Ahmet Burak Kurtulmuş – 28996

# CS 405 Project 3 Report

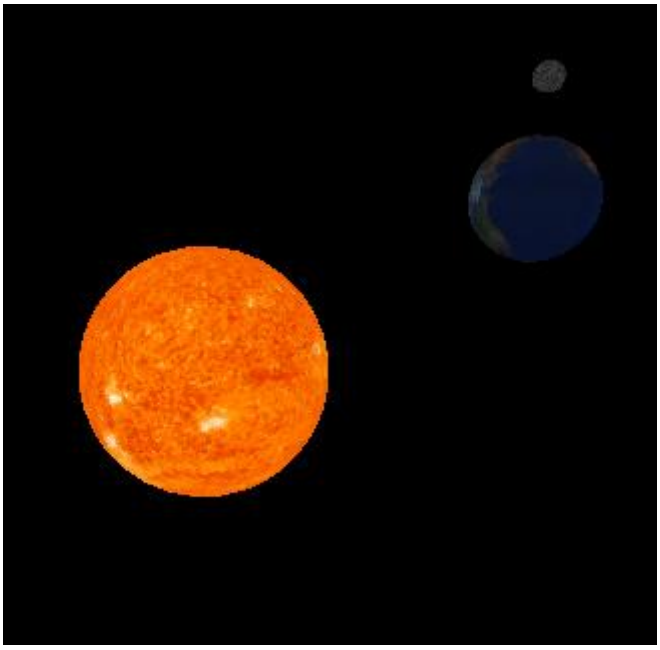
## Task 1:

In the first task, I began by invoking the getTransformationMatrix() method from the trs.js file to obtain the transformation matrix of the node. Since we are going to apply some transformation operations within the draw function, I multiplied the model matrix obtained here with this transformation matrix to obtain the transformedModel. This allowed me to obtain the position of the model in world coordinates along with transformedModel.

Then, using the modelViewMatrix for the model's position, I obtained the transformedModelView. After that, by using the getNormalMatrix method from the utils.js file, I obtained the normalMatrix, which is essentially the inverse of the transformation matrix.

Finally, to complete the transformedMvp, I multiplied the model's matrix to transform it into 2D coordinates on the screen.

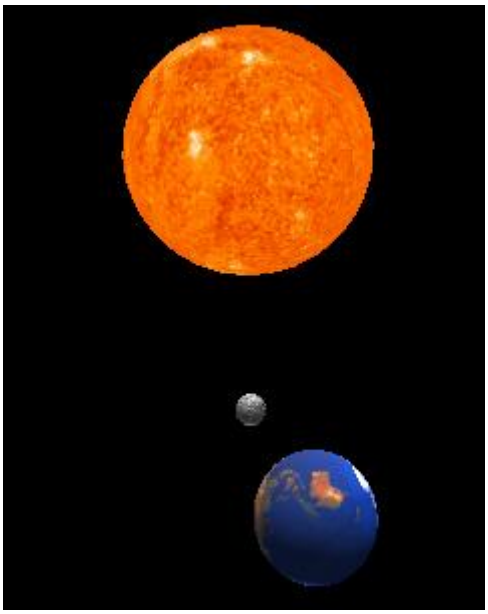
In the end, the following image was generated on the screen.



## Task 2:

In the second task, just as we did in the 6th-week recitation, we needed to make lighting adjustments with diffuse and specular lighting. I started with diffuse lighting. I controlled the diff, specified as float diff = 0.0, by examining the angle between the surface normal and the light direction for illumination using the max function. Then, I calculated specular lighting. Similar to what we did in the 6th-week

recitation, I first wrote viewDir. Then, using the reflect function to find the direction of the light reflected on the surface, I calculated reflectDir. After this process, I adjusted the reflection brightness with the pow function. Ultimately, the resulting image was as follows:



## Task 3:

In the third section, I integrated the lines as follows:

I created an object from the MeshDrawer class, called marsMeshDrawer, to draw the 3D model of Mars. Then, I set it with texture coordinates and normal information using setMesh. After that, I assigned the image from the given link as a texture. Next, I created an object that holds the position, rotation, and scaling matrices of Mars.

I shifted the position of Mars by -6 units along the X-axis with respect to the Sun. Then, I scaled it by 0.35 in the X, Y, and Z axes. Finally, I created the SceneNode object for Mars. In doing so, I positioned Mars as a child of the Sun.

Finally, within the renderLoop method, I needed to rotate Mars around the z-axis of the Sun 1.5 times. I achieved this with marsNode.trs.setRotation(0, 0, 1.5 * zRotation). Ultimately, the resulting image was as follows: