CS405 Project 3

Ceylin Ekinci

28369

Task-1

The 3D model linked to the current item and its offspring are drawn by this function. After transforming the model's vertices, normals, and other pertinent data through the required matrix multiplications, it assigns the meshDrawer object to do the actual drawing. The drawing is applied to the entire object hierarchy thanks to the recursion over children.

```
draw(mvp, modelView, normalMatrix, modelMatrix) {
      var transformedMvp = MatrixMult(mvp, this.trs.getTransformationMatrix());
      var transformedModelView = MatrixMult(modelView,
this.trs.getTransformationMatrix());
      var transformedNormals = getNormalMatrix(transformedModelView);
      var transformedModel = MatrixMult(modelMatrix, this.trs.getTransformationMatrix());

      if (this.meshDrawer) {
         this.meshDrawer.draw(transformedMvp, transformedModelView,
transformedNormals, transformedModel);
      }

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

```
      var transformedMvp = MatrixMult(mvp, this.trs.getTransformationMatrix());
      var transformedModelView = MatrixMult(modelView,
this.trs.getTransformationMatrix());
      var transformedNormals = getNormalMatrix(transformedModelView);
      var transformedModel = MatrixMult(modelMatrix, this.trs.getTransformationMatrix());
```

These matrix multiplication functions stores the transformation matrices in the defined variable.

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

This function executes the draw method recursively for each child.

Task-2

```
    float diffInt = max(dot(normal, lightdir), 0.0);
    diff = diffInt;

    vec3 viewDir = normalize(-vPosition);
    vec3 reflectDir = reflect(lightdir, normal);

    float specIntensity = pow(max(dot(viewDir, reflectDir), 0.0), phongExp);
    spec = specInt;
```

- Determines the diffuse intensity by taking the dot product of the normalized light direction and the normalized normalized normal vector.
- Uses the Phong reflection model to calculate the specular intensity by taking into account the Phong exponent, the view direction, and the reflection direction.

Task-3

```
sunNode.draw(mvp, modelViewMatrix, normalMatrix, modelMatrix);
requestAnimationFrame(renderLoop);
```

- Invokes the sunNode's draw function while providing a number of transformation matrices
- The renderLoop function is scheduled to be invoked on the following animation frame by this line. In web-based graphics programming, this is a standard pattern that synchronizes with the browser's rendering loop to ensure efficient and fluid rendering.

```
marsMeshDrawer.setMesh(sphereBuffers.positionBuffer, sphereBuffers.texCoordBuffer,
sphereBuffers.normalBuffer);(1)
setTextureImg(marsMeshDrawer, "https://i.imgur.com/Mwsa16j.jpeg");(2)
mars_trs = new TRS();(3)
mars_trs.setTranslation(-6, 0, 0);(4)
mars_trs.setScale(0.35, 0.35, 0.35);(5)
marsNode = new SceneNode(marsMeshDrawer, mars_trs, sunNode);(6)
renderLoop();(7)
```

(1): The mesh data for a marsMeshDrawer object is set by this line. It seems to be related to drawing a spherical object that provides texture coordinates, normal buffers, and position possibly representing Mars.

(2): The marsMeshDrawer object's texture image is set by this line. During rendering, the image referenced by the URL "https://i.imgur.com/Mwsa16j.jpeg" will be added to the Mars mesh's surface.

(3): A new TRS object is created by this line. Translation, rotation, and scaling, or TRS for short, refers to a collection of transformations that can be used on an object in three dimensions.

(4): With the help of this line, the Mars object will be moved 6 units to the left along the x-axis by the translation component of the marsTrs transformation.

(5): This line configures the marsTrs transformation's scale component.

(6): A new SceneNode object called marsNode is created by this code. The parent node is set to sunNode, the transformation (marsTrs), and the Mars mesh drawer (marsMeshDrawer) are associated. This points to a scene where the Mars object is a child of the sun and a hierarchical structure.

(7): The logic to update and redraw the scene repeatedly would probably be included in the renderLoop implementation itself.