Air Smash

By

Ali Tariq

Saad Abdullah Munir

Zaeem Saeed Toor

Submitted to Faculty of Department of Computer Software Engineering National University of Sciences and Technology, Islamabad in partial fulfilment for the requirements of a B.E Degree in Computer Software Engineering, May 2017

In the name of Allah, the Most Beneficent, the Most Merciful

ABSTRACT

Air Smash

The era of information technology brings with it a plethora of entertainment packages, the greatest of which is the video game industry. This billion-dollar industry, has not only engulfed the entire globe and brought to life all scenarios, it is an avenue worth marching on. Over a passage of time the contents have been molded and shaped on depicting western society's culture and views.

Air Smash is a table tennis simulation game. Virtual Reality (VR) headset assisted by a smartphone is used as a visual device to provide VR features and another smartphone is used as a table tennis racket. Accelerometer and gyroscope sensors embedded in the phones will play a major role in the gameplay. This game replicates the gaming experience of expensive gaming consoles such as X-box Kinect and Nintendo Wii by providing motion sensing features keeping cost effectiveness as a major concern.

The motion sensors embedded in the smartphone being used as a controller (racket) detect movement of the hand and orientation is changed accordingly in the display phone mounted inside the headset.

CERTIFICATE FOR CORRECTNESS AND APPROVAL

It is certified that work contained in the thesis – Air Smash carried out by Ali Tariq, Saad Abdullah Munir, Zaeem Saeed Toor under supervision of Dr. Seemab Latif for partial fulfilment of Degree of Bachelor of Software Engineering is correct and approved.

Approved By

Dr. Seemab Latif

Department of CSE, MCS

Dated: _____

DECLARATION

No portion of the work presented in this dissertation has been submitted in support of another award or qualification either at this institution or elsewhere.

DEDICATION

In the name of Allah, the Most Merciful, the Most Beneficent.

To our parents, without whose unflinching support and unstinting cooperation,

a work of this magnitude would not have been possible.

To our supervisor, Dr. Seemab Latif who has given us great support and valuable suggestions throughout the implementation process.

And finally, to our Friends and siblings for their encouragement.

ACKNOWLEDGEMENTS

There is no success without the will of ALLAH Almighty. We are grateful to ALLAH, who has given us guidance, strength and enabled us to accomplish this task. Whatever we have achieved, we owe it to Him, in totality. We are also grateful to our parents and family and well-wishers for their admirable support and their critical reviews. We would like to thank our supervisor. Dr. Seemab Latif, for her continuous guidance and motivation throughout the course of our project. Without their help, we would have not been able to accomplish anything.

Table of Contents

Chapter 5 Project Test and Evaluation

Chapter 6 Future Work

Bibliography

Appendix

Table of Figures

Chapter 1: Introduction

1.1 Overview

The product of this project is an android game which can be installed using '. apk file'. This game turns your phone into either a motion sensing device or a VR Headset. In this game, the motion sensing device acts as a racket in the player's hand. The racket phone detects the orientation of the racket and the force by which the player swings the racket.

The VR Headset acts as a display through which the player can see the game. The display phone automatically adjusts the angle of the view depending upon where the player is looking. To the player, it seems as he/she can view the virtual playing environment by 'looking around' normally.

Both are connected to each other via Wi-Fi. This can either be achieved by connecting both phones over the same network or by using one phone's Wi-Fi hotspot and connecting the other to it.

1.2 Problem Statement

Virtual reality is the emerging trend of the era. Nearly every industry, may it be medicine or gaming, is trying to make use of this new mode of media. One of the problems these industries might face is the massive cost. Such is also the case with motion sensing controls. Some expensive devices used for motion sensing controls for games include Nintendo Wii and Oculus Rift.

Virtual reality products should be available to the people from multiple public sectors. The price should range from an affordable rate for low end devices to more expensive rates for high end and sophisticated devices.

We plan to cut down on the cost of VR by using something that is readily available in every household, 'Smartphones'. The user also needs to buy a Google Cardboard device which comes in a variety of different quality and price so that each user can select a device suitable to particular needs and financial situations.

1.3 Approach

The aim is to provide the user with Virtual Reality experience and motion sensing controls in one package. We propose a game in which the player can use two Android smartphones, one inside the Google Cardboard device (display phone) and the other in his or her hand (racket phone), to play a table tennis game.

The Gyroscope, Accelerometer and Magnetometer sensors in the racket phone allow the game to sense motion as the user plays the shot. The Gyroscope sensor in the display phone allow the game to track head movement of the player and change the view of the game depending upon where the user is looking.

The two smartphones will be connected via Wi-Fi. This can be achieved by either connecting the smartphones to a common Wi-Fi network or by using one smartphone's Wi-Fi hotspot. The Unity 3D game development engine is used to develop the game. This game will be fully playable at the end of this project. It is also a proof of concept to encourage the development of other such applications in different domains.

1.4 Scope

Air Smash is a table tennis simulation game. Virtual Reality (VR) headset assisted by a smartphone is used as a visual device to provide VR features and another smartphone is used as a table tennis racket. Accelerometer and gyroscope sensors embedded in the phones will play a major role in the gameplay. This game replicates the gaming experience of expensive gaming consoles such as X-box Kinect and Nintendo Wii by providing motion sensing features keeping cost effectiveness as a major concern.

1.5 Objectives

This project has the following key objectives:

- To develop a fully functional game for Android devices.
- To incorporate 3D Virtual Reality in the game.
- To provide the user with 360 degrees view into the game.
- To provide the user motion based controls in the game.

- To keep the controls realistic enough for the gameplay to resemble real life table tennis experience

1.6 Deliverables

Sr.	Tasks	Deliverables
1	Literature Review	Literature survey
2	Requirements Gathering	Software Requirements Specification document (SRS)
3	Application Design	Software Design Specification document (SDS)
4	Implementation	Project demonstration
5	Testing	Evaluation plan and test document
6	Deployment	Complete application along with necessary documentation

1.7 Document Conventions

This section describes the standards followed while writing this document. Following are some words used frequently

- **VR** that stands for Virtual Reality
- **Racket phone**: Android smartphone used as a racket
- **Display phone**: Android smartphone used inside a VR headset for display

1.7.1 Headings:

Heading are prioritized in a numbered fashion, the highest priority heading having a single digit and subsequent headings having more numbers, according to their level.

All the main headings are titled as follows: single digit number followed the name of the section (Times New Roman, size 18).

All second level sub headings for every sub section have the same number as their respective main heading, followed by subsequent sub heading number followed by name of the sub section (Times New Roman, size 16).

Further sub headings, i.e. level three and below, follow the same rules as above for numbering and naming, but different for font size (Times New Roman, size 14).

1.7.2 Basic Text

All other basic text appears in regular, size 12 Times New Roman. Every paragraph explains one type of idea.

1.8 Intended Audience and Reading Suggestions

The intended audiences for the Air Smash include the project supervisor, the BESE 19 FYP group (developers), UG project evaluation team, and other persons at MCS CSE Department.

1.8.1 Project Supervisor

It will help the supervisor to supervise the project and guide the team in a better way. This document will be used by her to check whether all the requirements have been understood properly or not. In the end, she will be able to evaluate whether the requirements have been implemented properly and completely or not.

1.8.2 Developing and Testing team

It will help the developer to develop the product and to trace back the functional requirements. The document will provide guidance to the developers to determine what the requirements are and how they should continue with the project.

It will help the testers to understand the constraints. Project testers can use this document as a base for their testing strategy as some bugs are easier to find using a requirements document. It will help in building up test cases for the testing process.

1.8.3 End user

This document can be read by the end users if they wish to know what the project is about and what requirements have been fulfilled in this project.

1.8.4 UG Project Evaluation Team

Evaluation committee which will evaluate the progress of UG Projects. This includes both external and internal evaluators. The document will provide the FYP evaluators with the scope, requirements and details of the project to be built. It will also be used as basis for the evaluation of the implementation and final project.

1.8.5 Reading suggestions

The document compromises of the overall description of the project which includes its product perspective, features, operating environment, design and implementation, testing and its results.

Readers interested in a brief overview of the product should focus on the rest of Part 1 (Introduction), as well as Part 2 of the document (Literature Review).

Part 3 (Software Requirements Specifications) offers further details in project, including information on the user interface as well as the hardware and software platforms on which the application will run.

Readers who wish to explore the design features of this project in more detail should read on to Part 4 (Design and Development), which expands upon the information of system architecture, multiple components involved and the classes that design the project.

Readers interested in just the working of project should focus of Part 5 (Test and Evaluation), comprising of testing techniques used and results achieved after carrying out multiple tests.

The Project Supervisor, evaluators, coordinator, panel are advised to go through the entire document.

1.9 Product Vision

Name	Air Smash.
Is	A real-time 3D table tennis game.
For	This game is being developed primarily for youth, but can be played by any age group who wishes to play.

What	The game will be played using a smartphone used as a racket. This racket will be simulated in another smartphone mounted inside a VR headset.
That	The application is intended to provide a real-time 3D gaming experience to anyone who is interested in playing real time simulating table tennis game on Android smartphone.

1.10 References

1.10.1 Unity 3D Development Home

- http://unity3d.com/
- http://unity3d.com/learn/documentation

1.10.2 Autodesk

- http://www.autodesk.com/products/

1.10.3 3D Studio Max

- http://www.autodesk.com/products/autodesk-3dstudiomax/overview

1.10.4 Adobe Photoshop

- http://www.adobe.com/products/photoshopfamily.html
- http://www.photoshop.com

Chapter 2: Literature Review

This project can be viewed as a combination of sub projects. These sub projects would include the different aspects of sensing the smartphone's motion in the hand of the player i.e. 'orientation detection', 'position detection' and 'Force detection'.

2.1 Hand Motion Sensing

Several research papers are written on applications of sensing hand movement via wearable inertial sensors and also smartphones. A summary of information collected by examining these research papers is presented for review:

2.2 Orientation Detection:

The android smartphone is fairly accurate when it comes to detection of orientation. It uses a fusion of two sensors namely Gyroscope and Magnetometer to estimate orientation. But as we will see further in this chapter orientation accuracy is a requirement for position estimation and even small errors of a fraction of a degree can result in accumulative position errors.

2.3 Position Detection:

The fundamental data about position is obtained by the help of accelerometer. It provides us with a reading of acceleration applied to the device. This reading can be used to determine position by integrating it twice. Another method would be to use the equation of motion:

$$S = vi * t + ½ * a * t^2$$

This equation gives us the change in distance S, in the past time interval given the initial velocity vi, the acceleration a, and the duration of time interval t. The velocity can be kept track of using the equation of motion:

$$vf = vi + a * t$$

This equation gives us the final velocity vf, at the end of each interval given the initial velocity vi, and duration of time interval t. The initial velocity for the very first time interval would be considered zero.

Finally, we can calculate the new position Pf, by adding the distance moved S, in the interval to the old position Pi:

$$Pf=Pi+S$$

Of course, all these calculations are to be made specific to the direction provided by the above-mentioned sensor fusion of gyroscope and magnetometer. For example, a movement of 0.1 meters in one direction would result in a different position than moving 0.1 meters in another direction offset by 1 degree. Each calculation made in one interval is used in the calculation of the next interval i.e. initial velocity of one interval is the final velocity of the previous interval and initial position of one interval is the final potion of the previous interval. Therefore, any error, however small it may be just adds to the error produced in the previous steps and the error grows.

