FACULTY OF INFORMATICS
TECHNICAL UNIVERSITY OF MUNICH

Bachelor of Science

Phone Based Motion Control Travel Technique in VR

Mohamed Ashry

FAKULTÄT FÜR INFORMATIK
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Bachelorarbeit in Informatik

Phone Based Motion Control Travel Technique in VR

Mohamed Ashry

Supervisor: Prof. Gudrun Klinker, Ph.D.

Advisor: Amal Benzina, M.Sc

Submission Date: 6. September 2010

I assure the single handed composition of this bachelor thesis only supported by declared resources.

Munich, 6th of September 2010 Mohamed Ashry

Acknowledgements

First of all, I would like to thank my family, far away from here, for their love, kindness, and never ending support during the thesis. And without them I could not have finish my thesis.

I would like to express my sincere gratitude to my supervisor Prof. Gudrun Klinker for her continuous encouragement and supervision, and I appreciate her patience in waiting for my results. I deeply thank my advisor, Amal Benzina, M.Sc for her endless support, guidance, efforts, and supervision was invaluable. Also I thank Fachgebiet Augmented Reality, Their hospitality and support during the period of this thesis is greatly acknowledged.

I would like to thank Prof. Dr. Slim Abdennadher for offering such opportunity to do my bachelor thesis here in TUM, also for his never ending support during the thesis.

Finally, I would like to express my deepest gratitude to my friends for their support and encouragement, especially I would like to thank Ady Aldaghma, Ahmed Hazem, Maram Akram, and Nada Mohamed for their support, help, and valuable feedback during the thesis work period.

Abstract

Immersive virtual environment have been used in many applications area recently and the key of success in the VE is the interaction within it and how intuitive for users. Users can interact with in the virtual environment via standard input devices such as keyboard and mouse, or multi-modal devices like gloves, PDA or handled devices. Interaction in VE focuses in three criteria which are travel techniques, system control techniques and finally selection techniques.

In this work mobile phone with android operating system is used to investigate different techniques for navigation and specifically travel techniques, way finding and guiding people within the VE is out of the work scope. The travel techniques focus on how users can start and end motion, indicate direction of movement and alternate movement velocity. The study mainly focuses on three interaction techniques. First, The touch based motion techniques simulate the finger movement on the phone screen and recognize the gesture of the user's finger. Second, Acceleration techniques, The three implemented techniques are manual speed, finger speed and gas pedal technique. Third, The steering based motion techniques shows how users can change the view point orientation via changing the mobile orientation and the four different techniques investigated are rotate by heading, rotate by roll, rotate by roll with fixed horizon, and merged rotation which is a sort of combination between rotate by roll and rotate by heading.

The methodology will be tested by applying it to the terrain3D which is a complex VE allowing the users to follow a path then fill a qualitative evaluation questionnaires. We represented two different user evaluation tests for acceleration techniques and steering techniques. The results of the evaluation and future development are discussed.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

Immersive virtual environments (IVEs) have been used recently in many application areas, Such as education, simulation, training, medical, and emergency [Bow98]. The Virtual Reality (VR) aim is to immerse people within the virtual world and allow them to interact with the virtual environment (VE) in a very intuitive way. In addition, the goal of the ideal VR system is to immerse users within the environment and let users have the feeling that they are performing the tasks in the virtual world. In addition to the visual components, auditory effect, and haptic feedback are added to the VR systems, and they provide for real world experiences and increase the users immersion within the VE. However, current VR systems still far from the ideal VR system [KIM05]. Due to the lack of the interaction techniques and user interface constructs usability [Bow98].

The interaction with the VE is a key of success for the VR system [Bow98, BKH97, KGMQ09]. The VE interaction techniques design differ from the standard GUIs design, as illustrated in the work of (Bowman, Koller, and Hodges, 1997) [SGTF06]. The navigation techniques strongly dependable on the interaction techniques. According to the study of (Bowman,1998) the interaction techniques falls into three categories viewpoint motion control, selection, and manipulation [Bow98]. In this work we focus on travel techniques which mean focusing on the user's viewpoint motion control. Travel is important and universal user interface task, and in order to increase the productivity and users' comfort within the VR system, the travel techniques need to be better understood and implemented [BKH97].

The interaction techniques required specific input devices to the VE systems. Many interaction techniques have been developed to support different types of navigation tasks, hence different travel tasks, Also to support natural locomotion in the VE, As the more natural locomotion increases, the sense of presence in the VEs increases too [KGMQ08]. There are Many approaches for navigation tasks in IVEs. Some approaches use physical locomotion , such as treadmill walking, while other use a special device, such as joystick and mice. And the success of the approaches which show physical locomotion is higher [KGMQ09].

User Interface (UI) is consisted of some components, such as input/output devices and interaction techniques. The effectiveness of a VR systems is based on the UI methodology. Generally computer users are familiar with the 2D interface input devices, such as mouse and keyboard. The 2D interface input devices are still of great importance for 2D space and

3Dinteractions. In the other hand the 2D interfaces are not always applicable for the IVE systems. And attempts to adapt the 2D interfaces to 3D interactions faced many difficulties and causes many problems. Hence many researches regarding new 3D interaction techniques are conducted [KIM05].

Some studied focused on increasing the intuitiveness and naturalness of the UIs, and trying to increase the users feeling of being immersed within the VE. In such techniques users are prohibited from looking on the UIs as long as they can. In addition such techniques required new developed UI concepts, such as using new 3D input devices.

The 3D interactions includes special devices which are used to provide information required for the VR system, such as where users are looking, and provide information regarding orientation and position. Hence the VR system is able to present the exact scene based on users' motions or actions. The main devices are the tracking equipments, used to track the users' position and orientation, ART or sensors could be used to provide such data [KIM05]. The ART is used to track input devices and provides continuous and accurate feedback regarding the position and orientation of the device. However, it requires that device is always being within the tracking cameras' range. In this work the inertial sensors of the device is used to provide feedback regarding position and orientation.

The handled devices such as mobile phones and multi-touch screen devices have been used recently in many interaction techniques. The multi-touch technologies have been available recently for various devices sizes. The technology varies from simulating with computer vision to using resistive and capacitive multi-touch screens [KGMQ09]. The first multi-touch device was lemuar in 2005 manufactured by Jazzmutant [Jaz], followed by the iPhone and iPod Touch developed by Apple [Appa, Appb]. The multi-touch technology results in number of new interaction techniques. For example the pinch zoom implemented in the handled devices [KGMQ08].

The study of (kim - Gracanin - Matkovic - Quek,2009) proposed the Finger Walking In Place (FWIP) interaction technique, where user simulates the physical locomotion using a pair of fingers on the iPhone or iPod multi-touch screen [KGMQ09]. However, the FWIP reduces the sense of presence in the VE as it does not provide the real vestibular cues plus the handled device has to be hold using both hands [KGMQ08]. As a result, The task of implementing a multi-touch interaction technique is far from trivial. And effective interaction techniques where human navigation and motion control are taken into consideration are crucial for the success of new interaction paradigm [BKH97, KGMQ09].

1.2 Aim of the project

In this work mobile phone is used as an input device for interaction techniques within the VE. The mobile phone used in the study is HTC desire, which supports multi-touch capabilities, built in inertial sensors such as accelerometer and orientation sensor, and the operating system is android [HTC]. The work focuses on the travel techniques within the VE, and not way finding. During the study we tried to solve problems related to naturalness and usability of the interaction techniques. The use of the handled device before in simulating FWIP techniques results in reducing the sense of presence within the VE[KGMQ08], and due to the fact that implementing a multi-touch technique is far from trivial[KGMQ09]. Therefore we focused on increasing the naturalness and the sense of presence in our approach, so a new methodology for traveling in the VE based on the single-touch techniques is implemented.

The virtual environment used in this work is the Terrain3D. The Terrain3D can run on multiple GPUs in parallel and multiple clients in a cluster. Hence, it could be drived in a cave system. The Terrain3D engine is capable of rendering immersive high-quality terrain landscapes in the CAVE in real-time [Kir09].

The aim of the project is to provide intuitive travel techniques and a usable User Interface for interaction within the VE, and mainly we focus on the viewpoint motion control. The study provides different interaction metaphors. In addition we focus on the users performance while using more degrees of freedom (DOF) or less degrees of freedom. The user has to hold the mobile phone in one hand and perform all the interactions using the thumb, and hence the user can use the second hand for another task.

The handled device touch screen is used to simulates the physical locomotion. As the more natural and physical the technique is, the higher the success of the technique is. And the inertial sensors of the mobile phone are used to provide information regarding the mobile orientation, instead of tracking the mobile phone using the ART for example.

The travel techniques falls into three categories the touch based motion control, the acceleration based motion control, and the steering based motion control. First, the touch based motion control simulates the physical locomotion on the handled device touch screen. Second, the acceleration based motion control provides different techniques for controlling the acceleration, three techniques are conducted, which are manual speed, finger speed, and gas pedal techniques. All acceleration techniques are based on the single-touch techniques. Third, the steering based motion techniques are responsible for the change in the user's viewpoint orientation within the VE. Four techniques are implemented Rotate by heading, Rotate by roll, Rotate by roll with fixed horizon, and Merged rotation. Some techniques provide users with more degrees of freedom than others, such as merged rotation.

The project conduct an evaluation to compare between the different techniques, and some quantitative measures are used for comparison such as speed and accuracy. Some Qualitative Evaluation Questionnaires are conducted to evaluate the usability, ease of use, and attractiveness of the metaphors, Also indicates how successful and intuitive the techniques are, The Qualitative Evaluation Questionnaires are SUS and Attrakdiff evaluations. The goal of the evaluation is to figure out which techniques are more suitable for users, and provides users with naturalness and the increases their sense of presence within the IVE.

1.3 Thesis Statement and Structure

The thesis presents new intuitive travel techniques for navigation in an IVE. It also focuses on viewpoint motion control. It provides a usable 3D user interface for interaction in VR systems.

The thesis explores literature about VE systems, travel techniques, and travel techniques quality factors. It also discussed some related work to our approach, the related work focuses on the different travel techniques and discussed the different quality factors. It Also explains the android orientation sensors, in addition to the differences in the coordinate systems for the android phone and the virtual environment (Terrain3D). It also argues why using a UDP for the server connection is more convenient. (In Chapter 2)

Our approach and the different techniques implemented regarding mobile phone tracking, the touch based motion control, the acceleration techniques, the steering techniques, and the Graphical User Interface (GUI) are discussed (In Chapter 3).

The thesis explores the implementation of the different interaction techniques and the user interface. It explains the touch based motion control, and how the translation and steering techniques are implemented. The network between the VE and the mobile phone is discussed too.(In Chapter 4)

The evaluation is discussed, and how the evaluation is implemented and the different techniques used to evaluate the acceleration and steering techniques. Also pointed to the Qualitative Evaluation Questionnaires. Then the results of the evaluation are investigated.(In Chapter 5)

Finally, The summary of the study, the contribution to the researches, and introduction to future work and the direction of the next researches are described (In Chapter 6).

2 Background and Related work

In this chapter background about navigation in the 3D virtual environment is discussed. In addition to that the travel techniques are investigated, also different locomotion metaphor. The second part discuss the related work to our project are discussed, Also the disadvantages and advantages will be shown, also discuss some researches related to different interaction techniques. After that a brief description about the orientation sensor used in the android phones, followed by a comparison between the different coordinate systems used for the mobile phone and the VR system. Additionally, a brief description for the networking part used as a sort of communication in the environment.

2.1 Background

The aim of the Virtual Environments -VEs- is to obtain a new HCI paradigm where users are active participants and immersed within a virtual 3D world generated by computers. VEs have been recently used in many criteria such as design, education, visualization, emergency, medicine, and military training. However, despite the rapid advances in the technology of the VEs, still immersive VE applications are not common outside the research laboratory. One of the reason is related to the interaction techniques and user interface constructs for immersive VEs, as they lack to usability and effectiveness [Bow98].

The CAVE system was originally designed and implemented in 1991. The computer generated illusions are displayed onto screen walls, and they are surrounding the user. The CAVE system usually equipped with head and hand tracking system. A stereo sound system is added to increase the user's immersion within the VE. The cave combines both real and virtual objects in the same space without any occluded view of their own bodies while they are interacting. User can physically move around and observe the environment. The cave system provides unique interaction experiences could be used for education, scientific, and entertainment [KIM05].

The cave systems are mostly used in scientific visualization. In Addition, as the cave system is room sized, hence a lot of technology could be immersed within the VE, for example a specific designed vehicles can be used for simulation and training programs [KIM05].

The success of the immersive VE for navigation are based on how successful the interaction technique is. The interaction is based on the VEs' input devices. There are many approaches for navigating in a VE, such as physical locomotion, and some devices, such as joystick. The more natural and physical the locomotion is, the more sense of presence in VEs, in addition the physical and natural locomotion increase the spatial awareness. Hence the acceptance of natural and physical locomotion are higher [KGMQ08, KGMQ09].

The interaction within the VE divided into three categories, which are viewpoint motion control, selection, and manipulation. Viewpoint motion control shows how the user interactively positions and orients the viewpoint within the VE, Different viewpoint motion and control techniques have been implemented. Some of the proposed metaphors are flying, eyeball-in-hand, and scene-in-hand [BKH97]. Selection refers to the picking of the virtual objects in the VE. Manipulation refers to changing the virtual object's position and orientation. Selection and manipulation are almost combined together. The fourth interaction category is system control which involves the users commands to perform the task within the VE, for example delete an object. However, at the low level phase it could be categorized as selection and manipulation [Bow98].

The navigation in the VE involves both the movement within the VE which is the travel, and the process of indicating the travel direction and target which is the way finding. Travel technique has to incorporate three basic components which are indicating movement direction, indicating the velocity, and the input condition which means how users are going to start and stop the movement. These components provides the structure for the taxonomy of travel techniques. Therefore Travel techniques and universal UIs need to be better understood and implemented ,so users comfort and productivity in VE systems could be maximized [Bow98, BKH97].

Traveling techniques are classified based on the different interaction techniques, such as physical locomotion, steering, target-based, and manual manipulation. Physical locomotion, straight forward represents transformation from real world to virtual world. For example, walking locomotion, such as walking in place, and real walking. It provides better spatial knowledge. However, these techniques are fatigue, hence users get less benefits. On the other hand, The steering techniques are based on flying locomotion technique. It provides easy techniques to travel in the VE. However, the spatial knowledge is less than the physical locomotion techniques [KGMQ08].

(Bowman,1998) identified a set of quality factors which could be used to evaluate travel techniques. These include quantitative measures such as speed and accuracy, ease of use and ease of learning, and more subjective measures such as spatial awareness, presence, and user comfort. (Bowman, Koller, and Hodges, 1997) proposed a list of quality factors, since applications have different requirements for the travel techniques used. application may not be applicable for all factors as some factors may be irrelevant [Bow98, BKH97].

The travel technique has to promote speed, accuracy, spatial awareness which represents the user's awareness of its position and orientation within the VE, ease of learning, presence which is the users feeling of immersion within the VE, and information gathering where users have to obtain information from the VE. For example, in the in building navigation environment, Spatial awareness, presence, and ease of use are more important than speed. On contrast, The action games focus on maximizing speed, accuracy, ease of use, and gathering information [BKH97].

2.2 Related work

A number of researchers have been conducted to investigate navigation and travel techniques in immersive VEs. And to build an effective travel techniques, some knowledge about human navigation and motion control would have a great impact on the naturalness and usability of the techniques.

The study of (Bowman, David, and Hodges,1997) compares two different travel techniques, gaze-directed steering and hand directed steering. gaze directed steering, the user's view is used as the direction of movement, where user's view is the head tracking orientation for example. Hand directed steering, the hand orientation indicates the movement direction and orientation. Also the study mentioned that the basics of the quality factors for travel techniques are speed and accuracy. Since these factors indicate how fast and accurate the user is to reach his target. In the other hand, the users get fatigued with the slow and inaccurate techniques, since the user has to hold the device or press some buttons for a lengthy period of time. The study showed that hand directed techniques performed faster than gaze directed technique. In addition the study showed that both techniques produced an equal accuracy. The study suggested that hand directed techniques can be used in application which required speed and accuracy [BKH97].

(Bowman,1998) The interactivity within the VE is a key of success of the VE. The study proposes a technique for design and evaluation the interaction techniques, the design and evaluation exhibit some analysis and categorization of techniques, multiple performance measures are used here. The study focuses on the users performance in VE interaction, Speed is a quantitative determination, and is the primary consideration while evaluating new technique. Also accuracy is important and it is a quantitative measurement. In addition to speed and accuracy the interaction technique has to evaluate ease of use, ease of learning, comfortability, and the presence within the IVE.

According to the study any travel technique has to include three components the direction selection, velocity selection, and input condition [Bow98].

The study of (Chang Ha Lee Alan Liu Thomas P. Caudell,2009) focuses on the usage of the IVE in the medical training, as it provides a good context for practicing team work. According to the study the locomotion methodology should be intuitive , and should not distract users away from the task itself. In this paper, four locomotion techniques were studied, Look & Go, Push & Go, Point & Go, and Grab & Drag. Objective metrics had been used to evaluate the navigational efficiency [LLC09].

First, Look & Go, HMD is used as a navigation device. A button is clicked and held on the joystick to move. To move the individual's virtual body, users had to turn the head left or right, hence the virtual body turns in an anticlockwise or clockwise arc respectively. Constant speed is used here. Looking straight moves the individual in a straight line, and looking up or down leads to up or down motion respectively. According to the paper this approach is highly intuitive, and it is classified as gaze-directed steering [BKH97, LLC09].

Second, Push & Go, This techniques simulates the action of a gaming joystick. User click and hold a joystick button, next the joystick is pushed in the travel direction. The speed is determined by the joystick displacement from the start button, where user held the button, to the current position. In order to rotate user had to twist the joystick. The technique is familiar for video game players.

Third, Point & Go, The user click a joystick button, then a laser beam is emitted from the user's position. The beam indicates the movement direction. This technique is categorized as pointing/gesture steering paradigms [BKH97, LLC09].

Fourth, Grab & Drag, In this technique the user remains in the same position. However the environment moves. To move a joystick button is clicked, then dragging to the travel direction. To move up or down, joystick has to be moved up or down. The rotation based on the joystick twisting.

The results of the study showed that some factors, such as the ease with which waypoint are located, affect the overall performance. It indicates that Grab & Drag matches the requirement the most. It was the most time efficient and could be used for large VE. It also suggests that Grab & Drag allowed users to locate targets much faster. It was easy to learn and required the least head movement amount. However it shows significantly slower to reach target. It also mention that the point & go technique has to be improved in a way such that the user face the target. Physically turning the head or body is time consuming, and methods to facilitate locating the next waypoint should be developed [LLC09].

