# CS405 Project 3

## Task 1

### sceneNode.js:

draw(mvp, modelView, normalMatrix, modelMatrix) {

/\*\*

\* @Task1 : Implement the draw function for the SceneNode class.

\*/

var transformedMvp = mvp;

var transformedModelView = modelView;

var transformedNormals = normalMatrix;

var transformedModel = modelMatrix;

// Apply the transformation of the current node

transformedMvp = MatrixMult(transformedMvp, this.trs.getTransformationMatrix());

transformedModelView = MatrixMult(transformedModelView, this.trs.getTransformationMatrix());

transformedNormals = MatrixMult(transformedNormals, this.trs.getTransformationMatrix());

transformedModel = MatrixMult(transformedModel, this.trs.getTransformationMatrix());

for(var child of this.children){

child.draw(transformedMvp, transformedModelView, transformedNormals, transformedModel);

}

// Draw the MeshDrawer

if (this.meshDrawer) {

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

}

}

The task is to implement a draw function for the SceneNode class, which renders the node and its children in a 3D scene. The function applies given transformation matrices to the current node's transformations and recursively calls itself for each child node. Additionally, it draws the mesh using a MeshDrawer if available.

## Task 2

### meshDrawer.js:

/\*\*

\* @Task2 : Update the fragment shader for diffuse and specular lighting.

// Calculate the diffuse and specular lighting below.

vec3 rDirection = reflect(lightdir, normal);

vec3 vDirection = normalize(-vPosition);

spec = pow(max(dot(rDirection, vDirection), 0.0), phongExp);

diff = max(dot(normal, lightdir), 0.0);

vec3 lightning = ambient \* vec3(1.0,1.0,1.0) + diff \* vec3(1.0,1.0,1.0) + spec \* vec3(1.0,1.0,1.0);

The task is to enhance the fragment shader with Phong reflection for diffuse and specular lighting. The code computes lighting based on a given light source position, encompassing ambient, diffuse, and specular elements. The shader supports textures and distinguishes between light sources and other fragments.

## Task 3

### project3.html:

marsNode.trs.setRotation(0, 0, zRotation \* 1.5);

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);

renderLoop();

Task 3 added Mars to our 3D solar system. Utilizing a scaled sphere mesh, Mars is positioned as a child of the sun with a -6 unit X-axis translation, 0.35 scale, and Z-axis rotation at 1.5 times zRotation. The texture is sourced from a provided link, enhancing the realism of our dynamic solar system model.