NTNU – Trondheim
Norwegian University of
Science and Technology

Navigation by the use of Maps in Virtual Reality portrayed by Oculus Rift

Haakon Gierløff

Master of Science in Engineering and ICT
Submission date: June 2014
Supervisor: Terje Midtbø, BAT

Norwegian University of Science and Technology
Department of Civil and Transport Engineering

NTNU
Fakultet for ingeniørvitenskap og teknologi
Institutt for bygg, anlegg og transport

Master thesis

(TBA4925 - Geomatikk, masteroppgave)

Spring 2014

for

Haakon Gierløff

Navigation by the use of maps in virtual reality portrayed by Oculus Rift

BACKGROUND

In 2012/2013 a new product for presenting virtual reality (VR) to the mass marked emerged. The product is named Oculus Rift, and is primarily intended for the gaming industry. However, the equipment, which has open programming interface, has showed potential for 3D visualization in different areas.

Different studies look on how people navigate by the use of maps, either as a traditional orthogonal projected 2D map, or as a 3D visualization of the environment.

In this thesis the candidate is going to look into how maps can be used as means to obtain better navigation in a VR model. This includes testing the proposed visualization in a virtual landscape.

TASK DESCRIPTION

Specific tasks:

- Study literature within relevant subjects as VR, geographic information, inner orientation and stereo vision.
- Suggest new and creative methods for including maps for better navigation in the VR space.
- Suggest and test different methods for the interaction with maps in Oculus Rift VR model.
- Make a 3D landscape model by the use of Oculus Rift development kit
- Test the different map realizations and solutions for interaction in this model.
- Analyze the results and estimate the usability of the methods.

ADMINISTRATIVE/GUIDANCE

The work on the Master Thesis starts on January 11th, 2014

The thesis report as described above shall be submitted digitally in DAIM at the latest at June 11th, 2014

Supervisors at NTNU and professor in charge:

Terje Midtbø

Trondheim, January 10, 2014. (revised: 02.06.2014)

Sammendrag

Da Oculus Rift ble introdusert i 2012, ble det forutsett at den kom til å bli en teknologisk enhet som ville forandre fremtidens underholdning. Fordi den er såpass rimelig i pris og utviklervennlig, har mange applikasjoner og spill med støtte for Oculus Rift blitt utviklet de siste to årene. Kreativ bruk av teknologien til Oculus Rift har så langt bare vist toppen av isfjellet når det kommer til potensial.

Ettersom Oculus Rift hovedsakelig ble laget for dataspill, så utmerker den seg når den brukes i førstepersonsspill. Denne oppgaven vurderer mulighetene for å vise kart på nye måter mens man navigerer i 3D omgivelser i virtuell virkelighet. Tidligere studier av stereoskopisk syn, sporing av hodebevegelser og grensesnittdesign blir diskutert, i tillegg til nåværende applikasjoner til Oculus Rift. Disse studiene blir så brukt til å vurdere nye metoder for å forbedre immersjon i virtuell virkelighet og brukervennlighet ved bruk av kart i 3D omgivelser. Det ble også laget et program i UDK (Unreal Development Kit) for å teste implementasjon av disse metodene.

Resultatene består av forskjellige teknikker og metoder. Disse er tilgjengelige for utviklere for å lage/forbedre kartlesing i 3D omgivelser kombinert med bruk av Oculus Rift. Nytteverdien til metodene blir diskutert, i tillegg til implementasjonsteknikker og tilpasningsmuligheter. Det blir konkludert med at alle metodene har potensiale til å forbedre kartlesing. Teoretisk er det mulig å implementere alle metodene, men begrensninger i UDK krever avansert forståelse av Oculus Rift for å kunne lage noen av dem.

Nøkkelord: Oculus Rift, virtuell virkelighet, kartvisningsmetoder, 3D, stereoskopi, UDK

Abstract

With the introduction of the Oculus Rift in 2012, it was predicted to become the new technological device that would change future entertainment. Being affordable and developer friendly, many applications and games with Oculus Rift support have been created the last two years. Creative uses of the Oculus Rift's technology have shown just the tip of the iceberg regarding its capabilities.

