GD1P04-3D Graphics Programming

Technical Demo 3 (40%)

Component code and name	GD1P04 - 3D Graphics Programming
Assignment name	Technical Demo 3
Weighting	40%
Submission deadline	Week 16
Week issued	Week 12

Brief

The objective of this assessment is to test students' understanding of the introductory OpenGL concepts and techniques and show them in a small technical demonstration. It covers the following learning outcomes:

- Utilize good programming practices
- Understand the uses of 3D graphics in game development
- Understand how 3D Math functions are used in OpenGL
- Initialize the OpenGL 3D pipeline
- Utilize OpenGL libraries
- Understand basic OpenGL rendering techniques
- Manage buffers and render primitives in OpenGL
- Create a camera system in OpenGL
- Understand and utilize the OpenGL lighting effects
- Apply textures using the OpenGL programmable function pipeline
- Use multi-texturing techniques with the OpenGL programmable function pipeline
- Utilize alpha blending techniques with the OpenGL programmable function pipeline
- Utilize, review and enhance software engineering design principles
- Utilize, review and enhance a software testing strategy

Instructions/Requirements:

Create a technical demo to showcase the following:

- **Lighting (45%)**
    - Create a scene using 10 or more 3D models. Each model needs to have correct normals, an appropriate texture, and use a Blinn-Phong lighting shader.
    - Add Lights to the scene:
        - 2 or more Point Lights with attenuation affecting the models with varying degrees.
            - Ensure the models are also different distances away from the point lights.
            - Represent each point light with a 3D object. Use a non-textured and non-lighting shader that colors each respective object to match the lights color.
        - A single Directional light.
        - A single spotlight using the cameras position and direction.
    - Use an additional class (e.g. LightManager) to manage the lighting properties and pass the lighting information into the shaders via uniforms, structures and arrays where appropriate.
    - Controls to allow each type of light to be toggled on/off:
        - Key press '1' to toggle the point lights on/off.
        - Key press '2' to toggle the directional light on/off.
        - Key press '3' to toggle the spotlight on/off.

- **Cubemaps and Reflection (25%)**
    - Create a Skybox class that can construct and render a cube map texture to the scene as a skybox with textures lining up together correctly and seamlessly.
        - Use perspective division to set the z-component to a value of 1.0f.
        - Set the depth function to 'less than or equal' for the skybox draw call.
    - Create a new 3D model and apply reflection:
        - Apply an appropriate texture and use a reflection map to control the reflection amount.
            - Part of the object must have a full/partial reflection.
            - Part of the object must have no reflection with only the object texture showing.
        - The reflection should update as the camera moves around the scene.
        - Apply Blinn-Phong lighting to the shader as well as reflection.

- **Free Camera + Control (20%)**
    - Create a free camera (perspective projection) with the following controls:
        - Forward/backward movement using the camera forward axis (W and S).
        - Strafing left/right movement using the camera right axis (A and D).
        - Up/Down movement using the global Y axis (Q and E).
        - Pitch and Yaw rotation using raw mouse motion input.
        - Zoom in/out by changing the cameras projection Field Of View (FOV) on the mouse scroll wheel (requires scroll callback function).
    - Extra Controls and capabilities
        - Allow a toggle for wireframe mode for the scene on a keypress.
        - Back-face culling is enabled.
        - Multi-Sampling Anti-Aliasing (MSAA) is enabled.

- **Programming Practices (10%)**
    - A ReadMe.txt file is included stating the functionality, controls, and any additional triggers or needed information for the project to showcase all included features.
    - Function headers are consistent and present across all files and functions.
    - Comments are used to clarify the purpose and use of data and functions demonstrating an understanding of the key areas of related code.
    - Classes and functions are appropriately used to create systems and demonstrate a higher understanding of modular code and proper C++ OOP concepts.
    - consistency of naming conventions, code formatting, and accessors to increase readability across all files.
    - No warnings are generated during the building that originate from student project files.
    - No Intermediate files are included.

All the above criteria are to:

- Use the appropriate OpenGL programmable pipeline with vertex and fragment shaders.
- Be visible in one scene.

Guidelines

Follow the guidelines, standards, and specifications regarding the tasks outlined in the instructions section.

Submission Guidelines

Place the work in a .zip file and submit it to Blackboard by the time and date specified (see Blackboard).

Naming conventions

The file structure and file names of the submission must follow the file hierarchy listed below.

- YYYY-MM-DD – GD1P04 – Assignment 3 – *Student Name*.zip
  - Source – *Student Name*
    - Assignment 3.sln
    - … Project and source code, etc.
  - Release Build – *Student Name*.zip.
    - … Include additional resources and DLLs

Submission structure

Source code folder:

- Solution file (.sln).
- Project file (.vcproj).
- Source files (.cpp, .h).

Release build zip:

- Standalone executable (.exe).
- Any additional files required to run the executable.
- Readme file (.txt).

Intermediate and repository files are removed to reduce file size.

Submission Policy

Submissions

The submission dates for the summative are provided with the assignment briefs along with the information required to pass that assignment. It is the student's responsibility to read the assignment brief carefully as the submissions need to comply with the naming convention supplied, artifacts required, and any other specific requirements outlined in the briefs.

It is the student's responsibility to submit their work on time and communicate with faculty regarding their submission.

Late Submissions

Late submission is bound with a 24-hour time frame from the stated submission time for this component and is **capped at 50%.** Students should make sure that they submit everything on time.

Resubmissions

A resubmission may be offered to a student who does not achieve a passing grade for this summative. If a resubmission is offered, it will be discussed on a case-by-case basis between the lecturer and the student. Resubmissions are **capped at 50%.**