One method proposed is to reset the value of the acceleration, velocity and position when it is almost certain. For example, in our case the player cannot the phone in one direction in constant velocity because of the dynamics of the game and nature of the gesture made by the player while swinging the racket. Hence, if accelerometer is showing a reading close to zero, it must mean that the device is stationary and velocity is zero rather than constant.

Another method is to use high pass and low pass filters where necessary. For example, the accelerometer data should be passed through a high pass filter to filter out any sudden changes in the reading as these sudden changes are almost always noise as compared to actual data. The extent to which we should filter the data can only be determined by hit and trail method. We would choose the filter parameters which would produce the best results.

2.4 Force Detection:

By force detection we can emulate how hard or how soft the ball is hit by the racket when the player swings the racket. The accelerometer sensor can be used to obtain this data. The accelerometer reading provided by the accelerometer can be simply multiplied by a constant to obtain force. This idea is inspired by the equation:

$$\mathbf{F = m * a}$$

Force F, is equal to mass m, times acceleration a. ideally this constant should be equal to mass of the smartphone used but we could use this constant to control the dynamics of the game and could use a value that produces best results.

2.5 Conclusion:

After much research and study, we conclude the final scope of the project to be able to detect the force and orientation of the devices to the best of our efforts. Position of the device can be narrowed down to some extent using filters and other correctional techniques. We will pursue the idea as farther as we can but will consider it an extended scope in this project.

Chapter 3: Software Requirements Specification

3.1 Introduction

The introduction of the Software Requirements Specification (SRS) document provides an overview of the entire SRS with purpose, scope, references and overview of the SRS. The aim of this document is to present detailed description of the Air Smash game keeping cost-effectiveness as a major concern.

3.1.1 Purpose

The purpose of this document is to present a detailed description of the Air Smash (a real-time 3D table tennis game). It will explain the purpose and features of the application, the interfaces of the application, what the application will do, the constraints under which it must operate. This document is intended for both the stakeholders and the developers of the system and will be proposed to the Evaluation Panel for its approval.

3.2 Overall Description

3.2.1 Product Perspective

The Air Smash game is being developed to provide advanced features in a simple table tennis game keeping cost-effectiveness as a major concern.

3.2.2 Product Features

- The game will be displayed on an Android smartphone mounted inside any type of VR headset.
- Another Android smartphone will be used as a racket.
- Racket movement will be simulated on the display phone.
- Sensors used for data collection during the gameplay are Accelerometer, Gyroscope and Magnetometer.
- Communication between the phones will be ensured using wireless media.
- On collision of racket and ball the system will compare racket phone motion and ball's motion before collision to determine the ball's motion after collision.

3.2.3 Operating Environment

3.2.3.1 Hardware

Android Smartphones

- One smartphone will be mounted in the VR headset displaying the gameplay.
- Another smartphone will be used as a racket of the game.
- Connectivity between both the phones will be done using Wi-Fi.

VR Headset

- VR headset will be used for providing Virtual Reality features to the gameplay.

3.2.3.2 Software

- Android OS 4.1(Jellybean) or above.

3.2.4 Design and Implementation Constraints

Since gaming is one of the most financially rich and technologically advanced industries therefore it requires a gist of highly advanced set of tools. Keeping in view the aforesaid, few shortcomings are:

Constraint	**Explanation**
Technical	• Magnetic interference can result in data disruption or noise that affects the accuracy of the gameplay. • Gyroscope provides accurate data for fast paced movements but gives erroneous readings if a particular orientation is maintained for a long period of time. • Accelerometer provides values of acceleration based on fixed intervals. Using double integration, we can acquire the position but due to the integration constant we face uncertainty in the acquired position. If somehow we overcome this problem of integration error, still we can't get continuous acceleration readings rather we get values based on the intervals with a difference of milliseconds. This will be taken care off in the extended scope.
OS Required	VR functionality can only run on android phones having OS version jellybean (4.1) and above.

3.3 External Interface Requirements

3.3.1 User Interfaces

- Game user interface will be displayed on the display phone.
- Racket will be seen on the display. Its orientation will be according to the orientation of racket phone.
- User will configure the game before wearing VR headset using a menu on the racket phone.

3.3.2 Software Interfaces

- Adobe Photoshop will be used to design graphics of 2D UI elements that can be imported to unity as images/textures.
- 3DS Max will be used to design 3D objects that will be used as 3D models with .(dot)fbx extension to be imported in unity as prefabs.
- The display phone can be configured to be used in VR headset by using scripts provided in Google VR SDK for Unity which can be imported as a package.
- Once the game has been fully designed, it can be compiled to .(dot)apk file by providing location of android SDK which in our case is located on the local machine

3.3.3 Hardware Interfaces

- The display phone will be mounted inside the VR headset that provides a single input which essentially is a magnet mounted on the side of the headset. A change in magnetic field is translated as binary input by the mounted device.

3.3.4 Communication Interfaces

- Communication between racket phone and display phone will be done using Wi-Fi.

3.4 System Features

Following are the system features of Air Smash with detailed descriptions.

- Displaying the gameplay
- Connectivity configuration between devices
- Simulation of racket
- Communication between devices
- Accurate gameplay
- Standard game rules

3.4.1 Displaying the gameplay

3.4.1.1 Description

The game will be displayed on a smartphone that will provide the user with a 3D view of the play area in a first-person perspective. The display phone shall be mounted on a VR headset to provide VR features.

3.4.1.2 Pre-conditions

The smartphone used is designed to support 3D display.

3.4.1.3 Post-conditions

The user can see the gameplay just like playing in real time as a result of virtual reality.

3.4.1.4 Interactions

Using the VR feature, user can look around to observe the different objects displayed in the game.

3.4.1.5 Functional requirements

REQ 1: The system shall display a play area having a table-tennis table, rackets and a ball as per the standards mentioned in “Appendix A”.

REQ 2: The system shall display a scoreboard displaying the player’s score.

REQ 3: The system shall display a play environment viewable in 360 degrees by looking around.

3.4.2 Connectivity configuration between the devices

3.4.2.1 Description

The game will be played by using another smartphone as a racket that will be held in user’s hand. The user will be guided at the start of the game to connect the racket phone before wearing the VR headset.

3.4.2.2 Pre-conditions

The smartphone used for racket should maintain connectivity with the smartphone used for display.

3.4.2.3 Post-conditions

The user will be able to hit the ball by describing the force and direction of racket by moving the racket phone with hand.

3.4.2.4 Interactions

User will be able to control the orientation and acceleration of racket while playing a shot.

3.4.2.5 Functional requirements

REQ 4: The system shall allow the user to connect both devices before wearing the headset.

3.4.3 Simulation of racket

3.4.3.1 Description

The movement of the racket will be simulated on the display. This will provide a real-time effect of playing the game.

3.4.3.2 Pre-conditions

The racket phone must consist of the essential sensors i.e. accelerometer and gyroscope.

3.4.3.3 Post-conditions

The racket phone's orientation and the force by which the shot is played will be simulated on the display phone.

3.4.3.4 Interactions

By moving the racket phone, the values of the sensors will change resulting in different simulations. User should be able to control the orientation of racket and will also be able to accelerate the racket while playing a shot.

3.4.3.5 Functional requirements

REQ 5: The system shall simulate the orientation of racket as per the orientation of the racket phone.

REQ 6: The system shall simulate the position of racket as per the position of the racket phone in limited 3D space i.e. 0.5m along the width and 1m along the length of the table (**Extended scope**).

3.4.4 Communication between the devices

3.4.4.1 Description

There must be steady and constant communication between the display and the racket phones without which the game will not function properly. Wireless media will be used for consistent communication between the devices.

3.4.4.2 Pre-conditions

The devices must be capable of Wi-Fi connectivity feature that will allow them to be connected on the same network.

3.4.4.3 Post-conditions

The network source provider must be strong enough to maintain a connectivity of at least 4 devices which will be fully utilized in the extended scope.

3.4.4.4 Functional requirements

REQ 7: The system shall send the sensor's data over wireless media from racket phone to the display phone.

3.4.5 Gameplay according to rules

3.4.5.1 Description

The game will be played based on standard rules of table tennis. These rules are mentioned in the appendix A.

3.4.5.2 Pre-conditions

User is playing the game.

3.4.5.3 Post-conditions

The player failing to play the game according to the rules will lose points according to game rules.

3.4.5.4 Functional requirements

REQ 8: The system shall enforce table-tennis rules during the gameplay.

REQ 9: The system shall notify user in case of loss or gain of point.

3.4.6 Accurate gameplay

3.4.6.1 Description

Accuracy in the gameplay refers to the racket's velocity which will decide the new velocity of the ball after being hit by the racket. Also, the correct orientation of racket according to phone movements using gyroscope. User experience will be enhanced by filtering the unrealistic data. In extended scope, accuracy will be the position of the racket in 3D space.

3.4.6.2 Pre-conditions

The racket phone consists of accelerometer and gyroscope and it is successfully configured with the display phone.

3.4.6.3 Post-conditions

By attaining the aforesaid accuracy conditions, the user will be able to enjoy a real-time gaming experience as if, actually playing the table tennis game.

3.4.6.4 Functional requirements

REQ 10: The system shall filter the sensor's data to avoid unrealistic movements of racket.

3.5 Other Nonfunctional Requirements

3.5.1 Safety Requirements

- The use of the software product has no harms whatsoever, nor does it have any possibility of loss or damage that might be inflicted to the devices used.
- However, use of VR device can cause harmful effects on the eyes due to which, each match of the table tennis can be either of 5 points or 10 and not more than that.
- Also, to avoid any physical damage to the display phone used in the VR headset, it must be fixed properly at the head so it doesn't slip off and the racket phone must not slip from the user's hand while playing an aggressive shot.

3.5.2 Security Requirements

- Application running on the smartphones should not need any additional information other than the collected data from the sensors during hand movement or already present data.
- Also, the network will be secured by password protection provided by default security for WLAN.

3.5.3 Software Quality Attributes

3.5.3.1 Usability

The graphical user interface of app is to be designed with usability as the first priority. The app will be presented and organized in a manner that is both visually appealing and easy for the user to navigate or play.

3.5.3.2 Reliability

This game should be reliable with minimum errors. The game should not crash frequently. Minimum crashes should be once in thousand times.

3.5.3.3 Portability

In API, portability can be defined as “compatibility of application with platform (Android’s version) upgraded or downgraded versions”. The game should be compatible for all versions of Android after 4.1 (Jellybean).

3.5.3.4 Availability

The application will be available from boot to shut down, provided mobile is in working state and the application is installed and configured properly.

3.5.3.5 Flexibility

The design and architecture of the application will be flexible enough for catering any new requirements, if any at some later stage or for the application enhancement.

3.5.3.6 Easy to use

Our game uses only those symbols and words that are immediately understandable by users. All the technical details should be hidden from the user.

Chapter 4: Design and Development

4.1 Introduction

4.1.1 Purpose

This software design specification (SDS) document describes the architecture and system design of the Air Smash (a real-time 3D table tennis game). It mostly contains different design diagrams and their explanation.

The document is intended to inform stakeholders of the details of the design and the design process. This document will help the developer(s) in implementation and maintenance of the application.

