

1. 机器人运动

(1) 在 move.js 中通过按键事件侦听实现键盘与机器人的交互。

```
move.html | Fire_fighting_robot | move.js | Fire_fighting_robot
46 var up = vec3.fromValues(0.0, 1.0, 0.0);
47
48 var currentKey = [];
49 // move object
50 function handleKeyDown() {
51     var key = event.keyCode;
52     currentKey[key] = true;
53     switch (key) {
54         case 65: // left//a
55             dxt -= stept;
56             break;
57         case 68: // right//d
58             dxt += stept;
59             break;
60         case 87: // up//w
61             dyt += stept;
62             break;
63         case 83: // down//s
64             dyt -= stept;
65             break;
66         case 90: // a//z
67             dzt += stept;
68             break;
69         case 88: // d//x
70             dzt -= stept;
71             break;
72         case 72: // h//ytheta-
73             dxm -= stepm;
74             break;
75         case 75: // k//ytheta+
76             dxm += stepm;
77             break;
78         case 85: // u//xtheta+
79             dym += stepm;
```

(2) 在 move.js 中通过 mat4.translate 和 mat4.rotateX (mat4.rotateY, Mat4.rotateZ) 分别实现平移和旋转。

```
move.html | Fire_fighting_robot | move.js | Fire_fighting_robot
166 function render() {
167     gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);
168
169     eye = vec3.fromValues(radius * Math.sin(theta) * Math.cos(phi),
170         radius * Math.sin(theta) * Math.sin(phi),
171         radius * Math.cos(theta));
172
173     mat4.lookAt(modelViewMatrix, eye, at, up);
174     mat4.translate(modelViewMatrix, modelViewMatrix, vec3.fromValues(dxm, dym, dzm)); //移动位置
175     mat4.rotateX(modelViewMatrix, modelViewMatrix, dxt); //旋转角度
176     mat4.rotateY(modelViewMatrix, modelViewMatrix, dyt);
177     mat4.rotateZ(modelViewMatrix, modelViewMatrix, dzt);
178
179     mat4.ortho(projectionMatrix, left, right, bottom, ytop, near, far);
180
181     gl.uniformMatrix4fv(modelViewMatrixLoc, false, new Float32Array(modelViewMatrix));
182     gl.uniformMatrix4fv(projectionMatrixLoc, false, new Float32Array(projectionMatrix));
183
184     gl.drawArrays(gl.TRIANGLES, 0, points.length / 3);
185
186     requestAnimationFrame(render);
187 }
188
```

2. 投影

(1) 在 move.js 中实现投影矩阵代码。

```
move.html | Fire_fighting_robot | move.js | Fire_fighting_robot

166 function render() {
167     gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);
168
169     eye = vec3.fromValues(radius * Math.sin(theta) * Math.cos(phi),
170         radius * Math.sin(theta) * Math.sin(phi),
171         radius * Math.cos(theta));
172
173     mat4.lookAt(modelViewMatrix, eye, at, up);
174     mat4.translate(modelViewMatrix, modelViewMatrix, vec3.fromValues(dxm, dym, dzm)); // 移动位置
175     mat4.rotateX(modelViewMatrix, modelViewMatrix, dxt); // 旋转角度
176     mat4.rotateY(modelViewMatrix, modelViewMatrix, dyt);
177     mat4.rotateZ(modelViewMatrix, modelViewMatrix, dzt);
178
179     mat4.ortho(projectionMatrix, left, right, bottom, ytop, near, far);
180
181     gl.uniformMatrix4fv(modelViewMatrixLoc, false, new Float32Array(modelViewMatrix));
182     gl.uniformMatrix4fv(projectionMatrixLoc, false, new Float32Array(projectionMatrix));
183
184     gl.drawArrays(gl.TRIANGLES, 0, points.length / 3);
185
186     requestAnimationFrame(render);
187 }
188 }
```

