School of Science and Engineering
Discipline of Computing
University of Dundee

AC40001 Honours Project

BSc (Hons) Computing Science

3D Arcade Android Game

WenJun Yu

Supervisor
Dr Iain Martin

May 5, 2020

3D Android Game

WenJun Yu
AC40001 Honours Project
BSc (Hons) Computing Science
University of Dundee, 2020
Supervisor: Dr Iain Martin

Abstract **-** With the development of portable devices(smartphones), mobile games are increasingly popular as entertainment. Smartphones are powerful enough to run games and allow the user to play at anywhere and anytime, especially with the 3-dimensional games due to its rich perspective and special effects.

The purpose of this project was to create a 3D Android game. The game is an endless run game and creates using the OpenGL ES and Java in Android Studio. User testing was carried out to help improve the output.

1 Introduction

The intent of this project was to design and develop a 3-dimensional game that is playable on an Android mobile device using OpenGL ES. OpenGL ES is a software used to render 2D and 3D graphics on embedded and mobile systems. It is a subset of OpenGL and has less functionality compared to OpenGL. [1]

To achieve the main objective, there were many steps that need to be obtained in order to success. With no experience in creating 3D Android game, there are some research that need to be done. There are many platforms exist for developing an android game. Unity is one of the most commonly used engines amongst developers. It had a very powerful visual editor to make the development much easier. Unity also provides an excellent implementation of the entity or component model that solve programming problem much simpler, and one most significant advantage is that it is easy to learn, which means it is most suitable for a beginner.[2]

Another platform is Android Studio. This is the official integrated development environment for Android application development. It has an interface called IntelliJ iDEA which provides an intelligent code editor that makes code more accurate.[3] It has also been decided that the programming language would be Java with the OpenGL ES. Many practice tutorials have been carried out to help familiar with the platform and language. Then is the game design and game development.

The implementation stage was the main part of this project, used to complete the initial requirements based on the design. Once the implementation is finished, user testing was used to gather feedback and make improvements with the time available.

The purpose of this document is to outline and describe the development process of the project.

2 Background

In order to successfully carried out all the development, background research was required to help understand and learn all the technique. It helps make the decision on what technique will be used and how will it be used to help to make an excellent 3D Android game.

2.1 Android Mobile Devices

The number of people nowadays using a smartphone are increasing every year. Android is one of the most popular mobile operating systems developed by Google.[4] Hence that many mobile games are designed to run on an android operating system.

2.2 Platform

To create a game for the android device, it will require to use game engines or Development platforms. There are many of them exist. [5] Unity is one of the most commonly used game engines. It is a cross-platform tool support systems such as IOS, Android, Windows Phone etc. Unity is easy to learn and use, had a very powerful Visual editor to make the development much easier. Unity can be used to create a 2D and 3D game with the development features it provided. One feature is it allows you to easily import any assets from 3D applications such as Blender and Maya.[2]

Another game engine is Unreal Engine. This platform is also very good for beginner to use as it contained lots of user-friendly features. Such as blueprints. Blueprints is the visual scripting system inside the Unreal Engine, it is good for prototyping and not require any programming

knowledge to assemble and adjust basic things. Instead of having write code line by line, blueprints allow developer to do everything visually such as: drag and drop nodes, set their properties in a UI and using wires to connect them together. The programming language it used is C++, often called Unreal C++. [6]

One of the tools for Android app development is Android Studio. It uses Java programming language.[7] Android Studio is not only used for Android app development but can also be used to create an Android Game.

2.3 Android Studio

Android Studio is an IDE (Integrated Development Environment) that designed specifically for Android.[3] It allows user to compile and edit code with Java programming language. It has an interface called IntelliJ iDEA which provides an intelligent code editor that makes code more accurate and faster with its advanced code completion refactoring and code analysis.[8] Android Studio also has Android SDK integrated to it.

SDK is a set of tools for Android development tools, and it is required in order to create any app for Android operating system. It is the most popular tool for app development process due to its available components (Library, code samples and lots other tools).

Android Studio allows the developer to set up the configuration for different virtual device to run on computer such as the API version, resolution by using the Android Emulator. This give the developer the ability to test their program on different virtual devices.

2.4 Java

Android Studio supports many different languages, such as Java, C++, and Kotlin. Java was chosen as the programming language used for this project as it is the official language of Android development. Java is Object Oriented.[9] This means it is based on the concept of class and objects, which makes the project easier and fast to execute. Another reason why Java was chosen is that Java is an easy-to-use language as it contains automatic garbage collection which automatically takes care of the memory management so the developer can concentrate on the logic and program.[10]

2.5 OpenGL

To program a 3D game will require to draw 3D objects this can be done by using the 3D graphics programming Language. OpenGL is the 3D graphics learned from the module 'AC41001-Graphics'.

OpenGL is an API (application programming interface) that allow computer programs to make use of 2D/3D hardware rendering features. OpenGL uses the geometric primitives and shaders to draw the object.[11] OpenGL application pass the data (vertex attributes and uniform data) to the vertex shader using the VBOs and then the vertex shader calculates the projected position of the vertex in screen space and sets their attributes. This then been pass to the primitive assembly to assemble triangles which then can be pass to the Rasterization step to output pixel fragments for each triangle and ready to be process in the fragment shader and output to the framebuffer. Framebuffer then can get drawn to the viewport. OpenGL has a subset called OpenGL ES, which is mainly designed to be used for the embedded system.

2.6 OpenGL ES

OpenGL ES is a subset of OpenGL which means it is also designed to be used to render 2D/3D graphics.[12] It is different from the OpenGL in terms of the number of the API functions it provides.[13] This is because any APIs and features that's not needed will be stripped out to make it suitable for a mobile platform.

