CS 405 Computer Graphics Project 3 Report

Ezgi Ergüney - 31211

Task 1: Implementing the draw function

```
10  class SceneNode {
25
26      draw(mvp, modelView, normalMatrix, modelMatrix) {
27          /**
28           * @Task1 : Implement the draw function for the SceneNode class.
29           */
30          var TransformationMatrix = this.trs.getTransformationMatrix()
31
32          var transformedModel = MatrixMult(modelMatrix, TransformationMatrix);
33          var transformedMvp = MatrixMult(mvp, TransformationMatrix);
34          var transformedModelView = MatrixMult(modelView, TransformationMatrix);
35          var transformedNormals = MatrixMult(normalMatrix, TransformationMatrix, );
36
37
38          // Draw the MeshDrawer
39          if (this.meshDrawer) {
40              this.meshDrawer.draw(transformedMvp, transformedModelView, transformedNormals, transformedModel);
41          }
42
43          for (const child of this.children) {
44              child.draw(transformedMvp, transformedModelView, transformedNormals, transformedModel);
45          }
46      }
47
48
49
50  }
```

In Task 1, we needed to implement the draw function so that any transformations applied to a node would automatically affect its children. This was important to make the celestial bodies move together in a realistic way. To do this, I first obtained the node's transformation by calling this.trs.getTransformationMatrix() and I stored the returned value in a variable. I then used the MatrixMult() function to multiply this matrix with the modelMatrix, mvp, modelView, and normalMatrix. This step was to apply the transformations. These updated matrices were passed to the meshDrawer.draw function (this part was already provided).

To make sure the transformations applied to all child nodes, I made the draw function recursive. For each child, I called the draw function again with the updated matrices. This way, any transformations applied to a parent were passed down to its children, and additional transformations of the children were applied on top of the parent's. For example, the Earth inherited the Sun's transformations, and the Moon inherited transformations from both the Earth and the Sun. This allowed the solar system to behave correctly with all bodies moving as expected.

Task 2: Calculate Diffuse and Specular Lighting

```
148  const meshFS = `
162      vec3 lightPos = vec3(0.0, 0.0, 5.0); // Position of the light source
163      vec3 lightdir = normalize(lightPos - fragPos); // Normalize the light direction
164
165      float ambient = 0.35;
166      float diff = 0.0;
167      float spec = 0.0;
168      float phongExp = 8.0;
169
170      ////////////////////////////////////////////////////////////////////////
171      // PLEASE DO NOT CHANGE ANYTHING ABOVE !!!
172      // Calculate the diffuse and specular lighting below.
173
174      vec3 viewDir = normalize(-fragPos);
175      vec3 reflectDir = reflect(-lightdir, normal);
176
177      diff = max(dot(normal, lightdir), 0.0);
178      spec = pow(max(dot(reflectDir, viewDir), 0.0), 8.0);
179
180
181      // PLEASE DO NOT CHANGE ANYTHING BELOW !!!
182      ////////////////////////////////////////////////////////////////////////
183
```

The task involved updating the fragment shader to add diffuse and specular lighting, along with the existing ambient lighting. I calculated diffuse lighting using the angle between the surface normal and light direction, with the line diff = max(dot(normal, lightdir), 0.0); to determine how much light hits the surface. For specular lighting, I calculated the reflection direction and the view direction, then used the dot product between them to determine how much light reflects toward the camera. The line spec = pow(max(dot(reflectDir, viewDir), 0.0), 8.0); controls the sharpness of the highlight. Note 0.8 in here comes from the phongExp variables value.

Task 3: Adding Mars Planet

```
cs405-project3 > project3.html > doctype > html > head > script > onload
 1   ml>
 3
 5
171  ript type="text/javascript">
196   window.onload = function () {
233       * @Task3 : Add Mars to the solar system
234       * Mars should be a child of the sun.
235       * Mars should use sphere as the mesh object.
236       * Mars should be translated by -6 units on the X axis with respect to the sun
237       * Mars should be scaled to 0.35 for x,y and z coordinates
238       * use the image on the link below as texture:
239       * @link : https://i.imgur.com/Mwsa16j.jpeg
240       *
241       */
242
243      marsMeshDrawer.setMesh(sphereBuffers.positionBuffer, sphereBuffers.texCoordBuffer, sphereBuffers.normalBuffer);
244      setTextureImg(marsMeshDrawer, "https://i.imgur.com/Mwsa16j.jpeg");
245      marsTrs = new TRS();
246      marsTrs.setTranslation(-6, 0, 0);
247      marsTrs.setScale(0.35, 0.35, 0.35);
248      marsNode = new SceneNode(marsMeshDrawer, marsTrs, sunNode);
249
250      renderLoop();
251  };
```

In this task we are asked to add planet Mars to the solar system, to do that I examined how other celestial bodies implemented and applied what I observed. To be able to view the Mars in our output I first created a meshDrawer object for mars node. Then I set the texture of the mars using setTextureImg() function. Next, I created a new TRS object for mars and then I set its size and translation with respect to its parent. After all the necessary information about Mars I created the Scene node for it and assigned the sun as its parent.

```
171  ript type="text/javascript">
172   function renderLoop() {
173       UpdateCanvasSize();
174       var timeOffset = (Date.now() / 1000) % 9;
175       var zRotation = timeOffset * 40 * Math.PI / 180;
176       gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);
177       gl.viewport(0, 0, canvas.width, canvas.height);
178
179       var modelMatrix = getIdentityMatrix();
180       var modelViewMatrix = MatrixMult(camera.getLookAt(), modelMatrix);
181       var mvp = MatrixMult(perspective.getPerspectiveMatrix(), modelViewMatrix);
182       var normalMatrix = getNormalMatrix(modelMatrix);
183       sunNode.trs.setRotation(0, 0, zRotation);
184       earthNode.trs.setRotation(0, 0, zRotation * 2);
185       /**
186        *@task3 : add rotation to mars on z-axis.
187           the rotation should be 1.5 * zRotation
188       */
189       marsNode.trs.setRotation(0, 0, zRotation * 1.5);
190
191       sunNode.draw(mvp, modelViewMatrix, normalMatrix, modelMatrix);
192       requestAnimationFrame(renderLoop);
193   }
194
```

In here I also examined how other celestial bodies perform their notation and applied something according to my observation. I set the mars rotation on z-axis using

setRotation() on marsNode.trs. I believe this sets the rotational speed of the celesetial body.