VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi-590018, Karnataka, INDIA

PROJECT REPORT
on

"VRNAV: POSITIONAL TRACKING FOR MOBILE BASED VIRTUAL REALITY SYSTEMS"

Submitted in partial fulfillment of the requirements for the VIII Semester

Bachelor of Engineering
IN
COMPUTER SCIENCE AND ENGINEERING

For the Academic year
2016-2017
BY

BHARAT GOGINENI	**1PE10CS021**
SIDDHANT PRIYADARSHI	**1PE12CS156**
S SAMPATH	**1PE13CS131**
VAISHAK R VELLORE	**1PE13CS174**

Under the Guidance of

Dr. Sarasvathi V
Associate Professor, Department of CSE
PESIT Bangalore South Campus

Department of Computer Science and Engineering
PESIT BANGALORE SOUTH CAMPUS
Hosur Road, Bengaluru -560100

PESIT BANGALORE SOUTH CAMPUS

Hosur Road, Bengaluru -560100

Department of Computer Science and Engineering

CERTIFICATE

Certified that the Project work entitled ***"VRNav: Positional Tracking for Mobile Based Virtual Reality Systems"*** *is a bonafide work carried out by* ***Bharat Gogineni, Siddhant Priyadarshi, S Sampath, Vaishak R Vellore*** *bearing USN* ***1PE10CS021, 1PE12CS156, 1PE13CS131 and 1PE13CS174 respectively****, students of* ***PESIT Bangalore South Campus*** *in partial fulfillment for the award of* ***Bachelor of Engineering*** *in* ***Computer Science and Engineering*** *of the* ***Visvesvaraya Technological University****, Belagavi during the year 2016-2017. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.*

Signatures:

Project Guide	Head Of Department Of CSE	Principal
Dr. Sarasvathi V Associate Prof, Dept. of CSE PESIT-BSC, Bengaluru	**Prof. Sandesh B J** Associate Prof and HOD, Dept. of CSE, PESIT-BSC, Bengaluru	**Dr. Suryaprasad J** Director/Principal, PESIT-BSC, Bengaluru

External Viva

Name of the Examiners	**Signature with date**
1.	
2.	

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without mentioning the people who made it possible, whose constant guidance and encouragement crowned our effort with success.

We are indebted to our Guide, **Dr. Sarasvathi V**, Associate Professor, Department of Computer Science and Engineering, PESIT Bangalore South Campus, who has not only coordinated our work but also given suggestions from time to time.

We are also extremely grateful to our Project Co-ordinators, **Dr. Snehanshu Saha**, Professor, **Mrs. Vandana M L**, Assistant Professor, Department of Computer Science and Engineering, PESIT Bangalore South Campus, for their constant support and advice throughout the course of preparation of this document.

We are greatly thankful to **Mr. Sandesh B J**, Associate Professor and HOD, Department of Computer Science and Engineering, PESIT Bangalore South Campus, for his able guidance, regular source of encouragement and assistance throughout this project.

We would like to express our immense gratitude to **Dr. Suryaprasad J**, Director and Principal, PESIT Bangalore South Campus, for providing us with excellent infrastructure to complete our project work.

We gratefully acknowledge the help lent out to us by all faculty members of the Department of Computer Science and Engineering, PESIT Bangalore South Campus, at all difficulty times. We would also take this opportunity to thank our college management for the facilities provided during the course of the project. Furthermore, we acknowledge the support of our family and friends.

Bharat Gogineni

Siddhant Priyadarshi

S Sampath

Vaishak R Vellore

ABSTRACT

Virtual Reality is currently a trending area of research and experimentation for both the industry and the academia. With the introduction of VR headsets like Google Cardboard, HTC Vive and Samsung Gear VR, anyone with a smartphone today can experience the magic of VR. Any good VR experience must allow a user to both navigate and interact with objects within the virtual world. However, neither of these two goals is easily achievable. Often, the use of external controllers and sensors reduces both immersion and freedom. The primary purpose of this project is to build a framework that allows users to both navigate and interact with objects in virtual worlds, without having to use a controller or any other similar touch input based mechanism used by existing implementations, while keeping in mind two important points. First, the setup must be designed in such way that these goals can be achieved using only a smartphone and a VR head mounted display, with no other external hardware. Second, the setup must be kept as cost effective as possible. Thus, anyone, from anywhere, with a smartphone and a low-cost Google Cardboard headset should be able to experiment with VR, and deploy our framework in any virtual world that they may design and build.

.

CONTENTS

Chapter 3

Chapter 4

Chapter 5

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Virtual Reality (VR) is defined as "an immersive and realistic simulation of a 360-degree three-dimensional environment, which has been created and built using interactive software and hardware, and controlled by the movement of the body", according to Wikipedia. VR is typically experienced through a head mounted display, that provides a visual output of the virtual world, and may also provide an audio output for the user to experience sound effects.

VR applications are usually designed for a specific use-case and include virtual worlds are modeled based on the real world. As an example, consider a VR app that allows the user to explore the Taj Mahal. The user would thus be able to get a first-hand experience of exploring the monument. These virtual worlds could also contain objects that users can interact with. In the case of the Taj Mahal example, there could be a camera that could be used to take pictures.

1.1 PURPOSE

The primary purpose of this project is to build a framework that allows users to both navigate and interact with objects in virtual worlds, without having to use a controller or any other similar touch input based mechanism used by existing implementations, while keeping in mind two important points. First, the setup must be designed in such way that these goals can be achieved using only a smartphone and a VR head mounted display, with no other external hardware. Second, the setup must be kept as cost effective as possible.

This project incorporates the concept of walking-in-place to achieve navigation, wherein the user jogs on the exact same spot where they are standing, without moving forward, or in any other direction. WIP is used to detect user activity, and effectively, the speed at which the user jogs in place is the speed at which the user moves in the virtual world. In addition to being able to navigate virtual worlds, the proposed framework implements a mechanism to allow users to interact with objects in the virtual world.

As a result, anyone, from anywhere in the world can virtually sightsee any other place, virtually play any sport or even virtually live another life, from their own room.

1.2 SCOPE

Since the primary goal of this project is to build a framework for VR systems, this project has a broad scope. This framework can be used to build VR applications that can simulate a wide spectrum of use-cases, ranging from learning to fitness to sightseeing, or even gaming. The framework has been written in such a way that developers can build on top of it, to implement additional functionality to enhance the framework's capabilities, thereby allowing a wide-range of applications. Also, given that VR is currently trending and is an active area of research, there is a potential for usage of this framework for research purposes.

1.3 DEFINITION, ABBREVATION AND ACRONYMS:

1.3.1 DEFINITIONS:

- **Unity:** Unity is a cross-platform game engine developed by Unity Technologies and used to develop video games for PC, consoles, mobile devices and websites, according to Wikipedia.

- **C#:** C# is one of many .NET programming languages. It is object-oriented and allows you to build reusable components for a wide variety of application types.
- **Blender:** Blender is a professional free and open-source 3D computer graphics software product used for creating animated films, visual effects, art, 3D printed models, interactive 3D applications and video games, as found on Wikipedia.
- **MATLAB:** MATLAB is a mathematical toolbox that supports a whole bunch of functions to perform various mathematical operations that primarily involve matrices.
- **Positional Tracking:** Positional tracking detects the precise position of the head-mounted displays, controllers, other objects or body parts within Euclidean space, as mentioned on Wikipedia. The goal is to track accurately how objects (like the head or the hands) move in real life in order to represent them inside VR.
- **Orientation Tracking:** Orientation tracking detects rotation of head-mounted displays about the three axes, also known as pitch, yaw and roll. It is precisely this type of tracking that allows users to look around in VR apps.