On the coding level, the OpenGL ES does not support any fixed-function and fixed-pipeline stuff such as glBegin/glEnd etc. whereas OpenGL can support this using a compatibility profile. These elements of OpenGL were covered in AC41001 - Graphics module. Although OpenGL ES doesn't support those, they can use vertex array and vertex buffer object instead. Vertex array objects is an object that stores all of the state needed to supply vertex data and vertex buffer object is used as a source for vertex array data. [13][14] Also, OpenGL ES only supports rasterization primitives such as points, line and triangles, but quads are not supported. [17]

In order to pass geometric data from program to GPU the OpenGL use the function glVertexPointer and glNormalPointer. glVertexPointer and glNormalPointer can be used to define the array of vertex and normal data, but on OpenGL ES those function is not supported instead it used glVertexAttribPointer to define an array of generic vertex attribute data including vertex and normal.[17] *Figure 1 – glVertexPointer and glNormalPointer Figure 2 – VertexAttribPointer*

```
void glVertexPointer(   GLint size,
        GLenum type,
        GLsizei stride,
        const void * pointer);
void glNormalPointer(   GLenum type,
        GLsizei stride,
        const void * pointer);
```

Figure 1 – glVertexPointer and glNormalPointer[29][30]

```
void glVertexAttribPointer( GLuint index,
                            GLint size,
                            GLenum type,
                            GLboolean normalized,
                            GLsizei stride,
                            const void * pointer);
```

Figure 2 – VertexAttribPointer[28]

Another example of the difference is that OpenGL has Utility called Glut which can be used for window control, drawing several geometric primitives (cubes, spheres etc.).[15] However, in OpenGL ES the Glut is not available, but there has a similar interface for the window manager known as EGL.[16] It is a lot like the Glut, but it has no APIs for drawing spheres whereas Glut does.

In conclusion, although there are has some features only exist in OpenGL, but OpenGL ES can provide an alternative solution to perform the same as OpenGL.

2.7 OpenGL ES 2.0

There are serval OpenGL ES versions exits.[18] OpenGL ES 2.0 was chosen for this project. This is the second version of the OpenGL ES version. When compare to OpenGL ES 1.0/1.1 API, OpenGL ES 2.0 provide faster graphic performance and a higher degree of control as it uses the shaders to provide a fully programmable pipeline in order to create effects whereas OpenGL ES 1.0/1.1 API it can be very difficult. OpenGL ES2.0 and OpenGL ES1. 0 both using the vertex and fragment shader, but OpenGL 2.0 give the ability to do more using the shaders such as handle extensive geometry. The reason why OpenGL ES 2.0 is chosen over 3.0 is because there is a lot of support and resource for help online whereas OpenGL ES 3.0 is the new version which lack of tutorial help the learning.

2.8 Why 3D Game

When making a game, there are decisions that need to make between 2D and 3D. In many ways they are identical, but 3D games offer a rich perspective, shadow and special effects that 2D wouldn't have.[3] and nowadays mobile phones tend to have a more powerful system that can handles 3d games smoothly and bring user more luxury gaming experience, hence that this game is a 3D game.

2.9 Game Type

This game is an Endless Runner Action Game.[19] An endless runner game is a non-stop running game where the player character cannot stop unless it makes impact with an enemy or obstacle. A popular example of this type of mobile game is "Subway Surfers".[30] The player can swipe left or right or up on the screen to control the character. This type of control is very easy for player to get started.

The game develop for this project was inspired by the Game "Boonk Gang".[20] Whereas the player will control character try to prevent been caught by enemies. *Figure 3 - Boonk Gang*

Figure 3 - Boonk Gang

2.10 Target Audience

Target audience is a group of people who are most likely to be interested in the product. Hence that define the target audience is very important before the implementation.[21]

This game is target at anyone who has an android mobile device as the game is developed for the android operating system so in order to play the game they must have an android device to install the game. There is no age and gender limit. The game has simple rules and game controls. Which makes it easier for the player whether they have game experience or not as it only takes a little effort and time to get familiar with the game.

2.11 Ethical

Ethical consideration was considered and researched as this game's target audience was wide. It is important that the game does not have any violence or stereotyping contents., so the game was not ethically affecting people who play it. [32]

2.12 Blender

Blender is a free to use 3D modelling program which can be used to create 3D models. It also allowed to import and export with different types of 3D model files.[22] There are a lot 3D modelling program exist, but Blender is chosen for this project because of the familiarity with blender as it was used in the module 'AC41001-Graphics'.

This is very useful for this project as the game is 3D, so it will require 3D models. There are two ways to create the 3D model for the game, one is using the OpenGL ES to code the 3D object from scratch, but this will be difficult when creating a complicate 3D object. Thus, why blender is used, it gives the ability to create models or import and export any model find online to the suitable file format that's ready to load to the game through OpenGL ES.

3 Specification

The initial specification was defined after the first meeting with the project supervisor. Once the background research stage of the project was conducted. The programming language and other required components were decided and understood. The original specification was constantly revised and updated throughout the project lifecycle. Show on *Figure 4 – A sample of the requirements* [See Appendix B.2 for Requirements Specification]

1. Player open the game
   The game shall have main menu with play, instruction choice displayed for the player
   The game shall allow player to select between them
   The game shall have background music.

Figure 4 – A sample of the requirements

3.1 Product Backlog

The agile methodology was adopted for this project to plan out the development and implementation stage of the game and specification, following the adopted agile methodology practices. A product backlog was created that contains all the features that need to be created within a project, in the form of user stories which can be found in the *Appendix B.1 Product Backlog*. Each requirement was given an estimated complexity and priority in order to determine which tasks to complete first. A Gantt chart was created to allocating the time for each section. Please see below in *Figure 5- Cropped Project Gantt Chart*

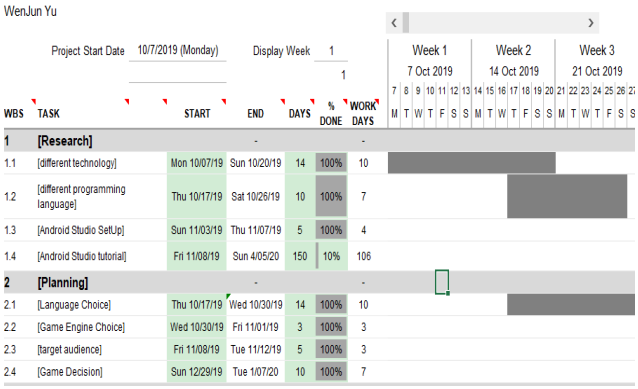

Figure 5- Cropped Project Gantt Chart

3.2 Game Specification

Figure 6 -Game Logo

The initial specification of this project was to let player control a character "pig" in the 3d world and try to not get caught by enemy character "farmer". The player's character will always move forward at a constant speed. The player only can control the character with left and right turn to change the direction. The player will lose the game if the enemy gets close and caught player. Player will earn a point every 10 seconds if alive. If the player is losing the game will be ended and end game screen will show up with the score. Player can either decided to play again or return to the main menu.

