

Computer Games Development

SRS (Software Requirements Specs)

Year IV

**Mantas Zalnierius**

**C00241626**

**28/10/2021**

# Declaration

# This examination/assessment will be submitted using Turnitin as the online submission tool. By submitting my examination/assessment to Turnitin, I am declaring that this examination/assessment is my own work. I understand that I may be required to orally defend any of my answers, to the lecturer, at a given time after the examination/assessment has been completed, as outlined in the student regulations.

**Contents**

[project abstraction 2](#_Toc54714373)

[project introduction 3-4](#_Toc54714374)

background [5-6](#_Toc54714375)

project description [5-6](#_Toc54714375)

philosophy [5-6](#_Toc54714375)

common questions [5-6](#_Toc54714375)

who is going to use this application [5-6](#_Toc54714375)

metrics [5-6](#_Toc54714375)

is there a precedent for this application [5-6](#_Toc54714375)

project milestones [5-6](#_Toc54714375)

project review & conclusions [5-6](#_Toc54714375)

references [5-6](#_Toc54714375)

# Project Abstraction

This project intends to create a standalone 2D platformer engine utilizing fast collisions and physics. So that if indie developers wanted to create a game, without the worry of hand-crafting their own engine from scratch. This engine will utilize fast collision responses for convex shapes via algorithms such as SAT/Diagonals and the use of ADSR envelopes for smooth movement.

# Project Introduction

In 2D platformer games, one of the biggest things that players want is immersive/responsive controls. Trying to craft a movement controller to achieve this is very time consuming, as you first need to understand some basic Newtonian physics and have the movement controller be mutable, i.e., let the developers change the max height, speed to see what looks and works best for their game.

Another thing that is very important is the responsive behaviour objects have when they collide off each other. This collision detection can be very hard, as you might believe that a certain algorithm works, but only if certain conditions are meet. An example of this would be AABB collisions, which do not consider the rotation of the colliding objects.

So, via my 2D platformer engine, it will remove the time to create these collision handlers and movement controllers, so that the developers have more time to spend on other parts of their game.

# Background

For background research, I investigated different types of collision detection, and why you would use one over the other. I looked at different papers/articles/videos and from doing this research I came across 2 algorithms for collision detection, SAT, and Diagonals. The reason I choose to implement two algorithms is because I wanted to showcase the differences between the algorithms. The differences I would consider is how fast one algorithm is over the other. Through my research I have learnt that SAT, is a good algorithm to use for collision detection for convex shapes, but one of the problems I saw, was that the collision response wasn’t very clean. By this I mean, the actual collision response had a jitter effect. But using the diagonal algorithm, this algorithm removes that jitter effect, but could lead to problems of how fast the algorithms is.

Another thing I researched was into different physics concepts I could use for a movement controller for my 2D engine. One concept I learnt was called the ADSR Envelope, this ADSR envelope is concept that utilizes different stages called (Attack, Decay, Sustain, Release) and using these stages using Newtonian physics, it will lead to a movement controller that is responsive to the user.

# Project Description

# Philosophy

**Philosophy Point #1**

This application is trying let the developers have more time in developing their game without the worry of implementing a collision library or physic library for their game. There is a multitude of different library’s that handle this collision/physics. My engine is just an alternative to what is already out there.

**Philosophy Point #2**

My goal for this project, is to have to as easy as possible for developers to create a movement controller that fits their game, this can be done by changing the values for max speed, acceleration etc. So, that it will lead for more time on getting a grasp of what type of movement they want.

# Common Questions

**Common Question #1** What is this project?

This project is a 2D platformer engine incorporating features such as Collisions and Physics. This engine can be used to create games such as Mario or Megaman and through this engine it will add extra time in developing other features for the game developers want to create.

**Common Question #2** Why did I choose this project?

The reason why I choose this project, was because after years of creating different types of platformer games, I found it quite hard to find an engine that was easy to understand and use for 2d platformer games. By doing this project, I hope of creating an engine that somebody would use in their own games.

**Common Question #3** What is the main functionality of your project?

The main functionality of this project is the utilization of collision detection and a movement controller to make it easier to create prototypes of games developers want to develop. Without the worry of creating an engine from scratch.

# Who Is Going To Use This Application?

The people that would use my application, will be developers that want to create a 2D platformer game with a worry of creating an entire engine for their game. These developers would be developers that are indie developers that don’t have the money to buy licences for other engines or don’t want to use entire engines like Unity/Unreal to create their game.

# Metrics

# Is There A Precedent For This Application?

There are a multitude of examples of different engines all around the world. What makes mine different than others are the ease of use my engine will have. When I am finished this engine, all that the developers will need to do, is just put the headers that are needed into their game. Whereas a lot of other engines have a very complicated process on how to set up their engines.

# Project Milestones.

# Project Review & Conclusions.

# References