(kim - Gracanin - Matkovic - Quek,2008) present the Finger Walking In Place technique -FWIP-, the technique based on using a pair of fingers to travel in a virtual environment while sliding on the multi-touch surface. The multi-touch display device used is Lemur, provided by JazzMutant in 2005 [Jaz]. The viewpoint could be translated and rotate by moving the fingers in place. The techniques used for travel are Walking in place and rotate in place. The techniques are designed for travelling in a plane [KGMQ08].

Walking In Place: is used for translation in the VE. In order to move forward, the first finger touches the screen and slides down. When user moves his finger up, the second finger touches the device immediately. Users can move forward by repeating the same process. In order to move backward, the users do the same, but this time the fingers slide up and not down. Hence, the first finger touches the screen, slides up, and next leaves the surface. Then the second finger touches the device.

Rotate In Place: Two rotation techniques have been implemented and evaluated. There is another touch area for rotation, Hence users how need to rotate and translate simultaneously have to use their both hands. First technique, simulates the walking in place for rotation, where the angle for the user's displacement is mapped to the rotation angle of the VE. Second, simulates the same technique used for rotation using the mouse, where the dragging distance is mapped to the rotation angle.

The results of the study showed that -FWIP- could be used as a travel technique with inaccurate motion detection. In addition it mentioned that users where unfamiliar with the rotation techniques [KGMQ08].

The study of (kim - Gracanin - Matkovic - Quek,2009) proposed iPhone/iPod Touch implementation for navigation interaction originally implemented for large multi-touch devices. The interaction techniques are implemented for navigation in a CAVE. The techniques used FWIP which is similar to the one used in their previous work described above which is FWIP [KGMQ08]. However, due to different sensing techniques and smaller touch surface of the iPhone comparing to the lemur, the techniques were modified. The rotation technique were modified too, users have to use two fingers for rotation, for example dragging. Hence, the user touches the screen with two fingers, and the two fingers leave the screen at the same time. The results suggest that users may navigate with sufficient precision [KGMQ09].

2.3 Android Orientation Sensor

The Android SDK provides an interface to monitor the mobile phone sensors. One of the sensors is the Orientation sensor which provides feedback about the mobile orientation around the three Axes.

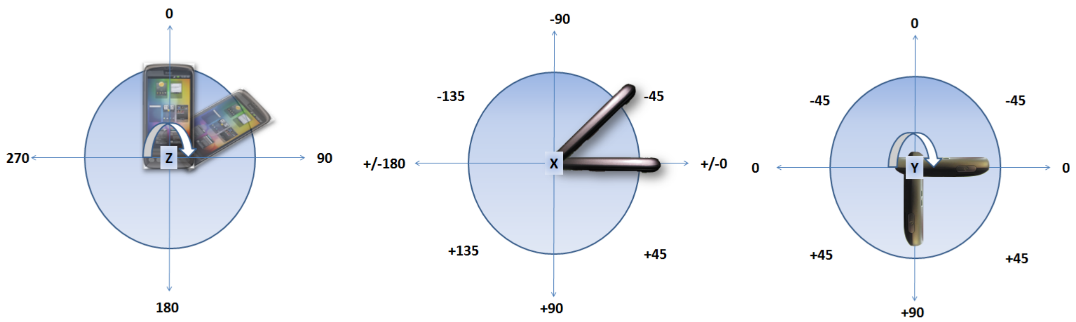
Coordinate Space is defined relative to the mobile phone screen in its default orientation. The origin of the coordinate system is the lower left corner of the screen, X-Axis horizontal and points to the right of the mobile, Y-Axis vertical and points up, and the Z-Axis extends outside the front face of the screen.

The Orientation sensors provide the mobile rotation angles around the three Axes, the angles are Azimuth, Pitch, and Roll.

Azimuth represents the mobile orientation around the Z-Axis, Where user rotates the mobile phone to the left or to the right around the Z-Axis. It represents the angle between the magnetic north direction and the Y-Axis, The angle ranges from 0 to 359 where 0 indicates the north, 90 the east, 180 the south, and 270 the west directions. The figure 2.1(a) shows the orientation of the mobile.

Pitch represents the mobile orientation around the X-Axis. Where user rotates the mobile phone up or down around the X-Axis. The angle ranges from -180 to 180, positive angle when the Z-Axis moves toward the Y-Axis. The figure 2.1(b) shows the orientation of the mobile.

Roll represents the mobile orientation around the Y-Axis. Where user rotates the mobile phone to the left or to the right around the Y-Axis. The angle ranges from -90 to 90, positive angle when X-Axis moves toward the Z-Axis The figure 2.1(c) shows the orientation of the mobile.



(a) The heading orientation (b) The pitch orientation (c) The roll orientation

Figure 2.1: Mobile Orientation

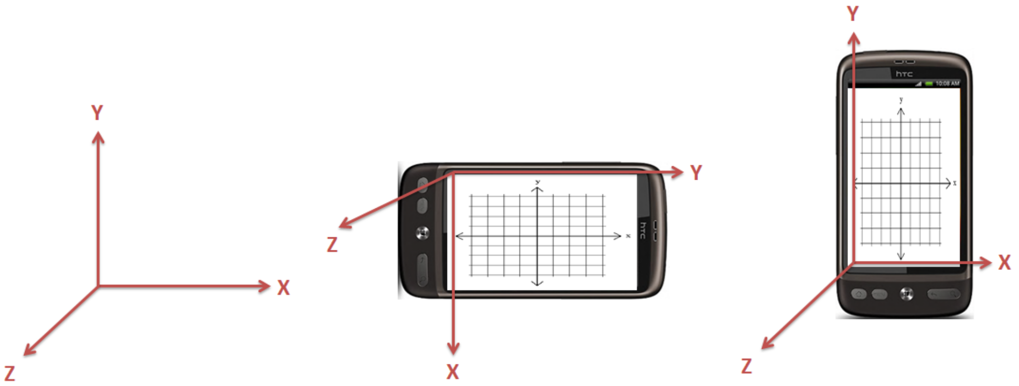
2.4 Coordinate System

The coordinate system for the android phones described in the previous section 2.3, The coordinate system for the mobile phone differs from the one used in the Terrain3D which is the VE used in this work.

Android phone The origin is the lower left corner of the screen. the X-Axis extends horizontally and to the right, the Y-Axis extends vertically and to the upper direction, and the Z-Axis extends outside the front of the screen. The figure 2.2 illustrate the coordinate system for the android phone.

The coordinate system is relative to the screen of the phone. The coordinates are fixed to the device itself. The coordinates are the same for both screen's orientation either portrait or landscape. The figure 2.2(c) shows the coordinate system of the mobile in the portrait orientation, and the figure 2.2(b) shows the coordinate system in the landscape orientation.

Virtual Environment - Terrain3D - The VE used in the work is an OpenGL version which uses a right-handed coordinate system [Kir09]. The figure 2.2(a) illustrate the right-handed coordinate system. However, According to the Terrain3D view, the coordinates are different, the X-Axis extends the depth of the screen, it points into the screen's center. The Y-Axis extends horizontally to the right of the screen's center. The Z-Axis extends vertically to the upper direction of the screen.



(a) VE Coordinate Systems (b) Landscape coordinate system (c) Portrait coordinate system

Figure 2.2: The Coordinate System

2.5 Networking

2.5.1 Transport Layer Protocol

The fundamental layer of the layered network architecture is the transport layer which provides a communication service between application running on several hosts in this work the IVE and mobile phone. There are different approaches to provide services by transport layer protocol such as TCP and UDP. The transport layer protocol affords logical communication between the different processes and the logically means that they do not have to be physically connected as they may be connected via different routers and connection types. Hence The processes are physically connected from processes' point of view and use the transport layer communication to send messages and communicate and ignore the fact how they are connected and how the messages are transmitted [KR04].

2.5.2 TCP - Transmission Control Protocol -

TCP service affords two services. First, Connection oriented service provides first a handshake phase between client and server, then full-duplex connection where both client and server can start sending messages to each other and at the end when application stops it has to tear down the connection. Second, Reliable transport service which guarantee the deliver of messages without errors and in proper order through acknowledgments and retransmissions protocol. TCP provides a congestion control mechanism which provides a fair share of bandwidth. TCP cannot guarantee a minimum transmission rate and delay guarantees [KR04].

2.5.3 UDP - User Datagram Protocol -

UDP service is a simple transport protocol and connectionless which means there is no handshake phase before communication starts. UDP provides no guarantee on deliver messages as it does not provide acknowledgments or retransmissions protocol and so UDP provides an unreliable data transfer service. There is no congestion control mechanism so there is no limits for data transferring rate. UDP does not provide delay guarantee. cannot guarantee a minimum transmission rate and delay guarantees [KR04].

2.5.4 Real time application and transport layer protocol

The main two differences between TCP and UDP from real time applications' point of view is congestion control and reliable transfer service which are provided by TCP protocol as mentioned. First, congestion control affects the transmission rate which affects the real time applications that require some minimum bandwidth. Second, reliable transfer service which provides acknowledgments and retransmissions can lead to slower transmission rate and that slow down the delivery of useful data. Hence UDP Service is used in real time applications as it is not providing either congestion control nor reliable service [KR04].

Due to the properties of the UDP service and it is more convenient for real time application then it is used in this work as a connection protocol between the client side which is the Mobile phone and the server side which is the Terrain3D environment.

3 Phone Based Motion Control Travel Technique

Travel techniques have to focus in three criteria which are how users can start and end the motion, indicate the motion direction, and finally alternate the velocity. In this chapter the different methodologies for traveling in the VE are discussed [BKH97, Bow98].

In this work we focus on travel technique and studying way finding and guiding people within the VE is out of the scope. There are three topics to be studied. First, Touch based motion control investigates how the user can translate within the VE using the handled device touch screen. Second, Acceleration based motion control techniques shows how user can control the translation speed within the VE. Third, Steering based motion control techniques shows how user can change their view point orientation within the VE.

Mobile tracking We study how the mobile phone is tracked, and provide information regarding the orientation of the mobile, in section 3.1.

Movement starts How users can start and end the displacement is discussed in the touch based motion control technique section 3.2

Movement direction How users can change the movement direction is explained in the touch based motion control techniques section 3.2.1

Movement velocity How users can control the movement speed. movement velocity is controlled differently from one technique to another. Three different techniques have been used to control the velocity and they are Manual control, Gas pedal control, and Finger speed control. The three techniques are studied in the acceleration technique section 3.3.

Graphical User Interface (GUI) The GUI design and the different approaches we followed are explored in section 3.5.

3.1 Tracking the mobile phone

In this section tracking the mobile phone is investigated. Here in the work tracking the mobile phone points to tracking its orientation and not its position. Many techniques to track the mobile phone are investigated.

Tracking techniques Two methodologies discussed to track the mobile phone. First, ART provides feedback about mobile orientation and position, uses infrared cameras to track the mobile. Second, Mobile phone inertial sensors used to track the mobile phone, for example accelerometer, orientation sensors, and magnetic sensor.

The work focus on the second option and how to track the mobile using the orientation sensor is investigated. How the orientation sensors work and its outputs is discussed in the background section 2.3. In the next subsections two different techniques to track the phone using the inertial sensors are discussed.

3.1.1 Mobile Orientation Sensor

The mobile orientation sensor enables to have a feedback about the mobile orientation around the three Axes. The three angles represents the mobile orientation are heading, pitch, and roll, as described before in the background section 2.3.

The application is using the mobile sensor feedback to get the mobile orientation around all axes. Therefore in this techniques the mobile phone is tracked using the mobile orientation sensor, no use for magnetic sensor or accelerometer.

3.1.2 Calculating heading for tracking

The methodology is to calculate the heading value based on the reading of the magnetic field sensor and a tilt sensor, the tilt sensor provides feedback about mobile orientation for the pitch and the roll.

The work of (Michael J. Caruso,2000) represents this technique, The tilt sensor determines the roll and pitch angles, the sensor senses the direction of the gravity, It includes for example an accelerometers, electrolytic based tilt sensors. the mobile magnetic sensor is three axes magnetic sensor [Car00].

The heading calculation is based on the three magnetic components. The compass with roll (Theta) and pitch (Beta). The magnetic readings for the X, Y, and Z is transformed to the horizontal plane (Xh, Yh) by applying the rotation equations shown in equation 3.1, where (Xh, Yh) are the earth's horizontal magnetic field components. The figure 3.1 shows an example for the compass system [Car00].

$$\begin{aligned} Xh &= X \times \cos(\beta) + Y \times \sin(\theta) \times \sin(\beta) - Z \times \cos(\theta) \times \sin(\beta) \\ Yh &= Y \times \cos(\theta) + Z \times \sin(\theta) \end{aligned} \tag{3.1}$$

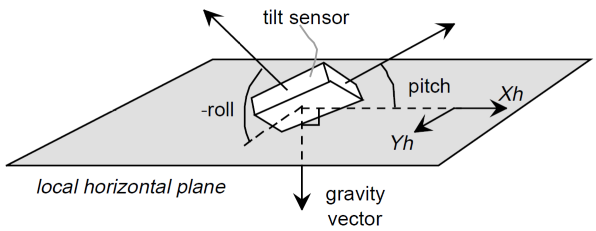


Figure 3.1: A Compass System shows tilt sensor angles referenced to the local horizontal plane [Car00]

The magnetic components are calculated in the horizontal plane defined by the gravity (Xh, Yh), The heading can be determined using equation 3.2. Then the heading calculations follows the sine and cosine lookup table, due to the arcTan limits, hence the heading calculations must account for the sign of the Xh and Yh readings as shown in 3.3.

$$Heading = \arctan\left(\frac{Yh}{Xh}\right) \tag{3.2}$$

$$\begin{aligned} Heading for\,(Xh < 0) &= 180 - \arctan\,(Yh/Xh) && (3.3)\\ for\,(Xh > 0, Yh < 0) &= -\arctan\,(Yh/Xh)\\ for\,(Xh > 0, Yh > 0) &= 360 - \arctan\,(Yh/Xh)\\ for\,(Xh = 0, Yh < 0) &= 90\\ for\,(Xh = 0, Yh > 0) &= 270 \end{aligned}$$
$$\tag{3.4}$$

The results of this technique was inaccurate and was not steady, the heading reading flickering too much, hence the orientation sensors provide more accurate and steady data comparing to this technique.

Tracking technique implemented Due to the fact that the second option for using the magnetic sensor and the orientation to get the heading leads to inaccurate and unsteady readings. Therefore the using of the orientation sensor to obtain mobile orientation around the axis are preferable. So the work uses the orientation sensor to get the mobile orientation around all the axis.

3.2 Touch based motion control techniques

Translation in the VE simulates the locomotion technique on the multi-touch display of the handled device using the thumb. The technique simulates the finger gestures on the screen, the finger displacement on the mobile screen will be mapped to a displacement in the virtual environment.

Movement starts when screen is pressed. The user can start moving just by touch the mobile screen and start moving while his finger is down in any direction. When the finger is up the deceleration phase will start and the displacement will decrease until finally reach zero and the displacement reaches to an end.

Movement direction is indicated by the finger movement direction on the screen. Once the finger is down the start point detected and as the finger move the displacement is calculated relative to the current point. After displacement calculated, the application gets the normal direction vector and hence the movement direction is represented by this vector.

3.2.1 2D and 3D Translation

The idea is moving the finger from one point to another point on the touch screen. The device will recognize the gesture and start moving. First the translation on the mobile is in 2D which means it's only in x and y axes. This displacement is mapped to a 2D displacement in the VE then it's only a displacement in a plane. The third dimension is added to have a 3D translation through using the mobile orientation to indicate the movement direction thus having a 3D movement in the VE.

3.2.2 How displacement works

The displacement will be separated into two phases for the mobile client and IVE server.

First, movement starts when finger is down on the mobile screen next the mobile will recognize the gesture and recognize both direction and speed, speed will be calculated differently depending on which technique is used, later on data is sent asynchronously to the IVE server. The equation for the mobile displacement vector is represented below 3.5.

$$MDVector = Speed \times NormalDirectionVector \tag{3.5}$$

Second, IVE simulates the displacement in two phases the movement phase and the deceleration phase. First, the movement phase focus on calculating the displacement vector as in equation 3.6, the vector is calculated by multiplying the displacement vector received from the mobile with the height rate - will be discussed in **section**3.3.1 - and the elapsed time between each two frames, the elapsed time is used so in each second we move with the displacement rate which means if the terrain has 60 frames per second accordingly displacement is divided upon the 60 frames. Second, deceleration phase starts when finger is up and it provides a smooth transition between movement and pause, the IEV will decelerate the speed till it reaches zero hence the movement reaches to an end, the start speed is defined with the mobile phone.

$$IVEDisplacement = MDVector \times HeightRate \times ElapsedTime \tag{3.6}$$

3.3 Acceleration based motion control techniques

Movement velocity is controlled differently from one technique to another. Three different methodologies have been used to control the velocity and they are Manual control, Gas pedal control, and Finger speed. The three techniques will be studied separately in the next subsections.

3.3.1 Height Rate

IVE height rate The displacement speed is in a direct proportion with the IVE height and hence near the ground the displacement is slower and as the user get higher and higher the displacement rate increases and continue increasing till the height reaches 10,000 KM. The height rate equation 3.7.

$$HeightRate = 1 + \frac{IVEHeight}{10000} \tag{3.7}$$

3.3.2 Manual control technique

Manual control gives the user the freedom to control the speed and fix it at anytime. The idea is to use a speed bar to control the speed values. The user select the desire speed and the application will use this speed as a factor to move with. The direction of the movement will be indicated by the normal direction vector of the finger. The speed bar indicates the speed in both direction the x and y axes.

In this technique no matter how far the user moves his finger as the normal direction vector is multiplied by the speed and this will be the displacement factor. The displacement factor is sent to the IVE and then will be multiplied with the time between each two frames. The speed bar values ranges from 0 to 100 .However, the speed used by this technique is the square of the speed bar values in order to increase the displacement value, and hence the speed ranges from 0 to 10000.

The whole equations for the displacement and speed are shown below. The increase in the speed is shown in the **figure** 3.2.

$$ManualSpeed = SpeedBar^2 \tag{3.8}$$

From equations 3.5, 3.6, and 3.8

$$\begin{aligned} MDVector &= SpeedBar^2 \times NormalDirectionVector \\ IVEDisplacement &= MDVector \times HeightRate \times ElapsedTime \\ &= \left(SpeedBar^2 \times N.D.V\right) \times HeightRate \times ElapsedTime \end{aligned} \tag{3.9}$$

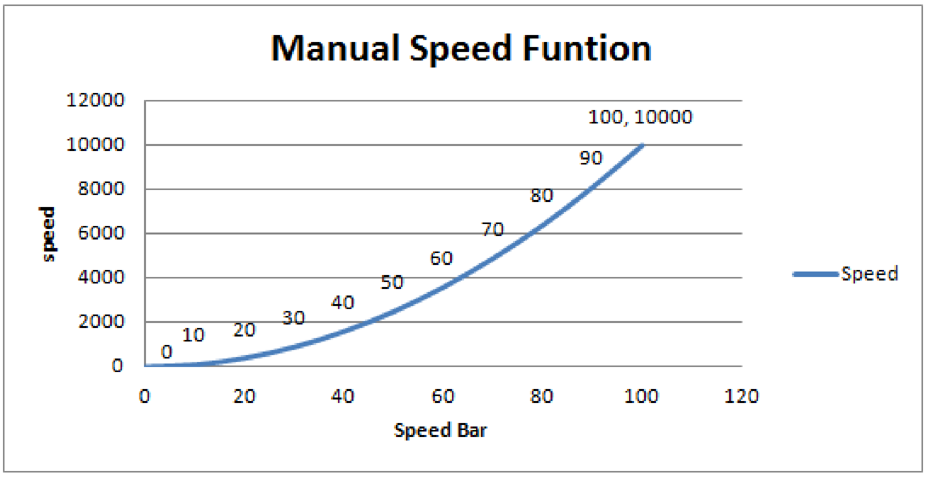


Figure 3.2: Manual speed function

3.3.3 Gas pedal technique

The car accelerates more and more as the driver presses the gas pedal harder. The same technique is used here. The far the user moves his finger from the start point, the higher the displacement speed is. Therefore the displacement speed depends on the displacement vector. Then users start traveling in the VE with this speed. The speed in any of the axes depends in the displacement in this axis.