This game is an endless running game, so there is no win condition apply to the game so the basis of this game is to keep player alive as longer as you can. Hence that every time player loses the game, the game over screen displayed the score player achieved and previous highest score and it only updates if the player gets a new high score.

3.2.1 Level

The game has two levels. The first level's difficulty is set to easy. Game will start with six enemies, and the speed of

the enemy will set to be the same. The second level will have eight enemies, and the speed will be increased to make the difficulty harder.

3.2.2 User Interface

The interface for this game is very simple and understandable. The score will be recorded at the top of the game screen. There was no button or anything else other than the game scenes. For the control, the play just needed to tap at the left side of the screen which will make the character turn left and if the player touches the right-hand side of the screen, the character will turn right. Hence there was no difficult control at all the user should be able to start playing the game and familiar the controls almost instantly.

The interface for the main menu is also simple. The main menu consists of game logo and two buttons (Play and instruction). The play button allows the play to select the level and instruction button takes the user to the screen with all the instructors, which help the user understand the rules and control of the game.

In terms of the main menu, the game was required to have an end game menu. This menu will show the score player have along with the previous highest score achieve by player. It also gives player to options to select: a play again button allows the game to load again with the same level player played before and a home button which was used to load the main menu so the player can make option again.

3.2.3 Enemy AI

In this game the only condition to determine if the game ends or not is when the enemy caught the player. So, the enemy must move toward the player. To do this, the enemies will have a basic AI where it will use the player's location to calculate the rotate angle so that the enemy can change direction and move toward the player.

3.2.4 Object Loading

It was decided that the objects would be loaded in to represent the player and enemies. The player is represented with a standing pig and the enemies is represented with a human. The model is found online and using the blender to modify the object file.

3.2.5 Skybox

A skybox was implemented to the game. This can be used to representing the 3d scene that can be seen in every direction by the camera. In this game is used to represent the background. This can be done by using cube or sphere. In this project it was decided to use the cube from the prototyping stage. The cube is rendered with detailed texture on each inside faces of the cube. It is similar to the skybox that author made for "AA41001-Graphic" 's coursework. The camera is kept inside the skybox and the cube is move with the camera, so the view is always inside the cube.

3.2.6 Sound

The sound effect is an essential part of a game. It can provide the atmosphere to captivate the player. For this game the background music is implemented to enrich the user experience.

3.2.7 Collision Detection

For the enemy to kill the player, the collision detection was implemented. This was implemented on the enemy and the player. It was also used to check if the player or enemy was at the edge of the scene or not.

3.2.8 Development Environment

The game had to meet a set of other requirements. Such as the game had to run on Android. It was decided to use the Android Studio with Java and OpenGL ES together to develop the game. This decision was made after the background research.

4 Project Management

4.1 Time Management

To successfully develop the project, it will require a plan for the various deadlines of the development. Good time management will help keep up to date of the progress for the project. A Gantt chart was therefore created and used help giving a clear idea of when each deadline was and ensured that the project would run smoothly.

4.2 Gantt

A Gantt chart [See *Appendix B.3 Gantt chart*] was created during the mid-progress report. [See *Appendix C: Mid Progress Report*] The Gantt chart gives brief of how much time each section of the project would take, this includes the background research, design, implementation and evaluation and lastly the final report. Figure 7 – Gantt chart

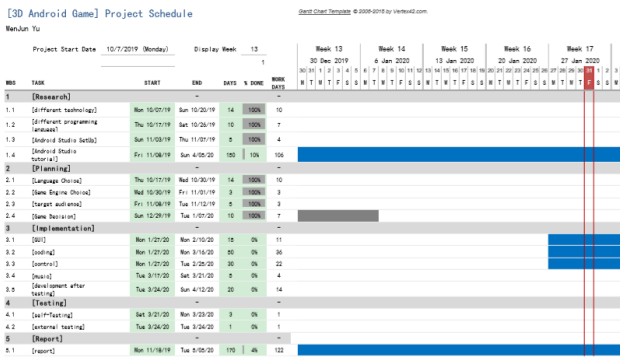

Figure 7 – Gantt chart

4.3 Version Control

Version Control is very important for the project. It helps track development stages. The GitHub was used for the project as the changes of the development can be easily tracked, and all the changes will have the time points associated with it. This is very useful as it maintains as an online backup if some mistake been made then it can help recover to the version you want.

4.4 Agile Methodology

An agile methodology was adopted for this project's implementation and development stage. It was chosen because the project was required to have regular weekly meetings throughout the project and evaluation stages. This methodology can help the project to be managed in an adaptable way so the progress can adjust to the change.

4.4.1 Regular Meetings

Regular weekly meeting with a supervisor was held throughout the project. So, the supervisor can monitor and advise the project progress. Author elaborate on what has been progressed and what problems were encountered. Then the supervisor can give feedback and advise on what to improve on and how to solve the problem. [*Appendix D: Meetings*]

4.5 User Testing

User Testing was used to help evaluate the game. The user testing plan will involve one session with around 10 participants after the implantation, so author can gain feedback on which the game could be improved upon.

In order to allow the user testing to take place, the author has completed the ethic form [*Appendix A: A1 Ethics*] and gain permission from the university.

User testing involved the participants read the information sheet which contains information about what they were taking part in and how they were being assessed in the user testing and agreed if they want to take part or not. Once they agreed, they were asked to complete the consent form. Then they were asked to play the game and then answer questions on the question sheet.

The question was simple and easy to understand. Example of some of the questions are below:

1. How did you feel about the gameplay?
2. How do you think about the difficulty of the game control? (does the control too confused)
3. What feature could be useful to add in the game?
4. Is there anything that you think could be improved?

Once the interview with each participant was finished, the feedback was analysed and group to one document so it can be used in the second implementation. From the feedback, author can then decide which feedback can give the project value with the time available.

This approach was chosen because it works well as participants might highlight parts that author never thought of during the design and implementation phases. It also gives a different perspective of how users of different experience level play the game as the game is aim for all type of audience. Overall, user testing is useful as it helps author identify issues and improve the quality of final products.