1.3.2 ACRONYMS:

- **VR:** Virtual Reality
- **DFD:** Data Flow Diagram
- **SDK:** Software Development Kit
- **WIP:** Walking in Place
- **WD:** Walk Detection
- **HMD:** Head-Mounted Display
- **VE:** Virtual Environment

1.4 LITERATURE SURVEY

Literature Survey is the most important step in any software development process. Before developing software, it is necessary to understand the domain, and then determine which operating system and language can be used for development. Various reference papers were looked into and modifications have been made accordingly. These papers helped us understand the current state-of-the-art systems and their drawbacks:

- **Sam Tregillus and Eelke Folmer, VR-STEP 'Walking-in-Place using Inertial Sensing for Hands Free Navigation in Mobile VR Environments' [1]:** Mobile based VR today is typically limited to using head tracking, which makes it difficult to perform complex tasks like navigation. As a result of this, immersion experience is lower, and simulation sickness is higher. They propose using WIP to perform navigation in such worlds. Based on their experiments, they discovered that using WIP did indeed reduce simulation sickness, while increasing the immersion factor by giving the player more freedom. However, their proposed approach limits the range of speeds at which users can move in the virtual world. As a result, after a particular threshold speed, no matter how much faster the user jogs, there would be no difference in speed of movement in the virtual world. Second, they do not discuss about any approach to interact with objects in virtual worlds, placing a strong emphasis on the navigation aspect alone. This paper also suggests some heuristics that can be used to evaluate VR systems.

- **Agata Brajdic, Robert Harle, 'Walk detection and step counting on unconstrained smartphones' [2]:**. The primary goals behind exploring this paper were to understand how pedometry (counting step while jogging/running/walking) works in the mobile context, and which algorithms would work best. Another goal was to understand whether the placement of smartphone at different parts of the body would affect the accuracy of pedometry. Based on experimentation, it was concluded that the use of standard deviation

thresholding (WD) and windowed peak detection (SC) algorithms were best, with error rates of less than 3%. Also, of the different placements of the smartphone on the body, only the back trouser pocket is found to degrade the step counting performance, resulting in undercounting for many algorithms.

- **Alistair Sutcliffe, Brian Gault, 'Heuristic evaluation of virtual reality applications' [3]:** The objective behind exploring this paper was to understand how VR systems are to be evaluated, since results can be measured qualitatively easily, than quantitatively. This paper presents a heuristic method for evaluating virtual environment (VE) user interfaces. Twelve heuristics are presented which address usability and presence issues. A few of these heuristics, combined with those proposed by **Tregillus et al.**, helped us evaluate our system.

1.5 EXISTING SYSTEM

Virtual Reality has been around for quite some time now and a lot of research and effort is being put into this field which is expanding at an alarming rate. If we survey existing VR systems, orientation tracking (support for 360 degree viewing of the virtual world) alone is supported. Positional tracking is not commonly implemented. Even in those systems that support navigation, external sensors or controllers are used to track body movement. Most existing implementations use such mechanisms for navigation primarily because when the user puts on their VR headset, they are no longer able to see the real world around them, thereby restricting the possibility for any movement. But this approach has its limitations, because the setup as a whole becomes complex and expensive, in addition to reducing the immersion experience. Now, think of way in which a person can actually walk in and around the virtual world, experiencing all this by whilst being in a single place?

1.6 PROPOSED SYSTEM

The key goal of the project is to build a system which allows navigation in VR worlds, based on a user's movements in the real world. As we have discussed in the previous section, existing systems lack this type of functionality. To achieve this goal, we use only a smartphone and a head mounted display, like Google Cardboard, keeping cost at a minimum. This effectively removes the need for any external sensors or even controllers for locomotion in the virtual world also making it easier to use. In addition to navigation, it is necessary to have some means to interact with objects in the VR world. We achieve this through head gestures. Thus, our framework essentially has four key modules; step detection, virtual locomotion, object interaction and head gesture recognition. Potentially, this framework can be used in a variety of VR applications like learning, fitness, E-commerce, to name a few. The project is described in detail in the following chapters which happens to put light onto some important points which will help in understanding the system as a whole.

1.7 SUMMARY

This chapter describes, in brief, the idea of the project. It begins with the explanation of the purpose of the project, the definitions of a few terms used in the document, the abbreviations used and the scope of the project. Later, we describe the literature survey and background preparations done to understand more about this project. Following that, we describe the existing system, its limitations and how it tries to improve the existing system. Finally, we explain the problem in hand for the system that is being designed

CHAPTER 2

SYSTEM REQUIREMENTS SPECIFICATION

A software requirements specification (SRS) is a comprehensive description of the intended purpose and environment for software under development. The SRS fully describes what the software will do and how it will be expected to perform.

2.1 OPERATING ENVIRONMENT

The users can focus on the specific system level design issues, without having to worry about low level design and infrastructure. The hardware and software requirements are as mentioned below:

2.1.1 HARDWARE REQUIREMENTS

Table 2.1: Hardware Requirements

Hardware Requirements
Smartphone, running Android 4.4 or above
Google Cardboard

2.1.2 SOFTWARE REQUIREMENTS

Table 2.2: Software Requirements

Software Requirements	
Operating System	Windows XP or above.
IDE Used	• Unity Daydream 5.4.2f2-GVR13. • Google VR SDK for Unity, v1.2 or above.
Programming Language	C#
Software	• MATLAB, a mathematical toolbox. • Blender, a 3D content creation platform. • Audacity, an audio processing tool.

2.2 FUNCTIONAL REQUIREMENTS

The functions of a system and its expected behavior in situations together constitute functional requirements. The functional requirements for our system are discussed below:

- **Step Detection:** The framework should accurately capture accelerometer reading values along the X, Y, Z axes on the smartphone and then process them to count the number of peaks, where each peak represented a step.

- **Virtual Locomotion:** The framework should effectively translate motion in the real world into motion in the virtual world, at a particular speed, based on the user's WIP speed.

- **Head Gesture Recognition:** The framework should detect gestures performed through head rotation, like nodding, to trigger appropriate actions.
- **Object Interaction:** The framework must allow users to interact with objects in the virtual world, while discarding the need for any touch input.

2.3 NON FUNCTIONAL REQUIREMENTS

Non-functional requirements are requirements which impose constraints on the design or implementation of the system. Various parameters were considered and described in detail.

- **Efficiency:** Efficiency is an essential characteristic for any system, and it is of utmost importance in a real-time application, such as this one.
- **Reliability:** Reliability is considered one of the main features to exploit Virtual Reality capabilities. It ensures constant operation of the system without disruption.
- **Immersion:** The system provides a truly immersive and realistic simulation of a 360-degree three-dimensional environment and this is considered to be one of the many important features for any VR system.
- **Freedom:** The system allows the user to explore the virtual world by letting him/her walk inside and interact with objects with the utmost freedom.
- **Learnability:** Learnability indicates the ease of learning to use the system.
- **Preference:** The system that is most preferred among the existing systems.

2.4 USER CHARACTERISTICS

There are no restrictions as such regarding the usage of the system. Anyone above the age of 5 can use the system. However, for users with history of epilepsy related problems, caution is advised, as with all VR systems.