The displacement factor is calculated by multiplying the speed with the normal direction vector. The speed ranges from 0 to 10000. To get the maximum speed user has to make a big displacement. Consequently, user has to start from the end of the screen to be able to reach the maximum speed.

The whole equations for the displacement and speed are shown below. The increase in the speed is shown in the **figure** 3.3 only for the Y-Axis.

$$gasPedalSpeedVector = 0.04 \times (displacementVector)^2 + 500 \tag{3.10}$$

From equations 3.5, 3.6, and 3.10

$$\begin{aligned} MDVector &= \left(0.04 \times (D.V)^2 + 500\right) \times NormalDirectionVector \\ IVEDisplacement &= MDVector \times HeightRate \times ElapsedTime \\ &= \left(\left(0.04 \times (D.V)^2 + 500\right) \times N.D.V\right) \times HeightRate \times ElapsedTime \end{aligned} \tag{3.11}$$

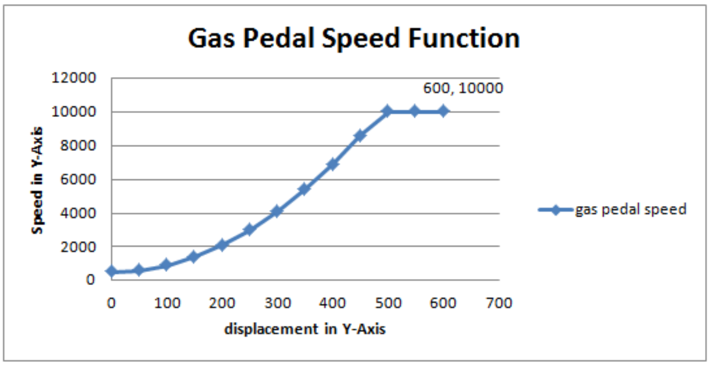


Figure 3.3: gasPedal speed function

3.3.4 Finger speed technique

In this technique the user can control the speed using his finger again. This time the application does not care about the displacement in fact it does care about the speed of your finger movement. Hence the faster the users move their fingers on the screen the faster the displacement speed is. Then user can move with this speed as long as his finger is down on the screen. The speed is the displacement over time, here the displacement is the distance moved by the finger and the time is the duration of this movement.

The displacement factor is calculated by multiplying the speed with the normal direction vector. The speed ranges from 0 to 10000. To get the maximum speed user has to move his finger fast enough. When the finger is up the deceleration phase will start and hence to keep moving user should not move his finger up from the screen.

The whole equations for the displacement and speed are shown below.

$$\begin{aligned} speed &= \frac{distance}{time} \\ fingerSpeed &= ((15 + speed))^2 \end{aligned} \tag{3.12}$$

From equations 3.5, 3.6, and 3.12

$$\begin{aligned} MDVector &= (15 + speed)^2 \times NormalDirectionVector \\ IVEDisplacement &= MDVector \times HeightRate \times ElapsedTime \\ &= (15 + speed)^2 \times HeightRate \times ElapsedTime \end{aligned} \tag{3.13}$$

3.4 Steering based motion control techniques

It's Critical how the user is going to control the system and change the view point orientation in the virtual environment.

In this study four techniques are studied and then be evaluated. Each technique has its own degrees of freedom and The techniques are based on the steering/flying technique. The four techniques are Rotate By Heading, Rotate By Roll, Rotate By Roll with fixed horizon, and Merged Rotation which is a sort of combination between rotate by heading and rotate by roll.

3.4.1 Overview

The view point motion orientation is controlled based on the mobile phone orientation. The orientation of the mobile phone is obtained from the mobile orientation sensor which provides angles for the azimuth , pitch and roll. The change in the mobile orientation will lead to the same or different change in the view point orientation.

Azimuth represents the rotation around the Z axis and it has values from 0 to 360. Pitch represents the rotation around X axis and ranges from -180 to +180. Roll represents the rotation around Y axis and ranges from -90 to +90 [Goo].

The implementation is divided into two parts the client side and the server side. The client side is more related to the user interaction with the phone. The server works with the IVE ,control and smooth the change in the view point orientation.

First the client side, The Rotation phase will start exactly when the finger is pressed in the screen so the start orientation of the mobile phone is indicated and a message will be sent to the server indicating the rotation phase. Then the application will listen to the orientation sensor and calculate the delta angles between the current orientation and the start orientation. A threshold values are added to control sending data to the server as sending data starts if the delta angle exceeds the threshold value which is normally 10 degrees. At last, the client will divide the delta angles with factor to smooth the rotation and send the data to the server depending on the orientation of the mobile since the change in the azimuth will differ if the mobile orientation is landscape or portrait. Normally the delta angles sent will differ depending on which technique is used. The application stops sending data when the finger is up and send a message to stop the rotation phase.

Second the server side, The server will start the rotation when it receives the message from the client and then it receives the delta angles. Then the server will multiply the delta angles with the elapsed time between each two frames and add the value to the orientation values of the IVE -azimuth, pitch and roll-. The aim of multiplication with the elapsed time is to get a factor to rotate with each frame and this will smooth the transition from one orientation state to another. The server will stop the rotation phase once it receives the stoppage message.

In general when user need to steer in the VE then the user has to press on the screen, and start steering with the mobile phone and as long as the orientation changes the user will keep steering. When user needs to stop steering, he has two options either to get his finger up or set the mobile back to its start position.

In the next sections each technique will be studied separately and point to the differences between each one of them.

3.4.2 Rotate By Heading Technique

The rotate by heading technique is based on the orientation of the mobile which means that the change in the azimuth of the phone will lead to change the azimuth value of the view point in the IVE and same goes for the pitch. The technique has two degrees of freedom the pitch and azimuth, The roll is the same so the horizon is always fixed in this technique.

This technique simulates more like a bicycle, since when user turns the bicycle to the left or the right, the view changes in the same way. And as long as the user is turning, the view is still turning till user turns back to the start position. Figure 3.4 illustrates the mobile orientation used in this technique (Appendix A.1.1).

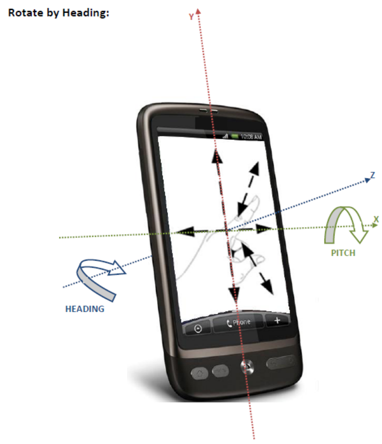


Figure 3.4: Rotate By Heading Orientation

At the client side when the rotation phase is started the delta angles of pitch and azimuth will be calculated and sent if they exceeded the threshold values, then delta angles are divided with the factors (1.5 for pitch and 3 for azimuth), then data is sent to the server depending on the orientation of the mobile since the delta azimuth will differ if the mobile orientation is landscape or portrait.

Second the server side, The server will multiply the azimuth and pitch delta angles with the elapsed time as explained before ,then the values will be added to the azimuth and pitch values for the IVE. There is no use of the roll value here.

3.4.3 Rotate By Roll Technique

The rotate by roll technique is simulating the flying technique . The change in the mobile roll will be mapped to change in both the azimuth and roll values in the IVE and change in pitch will change the IVE pitch. The flying technique is simulated here as it is more like flying in a plane where the horizon changes when the plane is turning. This technique has two degrees of freedom. Figure 3.5 illustrates the mobile orientation used in this technique (Appendix A.1.2).

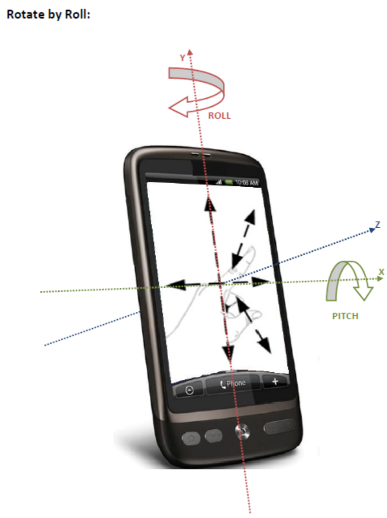


Figure 3.5: Rotate By Roll Orientation

At the client side when the rotation phase is started the delta angles of roll and pitch will be calculated and divided with a factor = 1.5 and then data is sent to the server depending on the orientation of the mobile since the delta roll will differ.

Second the server side, The server will multiply the roll and pitch delta angles with the elapsed time as explained before ,then the value of delta pitch after multiplication will be added to the pitch value of the IVE and the roll value after multiplication will be added to both the roll and azimuth values of the IVE. However, the rolling angle of the IVE is limited to only 10 degrees.

When the mobile is in the horizon a message will be sent to the server to reset the roll value to the horizon again. So if the user is rotating the mobile roll for example to the left and then getting back to the horizon then a message will be sent to the server to reset the roll back to zero. When the server receives the reset roll message it starts getting the roll back again to zero by adding a factor each frame using the same technique used for changing the IVE orientation, but this time with delta angle equals to 10, explained in details in the section 4.6.2.

3.4.4 Rotate By Roll With Fixed Horizon Technique

The rotate by roll with fixed horizon technique is similar to rotate by roll technique, the only difference is that the horizon is all the time fixed. Hence the mobile roll is mapped to the azimuth of the IVE and mobile pitch is mapped to the pitch of the IVE. This technique simulates the car steering technique as while rotating the horizon is always fixed not like the plane where horizon changes. Figure 3.6 illustrates the mobile orientation used in this technique (Appendix A.1.3).

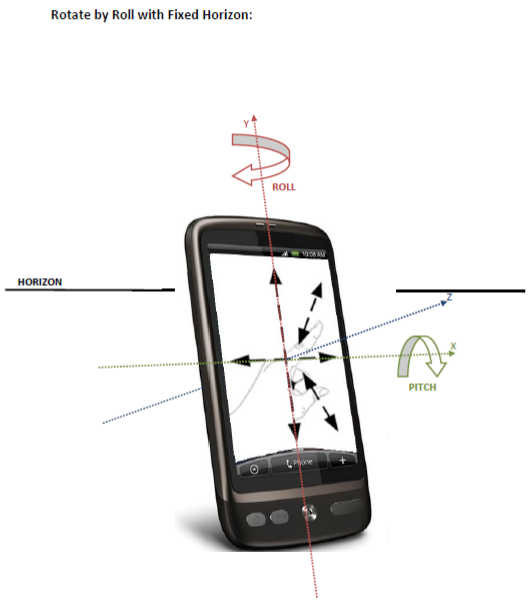


Figure 3.6: Rotate By Roll with fixed horizon Orientation

At the client side when the rotation phase is started the delta angles of roll and pitch will be calculated and divided with a factor = 1.5 and then data is sent to the server depending on the orientation of the mobile since the delta roll will differ.

Second the server side, The server will multiply the roll and pitch delta angles with the elapsed time as explained before ,then the value of delta pitch after multiplication will be added to the pitch value of the IVE and the roll value after multiplication will be added only to the azimuth value of the IVE.

There is no need for reseting the roll here as the horizon is all the time fixed.

Rotate by roll with fixed horizon is suitable for in building navigation, where the user needs the horizon to be all the time the same. As any change in the horizon may affect the view of the users and distract him while navigating.

3.4.5 Merged Rotation Technique

This technique is a sort of combination between rotate by heading and rotate by roll techniques. The only difference between this technique and rotate by roll is in the azimuth. The change in the mobile orientation will be mapped to change in the same orientation of the IVE except for azimuth. The delta azimuth will be calculated based on the change in both the roll and azimuth values of the mobile.

In this technique user has more degrees of freedom comparing to the other techniques where degrees of freedom are restricted. Figure 3.7 illustrates the mobile orientation used in this technique (Appendix A.1.4)

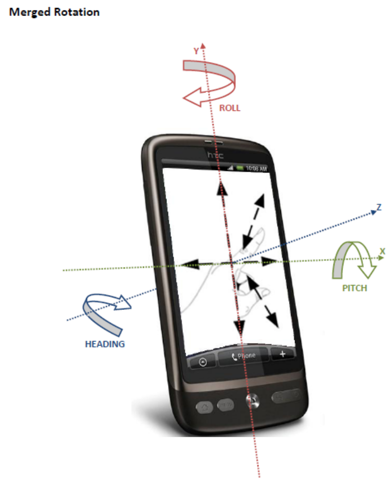


Figure 3.7: Merged Rotation Orientation

First client side, the same technique will be used here too. The delta angles will be calculated using the orientation sensor as usual. The delta angles will be sent to the server as before except foe the azimuth since the azimuth in this technique will be considered as an interpolation between the delta roll and delta azimuth, explained in details in the section 4.6.1. The delta azimuth sent to the server will be an addition of both change in azimuth and roll of the mobile orientation. Then the data will be divided by a factor (3 for azimuth and 1.5 for the rest) and sent to the server depending on the orientation of the phone as both the delta azimuth and delta roll will differ. When the mobile is in the horizon a message will be sent to the server to rest the IVE to horizon (roll = 0) as the rotate by roll technique.

Second the server side, The server will multiply the azimuth, roll ,and pitch delta angles with the elapsed time as explained before ,then the values of delta angles after multiplication will be added to the IVE angles to change the view point motion orientation. However, the rolling angle of the IVE is limited as the rotate by heading technique. When it receives the message for resetting the roll it uses the same technique as rotate by roll technique.

3.5 Graphical User Interface

This section focus on the Graphical User Interface -GUI- for the handled device application. The aim of the study is to make the GUI as simple as possible. Hence users can interact easily with the application. The GUI has to provide the user for all relative information.

3.5.1 Design

The design of the GUI has to be simple, so users can understand the main functionality of the application and how to interact with it. The Design is shown in the figure 3.8. The main view provide the start and stop buttons, touch area, and feedback area. First, start button, starts the application and initializes the connection with the server of the VE, application starts listening to the sensors hence application is ready for users interaction. Second, stop button to stop the application so the application does not listen any more to the sensors and stops sending data to the VE server. Third, touch area is used for the user interaction with the VE, so this is the area provided for user to navigate in the VE. Fourth, feedback or debug area provides users for more information regarding the mobile orientation, selected acceleration technique, selected steering technique, and finally navigation speed.

Feedback The GUI provides users with instance feedback for each performed action, for instance if user presses on the start button, then button colors changes to dark green. Same feedback goes for the stop button as it turns to dark red. However, it also shows a pop menu asking the user if he really wants to stop the application or just pressed by mistake. There is a pop dialog also shown when the server is connected or there is any error connecting to the server.

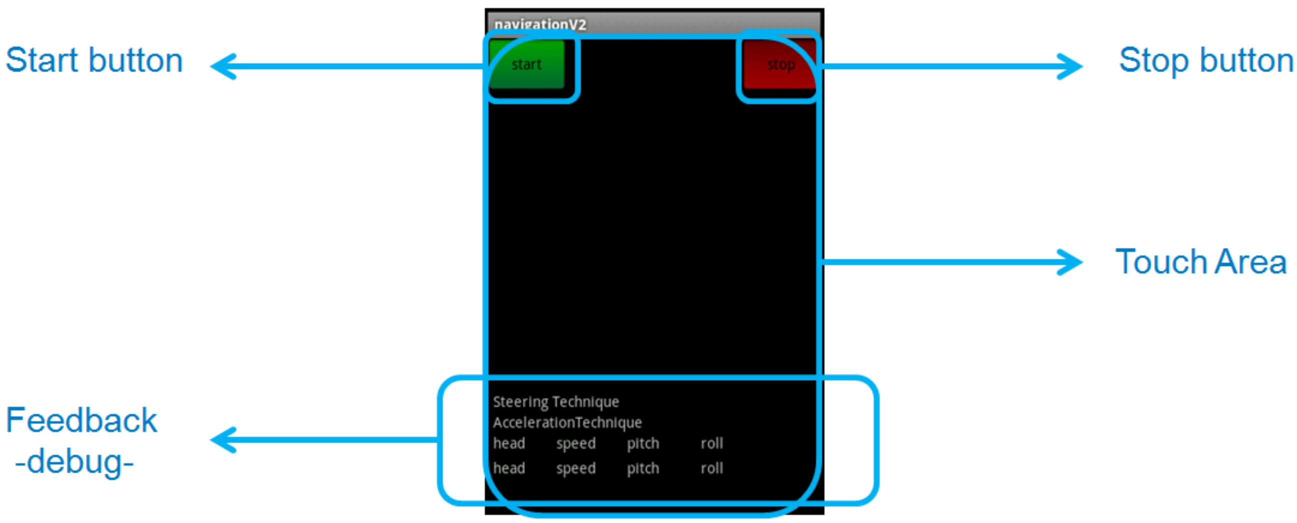


Figure 3.8: Handled device GUI

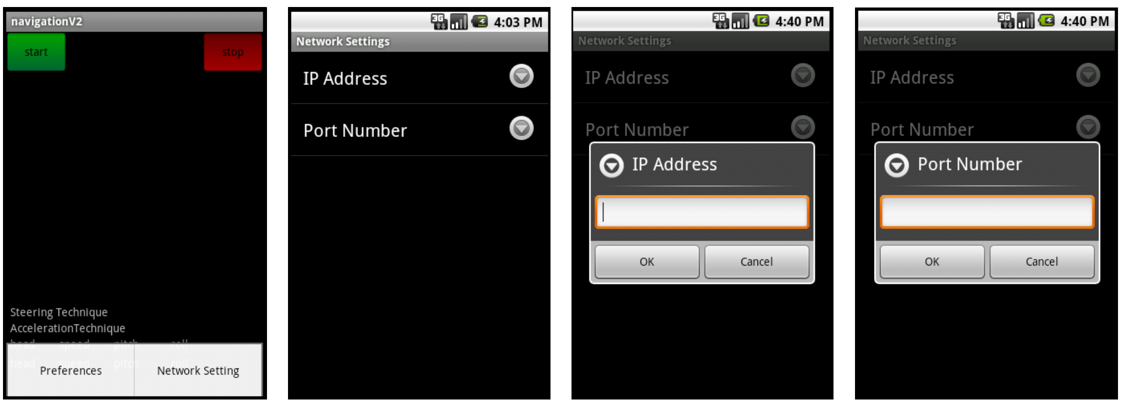
The GUI design will defer depending on the acceleration technique used, as for the manual technique a speed bar will be required, and hence the only difference in the GUI exists in the manual speed. All other acceleration techniques share the same interface. Figure 3.9 showed the differences in the two interfaces.

