CG

CGP600 – AE2 Individual project

Ethan Bruins – Q12192287

Solent University

Computer Games Software Development

# Introduction

In this report I will go into detail about the project I created in C++ with DirectX 11 for advanced games programming. The original design for this project can be found in the appendix under [Appendix A](#_Software_Design_Document).

# Implementation

In this section I will go into detail about the parts that I implemented in this project.

## Camera

For the camera, I had designed the game to use a third person perspective. The way I did this was to determine a target for the camera to look at. I then used this target to determine the cameras position and rotation. I also used the target to calculate the forward, right and up vectors relative to the camera. This would allow the player to use the cameras forward direction to move the player forward in relative space.

An additionally camera I made was a top down perspective one. This will be used with the mini map UI that I had previously talked about in my software design in [Appendix A.](#_Additional_Functionality)

## Collision

For my collision implementation I didn’t want to just have spherical colliders as this would result in a very clunky game if I had custom models. Instead I’ve gone for a joint Axis Aligned Boundary Box and Spherical collision. This allows me to have some freedom as to what type of colliding each object has. For implementation I took my knowledge of AABB collision and combined it with Sphere collision detection to allow for both collision types to collide with each other.

# Software Design Document – Appendix A

## Game Design – Ethan and Paulo

This document will help clarify the mechanics that will be implemented in the second assignment for this module. Below are two lists with the project requirements and the additional functionality we will use. After that will be a brief of the game

### Project requirements

* 3D Game
* Player should be able to move around
* Environment should have static and moving obstacles
* Some obstacles should be moveable
* Simple Textures
* Simple Lighting
* Collisions
* NPCs represented by models
* NPCs collide with objects and perform action (change direction)
* Interaction between player and NPC

### Additional Functionality

* Player can jump
* Enemies have simple pathfinding
* Mini-map UI
* Advanced shader for dissolving the player when they die
* 3rd person camera

### Game Brief

We have chosen to create a 3rd person puzzle game in DirectX 11 with C++. The game will consist of a minimum of 5 levels with each level having increasing difficulty. The game will have two modes, timed and unlimited time. In the timed mode users will have a certain amount of time to complete each level, whereas the unlimited time mode will have no time limit. Both modes will have a leader board showing the fastest players in each.

The user will be able to move around the level using basic movement as well as jumping, this will be done using the ‘WASD’ and ‘Space Bar’ keys. The user will be able to rotate the camera around the player to see around the level, this will be done with mouse input.

When the user completes a level, there will be a loading screen to transition them into the next level. There will also be some particle effects to pronounce the fact they have completed the level.

### Game Inspirations

We had some influences when thinking of the design for the game. The first influence we had was Portal. Portal is a great example of a puzzle game as it gives players a sense of accomplishment when they create a room. We will not be using the same mechanics as portal but it’s a great inspiration for our level design.



*Figure 1 Portal 2, image source -* [*https://www.rockpapershotgun.com/2007/10/10/rps-verdict-portal/*](https://www.rockpapershotgun.com/2007/10/10/rps-verdict-portal/)

Another influence that we talked about while designing was the game Q.U.B.E 2. Q.U.B.E 2 has good art and level design which could be incorporated into our design. They also have a great way of using lighting to direct the player in the correct direction for the puzzle.



*Figure 2 QUBE 2, Image source -* [*https://www.gamasutra.com/view/news/316731/How\_the\_QUBE\_2\_devs\_built\_a\_better\_massive\_3D\_puzzle\_labyrinth.php*](https://www.gamasutra.com/view/news/316731/How_the_QUBE_2_devs_built_a_better_massive_3D_puzzle_labyrinth.php)

# References