2.5 APPLICATIONS

VR can be applied to many fields and some of them are discussed below as follows,

- **Military:** Virtual Reality has been adopted by Military – Army, Navy and Air Force to train the soldiers for combat and survival without putting them at risk.
- **Entertainment:** VR is already being used extensively for creating immersive games and movies. Gaming consoles are being modified to incorporate VR features.
- **Real Estate and Architecture:** VR allows the builder as well as the buyer to virtually explore the construction and the design of buildings. Architects can virtually analyse and improvise the design.
- **Sports:** Training of the players in sports can be done more efficiently and effectively with the use of VR. For instance, football players could be trained through VR to view the game from different perspectives in order to react and respond in a better way.
- **Medical Science:** VR has one best use for the overall betterment of humankind through medical science by training students through VR. Medical students could be prepared for any kind of surgery or treatment through Virtual Reality.

2.6 SUMMARY

This chapter talks about the Software Requirement Specification, while introducing the hardware and software requirements and the operating environment needed to run the system. It also talks about potential users that can user the application, with application in particular fields. It gives an outline that is needed to design, develop and test the application.

CHAPTER 3

HIGH LEVEL DESIGN

In this chapter we give an overview of the design of the system and how it is organized and the flow of data through the system. By reading this document the user should have an overall understanding of the problem and its solution. We have also discussed the problems encountered during the design of the system and justified the use of the design.

3.1 DESIGN CONSIDERATIONS

There are many aspects to consider in the design of a piece of software. The importance of each should reflect the goals the software is trying to achieve. Some of the aspects like efficiency, reliability, immersion, usability, freedom etc. should be fulfilled by the design.

3.1.1 ASSUMPTIONS AND DEPENDENCIES

- **Unity 5.4.2**, a cross-platform simulation creation engine installed.
- **Google VR SDK for Unity**, to implement VR features.
- **Smartphone**, running Android 4.4 or above.
- **MATLAB**, for analyzing the accelerometer input signals installed on the smartphone.
- **Blender**, a 3D content-creation program for building VR worlds installed.
- **Audacity**, an audio processing tool to create and edit sound effects.

3.2.2 GOALS AND CONSTRAINTS

GOALS

- To develop a positional tracking system for smartphone based VR Head Mounted Displays (HMD's) that tracks body movements for user interaction and navigation within the VR environment, independent of any controllers or similar hand-held devices.
- To create a naturally immersive VR experience at a low-cost.

CONSTRAINTS

- Rendering 3D frames on a mobile device for VR requires very high graphical processing power, often slowing down other concurrent processes.

3.2 SYSTEM ARCHITECTURE

System architecture gives a clear description of the operations that take place between the different modules of the system, and the relationships between them. The main idea of the proposed system is to create a naturally immersive VR experience at a low-cost. The architecture of the proposed system is shown in figure 3.1

The VR framework consists of four modules and they are described briefly,

- **Step Detection:** The phone's accelerometer reading values along the X, Y, Z axes and are captured and their combined magnitude is computed. Next step is to subtract the average mean of acceleration values from the current input value to remove any constant effects like gravity. The number of peaks is then counted and each peak represents a step.

- **Virtual Locomotion:** Computes velocity of walking in the virtual world based on the WIP speed i.e. instantaneous speed and the average velocity.

- **Object Interaction:** A reticle is used to look at and identify various objects present in the virtual world. Upon looking at an object, a timed gaze progress bar is shown and the objects are manipulated by triggering an action.

- **Head Gesture Recognition:** Since Google VR SDK supports orientation tracking natively, we decided to experiment with head gestures and the system is able to detect rotation along the X & Y axes most effectively. Two independent head rotation actions were seen.

    - Action 1: Move up and down/Rotate about X axis (Yes).

    - Action 2: Move left and right/Rotate about Y axis (No).

The framework thus developed could be deployed into any fields and so as to demonstrate we were able to use it in two such fields namely,

- **Treadmill App:** The goal of this app is to simulate exercise on a treadmill, while users jog in place on any surface, anywhere. The treadmill app keeps track of the number of steps, current speed, average speed and the current state and exercise time.

- **E-Commerce App:** The simulation features a shopping store and allows the user to explore the shop by letting him/her walk inside and they are able to interact with objects present, view information about it and choose whether or not to buy it.

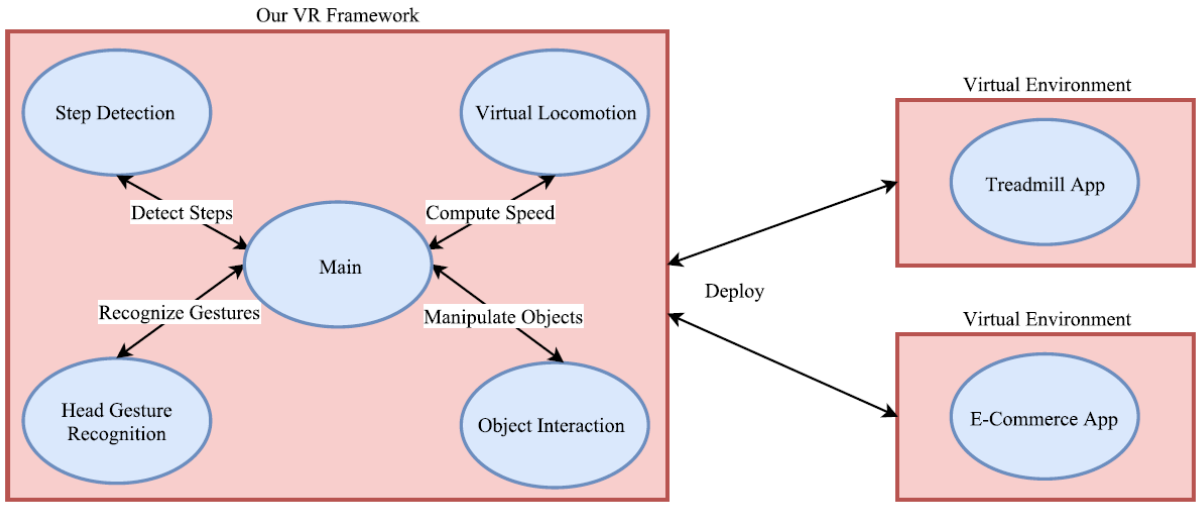

Figure 3.1: System Architecture

3.3 DATA FLOW DIAGRAM

Data flow diagrams indicate the flow of data through the system. They help provide a clear picture of how inputs are transformed into outputs.

The figure 3.2 shows the data flow diagram for Step Detection. The input is taken from the accelerometer on the smartphone. This input data is then used to compute the current acceleration i.e. acc_{cur} and at the same time the average acceleration acc_{avg} is updated. Using both the above values acc_{nog} is calculated using the equation as shown in the data flow diagram. Once the acc_{nog} is known, a check is done to see if it is a peak. If it is a peak then the step count is incremented and the process continues.