4.1.2 Document Overview

This document shows the design and working of Air Smash application. It starts from higher level details for a non-technical reader to understand just by seeing the diagrams to the lower level details that aid the developer to code and understand other technical details of the application.
Section 2 the **System Architecture** Description gives a detailed overview of the application. It shows the main component of the application and their inter-relationships. Also, it shows the higher-level details of the system by means of **System Block, Activity, State Transition** and **Use Case** diagrams. Lower level details are described using the **Class** and **Sequence diagrams**.
In **Section 3**, **Detailed Description of Component** is given to show the working of modules with low level details. It shows the purpose, function, subordinates, dependencies, interfaces, resources, processing and data of the components and their relationships with each other.
Section 4 shows the **Reuse and Relationship to other Products** i.e.; information about work done in the same project before and any reuse of the same work. The section also provides a key to reuse this system for further upgrades.
Section 5 Design Decisions and Tradeoffs shows the architecture style, while in the **Section 6** the **Pseudo Code** is given in for human reading rather than machine reading. **Appendix** is included in **section 7.**

4.1.7 Work Breakdown Structure (WBS)

WBS is utilized at the beginning of the project to define the scope, estimate costs and organize Gantt schedules. Work breakdown structure, WBS, captures all the elements of a project in an organized fashion.

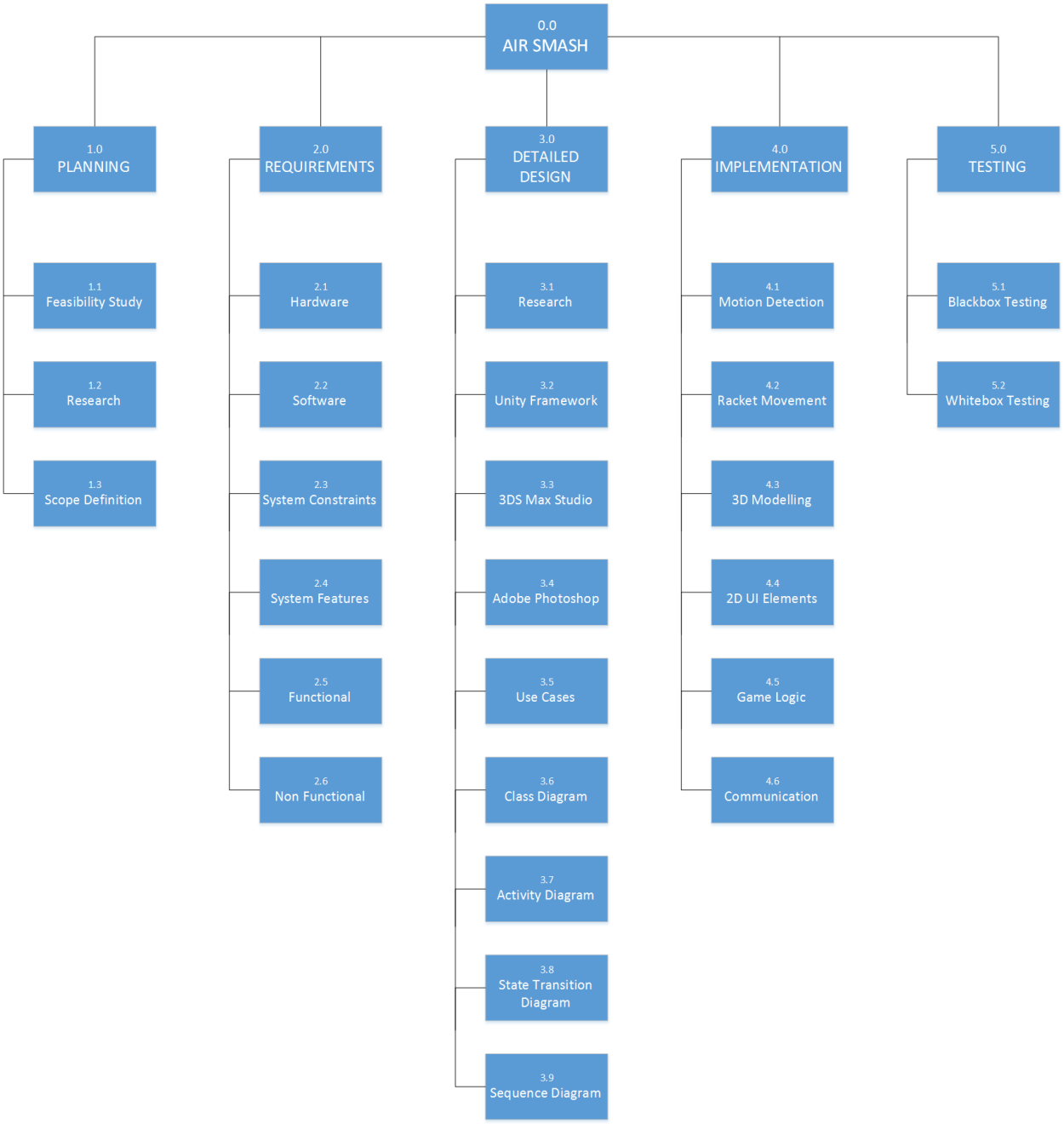

Figure 4.1 Work Breakdown Structure

4.2 System Architecture Description

4.2.1 Architecture Diagram

The communication of our project is based on client-server architecture. The racket phone acts as the client and the display phone is the server. The racket phone sends the orientation data to the

display phone.

