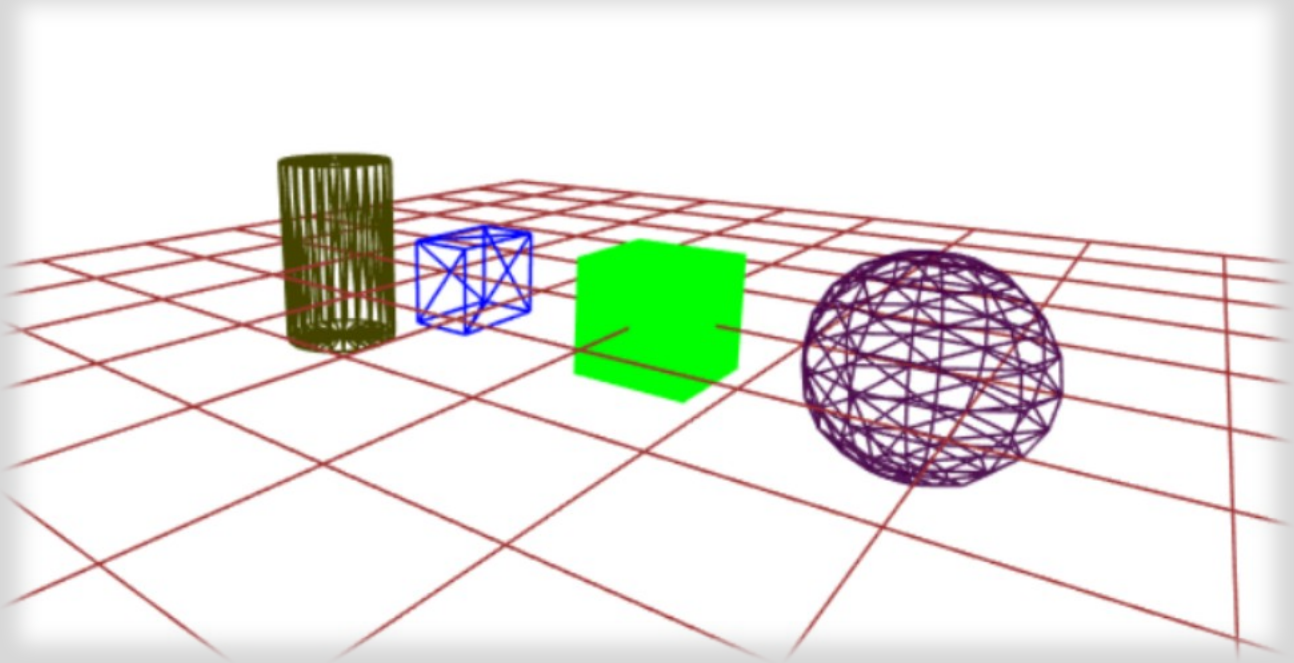
WebGL 三维可视化 Three.JS

控制器

贴图

选择器

控制器



①

- <script src="OrbitControls.js"></script>
- https://g14n.info/three-orbitcontrols/

添加 DIV 放画布 Canvas 区域

② ◦在 BODY 里添加 DIV

◦

③ ◦在 JS 里选择它

◦var threeDiv = document.getElementById('canvas-frame');

④ ◦//document.body.appendChild(renderer.domElement);

◦替换为： threeDiv.appendChild(renderer.domElement);

清除背景颜色

⑤

- var renderer = new THREE.WebGLRenderer({antialias: true, alpha: true});//WebGLRenderer();
- renderer.setClearColor(0xffffff, 0);
- renderer.setSize(window.innerWidth, window.innerHeight);

```
var orbitControls = new THREE.OrbitControls(camera, threeDiv);
orbitControls.target = box[6].position;//控制焦点
orbitControls.autoRotate = false;//自动旋转
scene.controls = orbitControls;


render();
function render()
{
    scene.controls.update();
    //object_selection.render(group, scene.userData.camera);
    renderer.render(scene, camera);
    requestAnimationFrame(render);
}
```

