CS 405 PROJECT 3

Sıla Özinan
28161

TASK 1

In task 1, I modify the draw function to properly see the objects in the solar system scene graph. First of all, I calculated the transformation matrix for the nodes. Then, I multiplied (matrix multiplication) this matrix with the matrices that are given as parameters. The draw function in the given if statement was for only the drawing parent node which is sunNode, in order to draw children nodes recursively I added a for loop.

```
draw(mvp, modelView, normalMatrix, modelMatrix) : void  {

    var nodeTransformationMatrix : []  = this.trs.getTransformationMatrix();
    var transformedMvp : []  = MatrixMult(mvp, nodeTransformationMatrix);
    var transformedModelView : []  = MatrixMult(modelView, nodeTransformationMatrix);
    var transformedNormals : []  = MatrixMult(normalMatrix, nodeTransformationMatrix);
    var transformedModel : []  = MatrixMult(modelMatrix, nodeTransformationMatrix);

    // Draw the MeshDrawer
    if (this.meshDrawer) {
        this.meshDrawer.draw(transformedMvp, transformedModelView, transformedNormals, transformedModel);
    }

    for (var i : number  = 0; i < this.children.length; i++) {
        this.children[i].draw(transformedMvp, transformedModelView, transformedNormals, transformedModel);
    }
}
```

TASK 2

In task 2, some modifications were made to meshDrawer.js to combine ambient, diffuse, and specular light. The given version already has the ambient light. For the diffuse and specular light, I made modifications. First I computed the value for diffuse lighting with the help of lightdir and normal that is provided. Then for the calculation of the specular lighting, view direction and reflection direction were needed and they were computed. With these values and provided phongExp specular ligt value is computed as well. Lastly, I equalized computed values to values that will be used for combining ambient, diffuse, and specular light.

```
// Calculate the diffuse and specular lighting below.

// Diffuse
float diffIntensity = max(dot(normal, lightdir), 0.0);

// Specular
vec3 viewDir = normalize(-fragPos); // assuming the camera is at the origin
vec3 reflectDir = reflect(-lightdir, normal);
float specIntensity = pow(max(dot(viewDir, reflectDir), 0.0), phongExp);

// Combine values
diff = diffIntensity;
spec = specIntensity;
```

TASK 3

In task 3, I added Mars to the designed Solar System as a child node of the Sun. I used the provided link for the texture of Mars. Additionally, it is scaled 0.35 for all x,y, and z coordinates and translated by -6 units on the x-axis. Its mesh object is the sphere and this methods are used as they applied to other objects such as the moon and Earth.

```
marsMeshDrawer = new MeshDrawer();
marsMeshDrawer.setMesh(sphereBuffers.positionBuffer, sphereBuffers.texCoordBuffer, sphereBuffers.normalBuffer);
setTextureImg(marsMeshDrawer, "https://i.imgur.com/Mwsa16j.jpeg");
marsTrs = new TRS();
marsTrs.setTranslation(-6, 0, 0);
marsTrs.setScale(0.35, 0.35, 0.35);
marsNode = new SceneNode(marsMeshDrawer, marsTrs, sunNode);
```

Also, in the renderLoop(), I added rotation to Mars on the z-axis with a degree of 1.5*zRotation(beforehand calculated value) to rotate around its z-axis 1.5 times the sun's rotation. Since marsNode is linked to sunNode we don't need to call the draw function for mars additionally, it will be drawn thanks to the draw of sunNode.

```
/**
 *@task3 : add rotation to mars on z-axis.
    the rotation should be 1.5 * zRotation
*/
marsNode.trs.setRotation(0, 0, 1.5 * zRotation);
```

Final result: