Project Proposal

Mohammad Anas, Audrey DeHoog

Project Goals

The goal of this project is to create a simple 3D solar system model created in C++, OpenGL, and GLFW. Through this project, we will build on techniques learned in class including topics such as model loading, camera view, and model transformation. We will also learn the new technique of applying texture mapping through applying textures to models and through environment mapping (skyboxes) and the advanced technique of *deferred* lighting.

Timeline

Week 1 (11/20-11/26):

- Project proposal
- Create GitHub repository
- Implement basic set up: import libraries, get 3D models + textures, have window opening, etc.

Week 2 (11/27-12/3):

- Create scene with planet movement
- Texture mapping: planet textures and skybox
- Integrate deferred lighting
- Add switching camera views

Week 3 (12/4-12/6):

- Finish up any remaining tasks/debugging
- Write final report

Literature Survey

OpenGL – the graphics library

OpenGL is a streamlined, hardware-independent interface for rendering graphics [1]. OpenGL performs major graphics operations such as constructing shapes from geometric primitives, arranging the objects in 3D space, calculating the colors of all the objects, and rasterization [1]. OpenGL can also perform other operations, such as eliminating parts of objects that are hidden by other objects [1].

The websites https://www.opengl-tutorial.org/ and https://learnopengl.com/ are sources of tutorials for learning concepts related to OpenGL. The OpenGL website also provides a page with other tutorials for learning OpenGL https://www.opengl.org/sdk/docs/tutorials/ .

GLFW – OpenGL framework

GLFW is a library for OpenGL development that provides a simple API for creating windows, contexts and surfaces, and receiving inputs and events [2].

GLM – OpenGL mathematics library

GLM is a C++ mathematics library for graphics software based on the OpenGL Shading Language (GLSL) specification [3]. GLM provides extended capabilities from GLSL, including matrix transformations, quaternions, half-based types, random numbers, etc. [3]. We will be using GLM to transform the planets to simulate orbits.

Deferred Shading

Deferred shading [4], a rendering technique, that decouples the shading process from geometry rasterization, diverging from the traditional forward rendering pipeline where lighting calculations occur per-fragment during rasterization [5]. This approach divides the rendering into two key stages: the geometry stage, which collects and stores information about the scene's geometry in a G-buffer, and the lighting stage, where efficient lighting calculations are performed using the stored data. This separation enhances the efficiency of handling lighting in scenes with numerous light sources and intricate geometry, making deferred shading a powerful tool for achieving realistic graphics. In our project with deferred shading, we render celestial bodies' geometry first, then efficiently compute lighting effects in a subsequent pass using precomputed information, ensuring a visually immersive representation

Group Member Responsibilities

Audrey:

1. Implement basic setup: Import libraries, acquire 3D models and textures, and establish the window.
2. Develop the scene with planetary movement.
3. Work on texture mapping for planet surfaces and skybox.
4. Implement switching camera views functionality – camera that can move freely around the model.
5. Assist in finishing any remaining tasks and debugging.
6. Contribute to the final report by documenting the implemented features and providing insights.

Mohammad:

1. Set up the GitHub repository, manage version control, and work with Audrey on essential setup tasks, such as creating a makefile and organizing vital project structures
2. Implement Integrate deferred lighting into the project.
3. Implement switching camera views functionality – camera that follows a planet.
4. Utilize GLM for transforming planets to simulate orbits.
5. Assist in finishing any remaining tasks and debugging.
6. Contribute to the final report by summarizing the project's progress and findings

References

[1] Shreiner, D. (2009). *OpenGL programming guide: the official guide to learning OpenGL, versions 3.0 and 3.1*. Pearson Education.

[2] GLFW. https://www.glfw.org/

[3] OpenGl Mathematics (GLM). https://www.opengl.org/sdk/libs/GLM/

[4] Deferred Shading. https://learnopengl.com/Advanced-Lighting/Deferred-Shading

[5] Forward Rendering vs. Deferred Rendering