As the Oculus Rift was mainly created for gaming, it excels when used in first-person view. This thesis considers the possibilities to view maps in new ways while navigating in a 3D virtual reality environment. Previous studies regarding stereoscopic vision, head tracking and interface design are discussed, as well as current applications for the Oculus Rift. These studies are used to consider new methods to improve immersion and usability when viewing a map in a 3D virtual reality environment. In addition, a program was made in UDK (Unreal Development Kit) to test implementation of the methods.

The results consist of different techniques and methods. These are available for developers to create and improve map reading in a 3D environment, while using the Oculus Rift. Their usefulness is discussed, as well as customization and implementation techniques. It is concluded that all methods have the potential to improve map reading. They can theoretically be implemented, but certain limitations in UDK require advanced understanding of the Oculus Rift to be able to create some of them.

Keywords: Oculus Rift, virtual reality, map-display methods, 3D, stereoscopy, UDK

Preface

This master thesis is a result of cooperation between me, student technician Haakon Gierløff, and the Norwegian University of Science and Technology. I have been studying the field of geomatics for the past three years, and specialized in Geographical Information Systems (GIS).

Geomatics is an interesting field of study and it has been very rewarding for me to learn more about it from working with this thesis. With my background in computing and ICT, it has also been rewarding to work with 3D modelling for the Oculus Rift.

I would like to thank Professor Terje Midtbø at NTNU for valuable help and guidance throughout this project. He has answered my questions and given useful feedback, which I am very grateful for. I would also like to thank NTNU and the Department of Civil and Transport Engineering for providing software and access to an Oculus Rift.

Haakon Gierløff, June 10, 2014. Trondheim

Abbreviations and definitions

Accelerometer	An electromechanical device that measures acceleration forces on an object by registering the displacement of a mass on sensors. This includes the gravitational forces (static acceleration), from which one can find the angle the device is tilted, and forces regarding how the device is moving (dynamic acceleration).
Gyroscope	A device for maintaining orientation, i.e. a device that assists systems to keep track of the system's direction relative to a surface.
Heads-up display (HUD)	The method for which additional information is visually relayed to the user as part of an interface in a simulation. This is usually in front of everything else on the screen.
Information Clutter	A negative effect when too much information is relayed to the user at the same time. This can cause frustration and significantly reduce task completion time and quality of results.
Magnetometer	A device to measure strength and/or direction of the magnet field around it. Can be used as a compass (among other applications).
Mesh	A three-dimensional model without any texture on it.
Monoscopic Vision	Not being able to view three dimensional images, i.e. no depth perception.
Software Development Kit (SDK)	A set of tools that allows applications to work on certain platforms, and/or gives the possibility to develop applications.
Stereoscopic Vision	The ability to capture two images of an object from slightly different angles, and thereby creating a three-dimensional image of what is viewed.
Texture	An image that is used to "wrap" around a mesh.
Virtual Reality Environment	The world the user of a virtual reality system is in. This includes landscape, physical laws and interactable objects.
Virtual Reality Landscape	The graphical landscape the user of a virtual reality system moves around in.

Table of Contents

List of figures

List of tables

1 Introduction

1.1 Background

Since the creation of the first computers, software developers have tried to simulate real world activities through games, and immerse users into fictional worlds. This is known as virtual reality. The goal is complete immersion, where the user is unable to differentiate the digitalized world from reality (Rheingold, 1992). The uses of virtual reality are many; a few examples are risk-free training simulations, therapy and immersive entertainment (Bowman & McMahan, 2007).

Virtual reality can be traced back to 1727, when Robert Barker (Matthew & Harrison, 2004) had an idea of panoramic murals shown on the inside of a cylindrical surface, with a person standing inside. Although he was not the first to make such a mural, he was the first to create them solely to immerse the onlookers in an artificial environment. The concept of virtual reality was not truly introduced before the 1950s, but finding whom to credit for realizing the idea is difficult. Virtual reality can be seen as a concept fronted by several pioneers; Morton Heilig, Jaron Lanier, Douglas Engelbart, Ivan Sutherland and Myron Krueger (Virtual Reality Blog, 2009).

Virtual reality systems can be divided into three types (Kalawsky, 1996): Non Immersive Systems, Fully Immersive Systems and Semi-Immersive Systems.