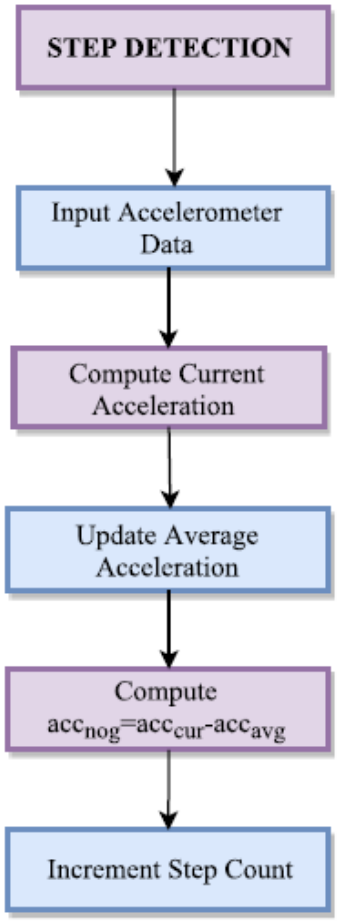

Figure 3.2 Step Detection Data Flow Diagram

Algorithm:

Step-1: Start.

Step-2: Input accelerometer data along the X, Y, Z axes. These are acc_x, acc_y and acc_z.

Step-3: Compute the magnitude of the current acceleration follows:

$$acc_{cur} = \sqrt{acc_x{}^2 + acc_y{}^2 + acc_z{}^2}$$

Step-4: Update magnitude of average acceleration, acc_{avg}, with the current acceleration value, acc_{cur}.

Step-5: In order to perform step detection, fluctuations are necessary in the accelerometer signal. To detect these, constant effects such as gravity can be eliminated as follows:

$$acc_{nog} = acc_{cur} - acc_{mag}$$

Step-6: Check if acc_{nog} is a peak. Every peak is a step. The number of peaks gives the number of steps.

Step-7: Stop.

The figure 3.3 shows a DFD for Head Gesture Recognition. The system is able to detect rotation along the X/Y axes and the current orientation angle is noted down. If the gesture is top to down then an event which specifies a 'Yes' is triggered. If the gesture is left to right then an event which specifies a 'No' is triggered and the next gesture is recognized.

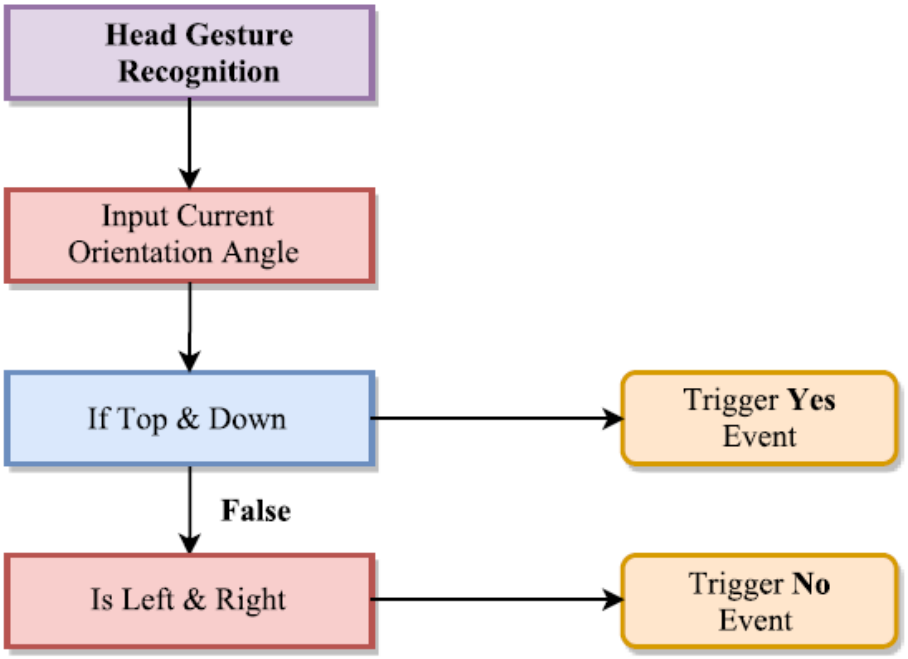

Figure 3.3 Head Gesture Recognition Data Flow Diagram

The figure 3.4 shows the DFD for Object Interaction. A reticle is able to detect the objects that are present in the virtual world. Using the time gazed progress bar along with the reticle pointer we are able to manipulate objects or trigger necessary actions.

Upon gazing at the object for more than 5 sec, an event can be triggered and this object interaction takes place. Object Interaction is a necessity for the E-Commerce App where a user is able to choose whether or not to buy an item.

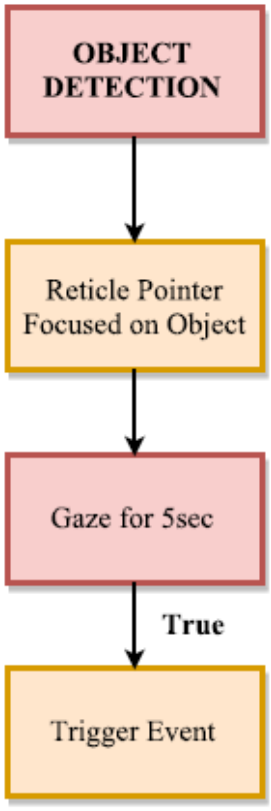

Figure 3.4 Object Detection Data Flow Diagram

The figure 3.5 shows the flow of events that take place. Each of the modules shown can work independently from one another but when used together the system reaches its goal of positional tracking and the core functionalities are met.

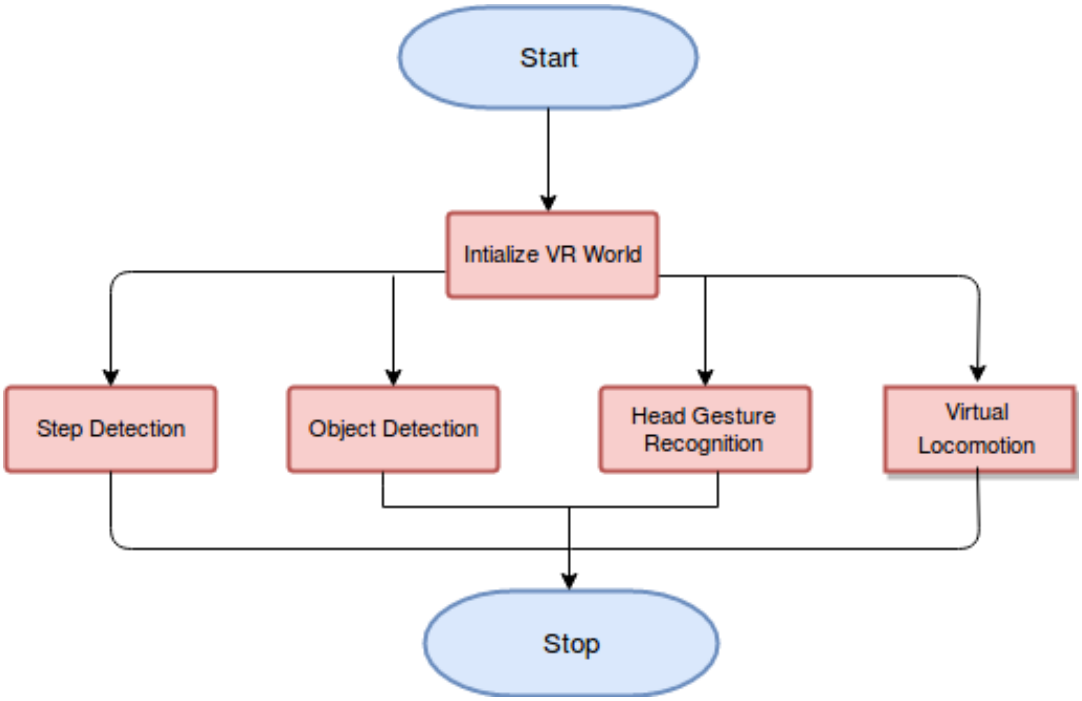

Figure 3.5 Final Flow of Events

3.4 SEQUENCE DIAGRAM

A sequence diagram shows the participants in an interaction and the sequence of messages among them. Fig 3.6 shows the sequence diagram for positional tracking in VR. The sequence diagram has 5 participants, User, Step Detection, Virtual Locomotion, Head Gesture Recognition and Object Interaction.