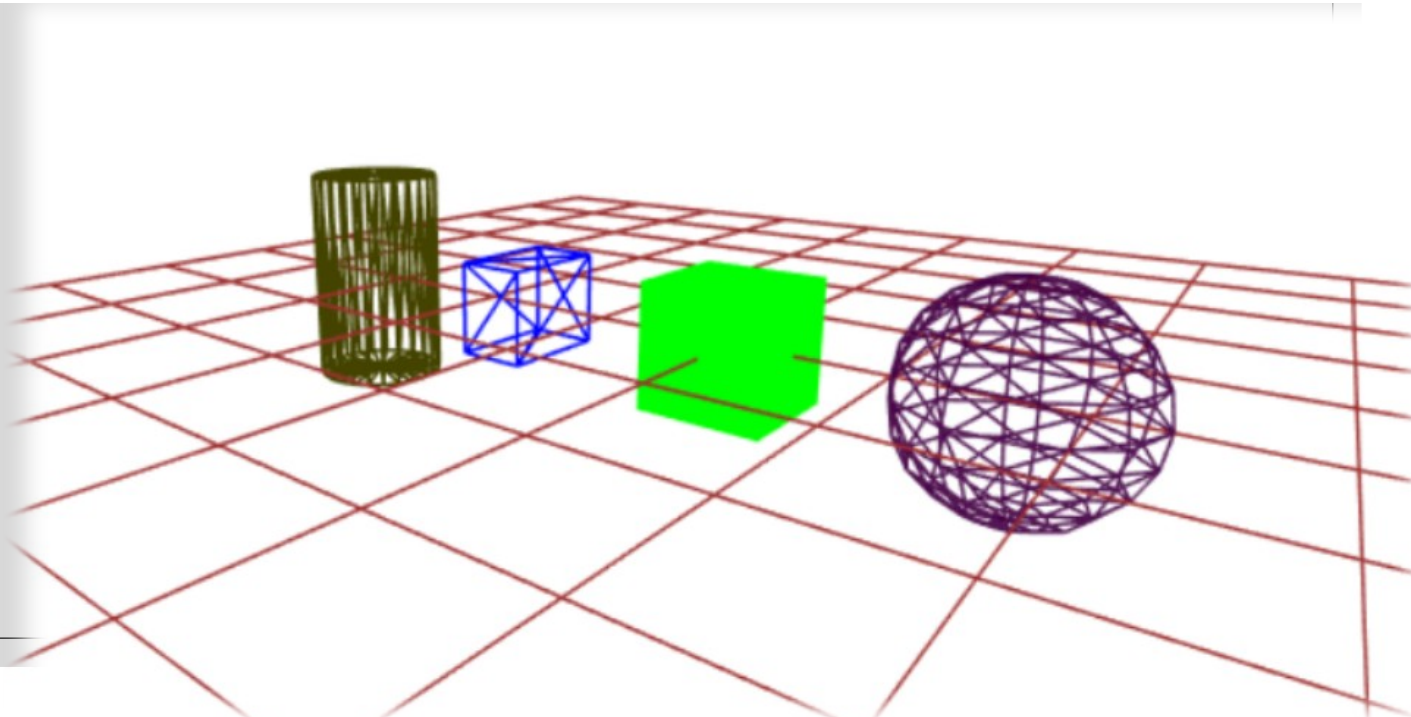
⑥ 控制器

⑦ 更新控制器

添加控制器

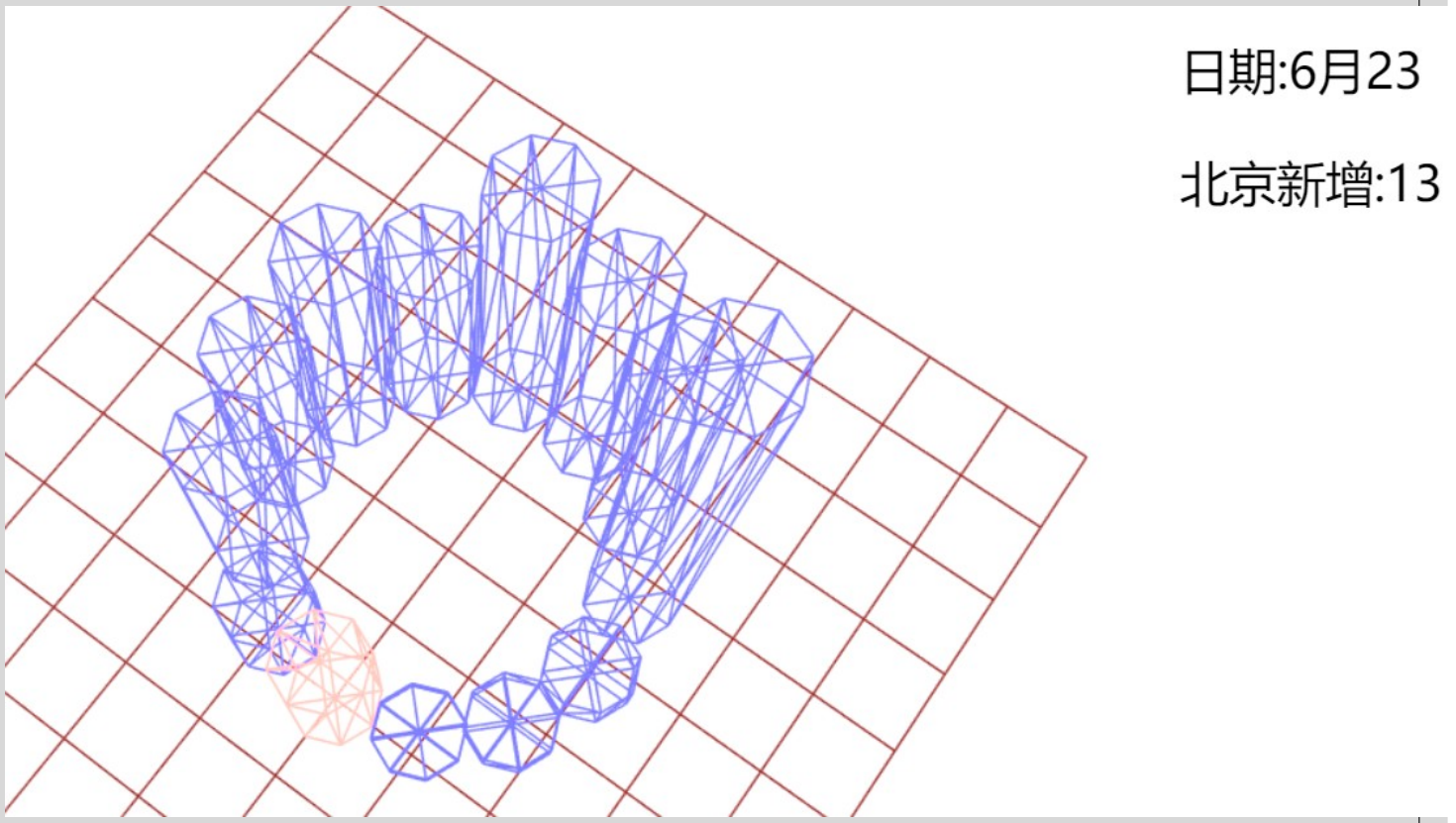
添加网格：地平线

```
initGrid();
function initGrid(){
    //平面
    var gridXZ = new THREE.GridHelper(1200, 20, 0xa23131, 0xa23131);//4个参数分别是：网格宽度、等分数、中心线颜色，网格线颜色
    gridXZ.position.set( 0,0,0 );
    gridXZ.material.linewidth = 10;
    scene.add(gridXZ);
}
```