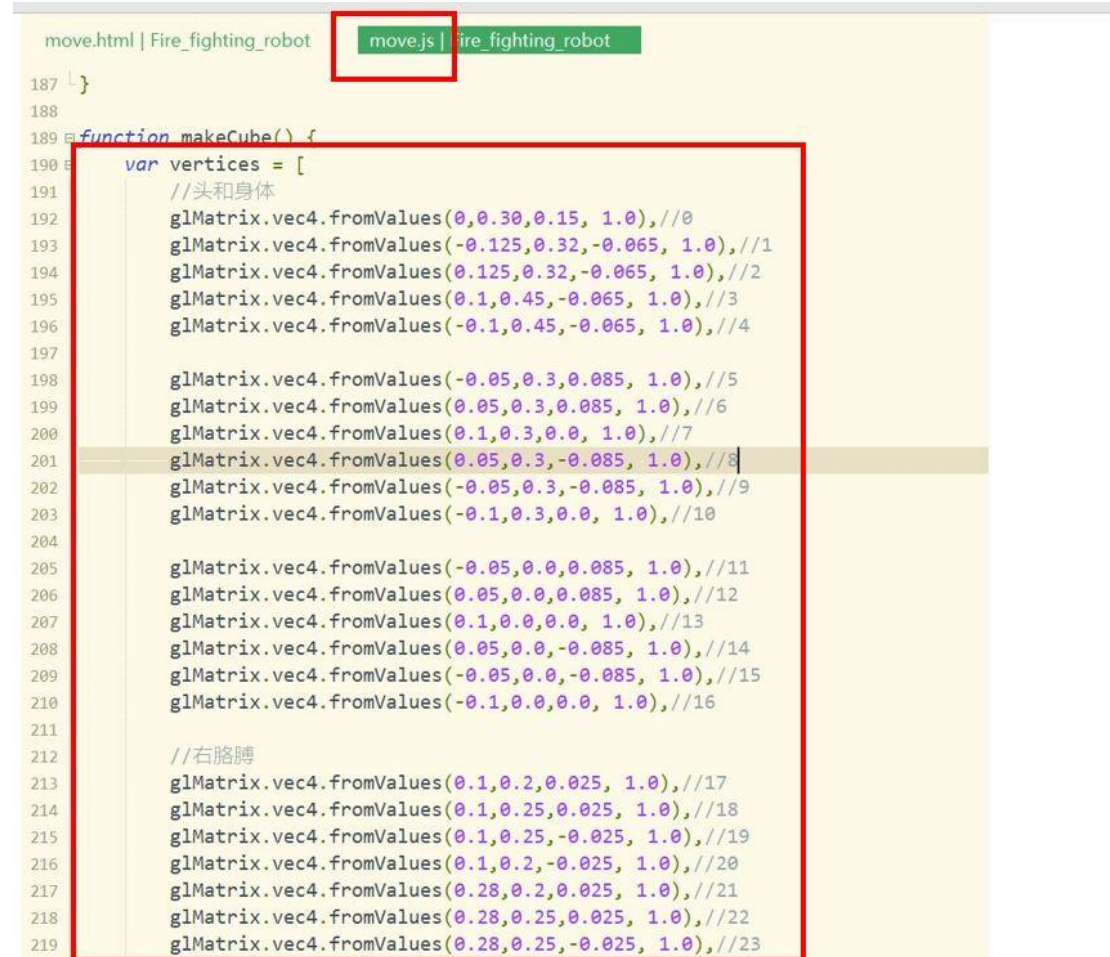
(2) 在 move.html 实现投影矩阵顶点着色器

```
move.html | Fire_fighting_robot | move.js | Fire_fighting_robot

22     background: url("1.png");
23     /* z层级设置低些 避免阻挡交互 */
24     z-index: -1000;
25     /* 背景铺满, 而不是重复平铺 */
26     background-size: 100% 100%;
27
28 }
29
30 </style>
31 <script id="vertex-shader" type="x-shader/x-vertex">
32     attribute vec4 vPosition;
33     attribute vec4 vColor;
34     varying vec4 fColor;
35
36     uniform mat4 modelViewMatrix;
37     uniform mat4 projectionMatrix;
38
39     void main()
40     {
41         fColor = vColor;
42         gl_Position = projectionMatrix * modelViewMatrix * vPosition;
43     }
44 </script>
45
46
```