5 Design

5.1 Block Diagram

Block diagram was used to show the overall design of the game and what technologies and resources was used. Shown on *Figure 8 – Block diagram*. Android Activity was used for the UI, and OpenGL ES2.0 was used for rendering graphics. [Appendix B: B.9 BlockDiagram]

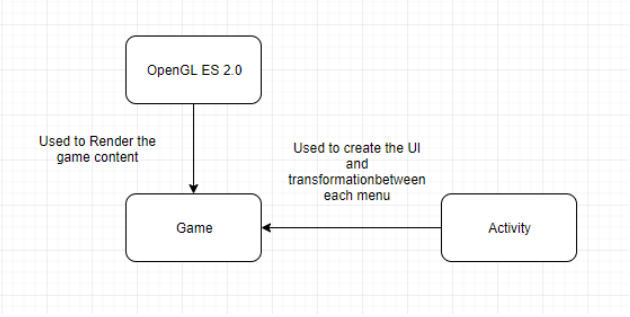

Figure 8 – Block Diagram

5.2 Activity Diagram

Several activity diagrams were created to show the workflows of the game. One of them was shown in *Figure 9 – Game Activity Diagram.* For all the activity diagrams can be found at [*Appendix B.4 MenuAT]* and *[Appendix B.5 GameAT]*

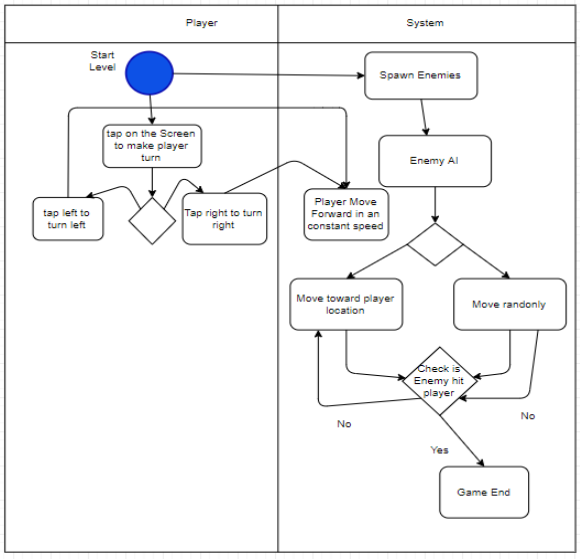

Figure 9 – Game Activity Diagram

In the diagram, it can be seen that when the game starts the system and player were involved. From the start point to the endpoint the player needs to decide the move direction of the character they controlled. The system part will control the enemy AI and player speed. It also checks if the game reaches the endpoint condition or not.

5.3 Class Diagram

A Class Diagram was created at the early stage of the project to map out some of the structure of the software. Shown in *Figure 10- Class Diagram.* [Appendix B: B.8 ClassDiagram]

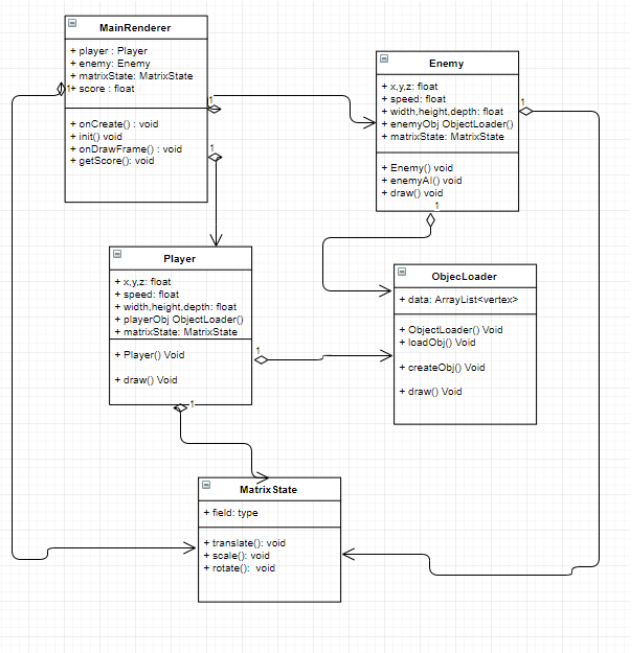

Figure 10- Class Diagram

For the design as shown in this class diagram, the MainRenderer and MatrixState class has many relationships to other class. MainRenderer is a central class where it controls all other parts to be drawn on the game screen whereas the MatrixState is a class where have all the transformation method ready to be used by other class.

The ObjectLoader class was design for loading object to the game and it is called in the Player and Enemy class to load each individual model associate with them.

5.4 UI Sketch

The UI sketches were considered and used to give a brief idea of how the menu screens and actual game would look like. *Figure 11 – Menu Sketch* shows the sketch for menu, and the game sketch can be found in [Appendix B: B.7 GameSketch]

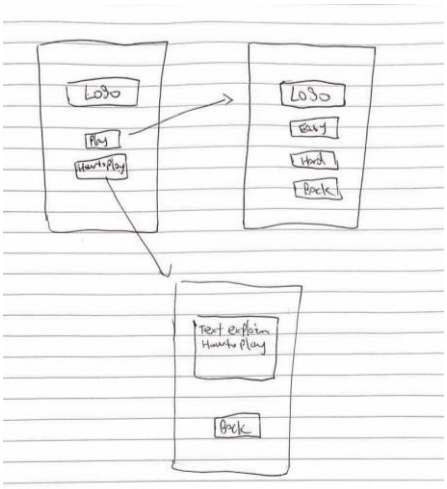

Figure 11- Menu Sketch

Those UI sketches then can be used in the developing state as a guide.

5.5 Prototyping

Several prototypes were created to learn about Android Studio and OpenGL ES. This will help author gain knowledge and experience about those techniques and prepare for the development.