(a) Manual Speed Technique (b) Other Techniques

Figure 3.9: Different GUI Design

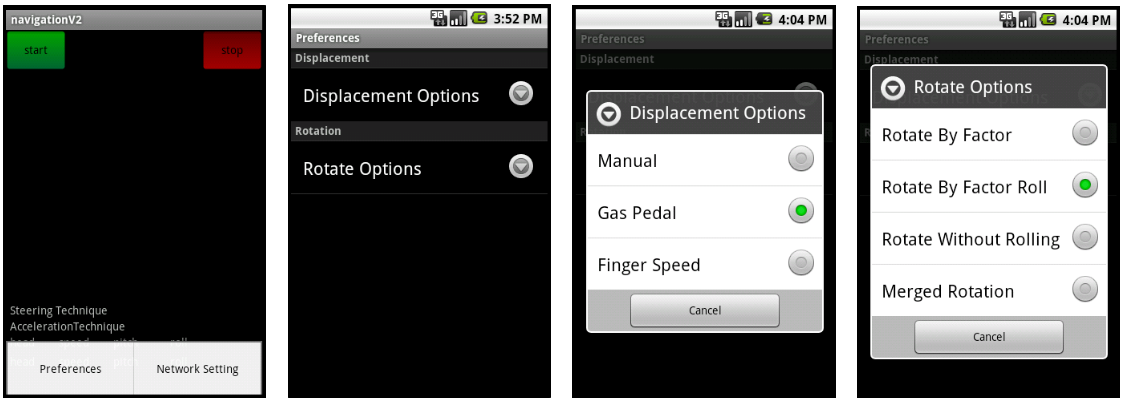
3.5.2 GUI Menu

The application menu is divided into two parts, first one for the networks settings and the second for the applications preferences. First, network menu includes two options the server IP and port number, both of them for the server connection, user has to provide the application with the server IP and port number to communicate with, illustrated in figure3.10. Second, preferences menu includes another two options, the steering technique and acceleration technique, user can change any of the techniques depending on his preferences, shown in figure 3.11.



(a) Menu (b) Network Menu (c) IP Menu (d) Port Numbers Menu

Figure 3.10: Network Settings Steps



(a) Menu (b) Preferences Menu (c) Acceleration Menu (d) Steering Menu

Figure 3.11: Preferences Settings Steps

3.5.3 Navigation Steps

Finally how users can start navigating in the VE. The user first presses the start button, hence the connection to the VE server initialized and application starts listening to the device sensors and tracking the device orientation. Next user can translate in the VE using the touch area provided in the application interface and he can steer by pressing in the touch area and start rotating the mobile phone depending on the steering technique used. At last if users want to stop navigating then he can press the stop button at any time, when the button is pressed application stop listening to the VE server and tracking the mobile orientation. If user wants to navigate again, then all what he needs to do is just to press the start button again.

4 Implementation

In this chapter, the implementation of the different travel metaphors used for interaction in VE is discussed here. The travel techniques are discussed in the phone based motion control travel technique (chapter 3). This chapter follows exactly the same structure as (chapter 3). First, the network part which illustrates the server and client connection in the VE and illustrates the server language. Second, discussion about the tracking techniques used. Third, How the touch based motion control, and the using of the touch screen for translating in the environment. Fourth, The different acceleration techniques to control the velocity. Fifth, we explore the implementation of the translation techniques and how it is mapped to the IVE. Finally, the implementation for the different steering techniques and shows the differences and similarities between them.

In this chapter the technical implementation details for the travel techniques used are discussed.

4.1 Network between mobile and Terrain3D

The work is divided into two parts the mobile phone and the IVE and a sort of communication between the two running processes has to be established. The transport protocol used is UDP as it is more convenient for real time application as discussed before in **section** 2.5.4. The network between mobile and IVE is illustrated in **figure** 4.1, the figure showed the network architecture. The mobile phone and the VE are connected wirelessly, therefore it enables mobility of the device. The client and server topology is used here, the server in this case is the VE, where the client is the mobile phone. Hence the mobile phone sends data a asynchronously to the VE through the network.

The Implementation of the server is multi threaded as the server is working in the background of the IVE and handle each message form the client in a separate thread, Consequently the IVE can minimize the overhead due to the handling and translating the messages. In the other hand, the implementation enables the VE to establish multiple servers. However, in this work the first server will be used. The server enables both protocols UDP and TCP. However, UDP is used as default and also used in the user evaluation.

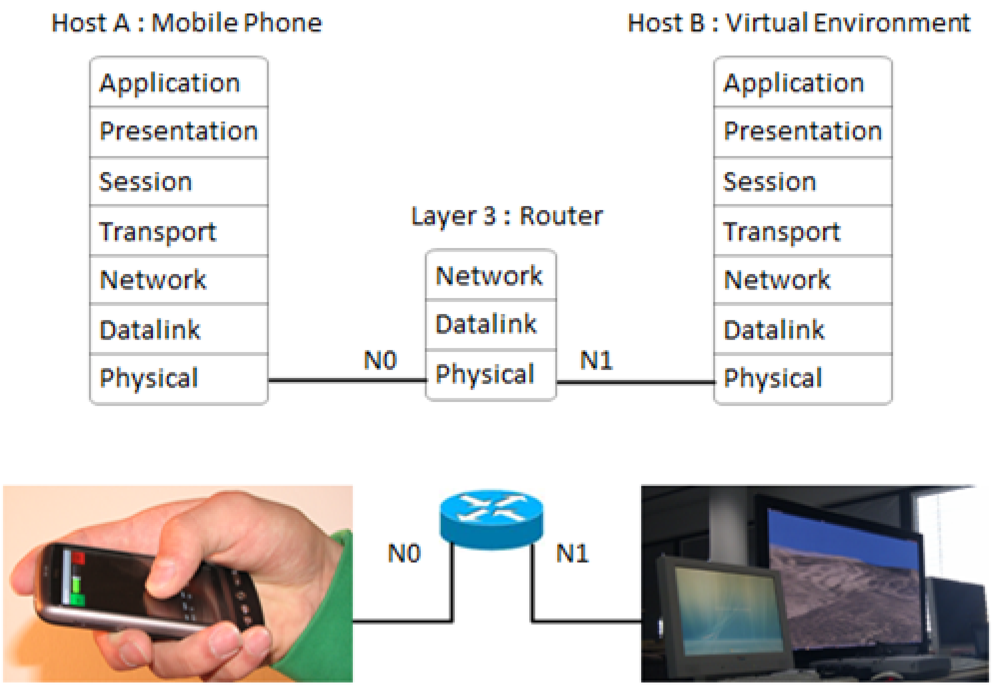


Figure 4.1: Network diagram between mobile and Terrain

4.1.1 Server task

The server provides the connection between the IVE and the mobile. The server provides the IVE with the whole information. First, server provides information about different techniques used for travel techniques. Second, Server provides data about the orientation of the mobile , delta angles for heading ,pitch and roll ,and finally send the rotation factors for the IVE view point orientation. Third, Server provides data about the motion direction and speed. It also supports the reset roll options, which is used by rotate by roll and merged rotation, this option reset the horizon back to its normal position which is zero.

4.1.2 Server language

The Language used between the server and client is showed in **the figure** 4.2, the figure also indicates which messages are supported by each technique. The client sends messages to server according to the following language. It includes the start of the application, acceleration techniques, steering techniques, rotation factors, displacement, and finally to stop the application.

Figure 4.2: Server Language

Server Message	Example	Description	Travel Technique	System Control Technique		
				Rotate By Factor	Rotate By Factor Roll	Merged Rotation
start	start	Sent when application started -start button pressed-	☑ Presented	☑ Presented	☑ Presented	☑ Presented
stop	stop	sent when application stopped -stop button pressed-	☑ Presented	☑ Presented	☑ Presented	☑ Presented
rotateByFactor	rotateByFactor	rotate by factor technique selected -from preferences-	☑ Presented	☑ Presented	☐ Not Presented	☐ Not Presented
rotateByFactorRoll	rotateByFactorRoll	rotate by factor Roll technique selected -from preferences-	☑ Presented	☐ Not Presented	☑ Presented	☐ Not Presented
mergedRotation	mergedRotation	Merged Rotation technique selected -from preferences-	☑ Presented	☐ Not Presented	☐ Not Presented	☑ Presented
startRotate	startRotate	Start Rotation Phase -finger pressed on screen-	☑ Presented	☑ Presented	☑ Presented	☑ Presented
rotateFactors 'H' 'P' 'R'	rotateFactors 0.0 5.0 20.0	sent when the mobile orientation changes H,P, and R represents delta angles for heading, pitch and roll	☑ Presented	☑ Presented	☑ Presented	☑ Presented
resetRoll	resetRoll	Sent when mobile is held horizontly -roll = 0-	☑ Presented	☐ Not Presented	☑ Presented	☑ Presented
stopRotate	stopRotate	Stop Rotation Phase -finger is up-	☑ Presented	☑ Presented	☑ Presented	☑ Presented
displacement 'X' 'Y' 'V'	displacement 500 100 1	sent when finger is moved in the screen. X,Y represents normal finger displacement vector. V represents the speed	☑ Presented	☑ Presented	☑ Presented	☑ Presented
deAcceleration 'D'	deAcceleration 50	Stop the displacement when finger is up and D represents the deceleration rate	☑ Presented	☑ Presented	☑ Presented	☑ Presented

4.2 Tracking the mobile phone

Tracking the mobile phone using only the orientation sensor is the technique used in this work, since it is more accurate and steady comparing to using the orientation and magnetic sensors as discussed in (section 3.1.2). The application will first register its context to listen to the orientation sensor, in order to listen to the sensor we first starts with unregister the application from listening to the sensor using the sensor manager, which provides an interface to access the device's sensors. After that the orientation sensor is registered to the application using the sensor manager [Goo]. The code for listening to the orientation sensor is shown below.

```
public void startSensors(){
        // un-register this activity from listening to the sensors
        sensorManager.unregisterListener(this);
        // register this context to listen to the orientation sensor
        // SENSOR_DELAY_UI is the rate of the orientation sample
        sensorManager.registerListener( this,
                                        SensorManager.SENSOR_ORIENTATION,
                                        SensorManager.SENSOR_DELAY_UI);
}
```

Second, the application starts listening to the sensor and reading its output, the sensor output is a float array of three values represents the readings of azimuth, pitch, and roll [Goo]. Next data are filtered using an average filter for some last previous readings. The code is shown below.

```
public void onSensorChanged(int sensor, float[] values) {

        synchronized (this) {

        float head=0,pitch=0,roll=0;

        if(sensor == SensorManager.SENSOR_ORIENTATION ){

                currentAngles[0] = averageFilterHead.getAverage((int)values[0]);
                currentAngles[1] = averageFilterPitch.getAverage((int)values[1]);
                currentAngles[2] = averageFilterRoll.getAverage((int)values[2]);
```

When user stop the application, there is no need to listen to the sensor. Hence application will unregister the context (application) from listening to the orientation sensor.

```
public void stopSensors(){
        sensorManager.unregisterListener(this);
}
```

4.3 Touch based motion control techniques

This section study the implementation of the touch based motion techniques, specially the translation techniques and how users can translate and control their displacement within the VE. As the translation depends on both the direction and speed according to the equation 3.5, The user has to control his movement direction, and the movement direction could be changed using two methodology. First, using the touch screen to have 2D translation. Second, using the mobile orientation to change the view point orientation direction. Consequently, the third direction is added, then we have 3D translation.

4.3.1 Touch Area Listener

The translation techniques are based on the touch screen and simulates the locomotion using the finger. Hence the first step is to register the touch area - label - in the GUI to listen to the touch listener.

```
((View)findViewById(R.id.view)).setOnTouchListener(this);
```

As a result of the touch listener, whenever the user touch the screen, the onTouch(View v,MotionEvent e) method is called. The method is called based on an action on the touch screen, the actions that matter for this work are ACTION_DOWN called when finger is down on the screen, ACTION_MOVE called when the finger moves on the screen, and ACTION_UP called when finger is up from the screen. The following code shows the actions taken when the onTouch method is called.

```
public boolean onTouch(View v,MotionEvent event) {
        ..
        switch (event.getAction() & MotionEvent.ACTION_MASK) {
                case MotionEvent.ACTION_DOWN:
                        ..
                        startMoveTime = System.currentTimeMillis();
                        start.set(event.getX() , event.getY());
                        ..
                        break;
                case MotionEvent.ACTION_MOVE:
                        ..
                        end.set(event.getX(),event.getY());
                        move(false);
                        ..
                        break;
                case MotionEvent.ACTION_UP:
                        ..
                        move(true);
                        ..
                        break;
        }
        return true;
}
```

the code shows that when the finger is down then we start the timer, which is used for getting user movement speed. Then the start point is set. While finger is moving, the end point is set and the method move(boolean lastMove) is called and its task is to get the movement direction and calculate the speed, then send data to the VE. When finger is up, the method move(boolean lastMove) is called again. However this time the boolean variable will be true, hence the application will send the deceleration message too, which will be explained in the next section 4.4.

4.3.2 Indicate movement direction

Based on the touch techniques user can have a 2D translation. Hence user can change his movement direction in a plane. the movement direction is indicated based on the normal movement direction of the finger on the screen, where the first point and end point of the movement indicate the vector, next the normal direction vector is calculated. The below code explains how the normal direction vector is calculated.

```
// The difference between start and end points
PointF disp = getDisplacementVector(end, start);
float displacement = (float) Math.sqrt(disp.x*disp.x+disp.y*disp.y);
PointF normalDispVector = new  PointF(disp.x/displacement, disp.y/displacement);
```

Where the method getDisplacementVector returns the displacement vector using the start and end points.

The 2D translation in the virtual environment is calculated based on the following equation 3.5, hence the normal direction vector indicates the translation direction in a 2D plane. The third dimension for movement is added using the steering techniques 4.6. The speed is explained in the section 4.4, and in the section 4.5 the translation techniques is explained.

4.4 Acceleration based motion control techniques

The acceleration techniques control the velocity of translation within the VE as explained in the section 3.3. The three acceleration techniques are manual control, gas pedal, and finger speed, they are based on the touch techniques. All the calculations for the speed is done in the client side (mobile phone). The implementation of each technique is described below.

Manual Control The code below indicates how the speed is detected using the speed bar. The application gets the speedbar value, then the speed is calculated based on the equation 3.8. The seekBarValue represent the value of the speed bar, and the variable Vf represents the speed of the translation.

```
private void move( boolean lastMove ){
        // finger movement speed &  deceleration
        float Vf,Vx,Vy,deAcceleration;
        ..
        // Manual speed control using the bar
        if(displacementOption == 0){
                ..
                // get the value of the seek bar
                float seekBarValue = (float)
                                        (((SeekBar)findViewById(R.id.SeekBar01)).
                                        getProgress());
                // get the speed of the finger
                Vf = seekBarValue*seekBarValue;
                ..
        }
        ..
}
```

Gas Pedal Control The code below shows the calculation of the translation speed. First, the finger displacement vector from the start and end points is calculated. Second, The speed in the two axes are calculated based on the equation 3.10. Next the threshold values are added to control the maximum speed. The variables disp represent the finger displacement vector, Vx and Vy represents the translation speed for the x and y axes respectively.

```
private void move( boolean lastMove ){
        // finger movement speed &  deceleration
        float Vf,Vx,Vy,deAcceleration;
        // The difference between start and end points
                PointF disp = getDisplacementVector(end, start);
        ..
        // gas Pedal Method to control the speed
        if(displacementOption == 1){
                ..
                // get the speed of the finger
                Vx =(float) (0.04*disp.x*disp.x+500);
                if(Vx>10000)
                        Vx=10000;
                Vy =(float) (0.04*disp.y*disp.y+500);
                if(Vy>10000)
                        Vy=10000;
                ..
        }
        ..
}
```

Finger Speed Control The finger speed technique depends on the finger movement speed. Hence The application starts with calculating the finger movement speed on the screen. First, time is started when finger is down as shown before in the subsection 4.3.1.

Second the displacement is calculated using the finger displacement vector -disp-, and the total time is calculated too. Next, the finger movement speed is calculated -Vf-. Third the translation speed is calculated based on the finger movement speed -Vf- used the equation 3.12. The translation speed is stored in the variable - tempSpeed -.

```
private void move( boolean lastMove ){
        // finger movement speed &  deceleration
        float Vf,Vx,Vy,deAcceleration;
        // The difference between start and end points
        PointF disp = getDisplacementVector(end, start);
        ..
        // MAPPING FINGER SPEED TO TRAVEL SPEED
        if(displacementOption == 2){
                ..
                // get the speed of the finger
                Vf = (float) ((Math.sqrt(disp.y*disp.y+disp.x*disp.x)) /
                                (System.currentTimeMillis()-startMoveTime))*2.5f;
                if(fingerSpeed<Vf)
                        fingerSpeed = Vf;

                float tempSpeed = (float) (Math.pow(15*fingerSpeed, 2));
                if(tempSpeed>10000)
                        tempSpeed = 10000;
                ..
        }
        ..
}
```

4.5 Translation Implementation

The translation in the VE is based on the simulation of user's locomotion using the user's finger. The translation techniques maps the user's translation on the handled device screen to translation in the VE. The displacement equation for the mobile and the immersive virtual environment are discussed in the section 3.2.2.

The implementation of the translation is divided into two parts. First, the client part which represents the user translation on the touch screen according to the equation 3.5. Second, the IVE part - server - which simulates the actual translation in the IVE according to the equation 3.6.

4.5.1 Mobile phone displacement

The direction vector is discussed in the previous section 4.3.2 and speed is discussed in the section 4.4. After the calculation of the normal direction vector and the speed, the displacement vector according to the equation 3.5 is sent to the IVE through the socket. When data is sent to the IVE, two cases have to be distinguished. First, if the finger is moving, which means the user is still translating, hence no need to start the deceleration phase. Second, if the finger is up, then user stops translating, therefore the deceleration phase shall starts and the application will send the deceleration message as well.

