

**Games Programming 3 Coursework 2019/2020: Report**

**Submitted for the Degree of: BSc Computer Games (Software Development)**

**Name: Jakub Szulwinski**

**Programme: BSc Computer Games (Software Development)**

**Matriculation Number: S1627131**

**GitHub link: https://github.com/JSzulwinski/GP3-Submission-2019**

***“****I confirm that the code contained in this file (other than that provided or authorised) is all my own work and has not been submitted elsewhere in fulfilment of this or any other award.****”***

**Signed by Student: Jakub Szulwinski Date: 30.09.2020**

Table of contents

[Overview 3](#_Toc49821213)

[Headers 3](#_Toc49821214)

[Object.h 3](#_Toc49821215)

[Light.h 3](#_Toc49821216)

[Cube.h 3](#_Toc49821217)

[Axis.h 3](#_Toc49821218)

[Vector3f.h and Vector4f.h 3](#_Toc49821219)

[Classes 4](#_Toc49821220)

[Main.cpp 4](#_Toc49821221)

[Object.cpp 4](#_Toc49821222)

[Light.cpp 4](#_Toc49821223)

[Cube.cpp 4](#_Toc49821224)

[Axis.cpp 4](#_Toc49821225)

[Additional framework 5](#_Toc49821226)

# Overview

This project presents foundations of a game engine. It is created using C++, OpenGL and Glut framework. It has Entity-Component System where all objects in the game world are represented by entities and their behaviours are represented by discrete components, textured 3D models making use of lighting model, collision detection and a demo scene that has a moveable perspective camera, models that take input from the player. Not all functionalities are present in the demo scene however, some are scripted to be easily modified implemented by a user.

# Headers

#### Scene.h

This header is mainly use to declare mouse and keyboard functions as well as camera, lights, vectors and GLU elements. Elements for creating the window such as windows width and height and its position are also declared here.

## Object.h

This header is used exclusively for handling the cubes and objects in the scene. It declares all their dimensions relative to the 3D vector, texture, collision box used for collision detection and draw functions.

## Light.h

Deaks with declaring function used for light handling. Position, direction, color and further properties such as focus and width of the cone.

## Cube.h

This declares the cube. This function is later used in other scripts.

## Axis.h

This header declares the axis function and length of the axis.

## Vector3f.h and Vector4f.h

Used for declarations of three dimensional and four dimensional vector properties – constants and operators used for further manipulations and interactions in the scene.

# Classes

## Main.cpp

Main class that defines the scene and all its dependencies, objects, lights, camera, textures and collision detection.

Establishes the controls on keyboard and mouse (GLUT functionality is used for that):

* Mouse moves the camera around
* space - coordinate system axes
* q/w – rotating the cube
* f - full screen
* ESC – exit the application
* arrows – moving cube in axes X/Y
* page up/down – moving cube in axis Z

Function main defines the Scene, sets the other functions, lights, applies textures, shows vectors, instantiates cubes.

#### Scene.cpp

Scene class is responsible for handling the demo level. First it establishes the window using its dimensions and glut functions. Quadric drawing configuration is set up to generate objects. Light are enabled and scene is initialized. Next couple of functions establishes the controls, sets frame rate and full screen settings. Rendering process uses projection matrices for perspective. Collision detection works in a way, that when objects collides with another, the collision box is shows to indicate that the objects have collided. Function checkCollision calculates whether the objects have collided.

## Object.cpp

The main function of this class is to enable handling and creating many new objects to add to the scene. Cubes can be replaced by spheres or other shapes for example.

## Light.cpp

This class sets all the aspects of light in the scene such as ambient, diffuse, specular. It can later be used in future projects to add more lights to the scene.

## Cube.cpp

This sets the size, color of the cube and draws it.

## Axis.cpp

Showing the axes in the scene was implemented to give more depth and provide some orientation. It may be useful for later projects to have this point of reference.

# Additional framework

The project uses OpenGL Utility Toolkit alternative – freeglut. In this engine it is mainly used to perform input/output operations (keyboard and mouse). It has also been utilised to enable drawing geometric shapes like cubes and spheres.

It has been chosen for this project because of the solutions it provides for two very important parts of the engine – input/output and drawing shapes.