5.5.1 First Prototyping

The first prototype was carried out based on the tutorial found online [24] as an instruction to OpenGL ES. The app was to use an OpenGL ES 2.0 to draw some 2D objects and went on to learn how to apply different colours to the object create. Shown in *Figure 12 – 2d Triangle*

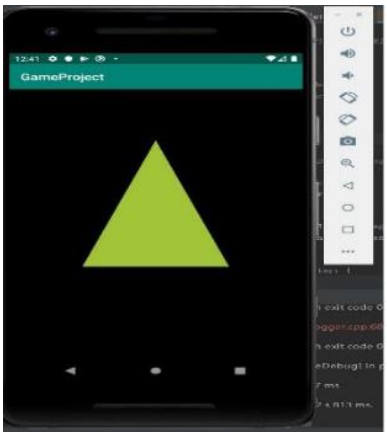

Figure 12 – 2d Triangle

5.5.2 Second Prototyping

The second app was to use the triangle create at first prototype to create a 3D Cube. This prototype was helpful as it gives an idea of how OpenGL ES handle with the 3D object and the basic of how the shader worked in OpenGL ES. Please see Figure 13 – 3D Cube

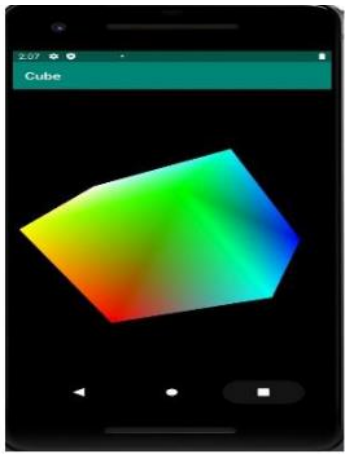

Figure 13 -3D Cube

5.5.3 Third Prototyping

Once the object is creating the other essential part of the game is the movement and texture. Hence that the third prototype was made. Based on the cube, add texture to it and then make the cube to be able to rotate and translate. Shown in *Figure 14 – RotateCube*

Figure 14 – RotateCube

It helps author gain more knowledge about the key control and transformation about matrixes. Adding textures to objects would be very useful for games, as it will make the model or scene more realistic when making 3D game. This prototype was very useful for understanding OpenGL ES and helped with the initial starting of the project.

6 Implementation and Testing

6.1 Sprint 1 02/07/20 to 2/24/20

The sprint 1 was to build a foundation for the game to be built on, it is mainly work on create the different screen for the menu and the main screen to show the game scenes.

The implementation was based on the design made earlier. The UI Sketches was used for visual guidance when creating different page for the main menu and the actual game. The class diagram and activity diagram and block diagram are all used to structure the software.

At this sprint, the class GLSurfaceView was create from the "Activity" class to attach an OpenGL ES renderer and implements the renderer interface.

At the first Sprint there were several challenges occur, and this has made the sprint to extend from 2 weeks to 3 weeks.

One biggest challenge was at the middle of the sprint 1, Author has encounter a situation where author has made a change to the code and them everything is breaking, but author doesn't have any backup of the program, So have to restart for part of the coding, hence why then author have decided to use GitHub as version control for this project throughout the project.

Another challenge was to get the player to constantly move forward base on the direction it is facing. This was hard as when the player was moving forward it also rotate based on the user control, so the direction was always changing. This problem was time consuming to resolve. It was eventually solved with the guidance from the supervisor and some research on this problem online. In order to make the player always move in the direction it faced, need to write a method using the player's current position and the angle it rotated to calculate the Euler angles. *Figure 15 – Euler Angles* Then, it can be used for the transformation to make the player always move in the right directions.

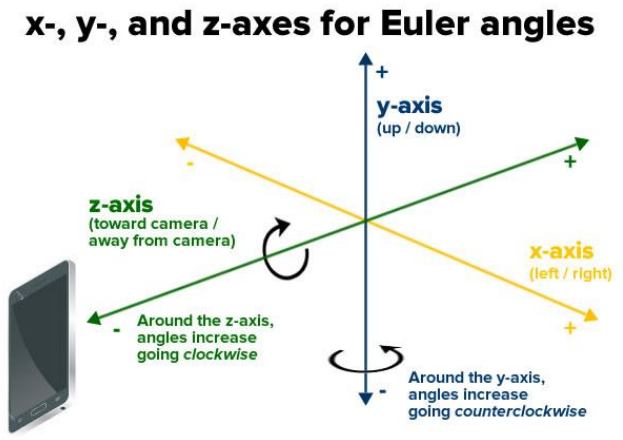

Figure 15 – Euler Angles [25]

Based on the design, the game should have different menus. In this sprint the main menu, level select menu and game end menu was made, but the problem of getting all the menu to link together was a challenge. There was no problem of linking the main menu and level select menu. As all the menu was create using the android activity, provide by the android studio. but the problem occurs when author tries to call the activity class "game end menu" from the non-activity class "GL Renderer". It took quite a lot of times to do the research and eventually found the solution by using the Intent library to instantiate an activity and call it from the render class.

Sprint 1 was challenge as author has mix up the OpenGL ES 1 and 2. This cause a lot of pains as needed to modify all the code and make sure all the code is using OpenGL ES 2.0 rather than OpenGL ES 1.

6.1.1 Testing

Testing was done on a regular basis throughout the development to ensure all the progress and changes are successful without any bugs or problems. Every time new functionality added to the project. Author will do the code testing to ensure that they work as intended. If the code work as expect and no bug, then it will be integrated with the project and another testing will be performed to ensure that it works with other units in the project. This helps reduce the risk of breaking the project.

6.1.2 Evaluation

The sprint 1 was successful as it has met the aim of the plan: All the menu screen was created and linked. The character was created with the ability to move and controlled. This provides the basic gameplay. As this was the first sprint, it was okay that not all the requirement was met.

6.2 Sprint 2 02/26/20 to 03/11/20

The second sprint was to develop the project base on the previous sprint's outcome. The aim of this sprint is to create the enemy and improve the gameplay.

The implementation was based on the design made earlier. The class diagram and activity diagram and block diagram are all used to structure the software. Create an enemy class base on the class diagram design.

A challenge face during this sprint was to getting multiple enemies to spawn. At the start of this implementation was consider spawning the enemy with different draw method. So, for each enemy using the transformation to specify the starting point. This solution was work perfectly but with a lot of multiplication of code and it was a problem when changing level. After the guidance from the supervisor and

some research on this problem online. The solution was to use a Java ArrayList library to create an array list of enemies and using the arrayList.add method to add enemy to the list to be drawn to the screen. To do this, firstly, need to set the XYZ value in the enemy class and the angle the enemy was facing. This was useful as author then can specify the position XYZ and other attributes of each enemy when they add to the array list. This was also very useful when making the AI of the enemy.

