**CS 405 Project Phase III Report**

**Introduction**

This project focuses on developing a solar system simulation using WebGL. The aim is to complete a series of tasks that illustrate the principles of scene graph structures, transformations, and lighting. Each task progressively enhances the simulation, resulting in an engaging and visually immersive experience.

**Task 1: Implement the Draw Function for the Scene Graph**

**Methodology:**

1. **Parent-Child Relationship Propagation:**

* Developed the draw function in the SceneNode class to effectively transfer transformations from parent nodes to their child nodes.
* Used local transformation matrices (trs.getTransformationMatrix) and combined them with the parent node's matrices for correct positioning and rotation.

1. **Matrix Multiplication:**

* Calculated transformedModel, transformedModelView, transformedNormals, and transformedMVP matrices by multiplying local transformations with parent transformations.

1. **Mesh Drawing:**

* Invoked the draw method of MeshDrawer with the transformed matrices.

1. **Recursive Drawing:**

* Ensured all children nodes were drawn recursively by iterating through the children array of the current node.

**Task 2: Update the Fragment Shader for Diffuse and Specular Lighting**

**Methodology:**

1. **Ambient Lighting:**

* Maintained the ambient lighting level at 0.35 to provide consistent baseline illumination.

1. **Diffuse Lighting Calculation:**

* Used the dot product between the normalized normal vector (vNormal) and the light direction (lightDir) to calculate the diffuse light intensity.

1. **Specular Lighting Calculation:**

* Calculated the reflection vector (reflectDir) and the view direction (viewDir).
* Used the Phong reflection model to compute specular highlights with a shininess exponent of 8.0.

1. **Light Source Condition:**

* Ensured the shader correctly handled light source nodes by assigning them full brightness while preserving their texture.

1. **Texture Mapping:**

* Combined the calculated ambient, diffuse, and specular components with the texture color to determine the final fragment color.

**Task 3: Add Mars to the Solar System**

**Methodology:**

1. **Creating Mars Node:**

* A MeshDrawer was initialized for Mars, which was assigned the sphere geometry along with the texture from the link (https://i.imgur.com/Mwsa16j.jpeg).

1. **Transformations:**

* Mars was translated by -6 units on the X-axis relative to the Sun.
* Its dimensions were scaled uniformly to 0.35 using the setScale method.

1. **Hierarchy Integration:**

* Mars was added to the Sun node as a child in the scene graph to maintain the hierarchical relationship.

1. **Rotation Logic:**

* The renderLoop function was modified to rotate Mars around its Z-axis at a speed 1.5 times that of the Sun's rotation.

1. **Texture Application:**

* The Mars texture was applied to the MeshDrawer to ensure it was rendered with the appropriate appearance.

**Conclusion**

Through these tasks, we successfully implemented a solar system simulation, showcasing advanced concepts such as hierarchical transformations, real-time lighting, and scene graph traversal. Each task reinforced critical WebGL techniques, culminating in a visually dynamic and interactive 3D environment.