文档二 详细设计

一、 系统完整功能描述

本文借助 objloader. js 文件，读取运动员模型的 obj 文件数据，然后通过鼠标或键盘改变相机的位置、视角对其实现平移功能，利用屏幕上的控件实现物体位置的平移，也可以选择不同的投影方式：正交投影、透视投影，平移、旋转功能是通过矩阵的运算实现的，它具有存储小，计算高效的特点。

二、 各模块流程、结构具体实现、关键函数、变量说明

平移模块：

```
function handleKeyDown(event) {
    var key = event.keyCode;
    currentKey[key] = true;
    if( changePos === 1 ){
        switch (key) {
            case 65: //left//a
                dyt = -90;
                dx +=step*Math.sin(dyt);;
                document.getElementById("xpos").value=dx;
                break;
            case 68: // right//d
                dyt =90;
                dx+= step*Math.sin(dyt);
                document.getElementById("xpos").value=dx;
                break;
            case 87: // up//w
                dy += step;
                dyt = 180;
                document.getElementById("ypos").value=dy;
                break;
            case 83: // down//s
                dy -= step;
                dyt =0;
                document.getElementById("ypos").value=dy;
                break;
```

```
case 81: // q -90~-180
    dyt = -120;
    dx += step*Math.sin(dyt);
    dy -= step*Math.sin(dyt);
    document.getElementById("xpos").value=dx;
    document.getElementById("ypos").value=dy;
    break;
case 69: //e 90~180
     dyt = 150
     dx -= step*Math.sin(dyt);
     dy -= step*Math.sin(dyt);
     document.getElementById("xpos").value=dx;
    document.getElementById("ypos").value=dy;
     break;
case 90: // a//z 0~-90
    dyt = -30;
    dx -= step*Math.sin(dyt);
    dy -= step*Math.sin(dyt);
    document.getElementById("xpos").value=dx;
    document.getElementById("ypos").value=dy;
    break;
case 88: // x xx0-90
    dyt = 60;
    dx -= step*Math.sin(dyt);
    dy += step*Math.sin(dyt);
    document.getElementById("xpos").value=dx;
    document.getElementById("ypos").value=dy;
    break:
```

① 通过移动物体的坐标移动物体的位置

读入文件，解析 obj 文件，选择改变物体位置，js 接受信息，得到 changePos=1，既可以使用键盘控制物体的位置，也可以用屏幕上的控件实现物体的转移。

键盘控制：通过 document.getElementById（）函数得到数值，利用 ASIIC 码分辨，并通过按键控制模型的位置和朝向（按键 WSADQEZX 分别代表方向上、下、左、右、左上、右上、左下、右下），得到最新的 **dx += step*math.sin(dyt)**，dy，dz 值，通过 mat4.translate(mvMatrix, mvMatrix,

vec3.fromValues(dx, dy, dz))函数得到 mvMatrix,，最后通过 gl.uniformMatrix4fv 传递给顶点着色器 modelViewMatrix，最新的 **modelViewMatrix** 参与矩阵相乘 gl_Position = projectionMatrix * **modelViewMatrix** * vPosition 便可得到物体的最新坐标矩阵，实现平移效果。

屏幕上的控件控制：通过 **document.getElementById**("xpos") 获取目标，得到 dx，dy，dz，其余步骤与键盘控制一致。

旋转模块：

```
        break;
    case 72: // h//ytheta-
        dyt -= stept;
        document.getElementById("yrot").value=dyt;
        break;
    case 75: // k//ytheta+
        dyt += stept;
        document.getElementById("yrot").value = dyt;
        break;
    case 85: // u//xtheta+
        dxt -= stept;
        document.getElementById("xrot").value = dxt;
        break;
    case 74: // j//xtheta-
        dxt += stept;
        document.getElementById("xrot").value = dxt;
        break;
    case 78: // n//ztheta+
        dzt += stept;
        document.getElementById("zrot").value = dzt;
        break;
    case 77: // m//ztheta-
        dzt -= stept;
        document.getElementById("zrot").value = dzt;
        break;
    case 82: // r//reset
        dx = 0;
```

旋转与平移的过程类似。上述过程中获得坐标位置的同时，也可以获得角度。

① 通过转动物体的坐标来实现物体旋转

键盘控制：通过 document.getElementById（）函数得到数值，利用 asiic 码分辨，利用 h/k/u/j/n/m 控制物体的上下左右前后旋转，得到最新的 **cxt -= stepct**，cyt，czt 值，

通过 mat4.rotateZ(mvMatrix, mvMatrix, **dzt * Math.PI / 180.0**)类似 dxt，dyt 作用到数组 mvMatrix 中，通过 gl.uniformMatrix4fv 传递给顶点着色器 modelViewMatrix，最新的 **modelViewMatrix** 参与矩阵相乘 gl_Position = projectionMatrix * **modelViewMatrix** * vPosition 便可得到物体的最新坐标矩阵，实现旋转效果。

屏幕上的控件控制：通过 **document.getElementById**("xrpos")获取目标，得到 dxt，dyt，dzt，其余步骤与键盘控制一致。

② 通过移动相机的位置实现物体旋转效果

鼠标控制：按下鼠标不放松，得到鼠标按下位置的坐标（lastMouseX，lastMouseY），移动鼠标，得到 deltaX = (newX - lastMouseX)和 deltaY = (newY - lastMouseY)，rthe，通过 vec3.set(**eye**, localRadius * Math.sin(rthe) * Math.cos(**rphi**), localRadius * Math.sin(**rthe**) * Math.sin(rphi), localRadius * Math.cos(rthe))得到 eye 的值，通过 **mat4.lookAt**(mvMatrix, **eye**, at, up)得到 mvMatrix，最后通过 gl.uniformMatrix4fv 传递给顶点着色器

modelViewMatrix，松开鼠标，最新的 **modelViewMatrix** 参与矩阵相乘 gl_Position = projectionMatrix * **modelViewMatrix** * vPosition 便可得到物体的最新坐标矩阵，实现旋转效果。

投影方式模块：

读入文件，解析 obj 文件，选择改变物体位置，js 接受信息，若得到 projectionType=1，通过 **mat4.ortho**(pMatrix, oleft, oright, oybottom, oytop, onear, ofar)函数，若得到 projectionType=2，则通过 **mat4.perspective**(pMatrix, fovy, aspect, pnear, pfar) 得到 pMatrix，通过 gl.uniformMatrix4fv(**projectionMatrix**, false, new Float32Array(pMatrix))把最新的 projectionMatrix 传给顶点着色器，进行矩阵相乘 gl_Position = **projectionMatrix** * **modelViewMatrix** * vPosition 便可得到物体的最新坐标矩阵。

三、实现工具，开发环境，工具库

实现工具：HBuilderX

测试环境：edge chrome 浏览器

开发环境：Windows 10

主要工具库：webgl-utils.js、initShaders.js

四、**小组分工及自评**

小组成员	自评分数	分工情况
李彦祥	90	功能需求设计分析，网页制作，代码实现
沈思宇	90	功能需求设计分析，文档书写，代码实现