圆柱体 3D 图表

- 选择元素
- 在画布上检测鼠标事件
- objSelection.js
- BY LiMin@Ruyi



```
THREE.ObjectSelection = function(parameters) {
  parameters = parameters || {};

  this.domElement = parameters.domElement || document;
  this.INTERSECTED = null;

  var _this = this;

  var callbackSelected = parameters.selected;//处理函数通过外界传入
  var callbackClicked = parameters.clicked;
  var callbackMousedown = parameters.mousedown;
  var callbackMouseup = parameters.mouseup;
  var mouse = { x: 0, y: 0 };

  //鼠标移入事件
  this.domElement.addEventListener( 'mousemove', onDocumentMouseMove, false );
  //获取当前二维坐标
  function onDocumentMouseMove( event ) {
    mouse.x = ( event.clientX / window.innerWidth ) * 2 - 1;
    mouse.y = - ( event.clientY / window.innerHeight ) * 2 + 1;
  }

  //单击事件
  this.domElement.addEventListener( 'click', onDocumentMouseClick, false );
  function onDocumentMouseClick( event ) {
    if(_this.INTERSECTED) {
      if(typeof callbackClicked === 'function') {
        callbackClicked(_this.INTERSECTED);
      }
    }
  }
```

```
  //在渲染中注意选择对象
  this.render = function(scene, camera) {
    var vector = new THREE.Vector3( mouse.x, mouse.y, 0.5 );
    vector.unproject(camera);//获取当前三维坐标

    var raycaster = new THREE.Raycaster(camera.position, vector.sub(camera.position).normalize());
    var intersects = raycaster.intersectObject(scene, true);//把scene作为检测对象

    if( intersects.length > 0 ) {
      if ( this.INTERSECTED != intersects[ 0 ] ) {
        if ( this.INTERSECTED ) {//若此类中实例对象不是选取对象，则颜色值任不变
          this.INTERSECTED.object.material.color.setHex( this.INTERSECTED.currentHex );
        }
        this.INTERSECTED = intersects[ 0 ];//依据检测，更换当前类中几何对象
        this.INTERSECTED.currentHex = this.INTERSECTED.object.material.color.getHex();//对象当前颜色
        // this.INTERSECTED.material.color.setHex( 0xff0000 );//移入对象则变红
        if(typeof callbackSelected === 'function') {
          callbackSelected(this.INTERSECTED);//调用选择处理函数，个性化处理
        }
      }
    } else {//若没有检测到scene中选择对象
      if ( this.INTERSECTED ) {
        this.INTERSECTED.object.material.color.setHex( this.INTERSECTED.currentHex );
      }
      this.INTERSECTED = null;//初始化
      if(typeof callbackSelected === 'function') {
        callbackSelected(this.INTERSECTED);
      }
    }
  };
};
```

```
THREE.ObjectSelection = function(para
  parameters = parameters || {};

  this.domElement = parameters.domEle
  this.INTERSECTED = null;

  var _this = this;

  var callbackSelected = parameters.s
  var callbackClicked = parameters.cl
  var callbackMousedown = parameters.
  var callbackMouseup = parameters.mo
  var mouse = { x: 0, y: 0 };

  //鼠标移入事件
  this.domElement.addEventListener( '
  //获取当前二维坐标
  function onDocumentMouseMove( event
    mouse.x = ( event.clientX / windo
    mouse.y = - ( event.clientY / win
  }

  //单击事件
  this.domElement.addEventListener( '
  function onDocumentMouseClick( even
    if(_this.INTERSECTED) {
      if(typeof callbackClicked === '
        callbackClicked(_this.INTERSE
      }
    }
  }
```

