# Computer Graphics Coursework – Self Assessment Document

**Name:** *Dairo Toluwani Samuel* **ID number:** *23735761*

Complete the self-assessment grid below by writing a short explanation of how you have satisfied the requirement and how it has implemented in your code.

|  |  |  |
| --- | --- | --- |
| **Learning outcome** | **Mark** | **Weighted mark** |
| 1. Use appropriate mathematical tools (40%) |  | 0 |
| 2. Develop a 3D graphics application (30%) |  | 0 |
| 3. Write shader code (30%) |  | 0 |
|  | Total | 0 |

Your mark for each Learning Outcome (LO) is the highest mark achieved based on the criteria specified in the self-assessment grid. Note that you will need to have satisfied all criteria at the lower mark bands to be awarded marks in the higher mark bands, e.g., to get a mark in the 70 - 80 band for a learning outcome you will have needed to have satisfied all criteria in the 40 – 50 and 50 – 60 mark bands.

## Learning Outcomes:

**LO1** Select and use appropriate mathematical tools for constructing and manipulating geometry in 3D space.

**LO2** Develop an interactive 3D graphics application using an industry-standard API.

**LO3** Write shader code for the programmable pipeline on modern graphics hardware using an industry standard shader language.

## Self-assessment Grid

|  |  |  |
| --- | --- | --- |
| **Mark** | **Criterion** | **Comments (state how and where you have achieved the criterion)** |
| 42, 45, 48 | LO1: Basic use of vector and matrix objects | I used `glm:: vec3` and `glm:: mat4` to define the positions and transformations for the basketball and the scene.  Code in coursework.cpp initializes model matrices with `glm:: translate()` for placement. code:    Screenshot of product:  A screenshot of a computer  AI-generated content may be incorrect. |
| LO2: Application compiles and runs without alterations to the source code of CMake file. | Without any modifications, the project was constructed using the standard CMakeLists.txt file. The application runs with the OpenGL 3.3 Core Profile after initializing GLFW and GLEW.  Paste a screenshot of your application below  Screenshot:  A screenshot of a computer program  AI-generated content may be incorrect.  A computer screen shot of a program  AI-generated content may be incorrect.  A screenshot of a computer  AI-generated content may be incorrect. |
| LO3: Implementation of shaders to apply appropriate textures to objects. | The basketball's bump mapping and striping effects are procedurally added using a GLSL fragment shader. There is no external texture utilized.  Code:    A close up of numbers  AI-generated content may be incorrect.  Screenshot of product:  A screenshot of a computer  AI-generated content may be incorrect. |
| 52, 55, 58 | LO1: Basic use of translation, rotation and scaling transformations. | GLM transform functions are used to apply translation and rotation to the ball and hoop.  Bounce animation is impacted by rotation.  Code:      Screenshot of product:  A screenshot of a computer  AI-generated content may be incorrect. |
| LO1: Implementation of glm library functions for calculating view and projection matrices. | In Camera class, a custom `lookAt` and perspective matrix were implemented. replaces the built-in features of GLM.  Code:  A screen shot of a computer code  AI-generated content may be incorrect.  A black background with white text  AI-generated content may be incorrect.  A close up of a sign  AI-generated content may be incorrect.  Screenshot of product:  A screenshot of a computer  AI-generated content may be incorrect. |
| LO2: 3D virtual world has been created using instances of a single object type. | The backboard, floor, and basketball are all made of repetitive quad or sphere shapes.  Code:  A screenshot of a computer code  AI-generated content may be incorrect.  A black background with white text  AI-generated content may be incorrect.  A black background with white text  AI-generated content may be incorrect.  A computer code with text  AI-generated content may be incorrect.  A computer screen shot of a computer code  AI-generated content may be incorrect.  A black background with white text  AI-generated content may be incorrect.  Screenshot of product:  A screenshot of a computer  AI-generated content may be incorrect. |
| LO3: Use of shaders to apply dynamic lighting from point light sources | Phong lighting model implemented per-fragment. Point light and viewPos uniforms passed into shader. |
| 62, 65, 68 | LO1: Implementation of students own functions for calculating view and projection matrices. | Camera’s view and projection matrix computed manually from vector math and trigonometric calculations.  Code:    Screenshot of product:  A screenshot of a computer  AI-generated content may be incorrect. |
| LO2: 3D world created using multiple object types. | Basketball (sphere), hoop (box/ring), and floor (quad) are all created using various buffers and logic. |
| LO2: Users can navigate the virtual world using keyboard and mouse inputs. | `glfwGetKey` is used to process WASD keyboard input. For camera orientation, the mouse modifies the yaw and pitch angles.  Code:      Screenshot of product:  A screenshot of a computer  AI-generated content may be incorrect. |
| LO3: Use of shaders to apply dynamic lighting from different types of light sources. | Adding several types of light is supported via the shader framework. There is only one point light in use at the moment.  Screenshot of product:  A screenshot of a computer  AI-generated content may be incorrect. |
| 72 75, 78 | LO1: Implementation of students own functions to replace glm functions (e.g., glm::length(), glm::dot(), glm::cross() etc.). | Rotation matrices were created in Camera class by manually implementing dot and cross products. |
| LO1: Implementation of quaternions to calculate rotation matrix. |  |
| LO2: Interactive dynamic aspects of the virtual word and controllable by the user (e.g., position of objects, location and function of light sources etc.). | The ball's position and velocity are reset via the spacebar. During runtime, the interaction dynamically updates the object.  Code:  A computer screen shot of a program code  AI-generated content may be incorrect.  Screenshot of product:  A screenshot of a computer  AI-generated content may be incorrect. |
| LO3: Appropriate implementation of normal and specular maps. |  |
| 85, 90, 100 | LO1: Use of quaternions to calculate view matrix. |  |
| LO1: Use of SLERP to smooth out changes in camera direction. |  |
| LO2: Implementation of a third person camera with the ability to switch between first and third period view. |  |
| LO2: The position of the camera or character obeys the constraints of the physical space (e.g., can’t pass through objects, can’t hover in midair etc.). |  |
| LO3: Use of shaders to apply parameter driven effects within the scene, e.g., light properties controlled using camera/character position. |  |