As the user may be pressing the screen or translating on the screen, the application has to separate the two situations.Therefore a threshold value for the displacement is added to distinguish between the two situations. As long as the displacement in any direction is less than 20 pixels. If the displacement is less than 20 pixels, then the displacement speed rate sent to the IVE is equal to zero. Consequently, the displacement vector is equal to zero. In the other hand, if the displacement is greater than 20, then data is sent. The disp is the displacement vector, and dispSent is the displacement vector supposed to be sent to the VE.

```
// check if there is no move - pressed to stop -
if(Math.abs(disp.x)<20&&Math.abs(disp.y)<20){
        sendData("displacement"+" "+dispSent.y+" "+dispSent.x+" "+0);
        return;
}
```

The following code shows sending data to the IVE when the finger is up and hence the application has to send the deceleration message which contains the deceleration rate. The deceleration rate changes depending on the acceleration technique used, where it is the same for Gas pedal and Manual speed equal to 0.5. However, it gets slower by half for the finger speed technique equal to 0.25, hence provides more smooth transition between displacement and stoppage, In addition the time for the stoppage increases and as a result the user can continue rolling his finger without stopping the translation in the gap between finger is up and down again.

```
private void move( boolean lastMove ){
        ..
        // when finger is up
        if(lastMove){
                deAcceleration = 0.5f;
                sendData( "deAcceleration " + (deAcceleration) );
        }
        ..
}
```

The following code shows sending data to the IVE when the finger is moving, It shows the process for all techniques. The dispSent represents the displacement vector which will be sent to the IVE, the dispSent will be set to the multiplication of the speed - Vf, Vx, Vy, and tempSpeed - with the normal direction vector - normalDispVector -. Then the dispSent is sent to the IVE, combined with the displacement speed rate which is 1.0 and this speed rate is decreases with the deceleration rate till it reaches zero and the displacement stops. The displacement vector is swapped when it is sent to the VE. As the X and Y axes are swapped in the IVE, where the mobile X-Axis is mapped to the IVE Y-Axis, and the mobile Y-Axis is mapped to the IVE X-Axis. Also the Y-Axis of the dispSent is multiplied with -1 to reverse the direction, due to the IVE coordinate system.

```
private void move( boolean lastMove ){
        ..
        // Actual Dispalcement to be sent
        PointF dispSent = new PointF(0,0);
        ..
        // Manual speed control using the bar
        if(displacementOption == 0){
                ..
                // calculate the displacement
                dispSent.x = (float) (Vf*(normalDispVector.x));
                dispSent.y = (float) -(Vf*(normalDispVector.y));
                // Send Displacement DATA
                sendData("displacement"+" "+ dispSent.y +" "+dispSent.x+" "+1.0);
                ..
        }
        // gas Pedal Method to control the speed
        if(displacementOption == 1){
                ..
                dispSent.x = (float) (Vx*(normalDispVector.x));
                dispSent.y = (float) -(Vy*(normalDispVector.y));
                sendData("displacement"+" "+ dispSent.y +" "+dispSent.x+" "+1.0);
                ..
        }
        // MAPPING FINGER SPEED TO TRAVEL SPEED
        if(displacementOption == 2){
                ..
                dispSent.x = (float) (tempSpeed*(normalDispVector.x));
                dispSent.y = (float) -(tempSpeed*(normalDispVector.y));
                sendData("displacement"+" "+ dispSent.y +" "+dispSent.x+" "+1.0);
                ..
        }
        ..
}
```

4.5.2 Immersive Virtual Environment displacement

As discussed before in the section 3.2.2, the translation phase is simulated into two different phases. First, movement phase where the equation 3.6 is calculated, and the user travels in the IVE based on the results of the previous equation. Second, deceleration phase where speeds factor decreases till it reaches zero and the stoppage phase is reached.

Movement phase The phase started when finger is down, This is a sample code from the camera class of the Terrain3D. The code shows how the Terrain3D environment mapped the mobile displacement to the IVE displacement. The position array contains the position of the viewpoint in the IVE, the variable rate indicates the height rate based on the equation 3.7, and the dispX and dispY indicate the displacement in the X and Y axes based on the mobile phone displacement. The two methods getDisplacementX() and getDisplacementY() returns the Mobile Phone displacement vector without any modification as long as the deceleration phase is not started. The dispX and dispY are multiplied by the elapsed time, height rate, and the direction vector using the equation 3.6, As the FElapsedTime indicates the elapsed time between each two frames. The two vectors m_vMoveDirection and m_vMoveRight responsible for translating backward and forward, and translating to the left and right respectively. Then data after multiplication - equation results - are added to the view point position represented in m_vPosition.

```
void Camera::Update(float fElapsedTime)
{
        ..
        // Navigation using android phone
        if(m_pServer->getDisplacementStatus()){
                float position [3];
                GetPosition(position);

                float rate = 1+(position[2]/10000.0f);
                if(rate >11)
                        rate =11;

                float dispX = m_pServer->getDisplacementX();
                float dispY = m_pServer->getDisplacementY();

                if(dispX!=0 || dispY!=0){
                        m_vPosition += m_vMoveDirection * rate * dispX * fElapsedTime;
                        m_vPosition += m_vMoveRight * rate * dispY * fElapsedTime;
                }
        }
        // End android navigation
        ..
}
```

Deceleration Phase The IVE displacement vector is calculated again based on the equation 3.6 and follows the same methodology explained before in the movement phase. However, the only difference is only in the methods m_getDisplacementX() and getDisplacementY(). The return values from these methods are different than the returned values in the movement phase. The displacement vector is multiplied by the displacement speed rate, the rate decreases in the deceleration phase by the deceleration rate based on the following equation 4.1 till it reaches zero. Hence the displacement vector will be equal zero. Therefore the translation phase reaches the stoppage phase.

$$\begin{aligned} DisplacementSpeedRate &= InitialSpeedRate + (DecelerateRate \times Time) \\ Time &= Time + 0.01 \end{aligned} \tag{4.1}$$

Time increases by 0.01 each frame

The following code shows the implementation of the above equation 4.1. The variable disp_Vf represents the current displacement speed rate, the disp_V0 represents the initial displacement speed rate, and disp_deAcceleration represents the deceleration rate received from the mobile phone, and the disp_t indicates the time shown in the equation 4.1. The current speed rate is calculated with each frame when the IVE calls the getter for the mobile displacement vector from the server class. the speed rate is decreased till it reaches zero, then the moveStatus is reset back to false, which indicates the stoppage of the translation phase.

```
float Server::deAccelerate(){

        disp_Vf = disp_V0+disp_deAcceleration*disp_t;
        disp_t += (float)0.01;

        if(disp_Vf<0){
                disp_Vf = 0;
                //dispX = 0;
                //dispY = 0;
                moveStatus = false;
                startDeAcc = false;
        }

        return disp_deAcceleration;
}
```

The following code shows the implementation of the two methods getDisplacementX() and getDisplacementY(). The dispX and dispY represents the mobile phone displacement vector, and the disp_Vf represents the current displacement speed rate. As shown the mobile displacement vector is multiplied by the displacement speed rate. The speed rate is decreased when the deAccelerate() method is called, It is only called in the deceleration phase. The deAccelerate() is called only once while getting the value of y for each frame, since it has to be decreased once per frame and not twice.

```
float Server::getDisplacementX(){
        return (disp_Vf)*dispX;
}

float Server::getDisplacementY(){
        if(startDeAcc)
                deAccelerate();
        return (disp_Vf)*dispY;
}
```

4.6 Steering based motion control techniques

The steering techniques allow user to change his viewpoint orientation in the VE as explained before in the section 3.4. The implementation is divided into two parts The client side and the server side and they are explained before in the subsection 3.4.1.

4.6.1 The Client side

The client side supports the user interaction within the VE. The four implemented techniques are Rotate by heading, Rotate by roll, Rotate by roll with fixed horizon, and merged rotation. All techniques' implementation are discussed separately below. The code shows the delta angles, threshold values, and the smooth factor. It also indicates the orientation of the mobile, for example the azimuth changes with the mobile orientation, either portrait or landscape.

Start and End Rotation Phase

The technique is explained in the section 3.4.1. The steering techniques start when user presses his finger down on the screen, and data is sent to the server which indicates the start of the rotation phase. When the finger is up, the steering phase stops again, and data is sent to the server indicating the stop of rotation and also reset the rotate factors to zeros. The code shows the implementation of the start and stop phase, and as it is related to the touch technique. Hence it should be added to the onTouch method.

```
public boolean onTouch(View v,MotionEvent event) {
        switch (event.getAction() & MotionEvent.ACTION_MASK) {
                case MotionEvent.ACTION_DOWN:
                        ..
                        rotateFirst = true;
                        sendData("startRotate ");
                        ..
                        break;
                ..
                case MotionEvent.ACTION_UP:
                        ..
                        rotate = false;
                        sendData("rotateFactors "+0+" "+0+" "+0);
                        sendData("stopRotate ");
                        ..
                        break;
        }
        return true;
}
```

Rotate By Heading The techniques is shown in the section 3.4.2. Also the tracking of the mobile orientation is discussed in the section 4.2. After getting the mobile orientation, then delta angles are available. The threshold values are checked which are 10 for pitch and 20 for the heading. The delta angles are divided by the smooth factors which are equal to 1.5 for pitch and 3 for heading. Then data is sent to the IVE server which represents the rotation factors, where rotation factor is the delta angle after division with the smooth factor. The variables currentAngles and startAngles represents the current and the start orientations of the mobile respectively, the start orientation is detected when the finger is down on the screen. The variables case1 and case2 represents the heading movement direction, either from 359 to 0 as in case1, or from 0 to 359 as in case2. The screenOrientation represents the mobile orientation either portrait or landscape. The heading, pitch, and roll represents the delta angles around the 3 axes.

```
int screenOrientation = ((WindowManager) getSystemService(WINDOW_SERVICE)).getDefaultDisplay().getOrientation();
// Rotate By Heading
if(rotateOptions == 0){
        float factor = 1.5f;
        // heading values
        if(Math.abs( startAngles[0]-currentAngles[0] )>20){
                //from 360 to 0
                if( case1 ){
                        head = (float) (360+(currentAngles[0]-startAngles[0]));}
                //from 0 to 360
                else if( case2 ){
                        head = (float) (currentAngles[0]-startAngles[0]-360);}
                else{
                        head = (float) -(startAngles[0]-currentAngles[0]);}
        }
        // pitch values
        if(Math.abs( startAngles[1]-currentAngles[1] )>10 && Math.abs(currentAngles[1])<150 ){
                pitch = (float) (-(startAngles[1]-currentAngles[1]));
        }
        // sending data
        if(screenOrientation==0){
                sendData("rotateFactors "+head/(factor*2)+" "+pitch/factor+" "+roll/factor);
        }else{
                sendData("rotateFactors ,"+ -head/(factor*2)+","+pitch/factor+","+ -roll/factor);
        }
}
```

Rotate By Roll The technique is shown in the section 3.4.3. After getting the mobile orientation, then delta angles are available. The threshold values are checked which are 10 for both. If the current roll is around the horizon and less than 5 degrees away from the horizon, then a reset roll message is sent to the server in order to reset the IVE horizon back to its normal, which is zero degree for the IVE roll value. The delta angles are divided by the smooth factors which are equal to 1.5 for both. Then data is sent to the IVE server. The method variables are the same as for the rotate by heading implementation which is discussed above.

```
int screenOrientation = ((WindowManager) getSystemService(WINDOW_SERVICE)).getDefaultDisplay().getOrientation();
// Rotate By Roll
if(rotateOptions == 1){
        float factor = 1.5f;
        // pitch values
        if(Math.abs( startAngles[1]-currentAngles[1] )>10 && Math.abs(currentAngles[1])<150 ){
                pitch = (float) (-(startAngles[1]-currentAngles[1]));
        }
        //roll values
        if(Math.abs(currentAngles[2])<5){
                sendData("resetRoll ");
        }else if(Math.abs( startAngles[2]-currentAngles[2] )>10 && Math.abs(currentAngles[2])<150 ){
                        roll = (float) ((startAngles[2]-currentAngles[2]));
        }
        //sending data
        if(screenOrientation==0){
                sendData("rotateFactors "+0.0f+" "+pitch/factor+" "+roll/factor);
        }else{
                sendData("rotateFactors "+ 0.0f+" "+pitch/factor+" "+ -roll/factor);
        }
}
```

Rotate By Roll with fixed horizon The technique is shown in the section 3.4.4. After getting the mobile orientation, then delta angles are available. It implements the same technique as the rotate by roll which is discussed above.The only difference is in the reset roll messages, As the horizon is always fixed. So the threshold values are 10 for both, and the smooth factor equal to 1.5 for both pitch and roll.

```
int screenOrientation = ((WindowManager) getSystemService(WINDOW_SERVICE)).getDefaultDisplay().getOrientation();
// Rotate By Roll With fixed horizon
if(rotateOptions == 2){
        float factor = 1.5f;
        // pitch values
        if(Math.abs( startAngles[1]-currentAngles[1] )>10 && Math.abs(currentAngles[1])<150 ){
                pitch = (float) (-(startAngles[1]-currentAngles[1]));
        }
        // roll values
        if(Math.abs( startAngles[2]-currentAngles[2] )>10 && Math.abs(currentAngles[2])<150 ){
                roll = (float) ((startAngles[2]-currentAngles[2]));
        }
        // sending data
        if(screenOrientation==0){
                sendData("rotateFactors "+0.0f+" "+pitch/factor+" "+roll/factor);
        }else{
                sendData("rotateFactors "+ 0.0f+" "+pitch/factor+" "+ -roll/factor);
        }

}
```

Merged Rotation The technique is shown in the section 3.4.5. After getting the mobile orientation, then delta angles are available. It combines the rotate by roll and rotate by heading implementation. The threshold values are 20 for heading and 10 for both roll and pitch. The smooth factors are 3 for heading and 1.5 for pitch and roll. Then the message with the rotation factors are sent to the IVE. However, the heading is calculated differently this time as it combines both the heading and roll.

```
int screenOrientation = ((WindowManager) getSystemService(WINDOW_SERVICE)).getDefaultDisplay().getOrientation();
// Merged Rotation
if(rotateOptions == 3){
        float factor = 1.5f;
        // heading value
        if(Math.abs( startAngles[0]-currentAngles[0] )>20){
                if( case1 ){
                        head = (float) ((360+(currentAngles[0]-startAngles[0])));}
                else if( case2 ){
                        head = (float) ((currentAngles[0]-startAngles[0]-360));}
                else{
                        head = (float) (-(startAngles[0]-currentAngles[0]));}
        }
        //pitch value
        if(Math.abs( startAngles[1]-currentAngles[1] )>10 && Math.abs(currentAngles[1])<150 ){
                pitch = (float) (-(startAngles[1]-currentAngles[1]));
        }
        // roll value
        if(Math.abs(currentAngles[2])<5){
                sendData("resetRoll ");
        }else if(Math.abs( startAngles[2]-currentAngles[2] )>10 && Math.abs(currentAngles[2])<150 ){
                roll = (float) ((startAngles[2]-currentAngles[2]));
        }
        // sending data
        if(screenOrientation==0){
                sendData("rotateFactors "+(head-roll)/(factor*2)+" "+pitch/factor+" "+roll/factor);
        }else{
                sendData("rotateFactors "+(head+roll)/(factor*2)+" "+pitch/factor+" "+ -roll/factor);
        }
}
```

4.6.2 The Immersive Virtual Environment side

The server part supports the change in the view point orientation within the VE. The server receives three different messages. First, the start message to indicate the start of steering. Second, The rotation factors message indicates the rotation angle factors for the view point orientation. Third, the stoppage message sent when the finger is up, then user stops steering.

As explained before in the section 3.4.1 the rotation factors are multiplied by the elapsed time between each two frames, then added to the IVE view point orientation angles. The factors are added based on which technique is used.

Rotate By Heading Factors are added normally as the mobile orientation angles are mapped to the exact IVE angles.

```
void Camera::Update(float fElapsedTime)
{
        ..
        if(m_pServer->getCameraStatus()){
                // Navigation using android phone
                ..
                if(m_pServer->getRotateByFactor()){
                        m_fHeading += m_pServer->getHeading()*fElapsedTime;
                        m_fPitch += m_pServer->getPitch()*fElapsedTime;
                        m_fRoll += m_pServer->getRoll()*fElapsedTime;
                }
                ..
        }
        // End android navigation
        ..
}
```

Rotate By Roll Factors of pitch and roll are added directly to IVE pitch and roll. In the other hand the IVE heading is manipulated differently, the factor roll is added to the IVE heading based on the IVE Roll. As explained in the section 3.4.3. From the code, the factors are added to the IVE heading, so if the horizon is turning to the right for example, then the heading values increases and rotates to the right. The reset horizon is explained below in the end of this subsection.

```
void Camera::Update(float fElapsedTime)
{
        ..
        // Navigation using android phone
        ..
        if(m_pServer->getCameraStatus()){
                if(m_pServer->getRotateByFactorRoll()){
                        m_fPitch += m_pServer->getPitch()*fElapsedTime;
                        m_fRoll += m_pServer->getRoll()*fElapsedTime;
                        if(m_fRoll>0)
                        {
                                m_fHeading += -1*(abs(m_pServer->getRoll()))*fElapsedTime;
                        }
                        else
                        {
                                m_fHeading += (abs(m_pServer->getRoll()))*fElapsedTime;
                        }
                }
        }
        ..
        // End android navigation
        ..
}
```

Rotate By Roll with fixed horizon Exactly the same as the rotate by roll except for the roll factor. As the horizon is all the time fixed, then there is no change to the IVE roll.

```
void Camera::Update(float fElapsedTime)
{
        ..
        // Navigation using android phone
        ..
        if(m_pServer->getCameraStatus()){
                if(m_pServer->getRotateWithoutRolling()){
                        m_fPitch += m_pServer->getPitch()*fElapsedTime;
                        if(m_pServer->getRoll()>0)
                        {
                                m_fHeading += -1*(abs(m_pServer->getRoll()))*fElapsedTime;
                        }
                        else
                        {
                                m_fHeading += (abs(m_pServer->getRoll()))*fElapsedTime;
                        }
                }
        }
        ..
        // End android navigation
        ..
}
```

Merged Rotation Exact technique as rotate by heading, as all the work to calculate the rotation factors is done in the client side (Mobile Phone).

```
void Camera::Update(float fElapsedTime)
{
        ..
        // Navigation using android phone
        ..
        if(m_pServer->getCameraStatus()){
                if(m_pServer->getMergedRotation()){
                        m_fHeading += m_pServer->getHeading()*fElapsedTime;
                        m_fPitch += m_pServer->getPitch()*fElapsedTime;
                        m_fRoll += m_pServer->getRoll()*fElapsedTime;
                }
        }
        ..
        // End android navigation
        ..
}
```

Reset The Horizon Back The same technique as the rotation is used here too. The horizon is getting back to its normal angle, using a rotation factor of 10 degrees, The factor is multiplied with the elapsed time and added to the IVE roll. The addition is based on the IVE roll, so either rotate to the left or to the right. The horizon is set back in two cases. First, when the mobile roll is around zero, mobile in horizon, Once the IVE receive the command message, the reseting starts, explained in the code 4.3(a). Second, When the finger is up from the screen, then the client send a message to stop the rotation phase, and the IVE resets the horizon back, showed in the code 4.3(b).

```
void Camera::Update(float fElapsedTime)
{
..
// Navigation using android phone
// Reset the roll value back to zero
if (m_pServer->getResetRollStatus()){
        if(m_fRoll>0){
                m_fRoll -= 10*fElapsedTime;
                if(m_fRoll<0){
                        m_fRoll=0;
                        m_pServer->setResetRollStatus(false);
                }
        }
        else if(m_fRoll<0){
                m_fRoll += 10*fElapsedTime;
                if(m_fRoll>0){
                        m_fRoll=0;
                        m_pServer->setResetRollStatus(false);
                }
        }
}
// End android navigation
..
}
```

(a) mobile is in the horizon

```
void Camera::Update(float fElapsedTime)
{
        ..
        // Navigation using android phone
        // Reset the roll value back to zero
        if(m_pServer->getCameraStatus()){
                ..
        }
        else{
                if(m_fRoll>0){
                        m_fRoll -= 10*fElapsedTime;
                        if(m_fRoll<0)
                                m_fRoll=0;
                }
                else if(m_fRoll<0){
                        m_fRoll += 10*fElapsedTime;
                        if(m_fRoll>0)
                                m_fRoll=0;
                }
        }
        // End android navigation
        ..
}
```

(b) finger is up

Figure 4.3: Different GUI Design

5 Evaluation

In this chapter the evaluation for the whole system is discussed. The main task of the evaluation is to evaluate the different techniques for acceleration and steering in the VE.