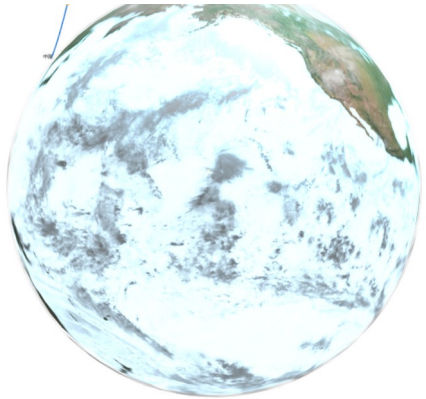
```
//在渲染中注意选择对象
this.render = function(scene, camera) {
  var vector = new THREE.Vector3( mouse.x, mouse.y, 0.5 );
  vector.unproject(camera);//获取当前三维坐标

  var raycaster = new THREE.Raycaster(camera.position, vector.sub(camera.position).normalize());
  var intersects = raycaster.intersectObject(scene, true);//把scene作为检测对象

  if( intersects.length > 0 ) {
    if ( this.INTERSECTED != intersects[ 0 ] ) {
      if ( this.INTERSECTED ) {//若此类中实例对象不是选取对象，则颜色值任不变
        this.INTERSECTED.object.material.color.setHex( this.INTERSECTED.currentHex );
      }
      this.INTERSECTED = intersects[ 0 ];//依据检测，更换当前类中几何对象
      this.INTERSECTED.currentHex = this.INTERSECTED.object.material.color.getHex();//对象当前颜色
      // this.INTERSECTED.material.color.setHex( 0xff0000 );//移入对象则变红
      if(typeof callbackSelected === 'function') {
        callbackSelected(this.INTERSECTED);//调用选择处理函数，个性化处理
      }
    }
  } else {//若没有检测到scene中选择对象
    if ( this.INTERSECTED ) {
      this.INTERSECTED.object.material.color.setHex( this.INTERSECTED.currentHex );
    }
    this.INTERSECTED = null;//初始化
    if(typeof callbackSelected === 'function') {
      callbackSelected(this.INTERSECTED);
    }
  }
};
};
```