User refers to the different sets of people using the system. The 4 other participants are modules which help in positional tracking and each of the messages between the various users is depicted in the picture.

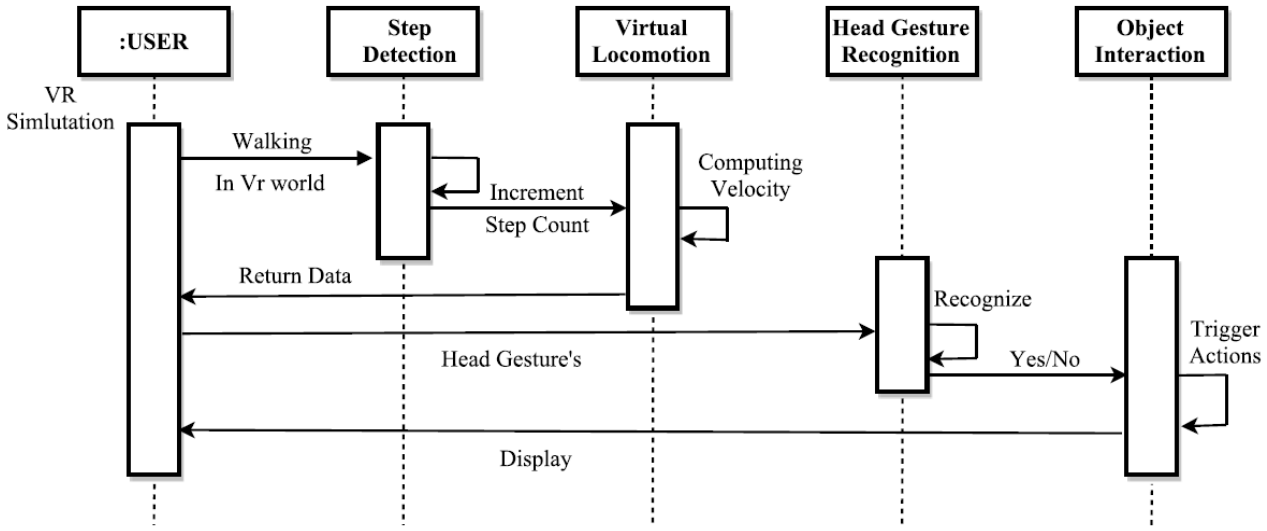

Figure 3.6 Sequence Diagram

3.5 SUMMARY

High-level Design provides an overview of an entire system, identifying all its elements at some level of abstraction. In this chapter, the goals and constraints of the project, system architecture and the sequence diagram is briefed and the communication between various entities is understood.

CHAPTER 4

DETAILED DESIGN

4.1 PURPOSE

Now that the high level design for the project has been explored, we dive into the depths to explore a more detailed design. While the goal of the previous chapter was to gain an overall understanding of the system from a birds-eye view perspective, this chapter contains a more involved discussion about the design of the system. It covers the various interfaces and modules of the project that are essential to the functioning of the system.

4.2 PROCESSING STEPS

In the process of building our VR framework, there are a few essential goals that need to be achieved. Each of these goals is achieved through independent modules:

- **Module 1:** Step Detection: Detect user's walking-in-place steps via the phone's accelerometer, and keep a count of the number of steps.
- **Module 2:** Virtual Locomotion: Translate the user's WIP steps in the real-world to appropriate movements in the VR world, while computing instantaneous speed and average speed.
- **Module 3:** Object Interaction: Allow the user to interact with objects in the VR world with the help of a mechanism to detect objects and trigger actions.
- **Module 4:** Head Gesture Recognition: Recognize rotational movements of the head, like nodding of the head, to trigger events in the VR world.

In addition to the above primary modules of the framework, a new module to is required within which the framework will be tested. This is called the Virtual Environment module.

4.3 USER INTERFACE DESIGN

We now proceed to a detailed discussion about the interfaces and the modules of the system. One of key components of any system is its User Interface. User Interface Design or User Interface Engineering is the design of user interfaces for machines and software and mobile devices, with the focus on maximizing the user experience. A good UI enables ease of learning and access of the systems core functionality, while at the same time being visually appealing and allowing the user to have a seamless experience using the system.

4.3.1 Software User Interface Design:

Being a VR project that is very much graphic oriented, it is of key importance to make it look visually appealing, while at the same time ensuring that there is minimal loss of performance. Unity Studio was used to design this project's UI and VR environment. Blender was used to create individual objects in the VR world, like treadmills, laptops. Google's Cardboard SDK was used to create Google Cardboard specific VR applications. Cardboard SDK was able to graphically convert a normal single-screen oriented Android app into one with two screens side-by-side, as required by Google Cardboard.

4.4 MODULES

4.4.1 Module 1 – Step Detection:

Steps are detected with the help of the phone's accelerometer as input, and some processing done at the application end. An accelerometer is an inertial device that measures acceleration along the three axes X, Y, Z. The acceleration value being measured is in terms of meters per second^2 (m/s^2). Values are sampled continuously by the accelerometer along the three axes, say acc, using which a combined magnitude is computed.

We are interested in fluctuations in this combined magnitude, which indicates some activity. So, we subtract the mean of acceleration values from the current input value, to remove any constant effects, like gravity. Following this, the number of peaks is computed, which then gives us the number of steps.

4.4.2 Module 2 – Virtual Locomotion:

Now that we are able to detect steps, the next big challenge is to design an appropriate scheme to translate movements in the real-world to movements in the VR world. This includes two broad goals in the VR world:

1) Computing speed of movement.
2) Translation along the appropriate axis.

Two different types of speeds are computed in the system; instantaneous and average. In order to compute the instantaneous speed, we begin by first computing the time taken between two consecutive steps. We then compute the instantaneous speed by taking the reciprocal of this time computed. If the time taken between steps exceeds 1 second, then the speed is set to 0, because the user is at rest. In order to compute the average speed, we divide

the total number of steps by the amount of time taken to cover those steps. Average speed is useful to analyze user behavior and usage of the system.

As for the next goal, now that we have the speed of the movement, it is a question of which direction to perform the movement in. For this, we keep track of the user's gaze direction, which is nothing but the direction in which the user is looking. We allow movement along both the X and Z axes, while keeping the Y axis position constant, to prevent unwanted actions like moving skywards or falling downwards. User's gaze direction is detected via rotational/orientation tracking about the Y axis, which provides for a full 360° rotation experience.

4.4.3 Module 3 – Object Interaction:

Movement in the VR world has been taken care of, thus the navigation part of the framework is taken care of. However, we realized the need to be able to interact with objects in the VR world, as an experience with just navigation seems incomplete. Given that we wanted to eliminate the use of any controllers, which may kill the immersion experience, we decided to experiment with a hands-free object interaction system.

We begin by keeping track of the user's gaze direction, as discussed previously. However, this time, we keep track of rotation about all three axes. In order to allow the user to specifically focus on an object, we implement the concept of a reticle, which is nothing but a small pointer, much like a cursor, that moves as you look around. The user can thus choose to interact with an object by first focusing on the object via the reticle.