1. Non Immersive Systems are the most commonly used today, with a field of view around 50°. An example is an explorable three-dimensional (3D) landscape on a computer desktop.
2. Fully Immersive Systems is what gives the user the greatest immersion, with a field of view at 360° and often accompanied by technology that excludes the "real world" around the user, e.g. CAVE Virtual Reality[1].
3. Semi-Immersive Systems are something between the two previous systems, with a field of view typically between 100° to 150° combined with virtual reality inducing technologies. Examples are professional flight simulators and advanced arcade machines.

Non Immersive Systems have been available at a reasonable price for a long time. However, in 2012 the Oculus Rift (see Figure 1-1) was introduced; an affordable Semi-Immersive System consisting of a head-mounted display with stereoscopic vision and head-motion tracking (Oculus VR, 2014). The Oculus Rift is also development friendly, allowing independent developers easy access to advanced development kits. This has caused a great interest in the advancement of

[1] www.mechdyne.com 30.05.2014

virtual reality, with many applications and games already created to utilize the Oculus Rift (Andersson, 2014).

Figure 1-1: Oculus Rift Development Kit 1 [2]

The Oculus Rift is still considered new technology, and so far there has not been much research delving into its capabilities in virtual reality. The Oculus Rift allows completely new ways to immerse the user, by adding a new control input (head-motion tracking) and adding a new level of detail (stereoscopic vision).

The Oculus Rift is well suited for use in first person perspective in a 3D landscape, and so far this has been the main focus for showing off the Oculus Rifts' capabilities. Once a user starts navigating through a 3D landscape, a map is usually provided, unless the game or application uses the absence of a map as part of the design (Darken & Cevik, 1999). The possibilities for new ways to view a map in Semi-Immersive Systems are unexplored, which leaves room for great advancements in displaying maps in 3D environments for the Oculus Rift.

[2] Image from https://www.oculusvr.com/order/dk1/ 29.05.2014

1.2 Objective

The main objective in this thesis is to establish new ways to view maps with the Oculus Rift and similar Semi-Immersive Systems. The following tasks are to be considered:

1. A literature study of the capabilities of the Oculus Rift and current games and applications utilizing it.
2. A literature study regarding the display of maps in 3D environments and in interfaces.
3. Evaluate the literature studies, and list different display methods for maps when using the Oculus Rift in a 3D virtual reality environment.
4. Create a 3D landscape where the different display methods for maps will be implemented and tested.
5. Evaluate the legibility and usability of the display methods, and their implementation.

1.3 Limitations

Certain limitations have been set to the extent of the task:

1. The study will only consider the Oculus Rift headgear, mouse and keyboard as input devices.
2. The Oculus Rift used for testing is the Rift Development Kit 1 (Oculus Rift DK1). When evaluating the results, it is the capabilities and functions of this version that is considered.
3. The display methods will be of a static map, and will not have position tracking.
4. The display methods will primarily be considered to be used with a first-person perspective in a stereo vision virtual reality environment.
5. The display methods discussed are those considered unique for the Oculus Rift and similar headgear, and methods not involving the Oculus Rift will not be considered.

This thesis consists of a literature study, discussing the Oculus Rift and previous research. The literature study is followed by a chapter about the implementation of map display methods in a 3D model using the Unreal Development Kit (UDK). The resulting map display methods are then discussed before the thesis work is concluded. Recommendations for further work are stated at the end.

2 Literature study

2.1 Oculus Rift

2.1.1 Technology

The Oculus Rift is in constant development. A new version, Rift Development Kit 2, was released March 2014 (Oculus VR, 2014). The version of Oculus Rift used for testing in this project is the older Rift DK1. This version of the headgear consists of two displays, one for each eye, with a resolution of 640x400 pixels. A different image is sent to each eye (Figure 2-1). The viewing optics allows a 100° angle field-of-view. In addition to the optics, the headgear also has a gyroscope, accelerometer and a magnetometer integrated to track the angle the users head is held and rotation speed.