3. 构建机器人

(1) 设置顶点



```
187 }
188
189 function makeCube() {
190     var vertices = [
191         //头和身体
192         glMatrix.vec4.fromValues(0,0.30,0.15, 1.0),//0
193         glMatrix.vec4.fromValues(-0.125,0.32,-0.065, 1.0),//1
194         glMatrix.vec4.fromValues(0.125,0.32,-0.065, 1.0),//2
195         glMatrix.vec4.fromValues(0.1,0.45,-0.065, 1.0),//3
196         glMatrix.vec4.fromValues(-0.1,0.45,-0.065, 1.0),//4
197
198         glMatrix.vec4.fromValues(-0.05,0.3,0.085, 1.0),//5
199         glMatrix.vec4.fromValues(0.05,0.3,0.085, 1.0),//6
200         glMatrix.vec4.fromValues(0.1,0.3,0.0, 1.0),//7
201         glMatrix.vec4.fromValues(0.05,0.3,-0.085, 1.0),//8
202         glMatrix.vec4.fromValues(-0.05,0.3,-0.085, 1.0),//9
203         glMatrix.vec4.fromValues(-0.1,0.3,0.0, 1.0),//10
204
205         glMatrix.vec4.fromValues(-0.05,0.0,0.085, 1.0),//11
206         glMatrix.vec4.fromValues(0.05,0.0,0.085, 1.0),//12
207         glMatrix.vec4.fromValues(0.1,0.0,0.0, 1.0),//13
208         glMatrix.vec4.fromValues(0.05,0.0,-0.085, 1.0),//14
209         glMatrix.vec4.fromValues(-0.05,0.0,-0.085, 1.0),//15
210         glMatrix.vec4.fromValues(-0.1,0.0,0.0, 1.0),//16
211
212         //右胳膊
213         glMatrix.vec4.fromValues(0.1,0.2,0.025, 1.0),//17
214         glMatrix.vec4.fromValues(0.1,0.25,0.025, 1.0),//18
215         glMatrix.vec4.fromValues(0.1,0.25,-0.025, 1.0),//19
216         glMatrix.vec4.fromValues(0.1,0.2,-0.025, 1.0),//20
217         glMatrix.vec4.fromValues(0.28,0.2,0.025, 1.0),//21
218         glMatrix.vec4.fromValues(0.28,0.25,0.025, 1.0),//22
219         glMatrix.vec4.fromValues(0.28,0.25,-0.025, 1.0),//23
```

(2) 设置顶点颜色

```
move.html | Fire_fighting_robot move.js | Fire_fighting_robot
271 glMatrix.vec4.fromValues(0.03,-0.15,-0.065, 1.0),//65,,
272 glMatrix.vec4.fromValues(0.08,0,0, 1.0),//66
273 ];
274
275 var vertexColors = [
276     //后脑勺
277     glMatrix.vec4.fromValues(0.57, 0.8, 0.918, 1.0),
278     glMatrix.vec4.fromValues(0.57, 0.8, 0.918, 1.0),
279
280     //四个侧脸
281     glMatrix.vec4.fromValues(0.86, 0.86, 0.86, 1.0),
282     glMatrix.vec4.fromValues(0.86, 0.86, 0.86, 1.0),
283     glMatrix.vec4.fromValues(0.57, 0.8, 0.918, 1.0),
284     glMatrix.vec4.fromValues(0.57, 0.8, 0.918, 1.0),
285
286     //身体 20个
287
288
289     //前2个一样颜色
290     glMatrix.vec4.fromValues(0.86, 0.86, 0.86, 1.0),
291     glMatrix.vec4.fromValues(0.86, 0.86, 0.86, 1.0),
292
293     //后2个一样颜色
294     glMatrix.vec4.fromValues(0.57, 0.8, 0.918, 1.0),
295     glMatrix.vec4.fromValues(0.57, 0.8, 0.918, 1.0),
296
297     //上
298     glMatrix.vec4.fromValues(0.86, 0.86, 0.86, 1.0),
299     glMatrix.vec4.fromValues(0.86, 0.86, 0.86, 1.0),
300     glMatrix.vec4.fromValues(0.86, 0.86, 0.86, 1.0),
301     glMatrix.vec4.fromValues(0.86, 0.86, 0.86, 1.0),
302
303     //底4个
```