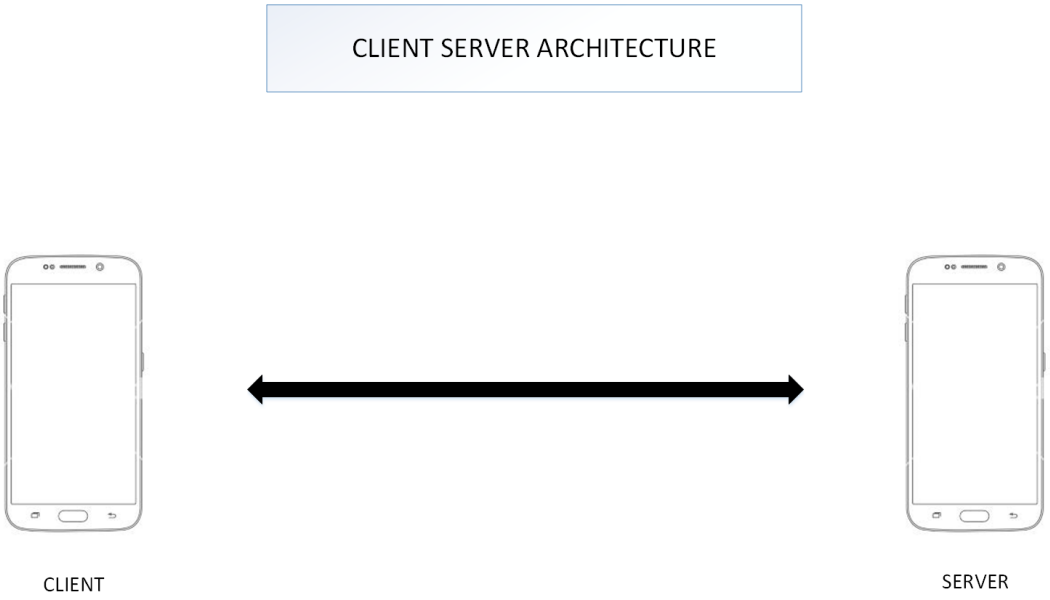

Figure 4.2 Architecture Diagram

4.2.2 System Block Diagram

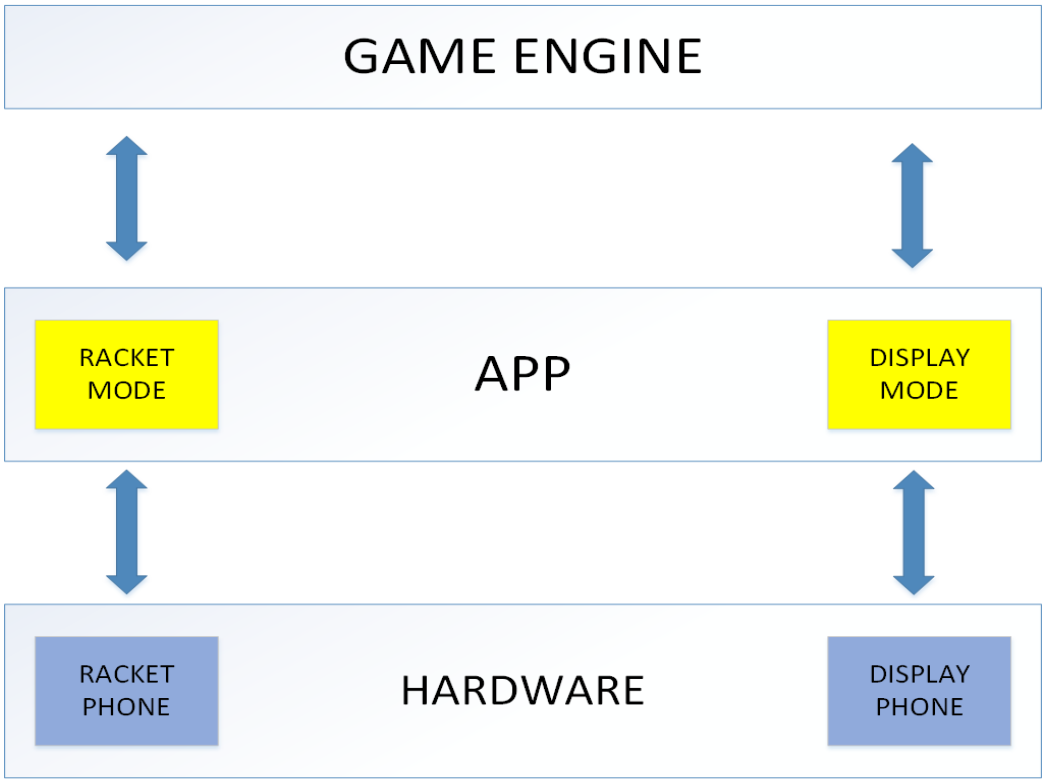

Figure 4.3 System Block Diagram

4.2.3 Component Diagram

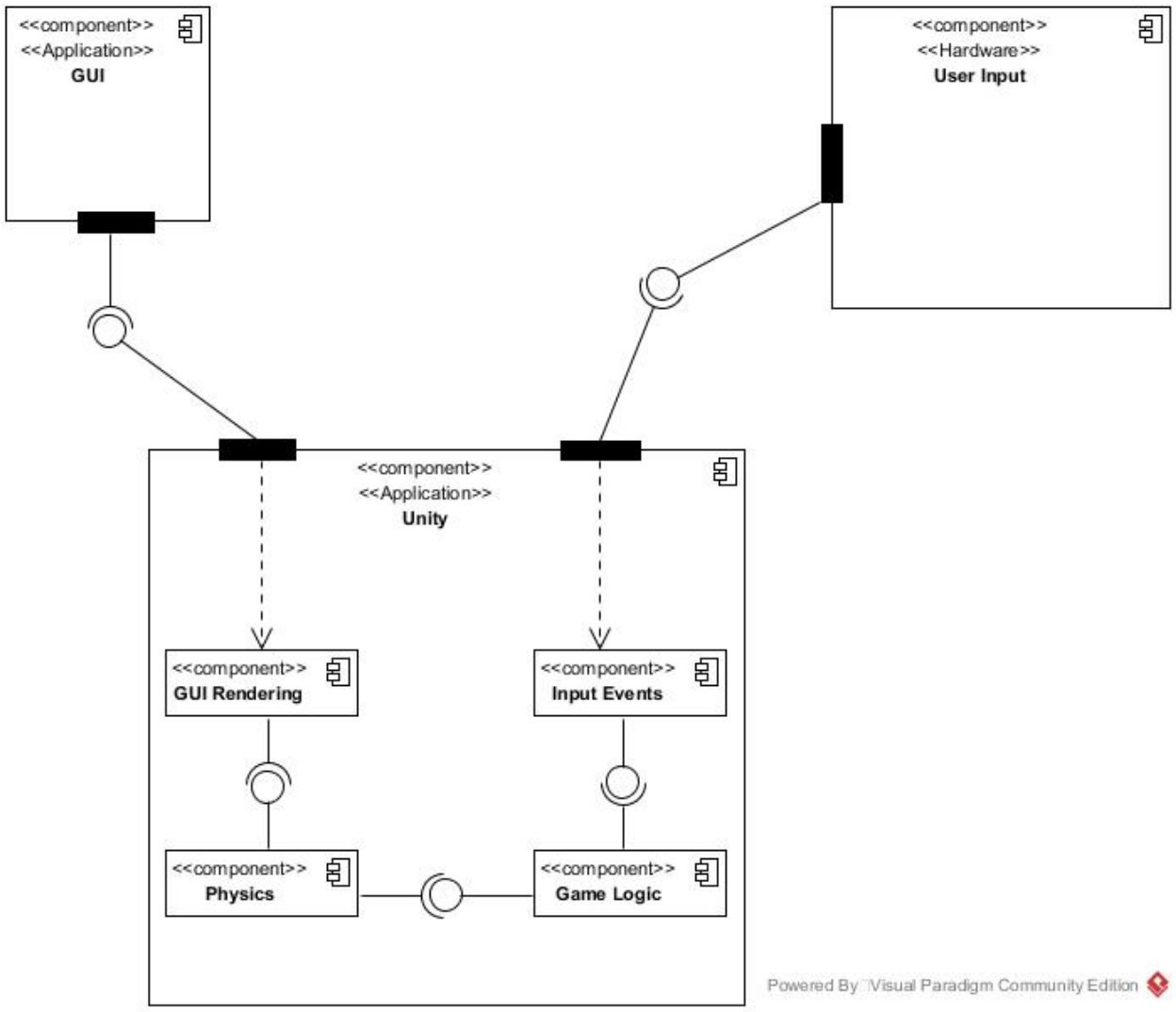

Figure 4.4 Component Diagram

The main components are

- User Input
- GUI
- Unity Engine

4.2.4 User View (Use Case Diagram)

Use cases tell how the user shall be able to interact with the system and how the system shall respond in return. They provide a means to show the functionalities of the system in a user-concentric manner, without paying attention to the dynamic behavior of the system or its underlying working.

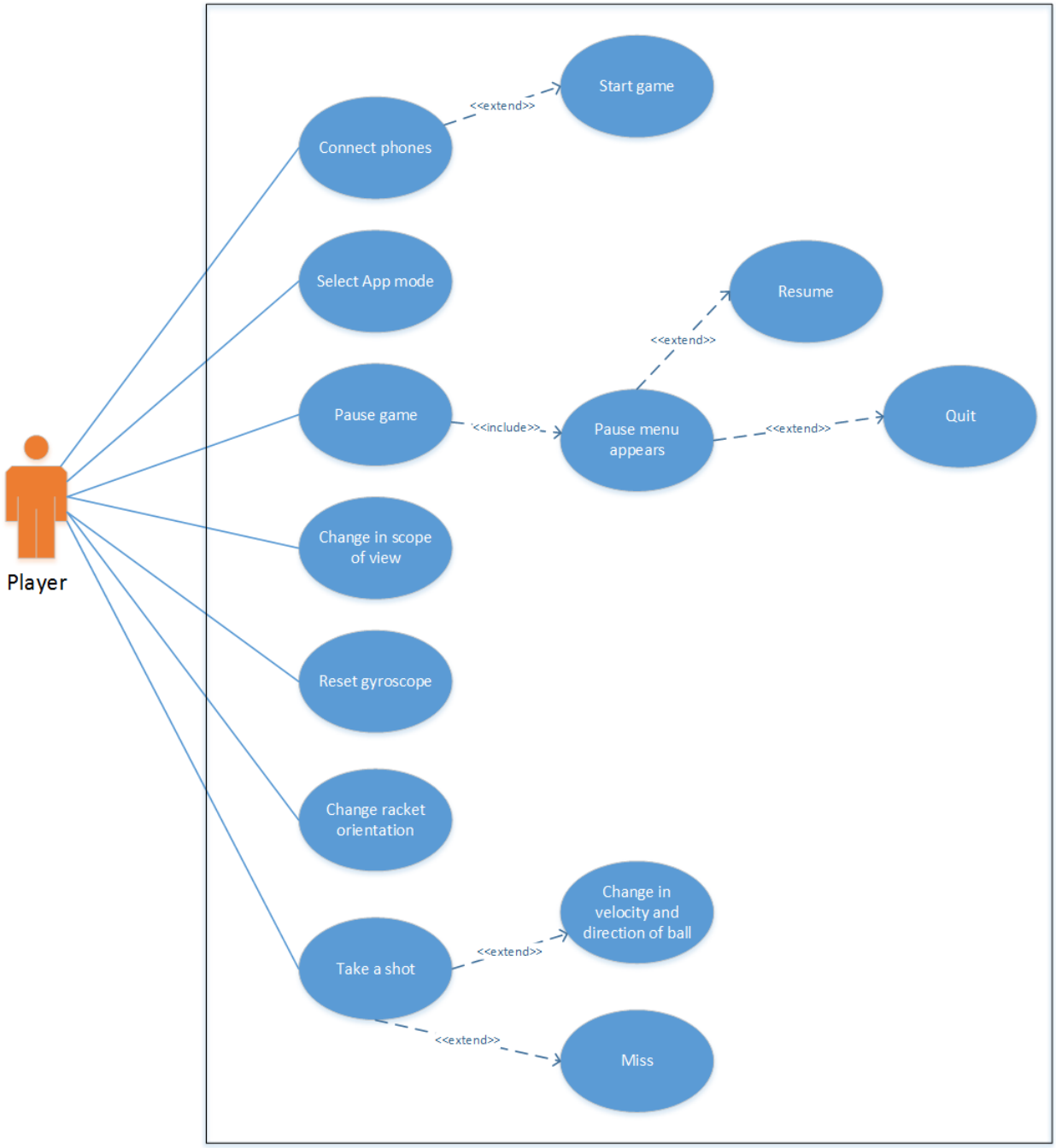

Figure 4.5 Use Case Diagram

4.2.5 Use Case Descriptions

The section will include the detailed description of all the individual use cases.

4.2.5.1 Use Case 1

USE CASE NAME	Connect phones
ACTOR	Player
NORMAL CASE	• Both phones have the application installed. • Network configuration is done on both for establishing connection.
ALTERNATE CASE	• The user will not be moved to next screen until devices are successfully connected. • A prompt will show up to try again if the devices are not connected.
PRE-CONDITION	The game must be installed on the mobile device.
POST CONDITION	The phones are connected
EXTENDS	Start game
INCLUDE	N/A
ASSUMPTIONS	Phones are configured correctly for connection and application is installed on them.

4.2.5.2 Use Case 2

USE CASE NAME	Start game
ACTOR	Player
NORMAL CASE	• Connection has been established between the phones • Application mode has been selected i.e. racket or display mode and • Start game option has been selected at startup
ALTERNATE CASE	• N/A
PRE-CONDITION	Connection has been established between the phones
POST CONDITION	The game has been started
EXTENDS	N/A
INCLUDE	N/A
ASSUMPTIONS	The player is displayed with both the modes to select one of them. After that start game can be selected.

4.2.5.3 Use Case 3

USE CASE NAME	Select App mode
ACTOR	Player
NORMAL CASE	• The player is prompted to select a game mode option. • From the two choices of racket mode and display mode, he selects one.
ALTERNATE CASE	N/A
PRE-CONDITION	Phones are connected to the network and the application has been launched.
POST CONDITION	The player has successfully selected a game mode and can start playing
EXTENDS	N/A
INCLUDE	N/A
ASSUMPTIONS	The player has been prompted with the game modes and a desired mode has been selected

4.2.5.4 Use Case 4

USE CASE NAME	Pause game
ACTOR	Player
NORMAL CASE	The user pauses the game by a swipe gesture on the screen of racket phone that pauses the game.
ALTERNATE CASE	The game does not respond to the swipe gesture and does not get paused.
PRE-CONDITION	The user has started playing the game.
POST CONDITION	The game is paused and options in the pause menu are displayed.
EXTENDS	N/A
INCLUDE	Pause menu appears.
ASSUMPTIONS	By pausing the game during gameplay, the game is successfully paused and pause options are displayed.

4.2.5.5 Use Case 5

USE CASE NAME	Pause menu appears
ACTOR	Player
NORMAL CASE	Two choices appear as a result of pausing the game
ALTERNATE CASE	N/A
PRE-CONDITION	The user pauses the game by a swipe gesture on the racket phone.
POST CONDITION	Two options of "Resume" and "Quit" are displayed
EXTENDS	• Resume • Quit
INCLUDE	N/A
ASSUMPTIONS	The player pauses the game and can select any one of the options displayed in the pause menu

4.2.5.6 Use Case 6

USE CASE NAME	Resume
ACTOR	Player
NORMAL CASE	At the pause menu, user selects the Resume option and the game continues from the same state in which it was suspended.
ALTERNATE CASE	N/A
PRE-CONDITION	Pause menu is appeared on the display phone screen
POST CONDITION	The game continues from the suspended state
EXTENDS	N/A
INCLUDE	N/A
ASSUMPTIONS	Player selects the resume option and the game continues successfully from the same suspended state

4.2.5.7 Use Case 7

USE CASE NAME	Reset Gyroscope
ACTOR	Player
NORMAL CASE	The player changes his position and performs a long-tap gesture on the racket phone screen which resets the gyroscope according to the new position
ALTERNATE CASE	N/A
PRE-CONDITION	Player is playing the game.
POST CONDITION	The gyroscope has been reset according to the position changed by the player.
EXTENDS	N/A
INCLUDE	N/A
ASSUMPTIONS	The player changes his position by turning right or left and long-taps the racket phone screen which resets the gyroscope and new position is acquired

4.2.5.8 Use Case 8

USE CASE NAME	Change in scope of view
ACTOR	Player
NORMAL CASE	The player moves his head while playing the game and due to the head gesture the scope of view is changed
ALTERNATE CASE	N/A
PRE-CONDITION	The player is playing the game
POST CONDITION	The scope of view is changed according to the head gesture made by the player
EXTENDS	N/A
INCLUDE	N/A
ASSUMPTIONS	The player moves his head and the and the scope of the display is rendered and changed accordingly

4.2.5.9 Use Case 9

USE CASE NAME	Change in racket orientation
ACTOR	Player
NORMAL CASE	The player moves the racket phone and its change in orientation can be seen in the display phone
ALTERNATE CASE	N/A
PRE-CONDITION	The phones are connected and the game is started
POST CONDITION	The racket orientation is changed and can be seen in the display phone
EXTENDS	N/A
INCLUDE	N/A
ASSUMPTIONS	The change in racket movement results change in its orientation on the display phone