The evaluation compares users performance using different techniques and it aims to figure the users accuracy and speed during navigation within the VE. It also points to the differences in users performance using more or less degrees of freedom. Additionally, the participants were asked to fill a qualitative questionnaire to measure the usability and attractiveness of the techniques. Eight participants participated in the evaluation.

Evaluation Environment The work is evaluated on the Terrain3D environment. A 3D TV is used during the evaluation, therefore users have the feeling that they are immersed within the VE. User has to stand in front of the TV during the evaluation, more like being in a cave. Users have to conduct two tests one for steering and the other for acceleration, then fill a questionnaire to evaluate the whole system. The questionnaires is added to the appendix A.2. Additionally, we handle drawings to users before starting the evaluation. These drawings show the different steering techniques, and indicates the degrees of freedom for each technique A.1.

Evaluation Environment for Steering techniques The user has to follow a tunnel through the VE. The Tunnel consists of straight paths and some curves to test the user abilities to steer while navigating. User has to be within the tunnel for as long as he can. The test starts when user enter the tunnel from the start point and ends at the end of the tunnel.

Evaluation Environment for Acceleration techniques The user has to follow a path, this time user has to finish the path as fast as he can. The path consists of spheres where user has to go from one sphere to the next. The spheres are green, red, and finally the blue. the test starts when user hits the green sphere and ends at the blue one.

In the next sections the two evaluations and the questionnaires are discussed.

5.1 Evaluation Technique

In this section the way of evaluating the users flight and recording the flight information is discussed. The user in each test has to follow a specific path in the VE, This path is a previous recorded flight using the mobile phone. While user travelling through the path, the users flights are recorded and all information regarding positions, orientation, and time are recorded too.

Recording flight Two recorded files are used here. First file includes all information about user's positions, and orientations. Second file includes samples of the flight. Those samples represents the user's flight. The recorded flight files store the following information shown in the table 5.1. Each file's line contains these informations.

Parameter	Description
m_dTime	current time
m_vPosition	3D vector to store the X,Y, and Z positions
m_fHeading	the view point orientation heading value
m_fPitch	the view point orientation pitch value
m_fRoll	the view point orientation roll value
m_fTurn	the view point orientation turn value

Table 5.1: Recorded flight file

Samples Technique Recording the sample of the flight is used in evaluating the steering techniques where the tunnel is used. The idea is to represents the user flight using some samples and not the whole flight, hence to compare only the users current positions (samples) to a reference positions. The tunnel consists of rings, these rings together constructs the tunnel. Therefore the center of these rings are used as a reference points.

The samples are taken on each ring, hence on each ring of the tunnel the user's position and orientation are compared to the ring's position and orientation. The idea is to cut the environments into planes a long the X-Axis, where each plane has a value in X. If this plane includes a ring of the tunnel, then this sample is recorded.

Recording samples starts when user hits the start position of the tunnel and ends when user reaches the end position of the tunnel.

5.2 Steering based motion control evaluation

This section focuses on the evaluation of the different techniques for steering which are Rotate By Heading, Rotate By Roll, Rotate By Roll Fixed Horizon, and Merged Rotation. The evaluation points to the users performance, while navigating within the VE.

The evaluation do the comparisons between all the techniques based on accuracy which is percentage of time within the tunnel, time it takes the user to finish the test (speed), and finally the user's standard deviation away from the centers of the tunnel.

5.2.1 Standard Deviation Comparison

Here the work focus on the users path during the flight. The work points to how far the users are away from the centers of the tunnel, As the test only cares about the position of user relative to the tunnel. First, The average distance away from the centers of the tunnel. Second, the standard deviation away from the centers of the tunnel during the whole flight. The tunnel radius is 1000 meters and if the standard deviation is less than 1000 meters, then the user is in average within the tunnel through the whole flight.

In the figure 5.1 the average distances for all users away from the centers of the tunnel plotted against the centers of the tunnel. The red arrow indicates the radius of the tunnel. The graph shows that in the Rotate by heading and Merged rotation techniques, blue and dark red curves respectively, users are outside the tunnel for long distances. However, for Rotate by roll and Rotate by roll with fixed horizon, green and purple respectively, users show much better performance and user stays within the tunnel for longer distances.

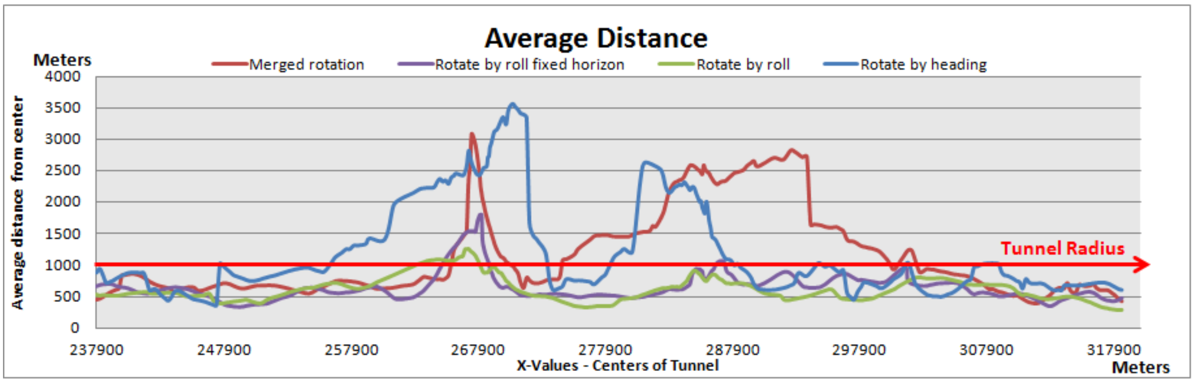


Figure 5.1: Average distance away from the centers of the tunnel

The Standard deviation is calculated for the distance away from the centers of the tunnel during the whole flight. All standard deviation for all users are shown in the figure 5.2, the graph shows that the rotate by roll and rotate by roll with fixed horizon, dark red and green curves respectively are consistent and show consistency for all users, It also shows that the standard deviation is almost less than the radius of the tunnel for all users, which indicates that users in average are within the tunnel for longer time. In contrast rotate by heading and merged rotation, blue and purple curves are not consistent for all users.

Figure 5.2 shows that using rotate by heading and merged rotation techniques lead to get higher standard deviation, Hence low accuracy. In the other hand rotate by roll and rotate by roll with fixed horizon get lower standard deviation and hence higher accuracy obtained.

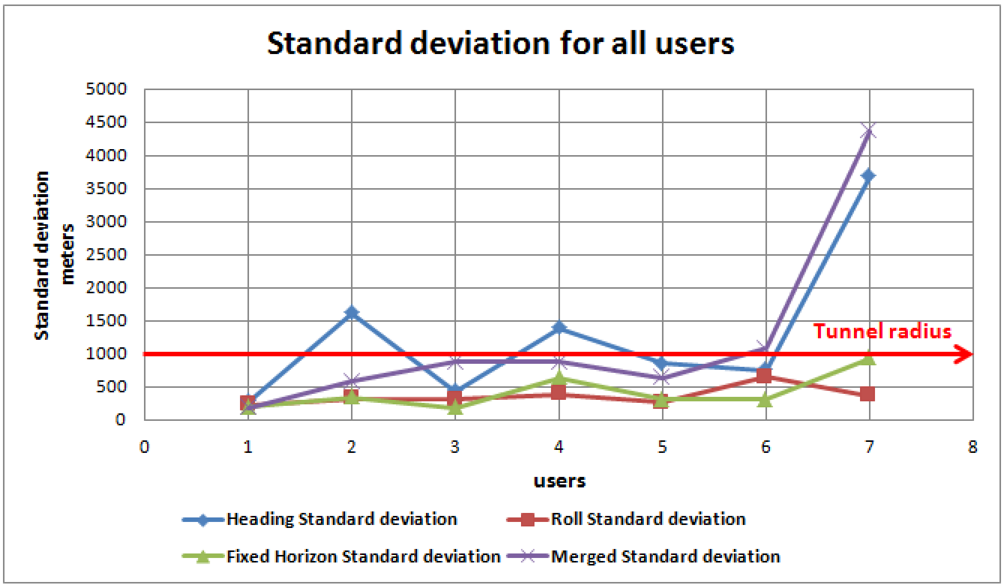


Figure 5.2: standard deviation per user

Mean Standard Deviation The table 5.2 shows the mean of the standard deviation for each steering technique. The figure 5.3 illustrates more that the mean standard deviation for the rotate by roll and rotate by roll with fixed horizon are less than 500 meters which means that users are in the middle of the tunnel in average. However, rotate by heading and merged rotation shows that the mean is greater than 1000 meters and hence users get out side the tunnel for long distances.

	Rotate By Heading	Rotate By Roll	Rotate By Roll With Fixed Horizon	Merged Rotation
Mean Standard Deviation	1248.487 meters	470.989 meters	433.549 meters	1401.549 meters

Table 5.2: Mean Standard Deviation per Technique

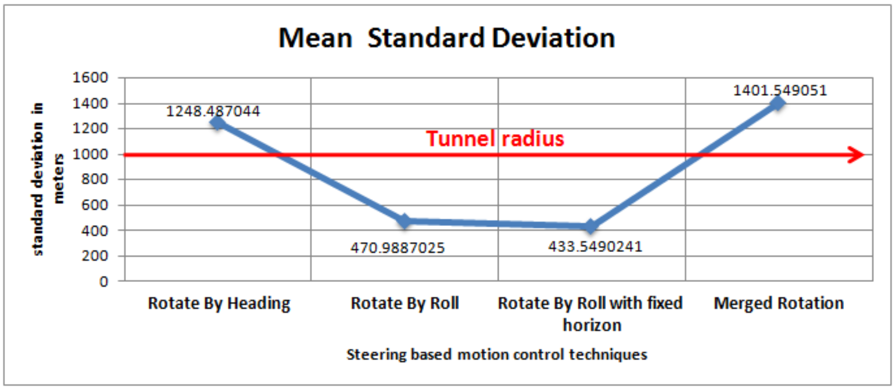


Figure 5.3: Mean Standard Deviation per Technique

5.2.2 Total Time Comparison

Here the evaluation focuses on the total time it takes user to finish the test, different comparisons between time within the tunnel and time outside the tunnel for all users are conducted. The test focus on the total time for user to finish the test. All this data shows how users are able to control the navigation and keeps themselves within the tunnel for as long as they can. In this subsection we focus only on the total time to finish the test, the comparison between time within the tunnel and time outside the tunnel in the next subsection 5.2.3.

The total time is calculated when the user enters the tunnel till he leaves it. The total time for all users using different steering techniques are shown in the figure 5.4, the graph shows that the rotate by roll and rotate by roll with fixed horizon, dark red and green curves respectively are consistent and show consistency for all users, as all users perform equally and finished approximately in the same time, It also shows that users finished the test faster using these techniques than using rotate by heading and merged rotation. In contrast rotate by heading and merged rotation, blue and purple curves are not consistent for all users. Also users finished the test in longer period of time comparing to rotate by roll and rotate by roll with fixed horizon.

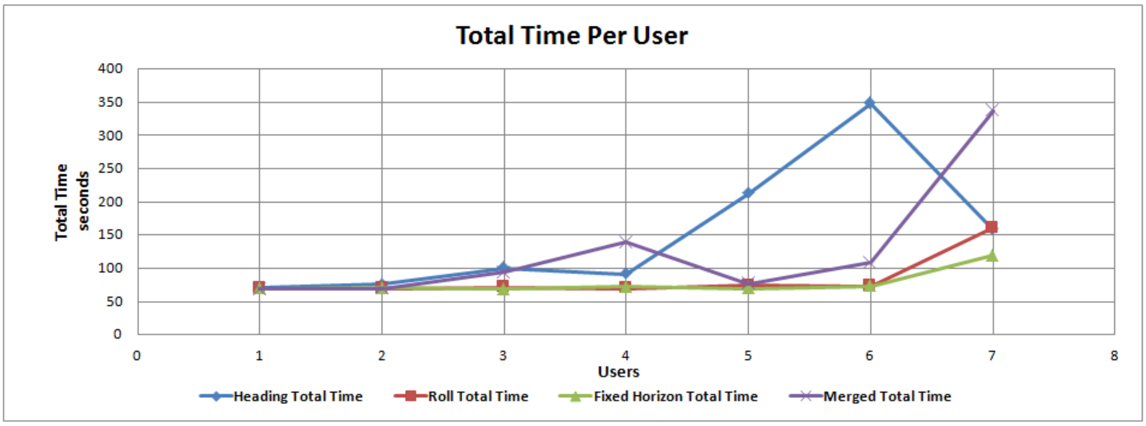


Figure 5.4: Total Time per user

Mean Total Time The Mean of the total time for all the steering techniques is calculated and shown in the table 5.3. In The figure 5.5 the mean total time for all users are plotted against each steering technique.The graph shows that users finish the test using rotate by roll and rotate by roll with fixed horizon techniques faster than using rotate by heading and merged rotation. Hence the time comparisons show that rotate by roll and rotate by roll with fixed horizon show much better performance.

	Rotate By Heading	Rotate By Roll	Rotate By Roll With Fixed Horizon	Merged Rotation
Mean Total Time	53.950 Seconds	12.999 Seconds	17.468 Seconds	34.719 Seconds

Table 5.3: Mean Total Time per Technique

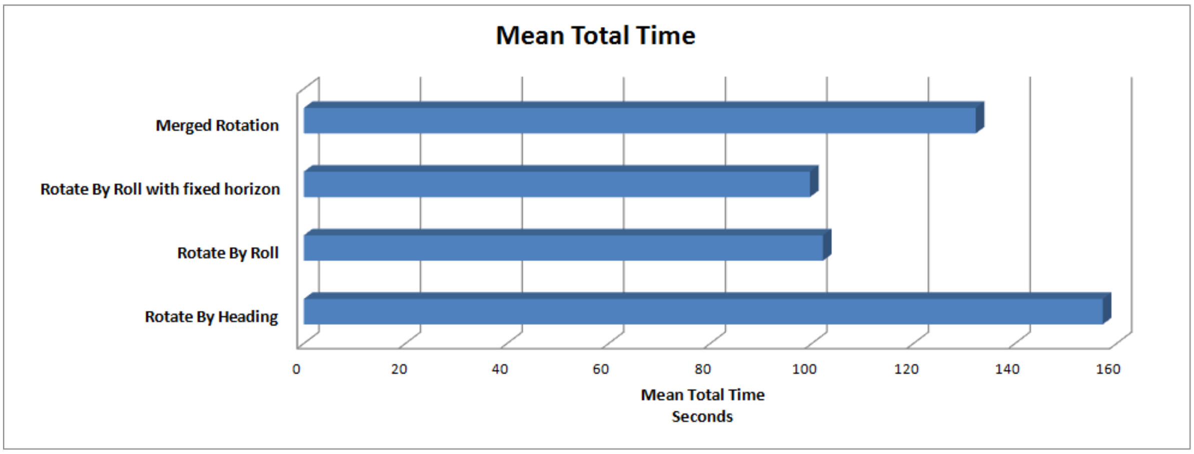


Figure 5.5: Mean Total Time per techniques

5.2.3 Accuracy Comparison

The Accuracy is the most important factor in this test. Accuracy is the percentage of the total time within the tunnel to the total time for the flight. So it does not matter how long it takes the user to reach the need of the tunnel. However, what is important is to be within the tunnel. The high percentage indicates that users are able to control the system and keep themselves within the tunnel for as long as they can.

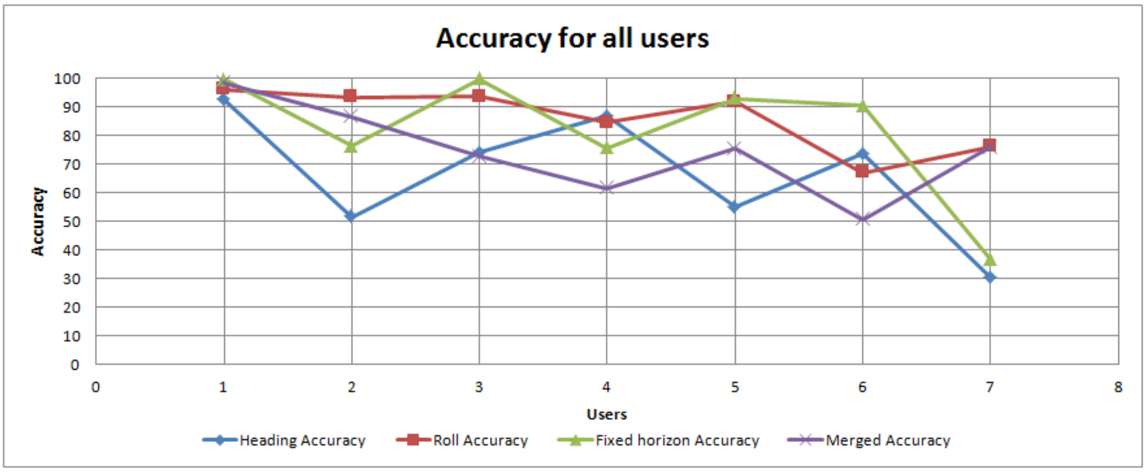


Figure 5.6: Accuracy per user

The Accuracy is calculated for all users using different steering techniques, then data is plotted against users as shown in the figure 5.6. The graph shows that users did much better performance and get a higher accuracy using rotate by roll and rotate by roll with fixed horizon techniques. As shown in the graph the red line represents rotate by roll and it shows a consistent data and users performed the same except for the last two users. In another hand rotate by heading and merged rotation, blue and purple curves are not consistent for all users. Also users performed less accurate than rotate by roll and rotate by roll with fixed horizon.