1- 参数

2- 事件

3- 参数

2024/2/22



地球贴图

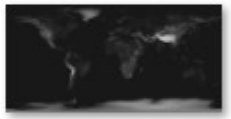
从球体到地球

球体 Mesh

(C:) › Tomcat9 › webapps › ruyi3D › earth › texture › earth


2_no_clouds_4k.jpg


earth_clouds_2048.png


elev_bump_4k.jpg

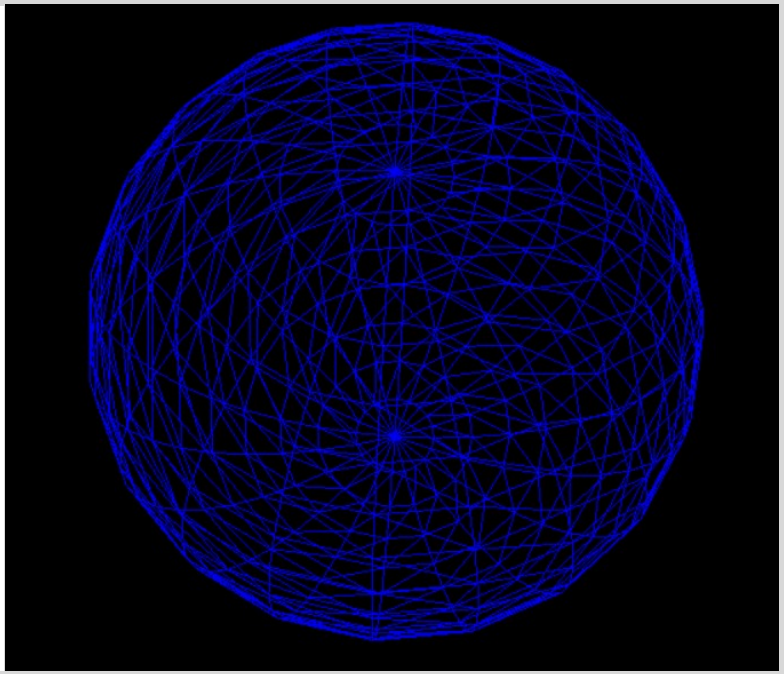

galaxy.png


grey.jpg


hong.jpg

- 贴图文件

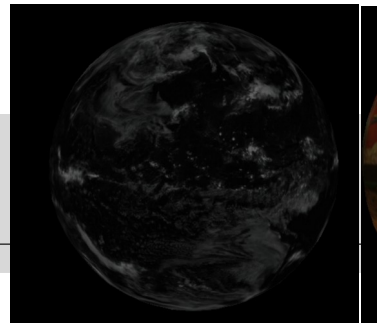
```
// 球体网格模型
var geometry = new THREE.SphereGeometry(100, 20, 20);


var material2 = new THREE.MeshLambertMaterial({
    color: 0xff00ff,
    wireframe:true,
    opacity:0.7,
    transparent:true
});

var sphere = new THREE.Mesh(geometry, material2); //网格模型对象Mesh
sphere.translateX(-100); //球体网格模型沿Y轴正方向平移120
scene.add(sphere);
```



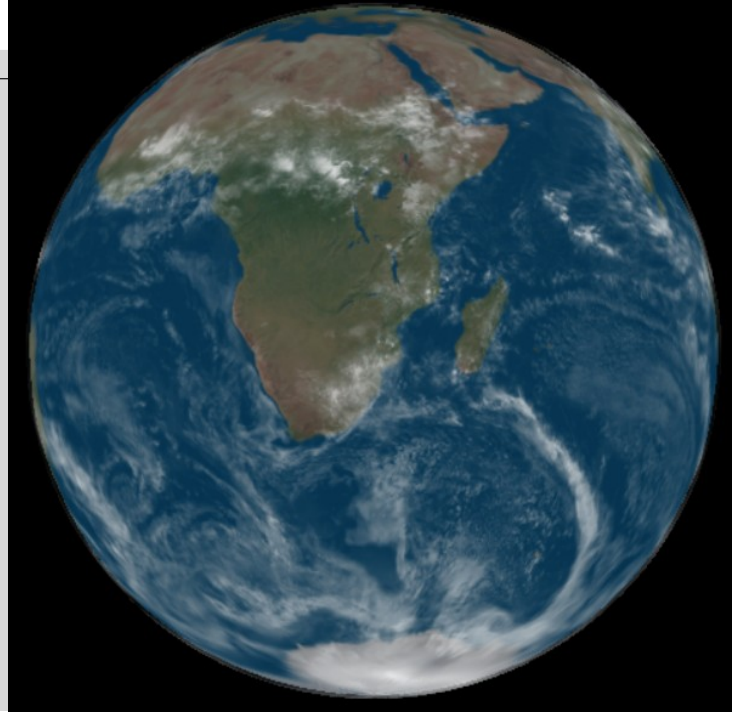
几何体贴图

```
//纹路加载函数
function getTexture(str){//str为纹理url
    var loader = new THREE.TextureLoader();
    return loader.load(str);
}
```

```
var geometry = new THREE.SphereGeometry(100, 20, 20);
var material = new THREE.MeshPhongMaterial({
        //wireframe: true,
        color: 0xFFFFFF,
        opacity:0.5,
        map:          getTexture('texture/earth/2_no_clouds_4k.jpg'),
        //map:          getTexture('texture/earth/earth_clouds_2048.png'),
        //map:          getTexture('texture/earth/hong.jpg'),
        //map:          getTexture('texture/earth/galaxy.png'),
        //bumpMap:      getTexture('texture/earth/elev_bump_4k.jpg'),//凹凸贴图
        transparent: true,
        emissiveIntensity: 0.05,
        //alphaMap:     getTexture('texture/earth/grey.jpg'),//透明度贴图，黑色：完全透明;白色：完全不透明

    });
var sphere = new THREE.Mesh(geometry, material); //网格模型对象Mesh
sphere.translateX(-100); //球体网格模型沿Y轴正方向平移120
scene.add(sphere);
```