However, just because a user looks at an object, it does not necessarily mean that they want to interact with the object. So to work around this issue, we introduce a timed radial progress bar that keeps track of the amount of time for which the user focuses on an object. If it exceeds five seconds, then we trigger an action associated with the object. Thus, an effective hands-free navigation system has been designed.

4.4.4 Module 4 – Head Gesture Recognition:

While we were experimenting with other methods of providing input to the system, we came across the concept of head gestures. The head, because is attached to the neck, has six degrees of freedom; three rotational and three translational. Given that the VR system is finally deployed in a Head-Mounted Display, it makes sense to try recognizing gestures of the head. Translational tracking of the head could be difficult to isolate from translational tracking of the body, as is there is a limited scope for the translation of the head, thus it would be ambiguous. We opt to work with rotational tracking instead.

Based on experimentation, we discovered that rotation along the X and Y axes were the most effective and unambiguous. This allows us to work with two independent head rotation gestures.

1) **Gesture 1:** Move up and down/Rotate about X axis (Yes).
2) **Gesture 2:** Move left and right/Rotate about Y axis (No).

Now that we have two gestures that can be recognized by the system, we can configure each of these to perform a trigger certain events based on the virtual environment of application. As an example, in the E-commerce application that we built as a proof of concept example, we configured Gesture 1 to work as a way to confirm a product purchase, while Gesture 2 was configured to work as a way to reject product purchase.

4.4.5 Module 5 - Virtual Environment:

Although much of the VR content out there today is used solely for the purpose of Entertainment, VR can be applied to many spaces like Fitness, E-Commerce, Learning and Architecture. Since we have built a comprehensive VR framework in this project, we decided to deploy it in two proof-of-concept virtual environments: Fitness and E-Commerce.

1) **Fitness:** We built a Treadmill VR application that keeps track of your fitness.

2) **E-Commerce:** We built a VR application that attempts to emulate the future of online shopping.

These two virtual environments give the user a chance to understand both the functionality, and the power of an immersive VR system. Our framework allows both navigation and interactions with objects in the VR world.

This goal of this Treadmill app is to simulate exercise on a treadmill, while users jog in place on any surface, anywhere. The treadmill app keeps track of the number of steps, current speed, and average speed, the exercise time and current state. According to My Fitness Pal, a 50 kg person jogging in place for 30 min loses up to 272 calories. This opens doors to using VR for fitness freaks, medical patients and just about anyone else.

The E-commerce app features a shopping store and allows the user to explore the shop by letting them walk inside. They can pick up objects, view information about it and choose whether or not to buy it. This object interaction is triggered using the time gazed reticle along with head gesture recognition. With the emergence of WebVR, this would be a potential app for E-commerce giants like Amazon, Flipkart to explore as VR adds a new dimension in a way to interact with customers.

4.5 SUMMARY

This chapter dealt with the various modules and interfaces of out VR framework in detail. Providing a good UI is key to VR apps. A good UI can greatly contribute to creating a very immersive experience, which is the number one goal of any VR system. In addition to providing a visually appealing experience, VR apps must also provide solid functionality and features, specific to the goals of that app, through well-designed modules, as discussed in this chapter.

CHAPTER 5

IMPLEMENTATION

In the previous chapter we discussed about the design of the system in-depth. With the design in place, we now focus on the implementation details of the system, which is perhaps the most exciting part of software development for any programmer. Details about the choice of programming language, platform, coding conventions and libraries used are all discussed in this chapter.

Being the only step in the Software Development Lifecycle that involves actual development of the product, it is extremely important that is follows the design guidelines as specified previously. All algorithms need to be implemented in the chosen language(s) of implementation. For our project in particular, we used a set of different tools to achieve our goal of building our framework.

5.1 PROGRAMMING LANGUAGE SELECTION

C# was used to build applications as a part of this project. C# is a completely object-oriented programming language, and is very similar to C++, in terms of syntax, thus making it easier for programmers to focus on the implementation, rather than syntax. Unity Engine, which we used to create VR applications in this project, uses C# as its scripting language. Unity Engine and C# together contain a lot of specialized functions, making it easier for the program to develop applications.

Preliminary implementation of algorithms was done in MATLAB, for the purpose of testing the system. MATLAB stands for MATrix LABoratory, and contains a whole bunch of functions that can be used to implement mathematical operations, especially those pertaining

to matrices. Programming in MATLAB is fairly simple, can be interfaced with other devices like mobile phones and it thus ideal for testing algorithms. After the algorithms were perfected, they were then reproduced in C#, for the final implementation.

5.2 PLATFORM SELECTION

Android was the selected platform of development, because Google Cardboard VR works with Android. All applications built were executables for Android. More specifically, two phones, both running Android 6.0 M were used for execution and demonstration of the application. Our application runs on all Android platform versions, starting from Android 4.4. This allows for a wide range of target devices, over 85% of the market has Android 4.4 or above, according to the Android Developers website.

5.3 GRAPHICAL USER INTERFACE

Unity Studio was used to create appropriate graphic interfaces. Being a heavily graphic-oriented project, much emphasis was placed on creating a visually appealing and simple-to-use user experience. Unity Studio was used to design this project's UI and VR environment. Unity is an extremely powerful cross-platform tool to build 3D applications. It allows developers to choose from a variety of platforms like Android, Windows, iOS, Xbox and Tizen, thereby allowing the developer to write once, port many times. When combined with Google's Cardboard SDK, it can be used to create Google Cardboard specific VR applications. Cardboard SDK was able to graphically convert a normal single-screen oriented Android app into one with two screens side-by-side, as required by Google Cardboard.

In order to create realistic objects in the VR world, like laptops and treadmills, Blender was used. Blender is a 3D modeling tool that allows designers to combine shapes

and objects, color them and add animations. While individual objects are modeled using Blender, they are put together to create a scene in Unity. All properties associated with objects are added in Unity.

In addition to the graphical aspect of the project, to enhance the VR experience and improve the immersion factor, appropriate sound effects, like the sound of footsteps, were added to the project, through Unity Studio.

Audio clips were edited using Audacity, a sound processing tool, and incorporated appropriately into the project. Audacity allows users to work with sound files, which can be edited to make them shorter or longer in length, slower or faster in speed or even softer or louder in terms of volume. This allows users to create appropriate sound effects for their projects.

5.4 SUMMARY

In this chapter, we discussed the implementation details of our project. Our application can thus be executed on any mobile device running Android 4.4 K or above. With a Google Cardboard headset to accompany the mobile device, a complete VR experience can be achieved, at low cost and at ease. Unity Studio is an extremely powerful tool to create 3D applications, and when integrated with Google Cardboard SDK, can be used to create stunning VR applications.

CHAPTER 6

TESTING

6.1 UNIT TESTING

Unit testing focuses on verification of the smallest unit of software design - the module. As specified in the previous chapters, our system comprises of five main modules. The testing procedure for each of those modules is described in this section.

6.1.1 Unit Testing for Step Detection Module

The primary factor to be checked is the accuracy of the step detection algorithm that is implemented. To do the same, we begin by manually jogging in place 100 times, and then identifying the number of steps detected by the algorithm. This would give us the number of steps detected by the algorithm, for every 100 steps taken by the user. For example, if the algorithm detects 97 steps, we have achieved 97% accuracy.