4.2.5.10 Use case 10

USE CASE NAME	Take a shot
ACTOR	Player
NORMAL CASE	The player performs a swing action on the racket phone and the racket's orientation changes leading to a shot played.
ALTERNATE CASE	N/A
PRE-CONDITION	The game has started and the player is about to hit the ball
POST CONDITION	The racket moves and the player plays a shot
EXTENDS	• Change in velocity and direction of the ball • Miss
INCLUDE	N/A
ASSUMPTIONS	The player swings the racket and a shot is played

4.2.5.11 Use Case 11

USE CASE NAME	Change in velocity and direction of the ball
ACTOR	Player

NORMAL CASE	The player plays a shot and the ball hits the racket resulting in a change in ball's direction and velocity
ALTERNATE CASE	N/A
PRE-CONDITION	The player takes a shot
POST CONDITION	The ball has a change in direction and velocity and moves towards the opposite end
EXTENDS	N/A
INCLUDE	N/A
ASSUMPTIONS	The player plays a shot and hits the ball

4.2.5.12 Use Case 12

USE CASE NAME	Miss
ACTOR	Player
NORMAL CASE	The player plays a shot and the ball does not hit the racket, rather bounces off the table resulting in a change in score
ALTERNATE CASE	N/A
PRE-CONDITION	The player plays a shot.
POST CONDITION	The shot misses and results in a change in score
EXTENDS	N/A
INCLUDE	N/A
ASSUMPTIONS	The player swings the racket according to ball's location but misses to hit it

4.2.5.13 Use Case 13

USE CASE NAME	Quit
ACTOR	Player
NORMAL CASE	The player selects the quit option from the pause menu and the game quits
ALTERNATE CASE	The player selects resume and continues playing

PRE-CONDITION	The game has started and the player pauses the current game state
POST CONDITION	The game exits
EXTENDS	N/A
INCLUDE	N/A
ASSUMPTIONS	The player selects quit and the game quits

4.2.6 Implementation View (Class Diagram)

These diagrams will show the main classes of our systems. It will also show the attributes and functions associated with each class. Class diagrams are integral for the description of low level components of the software that include data storage and state details. Classes provide the means for the system to perform its functions.

4.2.6.1 Class Diagram

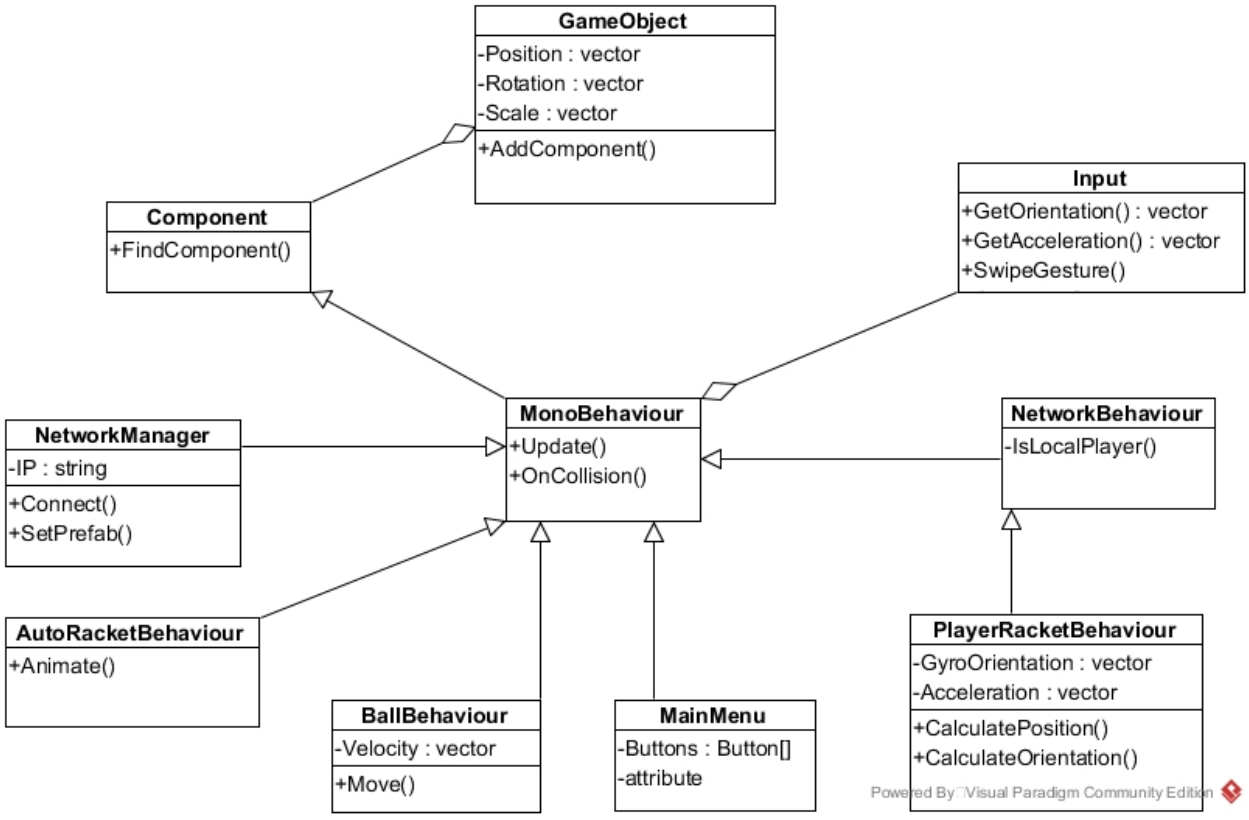

Figure 4.6 Class Diagram

4.2.6.2 Class Diagram Description

Name	Description
GameObject	This class is responsible for defining a unity game object. This class contains a series of properties which can be used to customize a unity game object. These properties are known as components. An example of a game object in this game is a racket. A simple example of a component is a 'transform' which defines a game object's position, rotation and scale.
Component	This class is a base class for everything that can be attached to a game object. This class provides the facility to store all the component connected to a game object in as a single structure.
MonoBehaviour	This class inherits form a component. This means that it can be attached to a game object. This class defines the behavior of the game object it is attached to. Multiple MonoBehaviors can be attached to a game object.
NetworkBehaviour	This class is an extension of class MonoBehavior. It is specialized for those objects whose transforms are to be synchronized over the network.
PlayerRacketBehaviour	This class extend from MonoBehavior it is responsible for the movement of player's racket in the game scene. It processes input for sensors such as gyroscope and accelerometer and translates these raw inputs to the rackets animation.
AutoRacketBehaviour	This class extends from NetworkBehavior. It is responsible for the animation of auto player's racket. It animates the racket to create a realistic simulation of player's racket while taking a shot.
NetworkManager	This class extends from MonoBehavior. It is responsible for the synchronization of selected game object's rendering on all devices connected to a networked game.
Input	This class responsible for exposing all kinds of user input for example accelerometer data, gyroscope data, touch input.

BallBehaviour	This class Extends from MonoBehavior. This class is responsible for the animation of ping pong ball. It processes the characteristics of collisions with racket and table and translates this to the new direction and velocity of ball.
MainMenu	This class is responsible for providing the user with a menu to navigate through different options in the game. It displays different buttons depending on the situation will trigger events accordingly.

4.2.7 Sequence Diagram

Sequence diagrams show how different objects are involved in the completion of a functionality of the system. They have a unique format that allows the reader to see how many objects are used and for how long for the completion of a system requirement.

A Sequence diagram is an interaction diagram that shows how processes operate with one another and in what order. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