(3) 定义机器人各面


```
move.html | Fire_fighting_robot | move.js | Fire_fighting_robot

421 |
422 | var faces = [
423 |     //脑袋
424 |     1,4,3,1,2,3, //背
425 |     3,4,0, //上
426 |     1,2,0, //底
427 |     1,0,4, //左
428 |     0,2,3, //右
429 |
430 |     //身体
431 |     5,6,11,11,12,6, //前
432 |     9,8,15,14,15,8, //后
433 |     10,5,9,6,8,7,5,6,9,8,9,6, //上
434 |     15,16,11,12,13,14,15,11,12,15,14,12, //底
435 |     10,5,11,11,16,10, //左前
436 |     6,7,12,12,13,7, //右前
437 |     9,10,15,16,15,10, //左后
438 |     7,8,14,13,14,7, //右后
439 |
440 |     //胳膊1右胳膊
441 |     17,21,18,18,22,21, //前
442 |     20,24,23,19,23,20, //后
443 |     18,19,22,19,23,22, //上
444 |     20,24,17,17,21,24, //下
445 |     18,19,20,17,20,18, //左
446 |     22,23,24,21,24,22, //右
447 |
448 |     28,27,25,25,26,27, //前
449 |     32,31,29,29,30,31, //后
450 |     25,29,32,32,28,25, //左
451 |     27,26,30,27,31,30, //右
452 |     28,27,31,32,31,28, //上
453 |     29,30,25,25,26,30, //下
```

(4) 顶点和颜色输入

```
move.html | Fire_fighting_robot | move.js | Fire_fighting_robot

490 |     61,62,59, //左外
491 |
492 |     //右脚
493 |     63,65,66, //右内侧
494 |     64,65,66, //后
495 |     63,64,65, //底
496 |     63,64,66, //右外
497 | ];
498 |
499 | for (var i = 0; i < faces.length; i++) {
500 |     points.push(vertices[faces[i]][0], vertices[faces[i]][1], vertices[faces[i]][2]);
501 |
502 |     colors.push(vertexColors[Math.floor(i / 3)][0], vertexColors[Math.floor(i / 3)][1], vertexColors[Math.floor(i / 3)][2], vertexColors[Math.floor(i / 3)][3]);
503 | }
504 |
505 | }
```

4. 喷水喷干粉

(1) 获取一个画布对象，并且设置其大小（以下所有步骤的代码都位于 **move.html** 中）

```
71 <canvas id="c" width="500" height="500"></canvas>
```

```
84 // 获取一个画布对象
85 var canvas = document.getElementById("c");
86 // 设置大小和颜色
87 canvas.width = window.innerWidth;
88 canvas.height = window.innerHeight;
89 //canvas.style.backgroundColor = "#ffffff";
90 // 将画布放置到body里
91 document.body.appendChild(canvas);
92 // 得到画笔
93 var context = canvas.getContext("2d");
94
```

(2) 获取绘制上下文

```
84 // 获取一个画布对象
85 var canvas = document.getElementById("c");
```

```
92 // 得到画笔
93 var context = canvas.getContext("2d");
94
```

