KV6003: Individual Project
Project Terms Of Reference

Cuthfbert Mutumbwa
Design and Implement a Rendering Engine

14th November 2018

1 Background

Creating A rendering engine is a task that requires one to understand multiple different topics and subjects, for example not only will I need to know how lighting works, but how it interacts with other objects, and how to represent it on screen. I will also need to learn how textures work, how to wrap them around an object and how the light will interact with the texture wrapped around the object. I will also learn how a modern GPU, by writing code for it, as well as understand what role the CPU will have when drawing the 3D images.

In college, I used two pieces of software that are called Unreal Engine and Blender. Unreal Engine was a game engine, and Blender was a 3D Creation suite that can be used as a rendering engine. My first question when using this software where "How does one Create 3D graphics software", then I realized I also was unsure of how someone designs this software.
I want to create a rendering engine that uses core rendering techniques and features, and Software engineering practices to design the engine. for the design aspect, I have opted to use different types of literature such as Designing a Modern Rendering engine [Bauchinger] to help me. I will also literature to choose which type of technique I will use to implement my features, this is because there are multiple different ways to implement features such as shadows, lighting, and even the way the engine will render a scene.

Rendering engines are still or more specifically rendering is a very interesting field of study. New rendering features are being made to improve quality in 3D animated films, and recently Nvidia released a new Graphics Card to allow Real-Time raytracing in Video Games[1]. Such an Achievement is one that is truly impressive as raytracing is known for being computationally expensive.

Creating a rendering engine is not only about knowing how to write good and efficient code, and knowing how lights, shadows, and texturing works are not enough. I will need to understand the underlining mathematical principles of computer graphics such as vectors, matrices and more. I will need to be able to

[1] `https://www.pcgamer.com/uk/what-is-ray-tracing/`

read mathematical notation, as well as choose which formulas I will use to achieve the results I want. I want to improve my mathematical skills, as well as know, learn what types of maths is needed to achieve this I have picked out literature and books such as Game Physics Cookbook [Szauer, 2017]. This book teaches the reader the maths needed to make the rendering engine.

2 Proposed Work

For the design phase I will use UML diagrams to help me conceptualize the rendering engine, and use them as a reference when creating the product. I will also incorporate a software Development life cycle to allow me to justify my choices, and allow me to see if I am on track to finishing the product. I will create a 3D scene environment that will showcase the features I plan on implementing. The user will be allowed to freely move around the 3D scene environment. The user will see different showcases on features within this scene, for example, there may be a location where the user is able to an object reflection from another object, and a location that demonstrates shadows by casting a light above the object.

Since there are many types of ways to implement one feature, in my design phase I have to consider what the best way to implement a specific feature will be. I need to make sure that it fits my criteria, and that I am able to implement the feature, while also making sure it is in scope. The analysis will be an essential part of not only the design phase but the implementation phase as well. To Aid me I will use literature such Learning OpenGL [2] I will also create small a prototype of the project while in the design phase, this will further help me define the scope of the project, as well as aid me in my design decisions.

I will be writing the rendering engine using C++, and the reason I chose this language is that I am the most familiar with it. I will be using the OpenGL API, this will allow me to send commands to different kinds of graphis devices[Bauchinger]. I also chose OpenGL because it is widely used and has many books and references such as OpenGL Programming Guide: The Official Guide to Learning OpenGLShreiner and Group [2009]. For debugging I have already chosen to use Nvidia Nsight for visual studio [3]. This debugging tool is compatible with my editor of choice (visual studio) and is also recommended by The Khronos Group [4] who are the contributor to the OpenGL API [5]

[2] https://learnopengl.com/book/learnopengl_book.pdf
[3] https://developer.nvidia.com/nsight-visual-studio-edition
[4] https://www.khronos.org/OpenGL/wiki/Debugging_Tools
[5] https://en.wikipedia.org/wiki/Khronos_Group

3 Aims and Objectives

Aims

- To Build a Rendering Engine
- To design a maintainable piece of software using modern design principles

Objectives

1. Create 3 UML diagrams.
2. Implement lighting.
3. Implement Shadows.
4. Implement Shaders.
5. Implement a Skybox.
6. Implement Reflections.
7. Implement Texturing.
8. Implement Model Loading.
9. Test and debug the system using tools specific for graphics programming.

The 3 UML Diagrams I will be creating will be Class Diagram, Sequence Diagrams, and State Machine Diagram. The 8 Features I will implement will be showcased within the scene environment. For testing and debugging I will generate a test plan.

4 Skills

1. Programming in C/C++ (from module KF4006)
2. Software Engineering (from module KF5012)
3. Algorithms (from module KF5008)
4. OpenGL API (Learnt enough to build a 3D scene)
5. Computer Graphics Theory (Currently Learning in Module KF6018)

I am confident I have a good foundation for each of these skills, however, I have many resources such as books that will not only aid me but further improve my knowledge on the subject.