Mean Accuracy The Mean of users accuracy, total time out of tunnel, and flight total time for all the steering techniques is shown in the table 5.4. In The figure 5.7 the average accuracy for all users are plotted against each steering technique.The graph indicates that rotate by roll and rotate by roll with fixed horizon shows a higher accuracy than rotate by heading and merged rotation.

Technique	Flight total time	Total time out of tunnel	Accuracy
Rotate by heading	157.2802555	53.95000186	61.16123865
Rotate by roll	102.1860039	12.99885757	87.11284394
Rotate by roll with fixed horizon	99.63975363	17.46786	80.43132982
Merged rotation	132.2902553	34.71942971	72.40885061

Table 5.4: Mean Accuracy per Technique

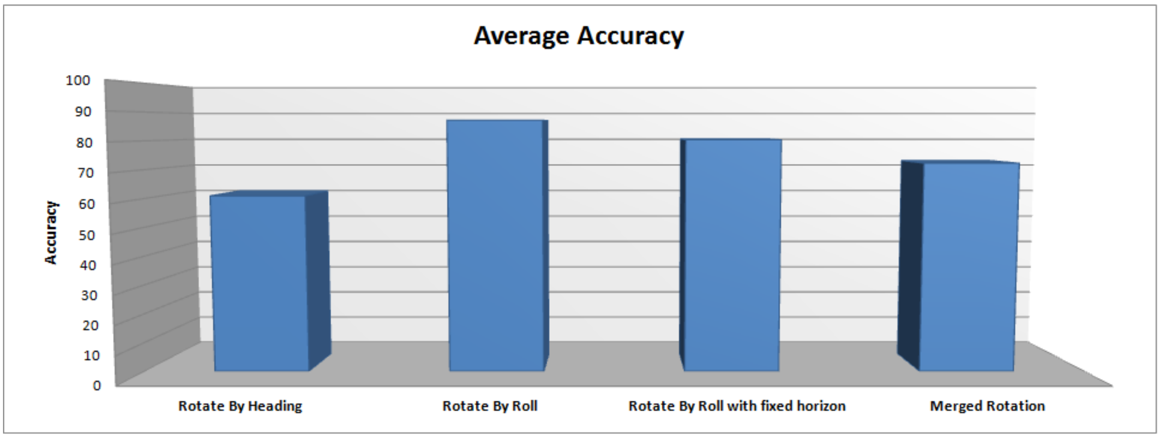


Figure 5.7: Mean Accuracy per techniques

Overall Performance The whole test shows that Rotate by roll and Rotate by roll with fixed horizon techniques show much better performance than Rotate by heading and Merged rotation in terms of accuracy, time -speed-, and finally how far away users are from the center of the tunnel. Rotate by roll technique shows higher accuracy equal to 87.11 comparing to 80.43 for rotate by roll with fixed horizon. However, rotate by roll with fixed horizon shows smaller standard deviation relative to rotate by roll, It is also clear that users could finish faster in rotate by roll with fixed horizon technique. The overall evaluation for steering technique points to the advantage of using less degrees of freedom, Users show much notable performance using less degrees of freedom represented in rotate by roll comparing to more degrees of freedom represented in merged rotation.

5.2.4 Users Feedback

The users evaluate the system overall and provide many feedbacks. Eight users evaluate the project. Five of the users choose Rotate by roll as their favorite technique for steering, two choose Rotate by roll with fixed horizon, and one goes for Merged rotation.

Rotate By Heading Two users mentioned that the technique is not intuitive at all. Also two users said the technique is fatigue. There is a lagging in the rotation reaction, the lag due to the reading of the heading value of the sensor and due to the large threshold value for the rotation. The reading of the sensor is inaccurate and unstable and once the orientation changes, the reading does not change instantaneously, and this is the reason behind the lagging.

Rotate By Roll More users go with the rotate by roll as their best technique. Users mentioned that movement of hand does not have to be as big as rotate by heading, Hence faster reaction. This technique was more intuitive for users comparing to the rotate by heading.

Rotate By Roll with fixed horizon Some users said that this technique does not add anything new to rotate by roll technique. However, one user mentioned that it is not that natural, as user expect the horizon to change while rotating the mobile.

Merged Rotation Again users complain about the heading lagging and the slow reaction for rotation. However, the majority of the users tend to use only the rolling to steer and some of the users said it explicitly that this he did exactly the same as the rotate by roll test.

Overall The majority of the users go with rotate by roll and rotate by roll with fixed horizon techniques. User complains about rotate by heading and merged rotation especially for sensors reading and the slow reaction. Rotate by roll is the most favorite technique according to users feedback which matches the evaluation results.

5.3 Acceleration based motion control evaluation

This section focuses on the evaluation of the different techniques for acceleration while traveling within the VE, the techniques are Manual speed, Gas pedal, and Finger speed. The evaluation focuses on users ability to control their speed during the navigation

The evaluation and comparisons between the different techniques are based on the users speed which means the time it takes the user to finish the test.

Users had been informed to do their best to finish the test as fast as they can. The user has first to hit the green sphere, then the red sphere, and finally the blue sphere. All the spheres are along the X-Axis in front of the users view so they do not have to steer in the VE, as the test focus only on the users speed.

5.3.1 Total Time Comparison

Here the evaluation focuses on the total time it takes user to finish the task,The task basically is to go through the three points within the VE. Total time for the whole flight from hitting the green sphere till reaching the blue sphere is calculated.

The total time for all users using different acceleration techniques are plotted for each user in the figure 5.8. The graph shows that users' speed is consistent using the gas pedal technique, and all users performed equally.The gas pedal is represented by the red curve. The graph also shows that Gas pedal technique get the smallest time which means the users where faster to finish the test, except for one user who was very slow due to the fact that user did not get the idea behind the gas pedal and user always mix the other techniques together, then we started to try to explain it. Regarding the other techniques data is not consistent for all users and some could finish fast and others were slower.

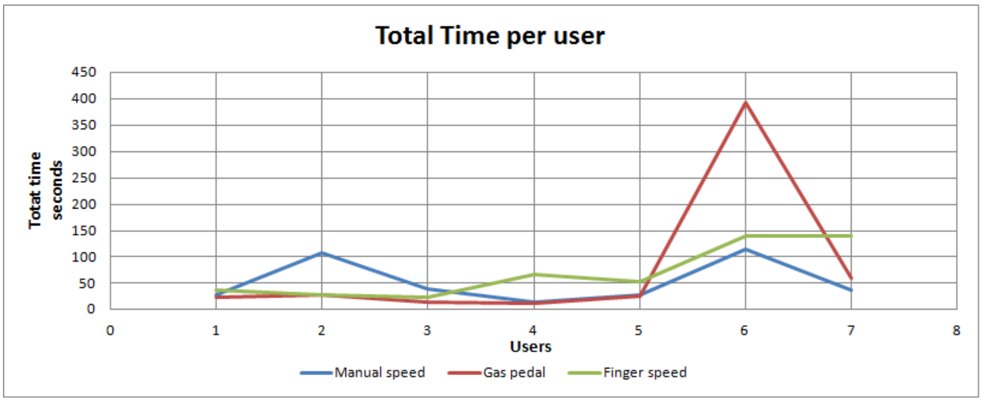


Figure 5.8: Total Time per user

Mean Total Time The Mean of the total time for all the acceleration techniques are calculated and shown in the table 5.5. In The figure 5.9 the mean total time for all users are plotted against each acceleration technique. From the chart it is clear that people were able to finish the test faster using the gas pedal technique. It takes user in average around 28 seconds to finish the test using the gas pedal relative to 42.8 seconds and 58 seconds for manual speed and finger speed techniques respectively.

	Manual Speed	Gas Pedal	Finger Speed
Mean Total Time	42.814 Seconds	27.941 Seconds	57.938 Seconds

Table 5.5: Mean Total Time per Technique

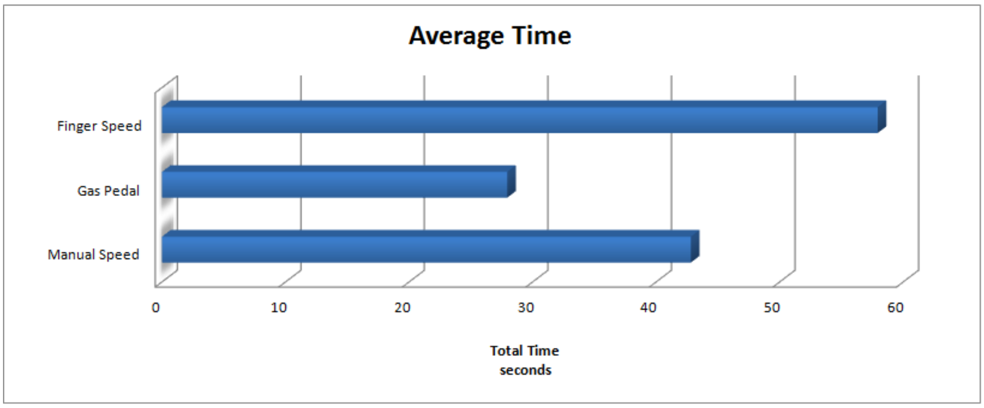


Figure 5.9: Mean Total Time per techniques

Overall Performance The whole evaluation shows that gas pedal technique performed much better than the other techniques and user still able to control the steering and accelerating at the same time. The finger speed shows the worst performance and users were not able to get used to it easily.

5.3.2 Users Feedback

The users evaluate the system overall and provide many feedbacks. Eight users evaluate the project. Seven of the users chose Gas pedal as their favorite technique for acceleration, while one user chose nothing.

Manual Speed It's better to select a medium speed and navigate with. The majority of user gave the same feedback which is that user have to stop, look at the screen, and then change the speed, and according to users this will lead to loose more time and loose the advantage that users want to speed. Users complain about the touch sensitivity of the speed bar and how hard is it to manipulate it will navigation. One user suggests using another member for acceleration.

Gas Pedal The majority of user chose gas pedal to be their favorite acceleration technique. Users provide that gas pedal is more natural and intuitive for users. The users feedback focuses more on the acceleration function, some mentioned that it has to be exponential acceleration, and other suggest to use absolute values on the handled device screen. Another suggestion that is common between many users that user who start from the middle of the screen should be able to reach to the maximum speed. Feedback regarding user's speed has to be shown on the mobile screen.

Finger Speed The user expects a continuous rolling even if the finger is up, more like going through a list in an iPod. This technique is less natural and hard for user to control the movement direction, steering, and accelerate at the same time. Finally, Three users found it as not good at all.

Overall The majority of the users go with gas pedal technique. User complains about the other techniques either for loosing time or hardness to control the speed. Therefore, gas pedal is the most favorite technique for users and this matches the evaluation results as users could finish the test faster using gas pedal technique

5.4 Overall phone based motion control feedback

Here we discuss the users feedback for the overall application and not only for one specific metaphor. Many of users feedback could be used in some future work and to the current version of the evaluated techniques.

One user suggests for Accelerating that we can use another member to accelerate, for example user could hold the mobile in one hand and holding a device to accelerate with his other hand, This technique is used in games for example.

When users Finger is up from the screen, the deceleration phase starts and the speed decreases smoothly till it reaches zero, some users mentioned that it is better to just stop when user get his finger up from the screen.

User has to get visual, audio, or haptic feedback when he pressed on the screen and start navigating. The visual and audio could be in the mobile phone screen and the VE. This feedback also indicates if user is navigating or not. The haptic feedback could be added when user press on the screen to start navigating, then the mobile vibrates. Visual feedback to illustrate each metaphor used and shows the user for example the possible rotations for this metaphor.

The translation in the X-Axis of the mobile screen has to be investigated, It may not be needed and the user could translate more like being in a plane where it can only move forward. It could also be like a car where user can translate forward and backward. Another suggestion is to reduce its sensitivity.

One of the user got motion sickness during the evaluation.

When phone is discharging the sensors reading were destructed and completely inaccurate. Also the mobile will switch the WiFi off. It could be to save energy, hence less power goes to the sensors.

5.5 Qualitative Evaluation Questionnaires

In this section we discuss the questionnaires users has to fill after completing the whole evaluation. The test users were asked to evaluate the whole system and not just one steering or acceleration metaphor, as we estimated that the qualitative evaluation questionnaires are not relevant for travel interaction in a VR environment, and it is also pointed out by some users during the evaluation.

5.5.1 Attrakdiff

Users evaluate the whole system. The figure 5.10 shows the Attrakdiff graph. The results of the Attrakdiff are above average, and the overall impression of the application is very attractive.

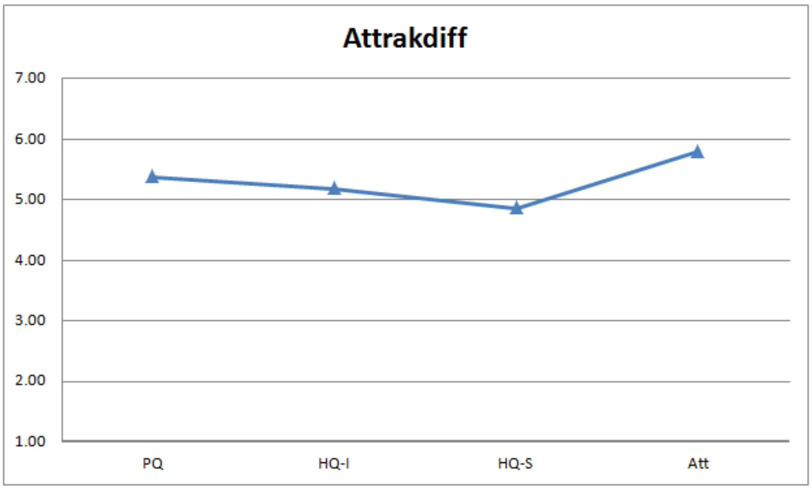


Figure 5.10: Attrakdiff Graph

Pragmatic Quality (PQ) Indicates the usability of the product, it also shows how successful users are in achieving their goals using the product.

Hedonic quality - Stimulation (HQ-S) People have an inherent to be developed and moved forward. This dimension indicates to what extent the product can support such needs, as it focuses on contents, interaction, presentation, novel, and how interesting is it for users.

Hedonic Quality - Identity (HQ-I) describes how users are able to identify themselves with the product.

Attractiveness (ATT) indicates the attractiveness of the product.

5.5.2 SUS : the System Usability Scale

Again Users evaluate the whole system due to the irrelevance of the qualitative questionnaire for interaction with the VE. The figure 5.11 shows the SUS scale. The result of the SUS is not bad at all. It ranges from 40 to 87.5.

The users tend to evaluate the application depending on their feedback of their interaction experience within the application and not evaluating the whole system itself. That is why some users tend to focus on the negatives with the application, while others focus on the positives.

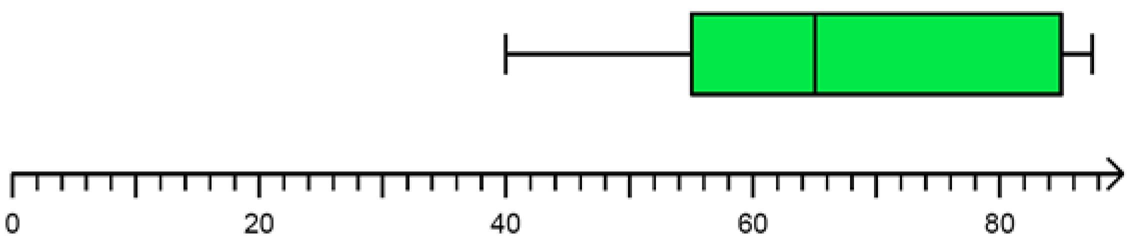


Figure 5.11: SUS Scale

6 Further improvement and Conclusion

In this chapter, we summarize the new travel techniques for navigation in a VE, which we proposed. Next we discuss more about some future improvement for the proposed techniques.

6.1 Conclusion

In the thesis we provide a new intuitive interaction techniques for travelling in a VE. Additionally, a study for a usable user interface is conducted. The proposed interaction techniques focus on how users can control their viewpoint within the VE. The interaction techniques used a mobile phone (HTC Desire) as an input device, The multi-touch screen is used for interaction within the VE and mainly used for the translation part. In addition the Mobile's inertial sensors are used for tracking the mobile phone, hence provide Mobile's orientation. We proposed different interaction techniques for viewpoint motion control, so we have three different interaction techniques for translation, velocity, and steering.

Translation Technique The translation techniques aims to change the viewpoint's position within the VE. The translation is based on the single-touch techniques, which means only one finger is used for translation. The translation on the mobile screen is 2D translation which is mapped to a 2D displacement in the IVE. The third dimension is added using the mobile orientation based on the steering technique used.

Acceleration Techniques It's critical how user can control his movement speed while travelling within the VE. We proposed three different techniques, so users can control his travelling speed. the techniques are manual speed, finger speed, gas pedal techniques.

Steering Techniques We focus on how user controls the orientation of the viewpoint. The mobile phone orientation is mapped to the VE viewpoint orientation. The different techniques varies in the degrees of freedom and how they are mapped to the VE orientation. The proposed techniques are rotate by heading, rotate by roll, rotate by roll with fixed horizon, and merged rotation.

The Interaction Technique User has to hold the mobile phone in one hand, he/she starts translating by moving his/her finger on the touch screen. From the menu users can choose the desirable acceleration and steering techniques. Once the user press his/her finger on the screen, he/she indicates the start of the translation or steering phase. The user can change the VE orientation by changing the mobile phone orientation while his/her finger is pressed on the screen.

Evaluation The study conducted an evaluation to compare the different techniques used, the evaluation focuses on the quantitative measures speed, accuracy, and ease of use. Also users were asked to fill qualitative questionnaires to measure the usability and attractiveness. We conducted a small evaluation of eight participants.

Steering Technique Evaluation According to the evaluation results the rotate by roll and rotate by roll with fixed horizon techniques show better performance in terms of the quantitative measures such as speed and accuracy comparing to rotate by heading and merged rotation. Also the users feedback suggests rotate by roll and rotate by roll with fixed horizon to be their favorite techniques. Additionally, it indicates that using less degrees of freedom represented in rotate by roll shows better performance comparing to more degrees of freedom represented in merged rotation. Finally, the thesis results in two intuitive and usable techniques, which supports the users needs and help them achieving their goals while navigating in the VE, the two techniques are rotate by roll and rotate by roll with fixed horizon.

Acceleration Techniques Evaluation The user evaluation and feedback suggest that gas pedal techniques is more intuitive, natural, and easy to use and learn. And the results showed that using gas pedal techniques, users were able to accelerate and steering simultaneously without any problems.

Qualitative Evaluation Questionnaires The questionnaires showed that the techniques are very attractive and the usability of the techniques are not bad according to the SUS test.