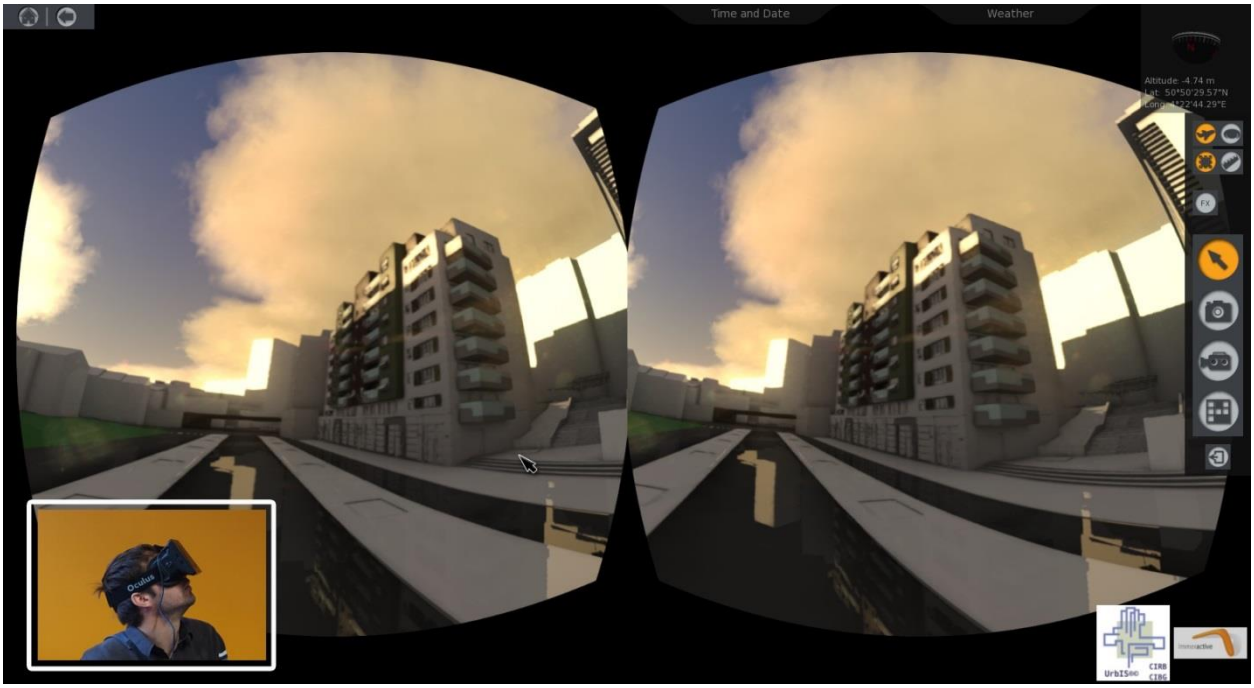


Figure 2-1: View of the images sent to each eye from the Oculus Rift [3]

This provides two new functions for the user: internal tracking and stereoscopic vision. In addition to improve the user's immersion, this also allows the Oculus Rift to enhance the control the user has in games and applications. With the internal tracking, the user's head-motions can now be used as an extra input device.

[3] Image from http://www.realareal.com/immeractive-uses-oculus-rift-for-immersive-virtual-walk-through-in-architectural-models 26.05.2014

The internal tracking allows:

1. Keeping track of head tilt angle.
2. Keeping track of horizontal head rotation angle.
3. Measuring head rotation speed: total speed, vertical speed and horizontal speed.

Stereoscopic vision with two displays adds a level of depth in the environment viewed by the user, adding a 3D effect without discoloring or flickering (Fauster, 2007).

This allows the user to:

1. Easily differentiate between close and faraway objects.
2. Estimate distance between the user and a point (as long as the destination is within his field of vision).

The Oculus Rift is already available for integration in several development tools (Oculus VR, 2014), most notably Unreal Development Kit (UDK) and Unity 4. Although integration through other programming languages and development tools is possible, the dedicated tools are set up for plug-and-play with minimal coding and adjustment needed from the developer.

2.1.2 Current application of the Oculus Rift

The Oculus Rift was originally designed for computer games (Kickstarter, 2014). Some games have implemented support for Oculus Rift after the release of the game, while some games were originally designed specifically for using the Oculus Rift. Its development kit was released together with "Doom 3: BFG Edition", as the Oculus Rift creators mainly designed it for first-person shooter games. However, as the popularity for the device increased, many game developers have found other types of games that benefit from Oculus Rift support (Andersson, 2014). Examples are third person games (where the camera view not only revolves around the playable character, but can additionally be rotated at its position) and two-dimensional (2D) games with a fixed camera (to create depth in the game world).