5 Resources

Hardware

- Computer

Software

- Visual Studio
- Nvidia Nsight

Visual Studio will be the IDE that will allow me to write the rendering engine code. Nvidia Nsight is a graphics debugger and will not only help me debugging the code but to also help me with the test plan.

6 Structure and Contents of Project Report

Planned Chapters

1. Designing the Renderer (Analysis)-This Chapter will explain what techniques and tools did I use to design the renderer.
2. Implementing the Rendering Engine (Synthesis)-This Chapter will explain how I actually created the renderer. It will allow me to justify my choices, as to why I chose to implement certain features the way I did. It will also allow me to show my understanding of the topic.
3. Evaluating the Rendering Engine (Evaluation)-I will furthermore justify my choices as well as evaluate on my work. I will talk about what I believe I did right, and what I would have done differently.

7 Marking Scheme

Project Type Software Engineering projects

Project Report Chapters

Analysis

- Designing the Renderer
- Analysing Rendering Techniques

Synthesis

- Implementing the Rendering Engine
- Creating the Scene environment
- Testing and Debugging using bespoke Software

Evaluation

- Evaluation and Conclusion of the Rendering Engine

Product Deliverables list

1. Source Code
2. Design Documents
3. UML
4. Test Plans

The design document includes anything related to design such as UML diagrams, requirement analysis etc.

Fitness of purpose and Build quality

Fitness of Purpose

1. Quality of Functionality
2. Meeting of Requirements as identified during project

Build Quality

1. Code Quality
2. Software is responsive
3. Test plan and results
4. Design documents were created correctly

The code quality must have multiple classes, and each class corresponding with the class diagram. Software must not stutter when the user attempts to move the camera view around the scene. Design documents must not use wrong notation. Test plans must be thorough and show which problems managed to be fixed.

Precentage Breakdown
Fitness for Purpose: 40%
Build Quality 50%

8 List of Appendices

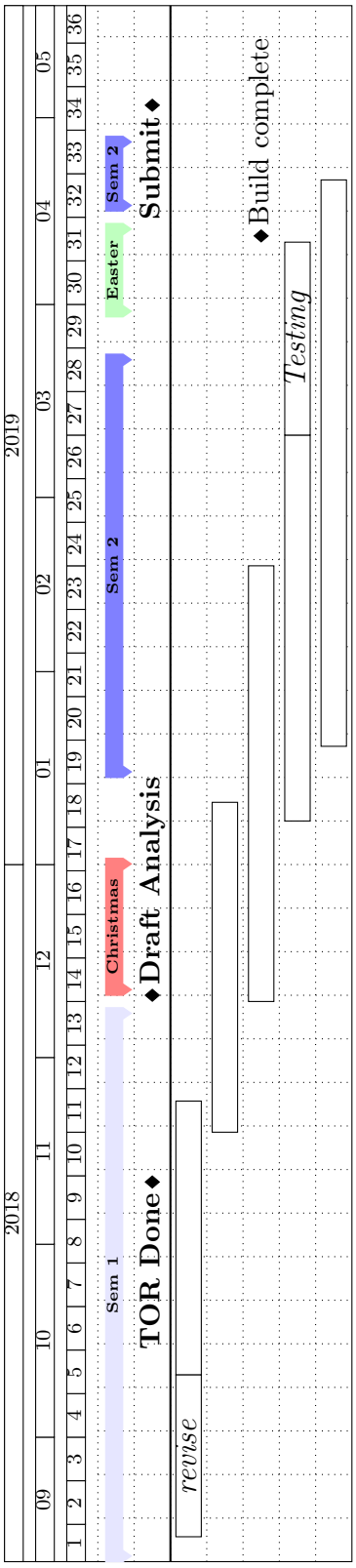
1. Finalised UML Diagrams
2. Test Plan
3. screenshots of the Rendering Engine

Having screenshots of the rendering engine will allow me to show and further explain certain aspects of my engine. Showing my finalized UML diagrams will allow me to show how helpful they were in the creation of the software.

9 Project Type

This will be a Software engineering project
Report: 40%
Product: 50%
viva: 10%
total 100

10 Project Plan



I have made sure no to have more than two task overlap with each other, as any more may result confusion when working.

11 References

Matthias Bauchinger. ausgeführt am institut für computergraphik und algorithmen der technischen universität wien.

Dave Shreiner and Bill The Khronos OpenGL ARB Working Group. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Versions 3.0 and 3.1 (7th Edition)*. Addison-Wesley Professional, 2009. ISBN 0321552628. URL `https://www.amazon.com/OpenGL-Programming-Guide-Official-Learning/dp/0321552628?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=0321552628`.

G. Szauer. *Game Physics Cookbook*. Packt Publishing, 2017. ISBN 9781787120815. URL `https://books.google.co.uk/books?id=frkrDwAAQBAJ`.