In Conclusion The thesis results in an intuitive travel technique with high usability for navigating within the virtual environment. The interaction techniques address the viewpoint motion control. The work suggests the rotate by roll technique for steering and gas pedal technique for acceleration as novel techniques and the future works have to target the two techniques. The provided technique is a milestone required more further improvement according to the evaluation results.

6.2 Future Work

As we proposed a new approach for travelling techniques for navigation within the VE. Still there are many challenges and open issues for discussion and further improvement. In this section we listed possible future directions based on our experience with the interaction techniques and the users feedback during the evaluation.

Develop the travel techniques Since the thesis results in novel techniques for travelling within the VE, and the rotate by roll, rotate by roll with fixed horizon, and gas pedal techniques are evaluated the best. Hence, the two techniques rotate by roll, and gas pedal techniques are subjected to more future work. The threshold values for rotate by roll techniques has to be investigated, as it is essential for interaction either to have a fast or slow reaction to user's movement. Regarding gas pedal techniques, the implementation of the technique has to be updated, so user who starts the displacement from the center of the screen is able to reach the maximum speed, so users should not start from the end of the screen. Additionally, it can be implemented so it is correlated to the distance between the initial thumb position and the higher point on the screen.

Investigate new Hardware sensors Since the Mobile's sensors show unstable and unsteady outputs, and this leads to slow reaction time. Hence investigating a new hardware is of great impact. An external sensor to provide the Mobile's orientation could be attached to the device. The new hardware is essential, as the value of the heading, pitch and roll are correlated, as the value of the head is dependable on both the pitch and roll. Consequently, having one sensor providing the three orientations leads to the unstable data.

Tracking the device The mobile phone could be tracked using the ART, and hence the application can get the Mobile's orientation from the ART, which is very accurate, and we can do a sort of comparison between using the inertial sensor and the ART tracking system. The two tracking techniques could be implemented together and a sort of filtering the two outputs can be implemented. It would be a great feature to implement a Kalman Filter for the inertial sensor and the ART tracking system. Hence we can use the inertial sensor in the absence of the ART tracking system for example.

Translation Technique The proposed translation technique includes translation in a 2D plane, thus the translation in the X and Y axes of the mobile phone is mapped to the IVE axes. In the future, the translation in the X axis could be investigated. And as a result, user can only translate in the Y axis, so two techniques could add great features. First, have a translation like the plane, which can only move forward. Second, it could be similar to the car movement, which can move forward and backward.

Audio, Visual, and Haptic Feedback Adding feedback would be of great interest, audio, visual, and haptic feedback could be added to both the mobile phone and the IVE. So user can have feedback when he/she starts navigating for example. Also visual feedback on the mobile phone screen to illustrate the techniques used would make it easier for user to learn the techniques faster. Haptic feedback could be added when users press his finger on the mobile screen to start the navigation. Therefore, adding audio, visual, or haptic feedback should be investigated in the future work.

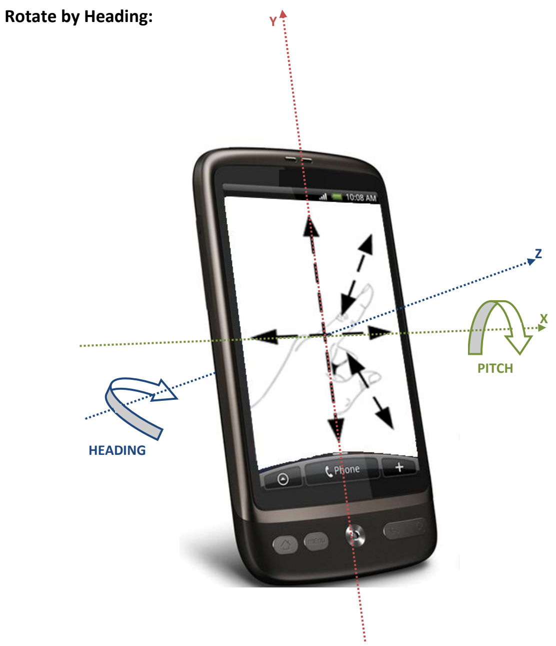
Qualitative Evaluation Questionnaires As mentioned in the section 5.5 that the qualitative evaluation questionnaires are not relevant for interaction in a VR system. Hence a future work to modify the qualitative questionnaires or have a new ones relative to the interaction within a VR systems would be of a great interest.

Online 3D Environment Future work could investigate merging the proposed travel techniques with a 3D VR systems, such as Google Earth 3D environment. Hence more users will test the new interaction metaphors, and more feedback from users will be available. And the users feedback will be of great interest for more future improvement or testing the usability and attractiveness of the new interaction techniques.

A Appendix

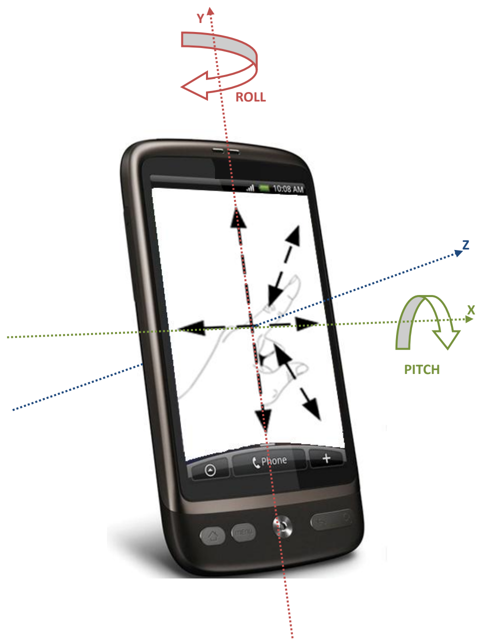
A.1 Steering techniques handled drawings

A.1.1 Rotate by heading

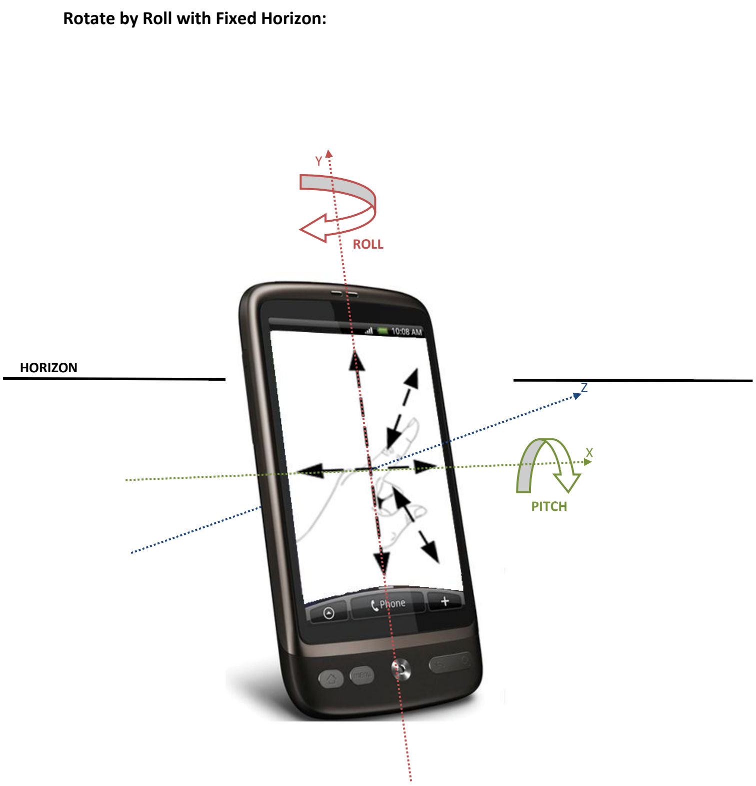


A.1.2 Rotate by roll

Rotate by Roll:

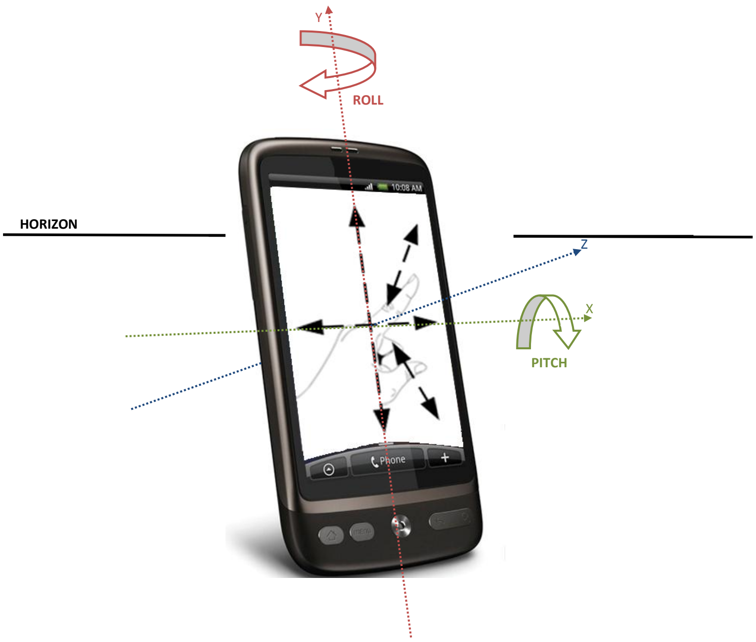


A.1.3 Rotate by roll with fixed horizon



A.1.4 Merged rotation

Rotate by Roll with Fixed Horizon:



A.2 Questionnaire

onlineFragebogen

33% ausgefüllt

2. Assessment of the System.

Please enter your impression using the proposed words to your chosen alternative.

Please click on a radio button for each line!

technical	human
isolating	connective
unpleasant	pleasant
conventional	inventive
complicated	simple
unprofessional	professional
ugly	attractive
impractical	practical
disagreeable	likeable
cumbersome	straightforward

Weiter

Es wurde noch kein Impressum eingetragen

oFb onlineFragebogen

50% ausgefüllt

3. Assessment of the System.

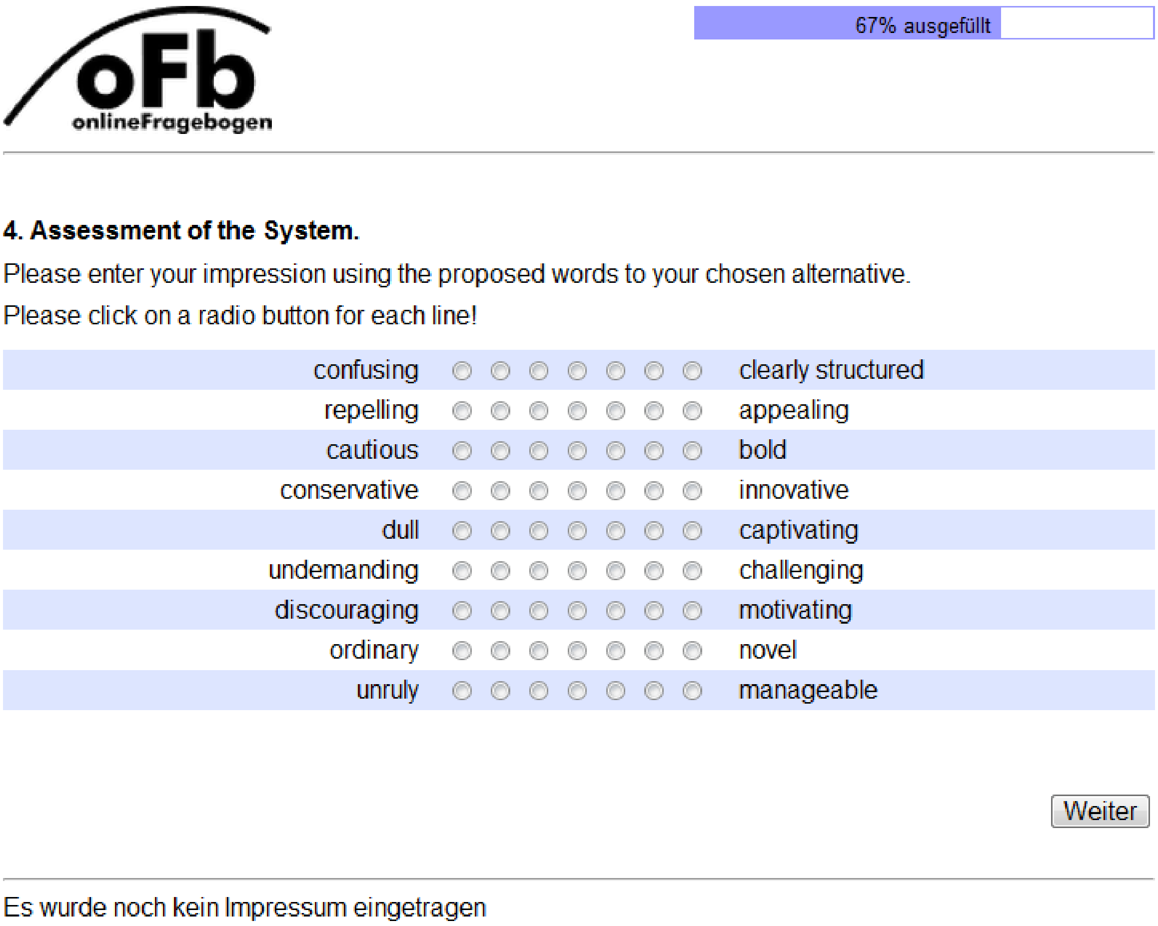
Please enter your impression using the proposed words to your chosen alternative.

Please click on a radio button for each line!

tacky	○	○	○	○	○	○	○	stylish
unpredictable	○	○	○	○	○	○	○	predictable
cheap	○	○	○	○	○	○	○	premium
alienating	○	○	○	○	○	○	○	integrating
separates me from people	○	○	○	○	○	○	○	brings me closer to people
unpresentable	○	○	○	○	○	○	○	presentable
rejecting	○	○	○	○	○	○	○	inviting
unimaginative	○	○	○	○	○	○	○	creative
bad	○	○	○	○	○	○	○	good

Weiter

Es wurde noch kein Impressum eingetragen



onlineFragebogen

67% ausgefüllt

4. Assessment of the System.

Please enter your impression using the proposed words to your chosen alternative.

Please click on a radio button for each line!

confusing	clearly structured
repelling	appealing
cautious	bold
conservative	innovative
dull	captivating
undemanding	challenging
discouraging	motivating
ordinary	novel
unruly	manageable

Weiter

Es wurde noch kein Impressum eingetragen

oFb
onlineFragebogen

83% ausgefüllt

5. Usibility Assessment of the System.

I think that I would like to use this system frequently.	○ ○ ○ ○ ○
I felt very confident using the system.	○ ○ ○ ○ ○
I thought the system was easy to use.	○ ○ ○ ○ ○
I would imagine that most people would learn to use this system very quickly.	○ ○ ○ ○ ○
I found the various functions in this system were well integrated.	○ ○ ○ ○ ○
I found the system unnecessarily complex.	○ ○ ○ ○ ○
I think that I would need the support of a technical person to be able to use this system.	○ ○ ○ ○ ○
I needed to learn a lot of things before I could get going with this system.	○ ○ ○ ○ ○
I found the system very cumbersome to use.	○ ○ ○ ○ ○
I thought there was too much inconsistency in this system.	○ ○ ○ ○ ○

Weiter

Es wurde noch kein Impressum eingetragen

oFb
onlineFragebogen

Thank you for your participation!

We want to thank you for your cooperation.

Invitation to SoSci Panel

Dear participant,
dear participants,

As non-commercial SoSci panel you would like to invite other scientific surveys. The panel respects your privacy, does not share your e-mail address to third parties and will be a year four invitations to high-quality studies send a maximum.

We would be pleased if you join the panel SoSci and thus continue to support the scientific research. The participation on the panel and the interviews are optional, you are obliged to do, and can be terminated at any time.

E-mail: [ ] Participate on the panel

You will receive a confirmation e-mail before your e-mail address, the panel included in. This will ensure that no one except you e-mail enters your.

The questionnaire, which you have just completed, was saved. You can also close the browser window of course, without the SoSci Panel to participate.

Bibliography

[Appa] Apple. *IPhone (Last accessed, September 2010).* http://www.apple.com/iphone/. [cited at p. 2]

[Appb] Apple. *IPod (Last accessed, September 2010).* http://www.apple.com/ipodtouch/. [cited at p. 2]

[BKH97] Doug A. Bowman, David Koller, and Larry F. Hodges. Travel in immersive virtual environments: An evaluation of viewpoint motion control techniques. *College of Computing, Georgia Institute of Technology*, 1997. [cited at p. 1, 2, 6, 7, 8, 13]

[Bow98] Doug A. Bowman. Interaction techniques for immersive virtual environments: Design, evaluation, and application. *College of Computing, Georgia Institute of Technology*, 1998. [cited at p. 1, 5, 6, 7, 13]

[Car00] Michael J. Caruso. Applications of magnetic sensors for low cost compass systems. 2000. [cited at p. vi, 14, 15]

[Goo] Google. *Android API from Android Developer (Last accessed, September 2010).* http://developer.android.com/reference/packages.html/. [cited at p. 20, 31]

[HTC] HTC. *Desire (Last accessed, September 2010).* http://www.htc.com/www/product/desire/overview.html. [cited at p. 2]

[Jaz] Jazzmutant. *Lemur (Last accessed, September 2010).* http://www.jazzmutant.com/. [cited at p. 2, 8]

[KGMQ08] Ji-Sun Kim, Denis Gracanin, Kresimir Matkovic, and Francis Quek. Finger walking in place (fwip): A traveling technique in virtual environments. *Virginia Tech, USA. and VRVis Research Center, Austria*, 2008. [cited at p. 1, 2, 3, 6, 8, 9]

[KGMQ09] Ji-Sun Kim, Denis Gracanin, Kresimir Matkovic, and Francis Quek. iphone/ipod touch as input devices for navigation in immersive virtual environments. *Virginia Tech, USA and VRVis, Austria*, 2009. [cited at p. 1, 2, 3, 6, 9]

[KIM05] JI-SUN KIM. Tangible user interface for cavetm based on augmented reality technique. Master's thesis, Faculty of the Virginia Polytechnic Institute and State University, December 2005. [cited at p. 1, 2, 5]

[Kir09] Andreas Kirsch. Multi-tile terrain rendering with ogl/equalizer. Master's thesis, Technical University of Munich, 2009. [cited at p. 3, 10]

[KR04] James F. Kurose and Keith W. Ross. *Computer Networking A Top-Down Approach Featuring the Internet*. Addison Wesley, third edition edition, 2004. [cited at p. 11, 12]

[LLC09] Chang Ha Lee, Alan Liu, and Thomas P. Caudell. A study of locomotion paradigms for immersive medical simulation environments. *Springer-Verlag*, March 2009. [cited at p. 7, 8]

[SGTF06] Alistair Sutcliffe, Brain Gsult, Kevin Tan, and Terence Fernando. Investigating interaction in cave virtual environments. *Centre for HCI Design, University of Manchester and Centre for Virtual Environments, University of Salford*, 2006. [cited at p. 1]