Another challenge was making the enemy trace the player. This problem was time consuming to resolve. After a lot of research about the basic AI of game. The solution was produced. The solution is to calculate the distance between 2 points and using this to move the enemy to the point. It was still complicated at the start because when the enemy tracing the player it should face the player. This was then be done by using the knowledge of Euler angle gain from previous player movement stage. It was using the x and z position to calculate the angle that the enemy should've to rotate in order to look at the player.

After the enemy AI was complete, need to do the collision detection. Collision detection is the essential part of the gameplay as this is used to determine is player died or not. There are many methods to perform collision detection; these include the sphere collision and AABB collision. The AABB stands for the axis-aligned bounding box. This method is to produce an invisible rectangle cube that covered the objects. This checks two objects by checking the range of the XYZ values for the corners of the invisible rectangle cube if they are intercepted then two objects collide with each other. *Figure 16 – AABB*

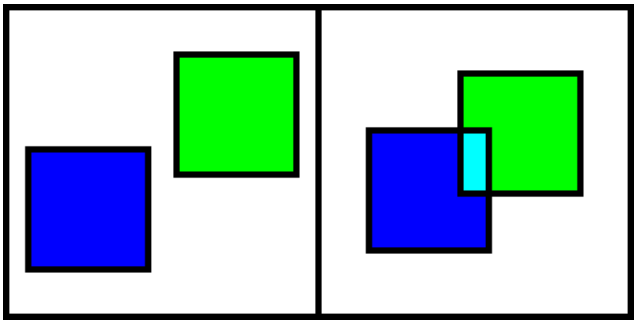

Figure 16 -AABB [26]

The Sphere collision detection is similar to the AABB method, but this method creates an invisible sphere covers the object with the radius value assigned to each object. This was then used to check if two objects distance is less than the radius or not, if yes then they were intercepted by each other. *Figure 17 -Sphere*

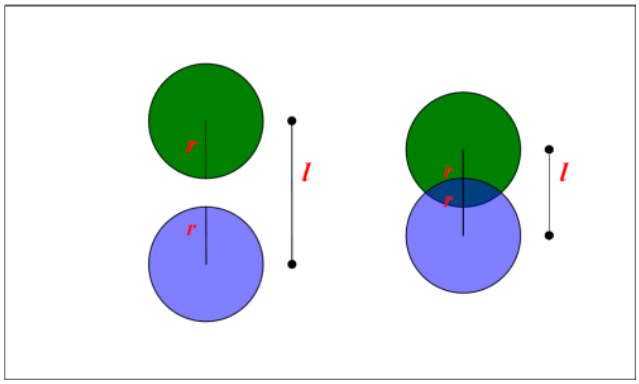

Figure 17 -Sphere [27]

After the meeting with the supervisor, it was decided to use the sphere collision detection for this project with time available, and the AABB method could be considered as a future improvement. Because this method was easy to implement, but the accuracy was not as good as AABB method.

6.2.1 Testing

The testing method used was the same as the previous sprint. After each new class or method was created, test it independently and if no bug, implement it to the game, test again to ensure no errors or crash occurs, so the game worked as intended.

6.2.2 Evaluation

As this is the second sprint, more of the requirements were met compare to sprint 1. All the unit develop on sprint 2 has successfully integrated into the project.

6.3 Sprint 3 03/14/20 to 04/13/20

The third sprint was to develop the project base on the previous sprint's outcome. The aim of this sprint is to fully complete all the requirement so the game can be ready for the user testing. This sprint was planning to last two weeks, but due to the coronavirus, the user testing has been delayed, so there is more time to implement the product hence the sprint was then taking about 4 weeks.

A challenge has occurred when author tries to load model for the player and enemies. Based on the research and the guidance from the supervisor. There are several different loaders created. One of the object loaders was using the vertex, texture and normal together to load the Obj file. This method was working perfectly independently, however, when author try to integrate it with other units, the problem occurs as the model could not load properly. Since that, another solution was considered. This was to load the model only using the vertex data. This method also worked and was successfully integrated to the project. However, the method was not perfect for load any type of object, hence it will require to using the Blender to import

the object find online and export the object with the data that can then be used for this method.

The texture was important for 3d model as it makes the model more realistic. However, loading object with texture was complicated and time consuming. Author find a solution which requires to read the .mtl file along with the Obj file to apply the texture to the model, but this solution was found to be difficult to implement after few days of work. Therefore, author decides to load the model without the texture, then write a method to apply the texture to the model manually.

In conclusion, although the second method was not perfect, but it works well with the time available, hence this method is considered as the final solution to load object problem. *Figure 18 -load Object*

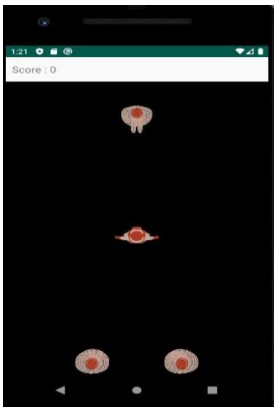

Figure 18- load Object

Another challenge faced was to show the score in the game screen and update the value every time it changes. It was decided to use the canvas to draw the score at the beginning of this implementation, however, after some research onto it, found that this method is hard to pass the value of score to the game end page. Hence that an alternative solution was produced as author found this can be done by using the android studio UI features to add the TextView. In order to show the score on the game scene, it then needs to perform an overlay view. This will show multiple views in multiple positions in a single view. Which was perfect for this problem.

The biggest challenge for this sprint is to have skybox and game objects rendered on the same scene. This problem is time consuming and author didn't find solution to achieves the goal. So, after the meeting with the supervisor, due to the limited time remaining, the skybox has been removed and decided that would be implemented in the future.

6.3.1 Testing

In term of testing, the testing method was the same as the previous sprint.

6.3.2 Evaluation

As this is the final sprint. all the requirements should be fully complete, and all the part in each sprint should link together and work as expected. The game should then ready for user testing.

7 Evaluation

In order to evaluate the game, user testing was used. There should be around 10 participants taking part for the user testing, however, because the situation occurs by the coronavirus, face-face is not available. The original plan could not carry out normally. The date for the user testing has been delay until another plan was produced. Author has considered about an online user testing using the emulator, but after meeting with the supervisor, this solution was reject as project was creating an android game, it needs the user to use the phone to play the game in order to give more accurate feedback about the control.