(3) 定义粒子类（水粒子、干粉粒子）

```
162 //定义水粒子类
163 function Particle1(x, y){
164     // 原坐标
165     this.x = x;
166     this.y = y;
167
168     // 初始出现的改变y的值
169     this.yVal = -6;
170     // 改变的x的值
171     this.xVal = Math.random() * 2 - 1;
172
173     // 定义一个下降的重力加速度
174     this.g = 0.15;
175     // 更新位置
176     this.updateData = function(){
177         // X值的变化
178         this.x = this.x + this.xVal;
179         // Y值的变化
180         this.y = this.y + this.yVal;
181         // 每次改变Y值速度的变化
182         this.yVal = this.yVal + this.g;
183         // 颜色设置
184         //context.fillStyle = "#828282";
185         context.fillStyle = "#02b9fa";
186         // 将更新位置后的圆绘制出来
187         this.draw();
188     };
189 }
```

```
190 // 绘图的方法
191 this.draw = function(){
192     // 开始画图
193     context.beginPath();
194     // 画圆
195     context.arc(this.x, this.y, 5, 0, Math.PI * 2, false);
196     // 结束画图
197     context.closePath();
198     // 填充
199     context.fill();
200 };
201 }
```

```

202 //定义干粉粒子类
203 function Particle2(x, y){
204     // 原坐标
205     this.x = x;
206     this.y = y;
207
208     // 初始出现的改变的y的值
209     this.yVal = -6;
210     // 改变的x的值
211     this.xVal = Math.random() * 2 - 1;
212
213     // 定义一个下降的重力加速度
214     this.g = 0.15;
215     // 更新位置
216     this.updateData = function(){
217         // X值的变化
218         this.x = this.x + this.xVal;
219         // Y值的变化
220         this.y = this.y + this.yVal;
221         // 每次改变Y值速度的变化
222         this.yVal = this.yVal + this.g;
223         // 颜色设置
224         context.fillStyle = "#828282";
225         //context.fillStyle = "#02b9fa";
226         // 将更新位置后的图绘制出来
227         this.draw();
228     };

```

```

230 // 绘图的方法
231 this.draw = function(){
232     // 开始画图
233     context.beginPath();
234     // 画圆
235     context.arc(this.x, this.y, 5, 0, Math.PI * 2, false);
236     // 结束画图
237     context.closePath();
238     // 填充
239     context.fill();
240 }
241 }

```

(4) 创建并显示水粒子与干粉粒子的方法

```

95 // 定义一个存放所有粒子的数组
96 var particles = [ ];

```

```

118 // 创建并显示 水粒子 的方法
119 function showParticle1(){
120
121     // 循环操作
122     setInterval(function(){
123         // 清空画布
124         context.clearRect(0,0,canvas.width, canvas.height);
125
126         // 创建粒子
127         var p = new Particle1(canvas.width * 1.18, canvas.height * 0.53);
128
129         // 将粒子装入存放粒子的数组
130         particles.push(p);
131
132         // 循环更新所有粒子的位置
133         for (var i = 0; i < particles.length; i++) {
134             // 更新位置
135             particles[i].updateData();
136         }
137     }, 70);
138
139 }

```

```

140 // 创建并显示 干粉粒子的方法
141 function showParticle2(){
142
143     // 循环操作
144     setInterval(function(){
145         // 清空画布
146         context.clearRect(0,0,canvas.width, canvas.height);
147
148         // 创建粒子
149         var p = new Particle2(canvas.width * 1.1, canvas.height * 0.53);
150
151         // 将粒子装入存放粒子的数组
152         particles.push(p);
153
154         // 循环更新所有粒子的位置
155         for (var i = 0;i<particles.length;i++) {
156             // 更新位置
157             particles[i].updateData();
158         }
159     }, 70);
160
161 }

```

(5) 通过按钮事件侦听实现喷洒不同粒子的效果

```

71 <canvas id="c" width="500" height="500"></canvas>
72 <button id="startAnimation">喷水</button>
73 <button id="startAnimation1">喷干粉</button>
74 <button id="stopAnimation">停止</button>
75

```

```

98 //通过按键设置 效果连接
99 var startbutton=document.getElementById("startAnimation");
100 var startbutton1=document.getElementById("startAnimation1");
101 var stopbutton=document.getElementById("stopAnimation");
102
103 //水start按钮控制
104 startbutton.onclick = function() {
105     // 调用显示粒子
106     showParticle1();
107 }
108 //干粉start按钮控制
109 startbutton1.onclick = function() {
110     // 调用显示粒子
111     showParticle2();
112 }
113 //stop按钮控制
114 stopbutton.onclick = function() {
115     location.reload();
116 }
117

```