# Creating a 3D Graphics Engine

Student: Billy Strange (1803524)

Supervisor: Dr. Michael Kampouridis

Second Assessor: Dr. Katerina Bourazeri

Degree Course: BSc Computer Science

# Acknowledgments

I would like to thank my supervisor Dr. Michael Kampouridis for the great meetings that we have had every Friday that have helped me with the creation of this project with his ideas and criticism. I would also like to thank my second advisor Dr. Katerina Bourazeri for helping me put my project on the right course through the oral interview. Lastly, I would like to thank the School of Computer Science and Electronic Engineering and Dr. Vishwanathan Mohan for providing advice and information during this module which helped me in the creation of my project.

# Abstract

Many of today's games and applications are created using a 3D graphics engine. Popular graphics engines such as unity, unreal engine, etc have created very popular and decorated games such as fall guys on the unity engine and Fortnite on the unreal engine. This project aims to create a game engine that will allow the user to create a simple application or game.

The user can create different types of objects, with a first-person camera being able to view the world the user can create. Also included will be many options such as changing the background colour, player collision, and more, which can be read on my poster.

All the coding the user will do will be within one class, with the user being able to create their game or application from there. Most of the operations that happen in the engine are hidden from the user. There is a wiki on how to use the graphics engine to ensure the user will be comfortable in using the engine.

# Contents

# Literature Survey

## Basic Concepts of a 3D Graphics Engine

To start any planning or coding of this project, I needed to research the basic concepts of creating a 3D graphics engine. This led me to read the first couple of chapters of two books. One book that gives a broad overview of computer graphics [1], whilst also going very in-depth, and another book that looks at the essential mathematics needed to create a game [2].

The book titled Computer Graphics: Principles and Practice [1] gave a great insight into how and what I should know when it comes to creating a 3D graphics engine. In chapter 1 it is great at explaining the core concepts of creating a 3D graphics engine, explaining the history of the subject area and how it is grown over the years since the first 3D games engine was created. This chapter also into the different kinds of graphics applications and packages available, which I found to be useful. The next couple of chapters went into detail in using 2D graphics which is a great stepping stone towards learning about 3D graphics.

The other book I read titled Essential mathematics for games and interactive applications [2] was great in explaining the core mathematics that will be required in creating a 3D graphics engine. In this book, I learned the basic maths needed to create 3D objects, like how points and vectors work, and how they will fit in on a three-dimensional space. Later, with this, I also learned how to correctly do linear and affine transformations.

## A Simple 3D Graphics Engine Example – Wolfenstein 3D

My research then took me to look and playing simple, classic examples of a 3D game created using a 3D graphics engine. This is when I researched and played Wolfenstein 3D [3][4]. Released in 1992, this game accelerated the concept of the first-person shooter (fps) to a new level and set a standard genre design model for a great amount of time.

The engine used to create Wolfenstein 3D was created by programmer John Carmack. John experimented with making a new 3D game engine that would be fast-paced compared to others at the time. This was achieved by restricting gameplay and viewpoint to a single plane.

The gameplay itself is simple. The game is broken up into levels, each of which is a flat plane divided into areas and rooms by a pattern of walls and doors, with all of them being equal in height. To finish a level, the player must traverse through the area to reach an elevator. While traversing the levels, the player must fight Nazi guards and soldiers, dogs, and other enemies whilst managing supplies of ammunition and health. The player can find weapons and ammo placed in the levels or can collect them from dead enemies. While the levels are presented in a 3D perspective, the enemies and objects are instead 2D sprites presented from several set viewing angles, a technique sometimes referred to as 2.5D graphics.

## What Programming Language and Graphics API to Use

For the final part of my research, I wanted to find out what language would best suit the project and what language I will be most comfortable with. My research led me to choose C#. I chose this language as although C++ would allow me to have greater control over

parameters, memory management, etc, C# is a lot simpler in terms of programming. I have also never coded in C# and would like to try it out on this project.

My research also led me to two different sites giving tutorials on how to start up a 3D graphics engine. The first site [5] gives an overview of some of the basic things I would need to know when it comes to creating a 3D graphics engine. The site gives tutorials on things like; rasterizing Line segments, circles, triangles, and quads, dynamic lighting, learning about spaces and culling, etc. I feel like this would be a useful site to visit for the basics of what I'm going to be programming.

The other website also has the same features [6]. It gives a tutorial on how to do basic functions needed in a 3D game engine. This one, however, teaches you how to use the OpenGL graphics API which is very useful. This website gives a detailed tutorial on how to use OpenGL, which is what I have chosen to be my graphic API.

# Project Planning

## The Jira Tool

Jira was very helpful when creating my project. It is very easy to list tasks and easy to keep track of what needs to be done next. It is a lot better than just writing some notes in a word document. Being able to move tasks into development phases was great because I have had a lot of work to do with my other modules. If I was unsure of what I was working on I can just look on Jira to see where I was and start off again from there.

## Jira In Action in this Project

For this project, Jira was very useful as the project methodology I used for my project was the agile methodology. My project's goal was clear; create a 3D graphics engine. However, the solution was not as clear. Therefore, I based my project on the agile methodology which allows me to complete short bursts of work where I can take some ideas, design, and then develop them. Jira was helpful in this process as it was easy to add a small group of tasks develop them and complete them, showing the progress along the way. It is also helpful that my supervisor was able to place feedback into my task area as I can see what needs improving quickly.