4.2.7.1 App mode and connection

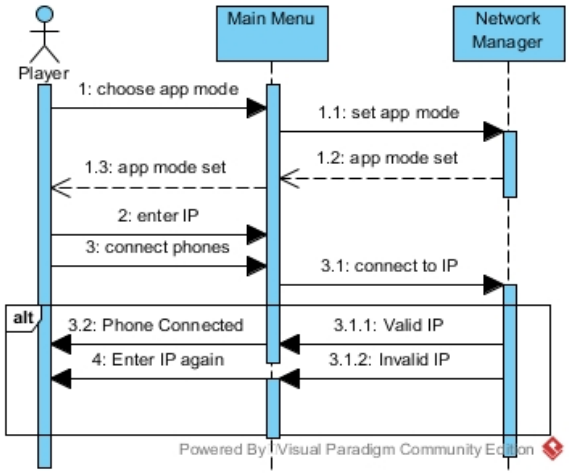

Figure 4.7 Sequence – App mode and connections

4.2.7.2 Start game

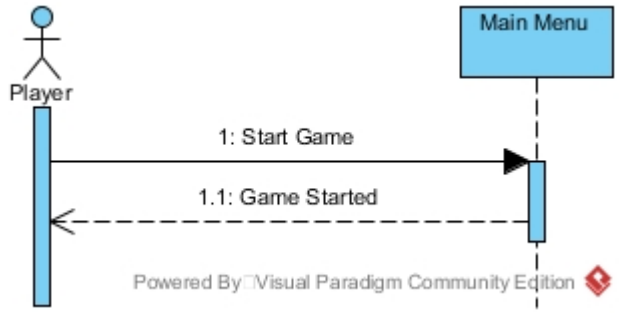

Figure 4.8 Sequence – Start game

4.2.7.3 Change racket orientation

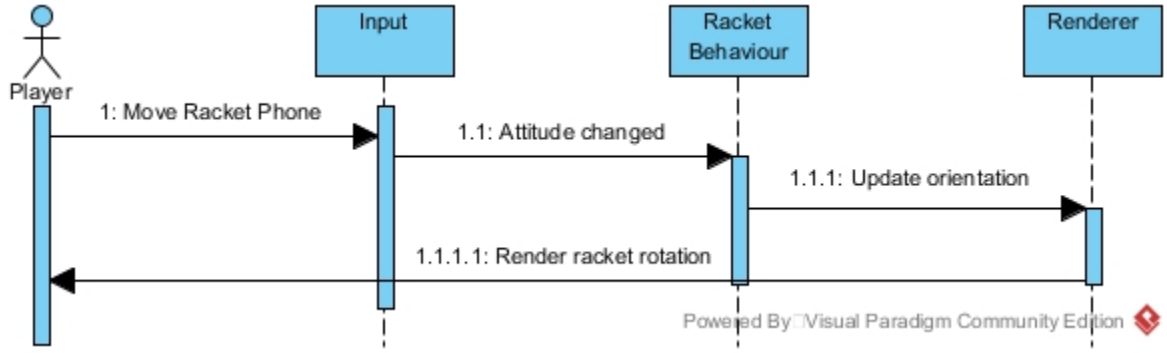

Figure 4.9 Sequence – Change racket orientation

4.2.7.4 Change scope of view

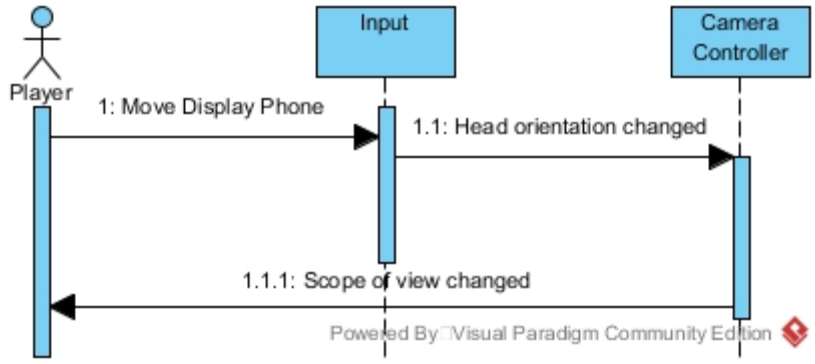

Figure 4.10 Sequence – Change scope of view

4.2.7.5 Serve from player

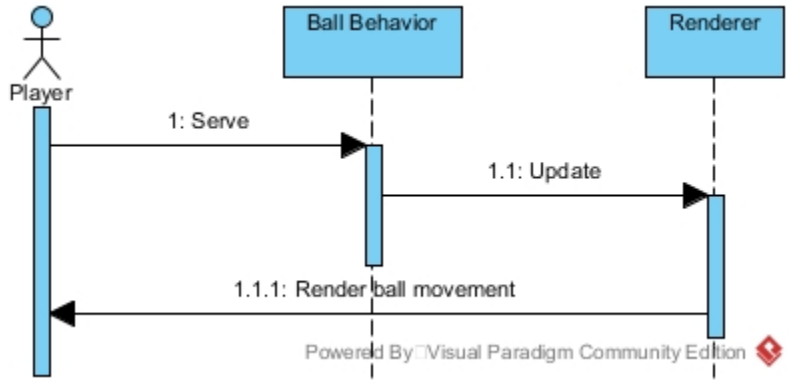

Figure 4.11 Sequence – Serve from player

4.2.7.6 Reset gyroscope

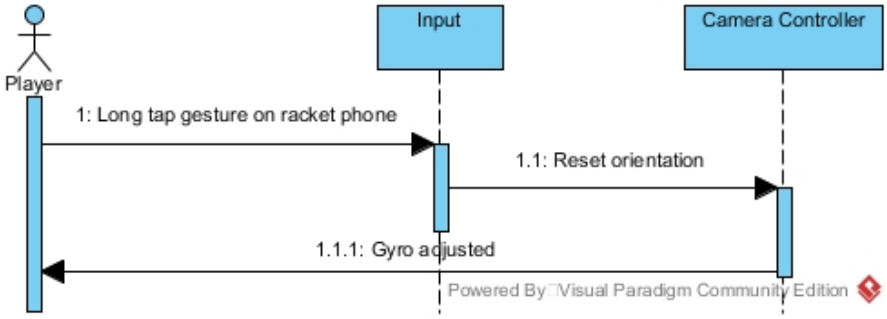

Figure 4.12 Sequence – Reset gyroscope

4.2.7.7 Pause game

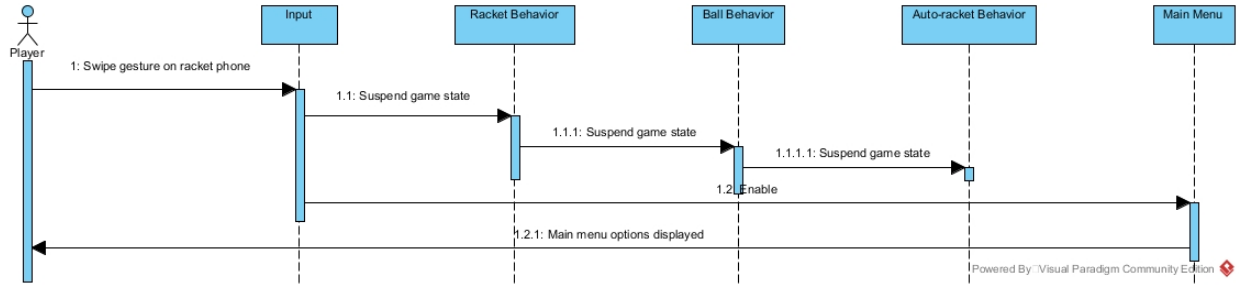

Figure 4.13 Sequence – Pause game

4.2.7.8 Traversal through pause menu options

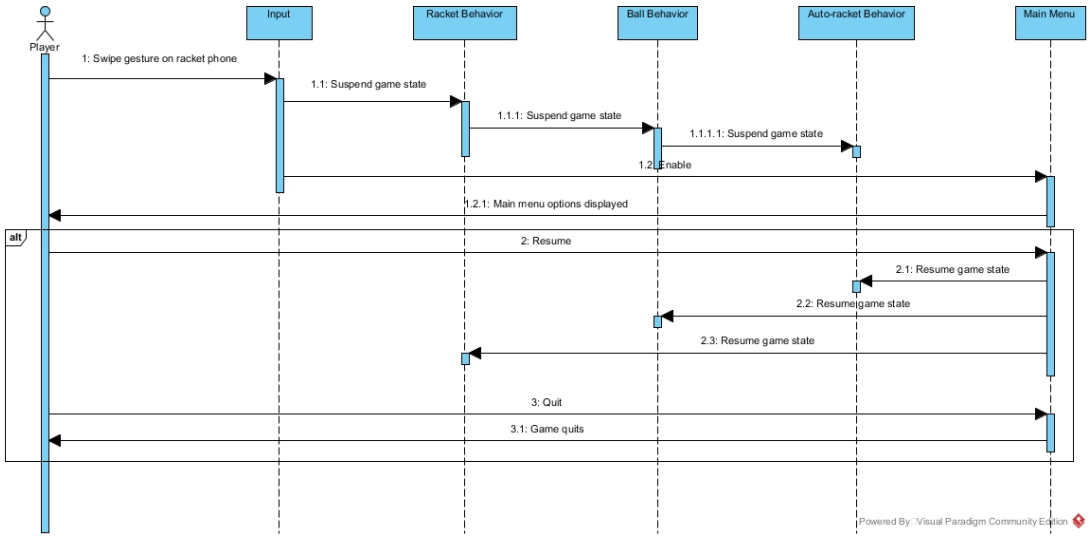

Figure 4.14 Sequence – Traversal

4.2.7.9 Shot played

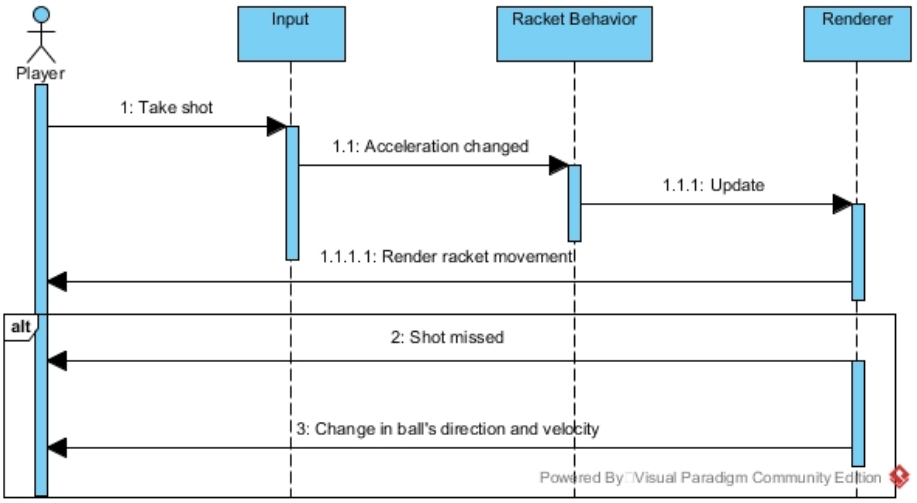

Figure 4.15 Sequence – Shot played

4.2.8 Activity Diagram

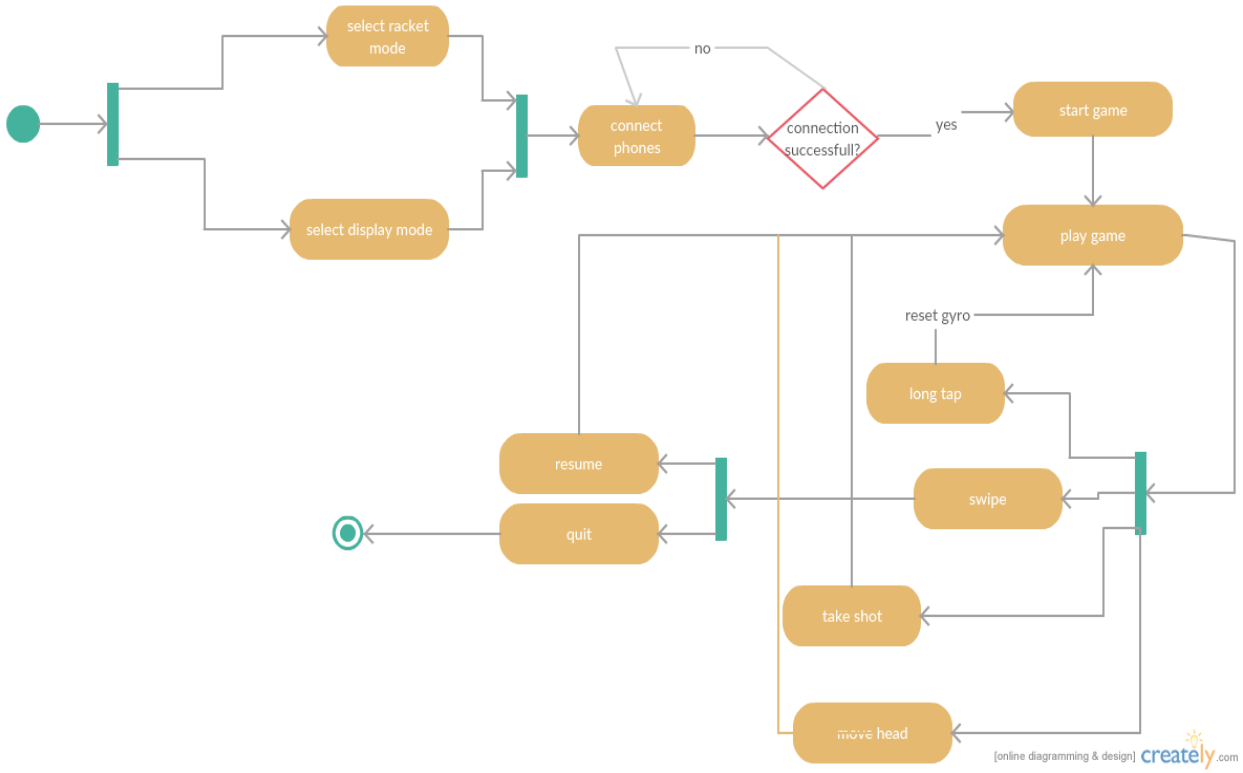

Figure 4.16 Activity Diagram

4.2.9 State Transition Diagram

This diagram will show the main states of the application and how one state will transit to another.

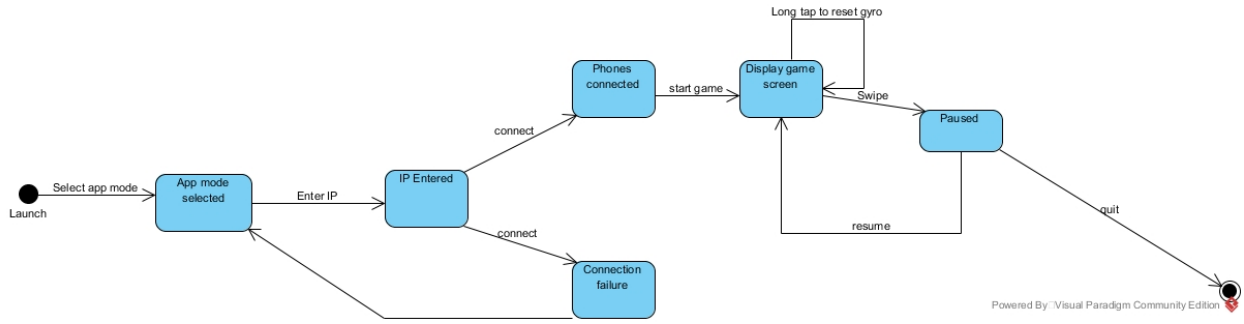

Figure 4.17 State Transition Diagram

4.2.10 User Interface

The game graphical user interface will look similar to these snapshots.