An unanticipated effect of the Oculus Rift's popularity was the rapid rise of developers using it for creating more than just games (Chacos, 2013). Oculus Rift supported simulations of real world experiences (such as sitting on a toured guide or going to the cinema) are now common, but developers are also delving into more sophisticated simulations. An example is Nonny de la Peña (Pyedog Productions, 2009) who creates "immersive journalism", a new form of journalism that lets the user experience the events reported in the news. By using real sound clips from the event and digitally rendering the events unfolding, the user is immersed and gets a much clearer understanding of what the reporter tries to explain happened. Another example is using the Oculus Rift to train amputees to use prosthetics before the prosthetic has been manufactured

(Chacos, 2013). Creating a customized prosthetic for a person is a time-consuming process, and this simulation may reduce recovery time for amputees.

2.1.3 First-person control types

When controlling from a first-person perspective, the Oculus Rift's internal tracking offer different control types. This has been thoroughly discussed between developers on the Oculus Rift official forum (Oculus VR, 2014). Although many control schemes are possible, these are the most common:

Table 2-1: Input type vs. control method for the most common control schemes

	Arrow keys or the W-A-S-D keys	Mouse	Oculus Rift head tracking
Standard virtual reality	Move character forwards/backwards and strafe left/right	Change front direction both horizontally and vertically	Rotate camera view vertically, Change front direction horizontally.
Alternative virtual reality	Move character forwards/backwards and strafe left/right	Change front direction both horizontally and vertically.	Rotate camera view both horizontally and vertically.
Standard first-person shooter	Move character forwards/backwards and strafe left/right	Change front direction horizontally. Vertical input disabled.	Change front direction both horizontally and vertically.
Adventure game 1	Move character forwards/backwards and rotate left/right	Mouse used to interact with the game world through point-and-click.	Change front direction both horizontally and vertically.
Adventure game 2	Move character forwards/backwards and rotate left/right	Mouse used to interact with the game world through point-and-click.	Rotate camera view horizontally, Change front direction vertically.

2.2 Previous studies and literature

In large-scale virtual reality environments, any map is better than no map (Darken & Sibert, 1996). The usefulness of a map in virtual reality is well documented. Many studies on maps in virtual reality have been done, but previous studies have focused on the design of the map and its readability, rather than on new ways to increase usability and immersion. Navigation in virtual reality is a recurring theme in said studies.

2.2.1 Stereoscopy

Arthur and Booth (Arthur, Booth, & Ware, 1993) studied the effects of using stereo vision in virtual reality. They concluded it had a positive effect; stereoscopic vision significantly increased task completion in an experiment involving participants performing a tracing task. However, it is often insufficient to rely solely on stereoscopic vision when viewing 3D objects. Another important factor for a user to determine depth with stereoscopic vision is the addition of shadows and textures (Barfield, Hendrix, & Bjorneseth, 1995).

Stereoscopy is not always helpful. Davis and Hodges discredit that stereoscopic vision always gives a better result, and state that monoscopic vision is more beneficial when using a flat plane view (Davis & Hodges, 1995). It has been concluded that although stereoscopic displays do enhance elevation judgments, azimuth judgments are not significantly enhanced (Barfield & Rosenburg, 1995).

2.2.2 Head tracking

Studies regarding the use of head tracking devices to register head movements as input have not concluded any negative results. Barfield, Hendrix and Bystrom (Barfield, Hendrix, & Bystrom, 1999) conducted experiments resulting in no decrease in task performance by using head tracking. To conclude if head tracking is better or worse than other input devices require further studies, and so far it is considered a valid alternative.

2.2.3 Maps in games

Most games where you navigate through a 3D landscape provide a map. Although the design and appearance of the different maps have been varied and innovative, the way the map is shown has not changed much.

The map is usually presented in two different ways:

1. Toggled on/off to cover the whole screen and thereby allowing interactions with the map (Figure 2-2). The game is either paused or made non-interactable, making the map serve as an interactable menu.
2. A mini-map in the Heads-Up Display (HUD) (Figure 2-3). Such a mini-map is non-interactable, and automatically follows the player's position and/or relevant area.