An alternative solution was considered as to send the code to the participant, but this will require all the participant to have the software and android phone and the knowledge to install the game which is unrealistic. Hence then decide to do the user testing with the people that live with author. As a result, only one participant take part in the user testing.

7.1 User Testing 04/20/20

The user testing has only one participant take part. As the number of participants was low, the feedback obtained from the user testing cannot be effectively analysed.

Based on the feedback obtained, the difficulty of the game controls was easy. The user can easily understand how to play the game without any help from author. Participant thinks the game was good to play, however, it is getting a bit boring when playing longer.

For the game feature part, participant thinks the enemy AI was most attractive. Participant was surprised with how the enemy was tracing the player.

When asking what feature could be useful to add in the game. Participant give feedback about adding music to the game as they think the game currently don't have any sound which is getting dull over time.

Participant also gives useful feedback about the improvement that could be made to the game. Participant found that sometimes the enemy hit player and the player doesn't die, and the game keeps going. This is useful feedback, as this will affect the gameplay. Another feedback participant gives about the movement of the player and enemy as they both don't have any animation

attach to them. Participant also suggest that the game was taking time to load. Participant need to wait for a while to let the game scene to load and show up.

After all the user testing feedback was reviewed, a table was created to sum up the requirement that need to be done for the game.

Requirement	Priority	Complexity	Complete
The game should have better collision detection	10	5	yes
The game should have background music	10	5	yes
The gameplay should improve	10	4	yes
The game should load scene smoother	10	7	no
The game should contain some animation	5	7	no

7.2 Implementation after user testing

There were improved made to the game from the user testing feedback.

From the user testing, the enemy's AI was improved. This was because the first version of the game has all the enemies to chase the player and this cause the game to become boring over time as the enemies will always behind the player and it is hard for player to die. To make change of this, another enemy was adding to the game. this type of enemy will only move forward and bounced back if they reach the edge of the game scene. This makes the game more interesting as increased of randomness of enemy movement.

A background music was implemented after the user testing. This was first considered to be feature work but after the research on this problem. Founded that Android Studio have a class called mediaPlayer can be used to control playback of audio/video files and streams. It took time to find the free music that can be used for this project. Eventually found the music and downloaded from https://www.dlsounds.com/?s=blazer.

A skybox was implemented to the game again. with more research and guidance from supervisor, author found the reason why skybox and objects cause the game to crash are because they use different shader program and author at first implementation didn't consider about this and make switch of the shader hence why this implementation was not completed. This was solved by create a class to store all the shader program and then in the skybox and objects using glUseProgram to make the switches between the shaders. *Figure 19 – Final Game*

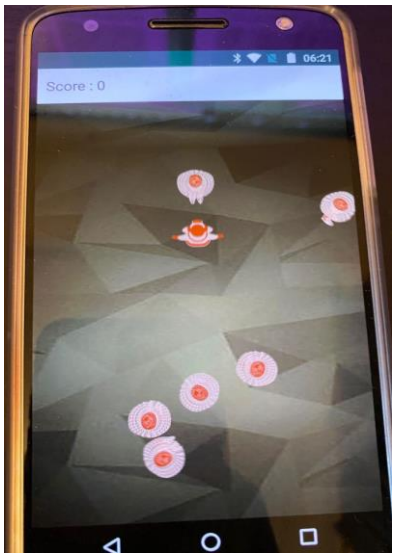

Figure 19 – Final Game

Furthermore, the accuracy of the collision detection was improved by applied more accurate value to the radius for objects. It was considered to used AABB detection if time available.

8 Description of the final product

A 3D arcade endless run game using OpenGL ES was successfully developed and produced. *Figure 19 – Final Game* and *Figure 20 – Final Game Menu* Where the player can use the touch screen to make the character turns to avoiding being caught by the enemy. If the user is still alive, they will earn points for every 10s. The score will update and show on the game screen. It was also get passed to game over screen.

Figure 20 -Final Game Menu

There is a main menu in the game to allow the user to select options such as the instruction option allows the user to a page of text content explain how the game is played and the controls, the play option will took the user to a level selection page where then user can select another option to make the choice of the game difficulty or the home option will take the user back to the main menu.

There is a game over menu implemented. This menu will only show up when player dies. It contains the score player have achieved and previous highest score. This page also gives the player two options to select: Play Again and Home. Play again option allow the user to play the level again and home option took the user to the main menu.

9 Appraisal

Overall the author evaluates the project as successful. The initial aims and objectives were all met, and a playable game was developed. However, if the author was to take on this project again, there would be couple thing that would be done differently.

1. Better User testing plan. The user testing for this project was only planned to take place once at the end of the implementation however this could be done different as the Agile Sprint was used. The user testing could have taken place at end of each sprint. This could be more helpful for author to develop a better game as author could have more time to solve the problem found after each user testing.

2. Better overall planning. When implementing some of the game mechanics and class, the planning for these components was not very suitable. As a result, some class and methods were poorly structured and might have duplicate code. With better planning, a lot of time would have been saved that could have been used to develop the optional requirements.

3. Better compatibility. The game was built specifically for the Android mobile device, however there are still a lot of different android devices such as tablet could have been considered. But this will require to adjust the layout and resolution which was time consuming.

10 Summary and Conclusion

A 3D Android arcade endless run game was successfully developed using OpenGL ES and Android Studio in May 2020. As the scope of this project was very open, a subset of requirement was created by author and achieved.

The development process led to the author creating an easy to play 3D game with easy to understand control and gameplay.

User testing was very useful as it allowed the game to be improved upon over time and produced better quality of game when compare to the initial version of the game.

Several new technologies were learned within this project, such as Java, OpenGL ES and Android Studio and they came with a very challenging learning curve. However, they are very useful throughout the project development.

The use of the Agile methodology was also very helpful due to the open scope of this project as the requirement changing throughout the project

Overall the project was successful by meeting the project goals of creating 3D Android game and gives the author a good experience in game developing. Learn new games programming skills and graphic language and being able to apply that knowledge to the project helped the author develop new skills. This experience is beneficial for the author's future development.