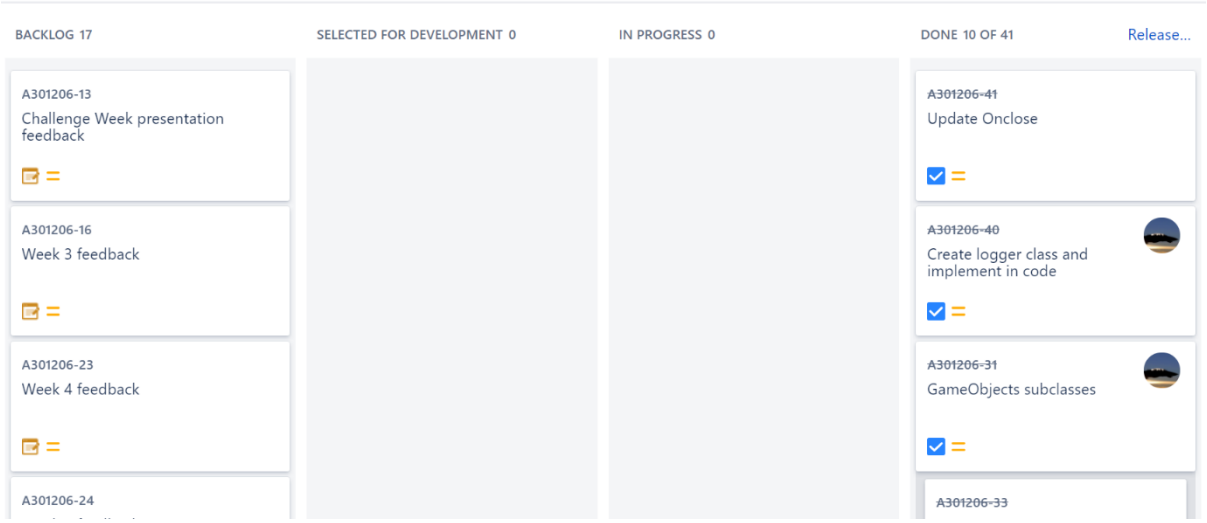


*Figure 1 An example of my Kanban board*

## Reflection on Project Planning

Using Jira helped me see where I was with my project at any time. I was able to monitor my progress easily and add ideas when I think of one. Overall, I think my project planning was good. There was the issue of trying to balance time between my other modules and this one but in the end, I got the hang of it and was able to have a steady work schedule.

## The use of GitLab

GitLab has been useful as a backup for me and my project work. When I have completed something major within my project, I updated my GitLab to accommodate that. I have also

been able to create a wiki for my project using GitLab which is helpful as the formatting is great.

# Technical Documentation

The complete technical documentation for this project is held on a University of Essex GitLab repository. Please follow this link to access the technical documentation. It is organized through a TOC readme file on the front page of the repository.

# Conclusions

In my project this year I have learned many things that that has improved my programming, maths, and overall project management skills. I liked working on my own project that I can improve in my own time and add to it when I get a new idea, or someone suggests an idea. I was able to research this idea and learn how to implement it, expanding the functionality of the Graphics Engine. I have also learned a lot about the maths side of graphics engines. I was able to use and be comfortable with working with 3D points and vectors, collision detection, which was successfully implemented, the use of quaternions, etc.

If I was able to have more time to work on this project, I would have liked to demonstrate my graphics engine by making a simple playable game. This would have enforced my point that a user can create a simple 3D game. I would also have liked to have a sphere object available to the user as well as there are currently only rectangular and triangular shapes available. This would have given the user another object to use. I was working on creating a sphere object using a few methods but was unable to create one by the time the project cycle finished. The closest I got was having an octahedron object which is on the right path to becoming a sphere however is not close.

I feel like my main goal was completed of being able to produce a simple yet functional 3D graphics engine that allowed the user to create a simple game or 3D model. I have implemented several objects for the user to be able to place in a 3D environment. Most background coding is hidden so the user only has to write a couple of lines of code to place said objects and can control various aspects of the engine such as placement and the first-person camera, toggle lighting, change background colour, etc. However, my most important achievement was having collision detection on most of the objects in the system. There is definitely more that can be added to this project, but I am happy with it and the progress I have made on it.

# References


[1] John F. Hughes, *Computer Graphics Principles and Practice*, 3rd ed. Willard, Ohio: RR Donnelley

[2] James M. Van Verth and Lars M. Bishop, *Essential Mathematics for Games and Interactive Applications*, USA: Morgan Kaufmann Publishers

[3] Ian Dransfield *The history of Wolfenstein* | https://www.pcgamer.com/the-history-of-wolfenstein/

[4] David Houghton The King of FPS - how Wolfenstein 3D changed video games forever | https://www.gamesradar.com/uk/8-things-wolfenstein-3d-gave-world/

[5] Kyle Sloka-Frey *Envatotuts+* | https://gamedevelopment.tutsplus.com/series/lets-build-a-3d-graphics-software-engine--gamedev-12718

[6] *NeHe Productions* | https://nehe.gamedev.net/