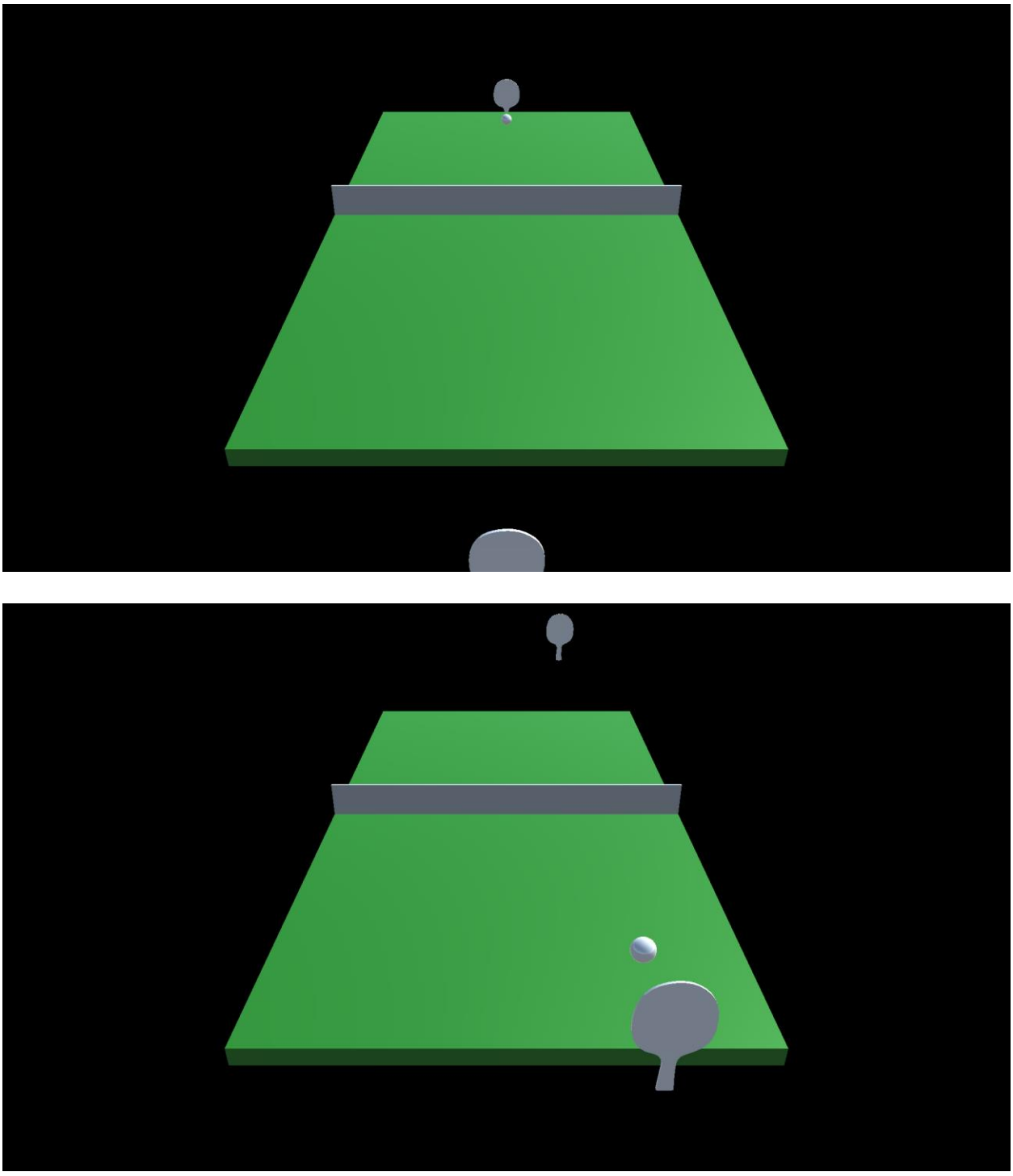

Figure 4.18 User Interface

4.3 Detailed Description of Components

4.3.1 User input

Identification	**Name**: User input **Location**: Application layer
Type	Component
Purpose	• The purpose of this component is to receive user input. • The input through touch screen of racket phone will be processed by this component and will be further sent to unity game engine. • The user head gestures and racket orientation will be received by this component.
Function	**What the component does:** Detect input in the form of phone orientation, head movement, hand gesture and touch inputs.
Subordinates	**Constituents of the component:** The component has no sub-components. **Functional Requirements:** **Requirement 1:** The user will be able to view the play environment in 360 degrees by looking around. **Requirement 2:** The user will be able to simulate the movement of racket as per the movement of the racket phone.
Dependencies	**Components using this component:** The sub component input events of the unity component will get its input from this module. This input will be further processed by input events component.
Interfaces	The external hardware interfaces interacting with this component will be: • Android smartphone **Error messages:** Connection not established (Invalid IP address).
Resources	The resources used by this component are touch screen (for touch inputs) and sensors (for racket movement).
Processing	The processing required for this component is receiving the player's input and giving this input to the input event module of the unity game engine
Data	Player inputs (touch, swipe, racket swings).

4.3.2 Graphical User Interface

Identification	**Name:** GUI (Graphical User Interface) **Location:** Application layer
Type	Component
Purpose	To display the game.
Function	**What the component does:** • The complete game environment is displayed by this component.
Subordinates	**Constituents of the component:** GUI basically contains all the game objects which includes: • camera • rackets • ball • table • scoreboard **Functional Requirements:** **Requirement 1:** The user will be able to see a play area having a table-tennis table, rackets, scoreboard and a ball. **Requirement 2:** The user will be able to connect both phones before wearing the VR headset.
Dependencies	GUI is dependent on unity engine. The rendering component of the unity will be providing input to the GUI.
Interfaces	**External interface requirement for GUI** An output display screen which it will be using to display all the objects created by the unity engine and all the menu and app mode screen.
Resources	• It will be using graphics engine of the system as per unity requirements. • Android smartphone screen
Processing	The processing required of this component is to respond and display results of player input during the game and also during navigation.
Data	User input (touch, hand gestures, head gestures)

4.3.3 Unity Engine:

Identification	**Name:** Unity engine **Location:** Processing layer
Type	Module
Purpose	Unity is a game development ecosystem. It is a powerful rendering engine fully integrated with complete set of intuitive tools and rapid workflows to create interactive 3D and 2D content. Unity built in physics engine provides component that handles physics simulation with just a few parameters setting. Physics can be controlled from scripts
Function	**What the component does:** The main functions of this module are as follows: • Handle game physics • Game play rendering • Manage game resources on system • Manage all other major components of game
Subordinates	**Constituents of the component:** The other components using this component are: • GUI • Player Input • Network Manager **Functional requirements:** **Requirement 1:** The player is able to connect devices. **Requirement 2**: The system will share the sensor's data over wireless media from racket phone to the display phone. **Requirement 3:** The application allows the player to reset gyro. **Requirement 4:** The application allows the player to pause game.
Dependencies	**Components using this component:** Player input controls GUI rendering
Interfaces	The external interfaces interacting with unity game engine are • touch screen and sensors to get input data

	• Speakers
Resources	The resources used by this module are RAM Graphic memory CPU usage sensors
Processing	The processing done by this module is that it: • Initiates game, game objects and all required global variables • Allocates required system resources for game • Manages memory requirements
Data	Float values, integer values, strings

4.4 Reuse and Relationships to Other Products

Air Smash (a real-time 3D table tennis game) is completely a new project. It is not an extension of already existing projects. However, materials already available on internet are being used to aid the modelling and enhance the graphics of the game. Special tools including 'Unity 3D' are used in our project.

Existing and available models are not used in our project as our project is according to our own requirements and choices.

4.5 Design Decisions and Tradeoffs

We have used a client server architecture for this game. The reasons are as follows.

Two devices:

The game uses two android phones. One acts as a display and the other as a racket. The communication between these phones is done via Wi-Fi because it is wireless and is faster than Bluetooth. Hence, we have chosen a client server architecture to implement this networked game.

Central storage:

The display phone provides the user with a view of the play environment and hence all the game logic is running on the display phone. The racket phone is merely being used as an input device

to fetch accelerometer and gyroscope data. This imbalance in responsibility led us to client server architecture.

Future Prospects:

Later, this project is expected to be a multiplayer game for which multiple devices will need to be connected. Keeping in view this future need we decided to implement our game with client server architecture.

Chapter 5: Project Test and Evaluation

5.1 Introduction

This document proposes the plan used for testing the gameplay of Air Smash (a real-time 3D table tennis game). The test plan proposed will ensure that the game will be working in a manner which was intended.

The purpose of this document is to describe the tests that will be conducted on the implementation of Air Smash. The aim is to check that the modules (system features) such as

- Displaying the gameplay
- Connectivity configuration between devices
- Simulation of racket
- Communication between the devices
- Accurate gameplay and
- Standard game rules

are performing the required operations. The user should be able to experience a 3D gaming environment error-free.

This document further includes the plan, approach, test items and the test deliverables. The pass/fail criteria are also defined for the items.

5.2 Test Items

The test items selected for testing include the following

- Performance
- Interface
- User control

5.2.1 Features to be Tested

The features of our game include the functionality mentioned in the use cases. Following features are to be tested keeping in view the test items and system features aforementioned

- Connection between the phones
- Selection of App mode

- Change in scope of view
- Change in racket orientation
- Change in velocity and direction of the ball after being hit
- Reset gyroscope after changing the position by the player

5.3 Approach

Functional Testing will focus on each use case that is included in the version currently being worked on. Testing will mainly consist of execution of test cases written to address the gap identified. It will focus on inputs, outputs and system changes due to the actions.

The testing strategy for Air Smash will be alpha testing. Our project is in modules so we will start the testing phase by testing the modules separately and then step by step integrating modules to test them with each other i.e. integration testing and then the complete application is tested as a whole in system testing. Black Box testing technique will be used for testing functionality of each module.

5.4 Pass/Fail Criteria for Test Items

The criteria are as follows

- The pre-conditions are met
- Inputs are carried out as specified
- Test case will pass if it produces the desired output for a specified input
- Test will fail otherwise

5.5 Test Suspension Criteria

Testing procedure will be suspended whenever a defect is found that restricts further testing. A corrective measure will be applied depending upon the criticality of the defect and testing will be resumed.

Efforts have been made to remove all and every chance of failure but there are certain unpredictable factors such as network issues, corrupt input data, or system failure that may lead to some issues. Error handling is applied more deeply to cover all these issues but unforeseen circumstances may happen.

5.6 Test Deliverables

5.6.1 Testing tasks

- Develop Test Cases.
- Execute tests based on the test cases developed.
- Report defects during tests if any.
- Manage the changes made after testing.