11 Future Work

In terms of future work, there are number of things would be improved. With the time available, some of the problem find in user testing didn't solved. Such as the time took to load the game was too long. This is because author didn't used the threads to separate the task. This cause the main thread deals to much task at once which then cause the performance to become slow. Another reason maybe because the overlay view was not as efficient as using the canvas to draw the score.

More works on the shader switch need to be done, as currently the game works well on the mobile device but not on the emulator, this cause the error and can cause the game to crash. This is very important as this suggest the game is not compatible. Research has done for this problem and author thinks the problem was occur because of the memory allocation (the task was asking for additional memory to be allocated to it and this request cause the crash).

It is also recommended to add the animation to the object, this can make them move more realistic. Also add more award object items to the game so when player collect the items, they gain points. This was considered at very beginning of the project as extension requirements which were not met due to the time available for the project.

In terms of the background sound, the game currently only has one background music. Author think by add more

sound effect such as option select sound and game over sound when player died and different background music for the game over screen would make the game more attractive.

In terms of the difficulty, currently there are only two level to select from. More level with more type of enemies can be add to the game. this will make the game more interest to play with.

Acknowledgments

The author would like to thank Dr Iain Martin for his supervision, feedback and guidance throughout the project.

The author would also like to thank all the participants who took part in the user testing for their valuable feedback and constructive criticism.

References

[1] "OpenGL ES"
https://www.khronos.org/opengles/

[2] "How Good Is Unity for Game Development"
https://www.redappletech.com/how-good-is-unity-for-game-development/

[3] "What is Android Studio and Android SDK tools"
http://androiddeveloper.galileo.edu/2017/03/29/android-studio-and-android-sdk-tools/

[4] "Android (Operating System)"
https://en.wikipedia.org/wiki/Android_(operating_system)

[5] "The Best 15 Mobile Game Engines/Development Platforms & Tools in 2020"

https://thetool.io/2018/mobile-game-development-platforms

[6] "What is the best game engine: is Unreal Engine right for you?" by Marie Dealessandri

https://www.gamesindustry.biz/articles/2020-01-16-what-is-the-best-game-engine-is-unreal-engine-4-the-right-game-engine-for-you

[7] "The Beginner's Guide to Android Game Development"

https://www.androidauthority.com/the-beginners-guide-to-android-game-development-692253/

[8] https://developer.android.com/studio/features

[9] "Java - What is OOP?"

https://www.w3schools.com/java/java_oop.asp

[10] "What is Java Garbage Collection? How It Works, Best Practices, Tutorials, and More"

https://stackify.com/what-is-java-garbage-collection/

[11] Shreiner, D. "OpenGL programming guide : the official guide to learning OpenGL, version 4.3". https://books.google.co.uk/books?hl=en&lr=&id=jG4LGmH5RuIC&oi=fnd&pg=PT27&ots=q3IPz6oIAZ&sig=aDzVBqWMrq6XCynu26eZFGrW3-Q&redir_esc=y#v=onepage&q&f=false

[12] https://www.khronos.org/opengles/

[13] Christophe Quarre, Haizhen Li, Chester Chan Kwok-Kee "OpenGL to OpenGL/ES translator and OpenGL/ES simulator"

https://patentimages.storage.googleapis.com/b5/3e/86/c6a694c6b97c1e/US8347275.pdf

[14] David Blythe, Aaftab Munshi "OpenGL ES Common/Common-Lite Profile Specification"

https://www.khronos.org/registry/OpenGL/specs/es/1.1/es_cm_spec_1.1.pdf

[15] "OpenGL Utility Toolkit"

https://en.wikipedia.org/wiki/OpenGL_Utility_Toolkit

[16] "EGL (API)"

https://en.wikipedia.org/wiki/EGL_(API)

[17] "Mali GPU OpenGL ES Application Development Guide"

http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dui0363d/CJAJHGHF.html

[18]https://developer.android.com/guide/topics/graphics/opengl

[19] Lindsay Grace “Game Type and Game Genre”

http://aii.lgracegames.com/documents/Game_types_and_genres.pdf

[20]https://play.google.com/store/apps/details?id=com.fyrehouse.boonk&hl=en_GB

[21] Tom Hallissey “What is a Target Audience, and why is it so Important?”

https://www.fatguymedia.com/inbound-marketing/target-audience

[22] https://www.blender.org/

[23] “2D Vs 3D Games: Differences, Benefits and Costs”

https://meliorgames.com/game-development/2d-vs-3d-games-differences-benefits-and-costs/

[24] “Build an OpenGL ES environment”
https://developer.android.com/training/graphics/opengl/environment?hl=en

[25] Joey deVilla “Augmented Reality in Android with Google’s Face API”

https://www.raywenderlich.com/523-augmented-reality-in-android-with-google-s-face-api

[26] “2D AABB Collision Detection and Response”

https://hopefultoad.blogspot.com/2017/09/2d-aabb-collision-detection-and-response.html

[27] Evan Closson “Efficient and Accurate Collision Detection Hierarchies with Collision Response”
http://www.evanclosson.com/projects/collisiondetectionandresponse

[28]https://www.khronos.org/registry/OpenGL-Refpages/es2.0/xhtml/glVertexAttribPointer.xml

[29]https://www.khronos.org/registry/OpenGL-Refpages/gl2.1/xhtml/glVertexPointer.xml

[30]https://www.khronos.org/registry/OpenGL-Refpages/gl2.1/xhtml/glNormalPointer.xml

[31] Shashank Bhardwaj, Nirupma Singh “COMPARATIVE STUDY ON MARKET ANALYSIS OF SUBWAY SURFER AND TEMPLE RUN”
http://www.gujaratresearchsociety.in/index.php/JGRS/article/view/1385/1144

[32] Victory Media “Ethical issues in the Games industry”
https://www.slideshare.net/jcolebrook/ethical-issues-in-the-games-industry

Appendices

Appendix A: Background Research and Analysis

A.1 Ethics

Appendix B: Design

B.1 Product Backlog

B.2 requirement Specification

B.3 Gantt chart

B.4 MenuAT

B.5 GameAT

B.6 MenuSketch

B.7 GameSketch

B.8 ClassDiagram

B.9 BlockDiagram

Appendix C: Mid Progress Report

Appendix D: Meetings

Appendix E :Source Code

Appendix E :Evaluation

Appendix F :User Manual