6.1.2 Unit Testing for Virtual Locomotion Module

The correctness of the instantaneous speed and average speed calculated by the algorithm is to be checked in this module. This verified by manually keeping track and calculating both the above parameters, and comparing with the algorithm generated values for correctness.

6.1.3 Unit Testing for Object Detection Module

Three key factors have to be checked for while testing the Object Detection module:

1) Detection of objects when a user looks at them.

2) Triggering of an action associated with an object after a fixed period of time.
3) Proper functioning of the radial progress bar.

The first two factors can be tested by trying the looking at objects and waiting for them to trigger associated actions after a fixed waiting period. As for the third factor, we can manually time the progress of the radial progress bar, and if it matches the manual timing, then it is functioning as expected.

6.1.4 Unit Testing for Head Gesture Recognition Module

Head Gesture Recognition module supports two gestures:

1) Moving head up and down.
2) Moving head left and right.

Each of these gestures can be tested by performing them multiple times and recording if the program can recognize them.

6.2 SYSTEM TESTING

We asked 15 people, 8 male and 7 female, of varying age groups between 10-50 years to participate in a survey. The study compared three different navigation systems and asked users to rate the best system based on six different parameters. These parameters are: Efficiency, Reliability, Immersion, Freedom, Learnability and Preference. We selected these parameters based on the work of Sutcliffe et. al and Tregillus et. al.

The other two systems, Look Down To Move (LDTM) and Click To Move (CTM) are two of the most popular approaches to navigating in a virtual world. LDTM requires users to look down at the ground to start moving, and look down again to stop. Similarly, CTM requires the user to click on the screen or a button to start moving, and click the button

again to stop moving. The biggest drawback with both these systems is that the user can move only at a constant speed, which reduces both freedom and immersion. The graph showing the comparison is shown in the Figure 6.1 as follows:

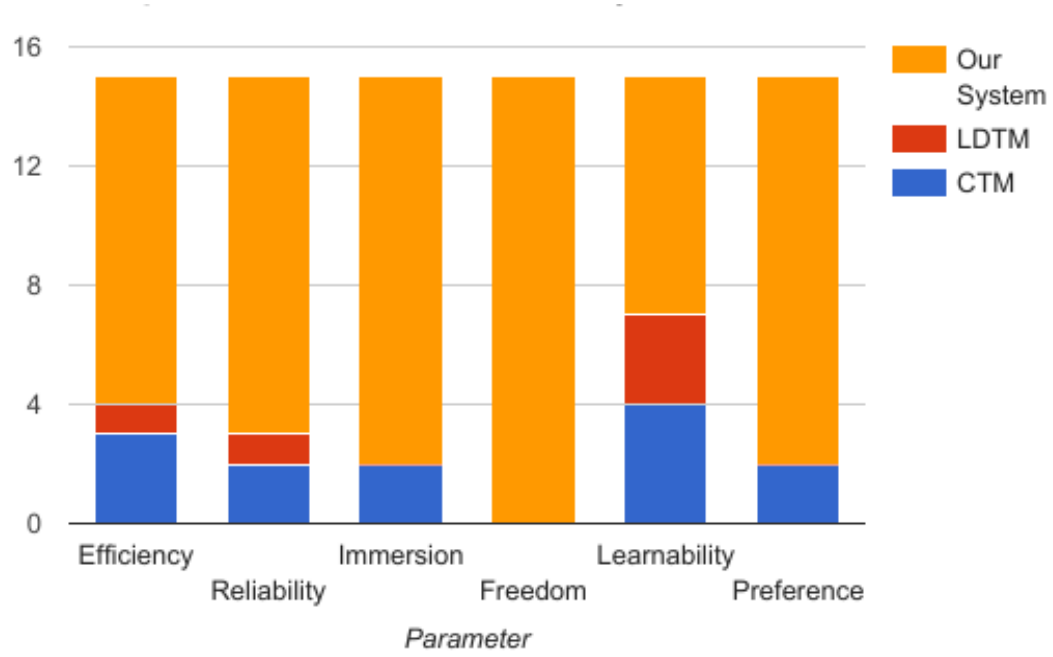

Figure 6.1 Comparison between various VR systems

As observed from above, our system outperforms the other systems in every parameter. According to this survey, users seem to experience maximum freedom while using our system, as expected. Similarly, our system seems to outperform the other systems in other parameters like efficiency, reliability and immersion. An observation was that users initially had some difficulty learning how to use our system. However after they were asked to try out a tutorial demo app, they felt much more comfortable. The overall consensus was that our system was the most preferred one, among the three. Being able to guarantee this level of freedom and immersion, while keeping in mind the cost factor as well as the constraint of not using any external hardware, is definitely both exciting and satisfactory.

6.3 RESULT SNAPSHOTS

The Figure 6.2 shows a snapshot of the E-Commerce App that was built. The simulation features a shopping store that allows users to explore the shop by letting him/her walk inside and can pick up objects, view information about them and choose whether or not to buy it.

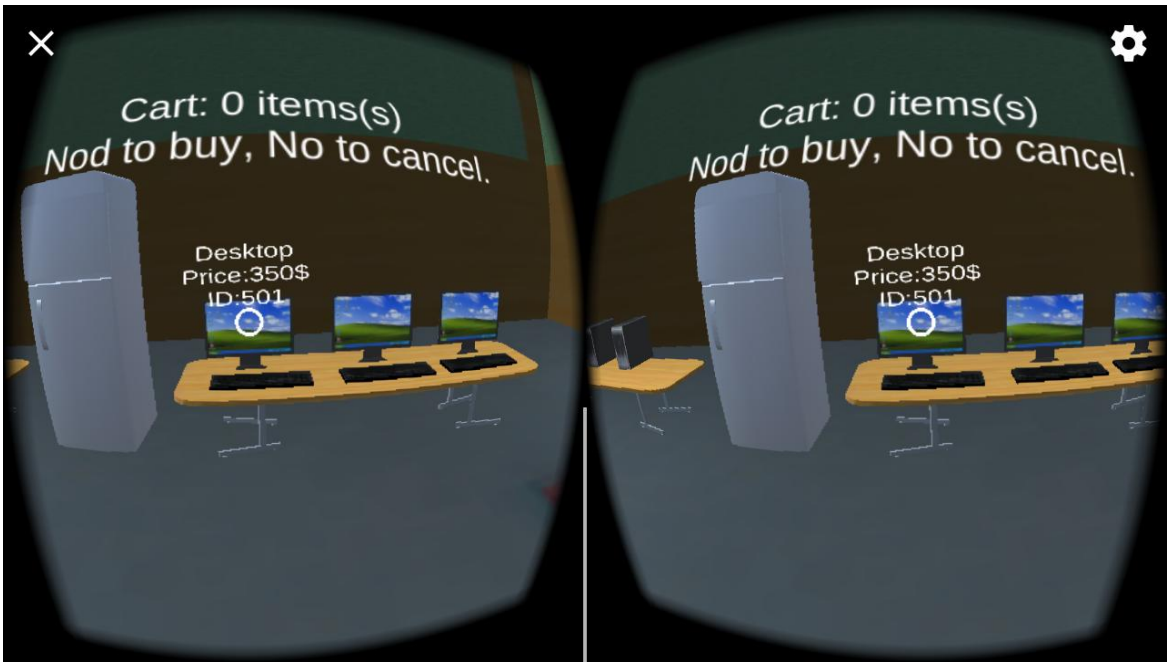

Figure 6.2: E-Commerce App

The Figure 6.2 shows a snapshot of the Treadmill App and the goal of this app is to simulate exercise on a treadmill, while users jog in place on any surface, anywhere. The treadmill app keeps track of the number of steps, current speed, average speed and the current state and time.