5.6.2 Test cases

5.6.2.1 Displaying the Gameplay

Test Case ID	TC 1
Test Procedure	The application is launched in the Display mode
Testing Technique used	Black Box Testing
Preconditions	The application has successfully launched
Expected output	User is able to see a 3D playing environment of a table tennis game.
Actual output	Playing environment is displayed
Status	PASS

5.6.2.2 Connectivity Configuration between Devices

Test Case ID	TC 2
Test Procedure	Both phones have the application launched and network configuration is done for establishing connection
Testing Technique used	Black Box Testing
Preconditions	Network configuration is done for establishing connection
Input values	Not sprcified
Expected output	Connection has been established between the phones
Actual output	Connection has been established between the phones
Status	PASS

5.6.2.3 Change in Scope of View

Test Case ID	TC 3
Test Procedure	The player moves his head while playing the game to look around in the playing environment
Testing Technique used	Black Box Testing
Preconditions	The player is playing the game
Input values	Gestures due to head movement
Expected output	The view spans left, right, up and down as player looks left, right, up, and down respectively

Actual output	The view spans left, right, up and down as player looks left, right, up, and down respectively
Status	PASS

5.6.2.4 Change in Racket Orientation

Test Case ID	TC 4
Test Procedure	The player changes orientation of the racket phone
Testing Technique used	Black Box Testing
Preconditions	The player is playing the game
Input values	Gestures on the racket phone due to hand movements
Expected output	The orientation of racket is changed and can be seen in the display phone
Actual output	The orientation of racket is changed and can be seen in the display phone
Status	PASS

5.6.2.5 Change in Velocity and Direction of the Ball

Test Case ID	TC 5

Test Procedure	The player plays a shot and the ball hits the racket
Testing Technique used	Black Box Testing
Preconditions	The player is playing the game
Input values	The racket is moved to play a shot according to the incoming position of the ball
Expected output	The ball has a change in direction and velocity according to the orientation and force of the racket at the time of collision.
Actual output	The ball has a change in direction and velocity according to the orientation and force of the racket at the time of collision.
Status	PASS

5.6.5.6 Reset Gyroscope

Test Case ID	TC 6
Test Procedure	The player looks in the direction he wishes to face while playing the game and performs a long-tap gesture on the racket phone screen
Testing Technique used	Black Box Testing
Preconditions	The player is playing the game
Input values	Long tap gesture is done on the screen of racket phone

Expected output	The gyroscope has been reset according to the direction changed by the player.

5.6.5.7 Gameplay according to Rules

Test Case ID	TC 7
Test Procedure	The gameplay is compared with respect to the standard game rules of table tennis
Testing Technique used	Black Box Testing
Preconditions	The player is playing the game
Input values	The player plays a shot • The ball is hit correctly • The player misses the ball • The ball is hit but leaves the table • The ball is hit incorrectly
Expected output	The behavior of the ball after the courses of actions aforementioned, will be according to game rules.

5.6.5.8 Scoreboard

Test Case ID	TC 8

Test Procedure	There will be a change in points of the player as the game proceeds.
Testing Technique used	Black Box Testing
Preconditions	The player is playing the game
Input values	The player is playing the game
Expected output	The scores will be displayed in the scoreboard and every time a change in the score will be notified to the player

5.7 Environmental Needs

5.7.1 Hardware

- Two android smartphones with gyroscope and accelerometer.
- Virtual Reality headset.

5.7.2 Software

- Android operating system 4.1 (Jellybean) or above.

5.8 Responsibilities, Staffing and Training Needs

5.8.1 Responsibilities

All developers of the project are responsible for the completion of all components testing and integration testing tasks.

5.8.2 Staffing and Training Needs

Basics knowledge of testing strategies and techniques is needed for the testing of the project.

Techniques such as Black Box testing, integration testing should be known to developers.

All the developers will be testing each other's work and will be actively participating in the development and testing of the project simultaneously.

Chapter 6: Future Work

6.1 Introduction:

As mentioned earlier in the introductory part of this thesis, this project has been developed with goals to open the community to the vast possibilities of Virtual Reality (VR) and motion sensing achieved via Smartphones. Therefore, while this project can be continued as is to incorporate additional features in the game, it is also encouraged to take it as a proof of concept and make use of the finding in multiple fields. The applications are endless. Some suggestions are made bellow regarding additional features that could be worth future attention. Also, some ideas are put forth the readers about how different applications of VR can be replicated using Smartphones.

6.2 Additional Features:

6.2.1 Multiplayer:

In Future Developments could be made so that more than one people can play the same simultaneously. Two players could pay against each other or maybe four players could play in teams of two. These players could either be in the same room or may be far apart connected through the internet.

6.2.2 Multiple Scenes:

The most attractive thing about VR is that we are able to experience something that is not there in reality. We could also let the players choose an environment of their liking.This way the people could relate it this game in multiple ways. For example, we could give the user the choice of playing indoors as well the outdoors. We could get creative and even create something the user has never seen before, some imaginary place.

6.2.3 Difficulty Level:

Another important feature to keep the players from getting bored would be introduction to difficulty levels. In this way the game can be challenging for experts and easy for starter at the same time.

6.3 Other Areas to Explore:

6.3.1 Medicine:

VR can be used to train medical students. Models can be made with high quality graphics to teach the students about surgery. It could be an app that students could download to their phones and use it to learn about how the body works while experiencing it around them.

6.3.2 Universal Controller:

A universal app could be developed so that no matter what VR application is being developed, it can easily be integrated to this universal app to provide motion sensing controls. This app could have an API that the developers could use while developing their own Virtual Reality games or applications. Support for apple TV and Android TV can also be added.

6.3.3 Military Training:

Military personal could be trained under carefully architected situations and could be taught the details of protocol and its benefits. Every possible scenario could be simulated to tell the military students how to react to the most unexpected situations.

Bibliography

Unity 3D Development Home

- http://unity3d.com/
- http://unity3d.com/learn/documentation

Autodesk

- http://www.autodesk.com/products/

3D Studio Max

- http://www.autodesk.com/products/autodesk-3dstudiomax/overview

Adobe Photoshop

- http://www.adobe.com/products/photoshopfamily.html
- http://www.photoshop.com

Appendix

Game rules

Table

- The table shall be in surface rectangular, 274 cm. (9 ft.) in length, 152.5 cm. (5 ft.) in width. It shall be supported so that its upper surface, termed the playing surface, shall lie in a horizontal plane 76 cm. (2 ft. 6 in.) above the floor.
- The playing surface shall be dark colored and matt, with a white line 2 cm. (3/4 inch) wide along each edge.
- For doubles, the playing surface shall be divided into halves by a white line 3 mm. (1/8 in.) wide, running parallel with the side lines.
- The playing surface shall be considered to include the top edges of the table, but not the sides of the table.

Net

- The playing surface shall be divided into two "courts" of equal size by a vertical net running parallel to the end lines.
- The net assembly shall consist of the net, its suspension, and the supporting posts, including the clamps attaching them to the table.

Racket

- The racket may be of any size, shape, or weight but the blade shall be flat and rigid.
- The surface of racket should be matt black on one side and bright red on the other.

Service

- Service must be done diagonally always from one player’s right-hand half to the opponent’s right hand half.
- Service changes after five serves done by one player.

Point

- If his opponent fails to make a good service.
- If his opponent fails to make a good return.
- If, after a good service or a good return has been made, the ball touches anything other than the net assembly before being struck by the opponent.
- If the ball passes beyond his end line without touching his court, after being struck by his opponent.
- If the ball is obstructed by the net assembly other than service or return.
- If his opponent strikes the ball twice successively.

- If his opponent strikes the ball with a side of the racquet blade having an illegal surface.
- If his opponent, or anything he wears or carries, moves the playing surface.
- If his opponent, or anything he wears or carries, touches the net assembly.
- If his opponent's free hand touches the playing surface.
- If, in doubles, his opponent strikes the ball out of sequence established by the first server and first receiver.
- If the umpire assesses a penalty point against his opponent.

Pseudo Code

AutoRacketBehaviour

Update ()

detectBall()

changePosition()

changeOrientation()

PlayerRacketBehaviour

update()

listenInput()

changePosition()

changeOrientation()

BallBehaviour

onCollision()

if (collider==table)

bounce()

if(collider==playerRacket)

detectForce()

changeVelocity()

if(collider==autoracket)

changeVelocityRandomly()

MainMenu

displayButtons()

connect()

setDisabled()

pause()

resume()

Glossary

VR: Virtual Reality

VR headset: A device assisted by a smartphone used to provide VR feature e.g. Google Cardboard

UI: User Interface

SDK: Software Development Kit

2D: 2-Dimensional

3D view: 3-Dimensional view

Availability: Present and ready for use; at hand; accessible.

Constraint: A restriction that is imposed on the choices available to the developer for the design and construction of a product.

API: Application Programming Interface

OS: Operating System

External interface requirement: A description of an interface between a software system and a user, another software system, or a hardware device.

Feature: A set of logically related functional requirements that provides a capability to the user and enables the satisfaction of a business objective.

Functional requirement: A statement of a piece of required functionality or a behavior that a system will exhibit under specific conditions.

Hardware: A computer and the associated physical equipment directly involved in the performance of data-processing or communications functions.

Implementation: Execution of a plan, idea, model, design, specification, standard, algorithm, or policy.

Interface: A point where two systems, subjects, organizations, etc., meet and interact.

Nonfunctional requirement: A description of a property or characteristic that a software system must exhibit or a constraint that it must respect, other than an observable system behavior.

Operating Environment: The circumstances surrounding and potentially affecting something that is operating.

Operating System: A collection of software that manages computer hardware resources and provides common services for computer programs.

Procedure: A written description of a course of action to be taken to perform a given activity, describing how the activity is to be accomplished.

Process: A sequence of activities performed for a given purpose. A process description is a documented definition of those activities.

References: List of any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.

Requirement: A statement of a customer need or objective, or of a condition or capability that a product must possess to satisfy such a need or objective. A property that a product must have to provide value to a stakeholder.

Requirements specification: See software requirement specification and specification, requirement.

Scope: The portion of the ultimate product vision that the current project will address. The scope draws the boundary between what's in and what's out for the project.

Software requirements specification: A collection of the functional and nonfunctional requirements for a software product.

Modifiability: The effort required to make changes in the software

Portability: The effort required to move the software to a different target platform

Reliability: dependable, how often the software fails.

Architecture: The complex or carefully designed structure of something.

Activity: A condition in which things are happening or being done.

Component: A part or an element of a larger whole.

Breakdown: An explanatory analysis of something.

Class: A description of a set of objects having common properties and behaviors, which typically correspond to real-world items (persons, places, or things) in the business or problem domain.
Dependency: A reliance that a project has on an external factor, event, or group outside its control.

Feature: A set of logically related functional requirements that provides a capability to the user and enables the satisfaction of a business objective.

Functional requirement: A statement of a piece of required functionality or a behavior that a system will exhibit under specific conditions.

Hardware: A computer and the associated physical equipment directly involved in the performance of data-processing or communications functions.

Implementation: Execution of a plan, idea, model, design, specification, standard, algorithm, or policy.

Interface: A point where two systems, subjects, organizations, etc., meet and interact.

Process: A sequence of activities performed for a given purpose.

References: List of any other documents or Web addresses to which this document refers.

Scope: The portion of the ultimate product vision that the current project will address. The scope draws the boundary between what's in and what's out for the project.

Transition: The process or a period of changing from one state or condition to another.

User: A customer who will interact with a system either directly or indirectly.