添加云层（比地表高一点）



```
creatAero(101);
//大气层
function creatAero(radius){
    var cloudsGeo = new THREE.SphereGeometry(radius,radius/2,radius/2);
    var cloudsMater = new
    THREE.MeshPhongMaterial({map:getTexture('texture/earth/earth_clouds_2048.png'),transparent:true,opacity:1});
    var cloudsMesh = new THREE.Mesh(cloudsGeo,cloudsMater);
    scene.add(cloudsMesh);
}
```

添加位置数据

```
createLink();

function createLink(){
    var locations = [
            {
                name: '中国',
                position: {x: 75.00796918081518, y: 61.91934379181622,z: -28.0733576398174 98},
            }, {
                name: '英国',
                position: {x: -0, y: 80.19051430548129, z: 57.366479276535955},
            }];
    var firstSprite;
    for(let i in locations){
        var sprite = createLocationSprite(locations[i].position);
        scene.add( sprite );

        var textSprite = createText(locations[i].position, locations[i].name);
        scene.add( textSprite );
        console.log(i);
        if(i == 0){
            firstSprite = sprite;
        }
        else{
            createLine(firstSprite, sprite);
        }
    }

}
```

```
function createLine(obj0, obj1, radius){
    var positions = [];
    positions.push(obj0.position);

    var midVector = obj0.position.clone().add(obj1.position.clone());
    if(midVector.length () > radius*1.5){
        midVector.multiplyScalar(0.8);
    }
    positions.push(midVector);

    positions.push(obj1.position);

    var curve = new THREE.CatmullRomCurve3( positions );//Create a smooth 3d spline curve from a series of points
    //https://threejs.org/docs/#api/en/extras/curves/CatmullRomCurve3

    lightMove(curve);

    var points = curve.getPoints( 50 );//从曲线中取出position，细分数为20
    var geometry = new THREE.BufferGeometry().setFromPoints( points );

    var material = new THREE.LineBasicMaterial( { color : 0x2376DD } );
    var curveObject = new THREE.Line( geometry, material );
    scene.add(curveObject);
}
```

```
function createLocationSprite(position){
    var spriteMaterial = new THREE.SpriteMaterial({
        map: getTexture('texture/sprite/snowflake7_alpha.png'),
    });
    var sprite = new THREE.Sprite(spriteMaterial);

    sprite.position.set(position.x*1.03, position.y*1.03, position.z*1.03);
    sprite.scale.set(5,5,5);

    return sprite

}
```

添加文字 & 光源

```
function createText(position, title){
    let canvas = document.createElement("canvas");
    canvas.width = 512;
    canvas.height = 512;
    let ctx = canvas.getContext("2d");
    ctx.fillStyle = "#000000";
    ctx.font = "40px Yahei";
    ctx.textAlign = 'center';
    ctx.textBaseline = 'middle';
    ctx.fillText(title, 256, 256, 512);

    let spriteMap = new THREE.CanvasTexture(canvas);
    spriteMap.needsUpdate = true;
    let spriteMaterial = new THREE.SpriteMaterial({map: spriteMap});

    let sprite = new THREE.Sprite(spriteMaterial);
    sprite.scale.set(30, 30, 1);

    sprite.position.copy(position).multiplyScalar(1.05);
    return sprite
}
```

```
function lightMove(curve){
    var that = this;
    var pointGeometry = new THREE.SphereGeometry(1, 20, 20);
    var pointMaterial = new THREE.MeshBasicMaterial({color: 0xFFCF71});

    addLightPoint();
    function addLightPoint() {
        var index = 0;
        var pointMesh = new THREE.Mesh(pointGeometry, pointMaterial);

        pointMesh.position.copy(curve.getPointAt(index));
        that.scene.add(pointMesh);

        function pointAnimate() {
            index+=0.001;
            if(index>= 1) {
                index = 0;
            }
            pointMesh.position.copy(curve.getPointAt(index));
            requestAnimationFrame(pointAnimate);
        }
        pointAnimate();
    }

}
```

三维可视化：场景导航、游戏、 VR/AR
对数据可视化来说： D3.JS 轻量级更适合