**Project 6: Specular Lighting, Objects, Illumination and Shaders**

Samson Becenti & Spencer Meren

CST-310: Computer Graphics Lecture & Lab

Professor Ricardo Citro

March 3, 2024

**Project 6: Specular Lighting, Objects, Illumination and Shaders**

**Project Background**

Our OpenGL program aims to demonstrate basic lighting effects using shaders by utilizing OpenGL techniques to render 3D cubes with varying shininess and reaction to light. The program uses transformation matrices like translation and rotation to position eight 3D cubes to demonstrate different lighting effects. Matrix operations are also used for transformations and for passing data to the shader programs. These shader programs are compiled and linked using vertex and fragment shaders, each for different lighting effects and rendering tasks. To display the shininess value of each cube, we utilize FreeType for text rendering. It is a library used for handling fonts and rendering text onto the screen. The main rendering loop is where all the rendering occurs, each cube is created with its own lighting conditions and transformations. Overall, this project covers various aspects of OpenGL including shader programming, vertex buffer objects, and text rendering, to demonstrate 3D lighting effects.

**Shininess Values Explained**

To recreate the provided “lighting graphic” showing the visual impact of different shininess values, we chose to use the same shininess values for each cube. Each cube is rendered in with a predetermined shininess value which corresponds to that cube’s lighting conditions.

**Flowchart of Algorithm**

**A diagram of a process

Description automatically generated**

Figure 1 Flowchart depicting the logic of the approach used to generate the cubes with varying shininess.

**Explanation of Algorithm for Illumination and Shininess**

The program renders multiple cubes with different positions, rotations, and lighting conditions. The lighting is implemented using Phong shading, consisting of three components: ambient, diffuse, and specular lighting. The program initializes multiple shader classes (‘lightingShader1’ to ‘lightingShader8’) in order to render the cubes with different lighting conditions. Vertex data for the cubes are setup in a array which contains the coordinates and upward vector of each vertex. Then, vertex array objects and vertex buffer objects are used to store the vertex data and attribute configurations. Inside the rendering loop of the code, each cube is rendered separately with different lighting conditions such as object color, lighting color, light position, and shininess. Then the corresponding shininess values are rendered under each cube using FreeType text renderer.

**Results**

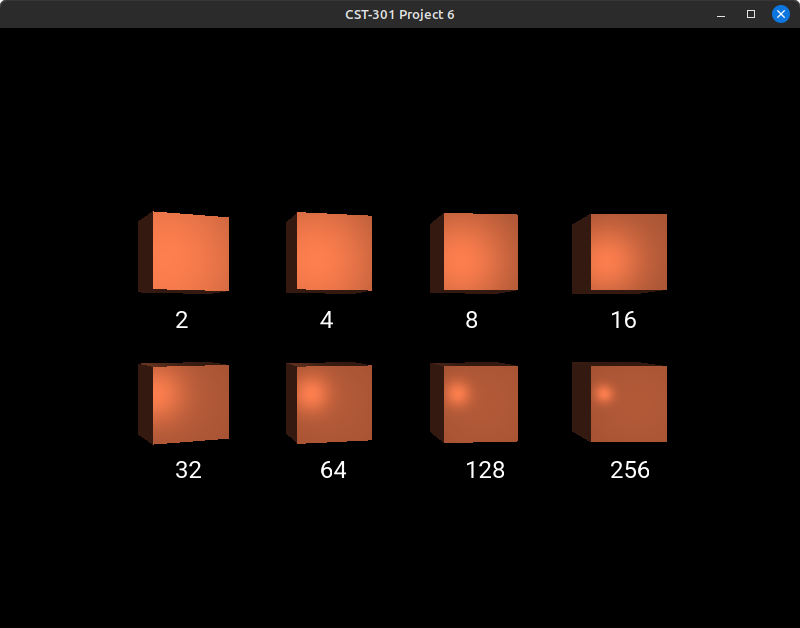


Figure 2 Program during execution.

**Github Link**

<https://github.com/SpencerM-2k1/CST-310-Lighting-Project>

**References**

“Specular Lighting- different shininess values” from the “Learn OpenGL” website <https://learnopengl.com/Lighting/Basic-Lighting>

Guha, S. (2019). *Computer graphics through OpenGL®: From theory to experiment*s (3rd ed.). Taylor & Francis, Inc. ISBN-13: 9781138612648