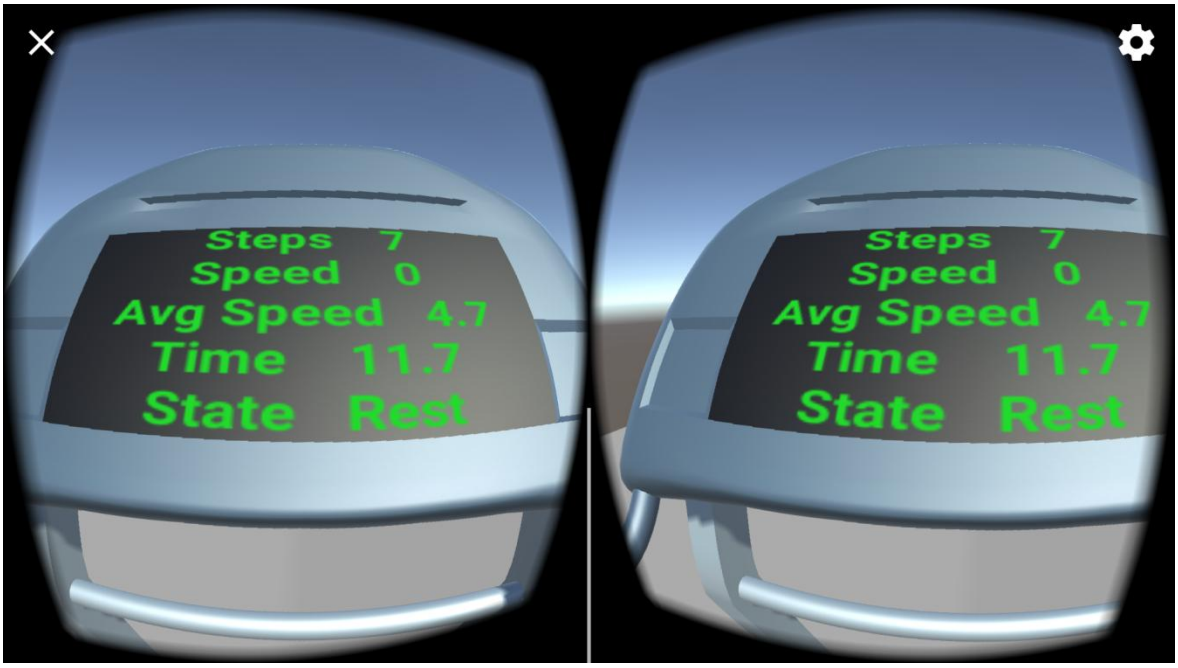

Figure 6.3: Treadmill App

6.4 SUMMARY

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. In this section, Unit testing and System testing of the project is covered. It can be observed from the survey described that our system is clearly the most preferred one amongst the existing implementations. Thus, our goal of creating an immersive experience for VR systems has been achieved.

CHAPTER 7

CONCLUSION

A lot of research is happening right now in VR, making it one of the hottest tech in the industry. VR has applications almost everywhere, from gaming and entertainment, all the way to sports and military. Much of today's VR systems use external sensors or controllers to track body movement or perform navigation in virtual worlds, and this has its limitations. Similarly, most of these support only orientation-tracking, and are yet to come up with exciting solutions for positional tracking. These systems are sophisticated and expensive.

The motive of the project was to build a framework which can be used for positional tracking in mobile based virtual reality systems, using only hardware that exists on your phone and making sure that the setup is as cost effective as possible. We were able to successfully build a framework that in addition to allowing users to navigate in virtual world, also allows them to interact with objects present in these worlds through a variety of mechanisms, including head gestures. This framework thus can be tested in a variety of test environments. It will definitely be an exciting future with VR systems such as this one being created for everyone.

7.1 LIMITATIONS OF THE PROJECT

Since VR applications consume a lot of graphical computational power, smartphones with lower graphic capabilities suffer from lag and screen stutter. As a result, VR worlds built to experiment with the framework cannot be overly sophisticated in terms of graphics. In other words, it is preferable to use objects with a low poly count to create VR worlds. But this makes the application less realistic, visually.

Another limitation of the positional tracking system is that it can only detect steps, and not other types of actions like jumping, crouching, and punching, primarily because these require use of additional sensors external to the phone.

7.2 FUTURE ENHANCEMENTS

A major enhancement could be the addition of an appropriate hand tracking mechanism that will also capture hand gestures and simulate appropriate actions in the virtual world. We were unable to include such a setup because existing hand tracking devices require a PC to function. Given that our setup is hands-free, it is an ideal feature to incorporate. Upcoming device Leap Motion VR could be a potential solution.

As mentioned in the Limitations section, an interesting idea could be to incorporate appropriate sensors and mechanisms to detect different types of actions like punching and crouching to make the experience more entertaining, and to give the user a higher level of freedom.

Another interesting challenge would be to integrate our framework with a proprietary application like Google Earth and Google Street View, which would potentially allow users to walk through streets and monuments anywhere in the world virtually, right from their own room.

REFERENCES

Papers:

[1] Tregillus, S., & Folmer, E. (2016, May). Vr-step: Walking-in-place using inertial sensing for hands free navigation in mobile vr environments. In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (pp. 1250-1255). ACM.

[2] Brajdic, A., & Harle, R. (2013, September). Walk detection and step counting on unconstrained smartphones. In Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing (pp. 225-234). ACM.

[3] Sutcliffe, A., & Gault, B. (2004). Heuristic evaluation of virtual reality applications. Interacting with computers, 16(4), 831-849.

Web Links:

- Virtual Reality, Wikipedia, Last Accessed on May 17, 2017 from https://en.wikipedia.org/wiki/Virtual_reality
- Unity Game Engine, Wikipedia, Last Accessed on May 17, 2017 from https://en.wikipedia.org/wiki/Unity_(game_engine)
- Blender Software, Wikipedia, Last Accessed on May 17, 2017 from https://en.wikipedia.org/wiki/Blender_(software)
- Positional Tracking, Wikipedia, Last Accessed on May 17, 2017 from https://en.wikipedia.org/wiki/Positional_tracking
- Calorie Counter, My Fitness Pal, Last Accessed on May 17, 2017 from https://www.myfitnesspal.com/

1. NAME: BHARAT GOGINENI USN: 1PE10CS021 CONTACT NUMBER: 9739591462 EMAIL ID: bharatgoku@gmail.com ADDRESS: 66, 1st cross, 2nd main, HAL 3rd stage, Bangalore 560075 .	2. NAME: SIDDHANT PRIYADARSHI USN: 1PE12CS156 CONTACT NUMBER: 7979968293 EMAIL ID: 1siddhant1@gmail.com ADDRESS:
3. NAME: S SAMPATH USN:1PE13CS131 CONTACT NUMBER: 9916312730 EMAIL ID: sampath_shanmugam@outlook.com ADDRESS: Ph-1, 309, Golden Park Apts, Bommanahalli, Bangalore 560076	4. NAME: VAISHAK R VELLORE USN: 1PE13CS174 CONTACT NUMBER: 9008030271 EMAIL ID: vaishak.vellore@gmail.com ADDRESS: #33/2, Aishwarya Amaze Apts, Block 1, Flat 004, BTM 4th , Bangalore 560076