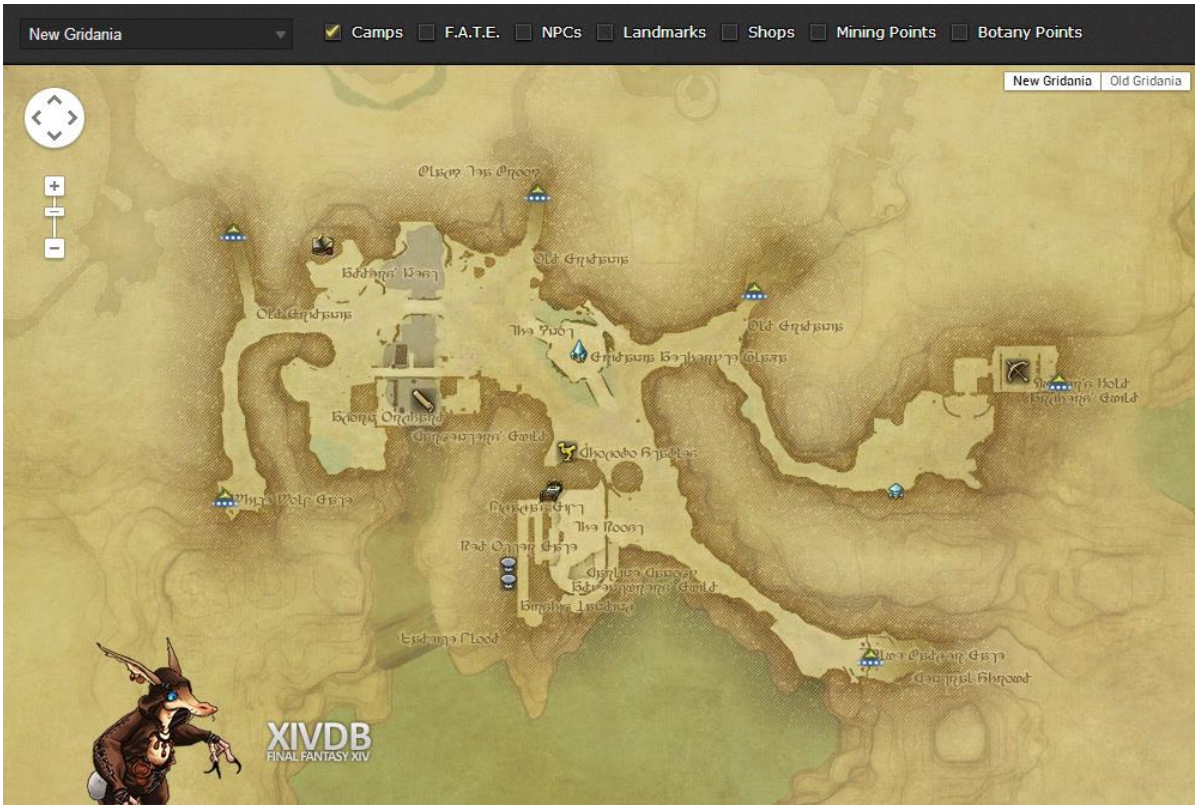


Figure 2-2: Map toggled on/off to cover the whole screen, taken from Final Fantasy XIV [4]

[4] Image from http://ffxivhub.tumblr.com/post/54543556536/a-realm-reborn-maps-on-xivdb-you-can-now-access 26.05.2014

Figure 2-3: Mini-map shown in upper right corner of the screen, taken from Final Fantasy XIV [5]

With the emergence of many MMORPGs (Massive Multiplayer Online Roleplaying Games), the need for a more efficient map has arisen. MMORPGs makes the user do immersive tasks that simulate living in another world completely, which also means the user often perform menial tasks for rewards (such as gathering resources). This has led to the creation of a third type of map; a map that covers a big part of the screen, allowing the user to quickly change between interacting with the map and the game world. With multiple input devices, a user can interact with both at the same time. These maps can often be turned transparent, thereby reducing the feeling that it covers up the screen (Figure 2-4).

Figure 2-4: Screen showing environment with and without transparent map, taken from Ragnarok Online 2 [6]

[5] Image from http://www.incgamers.com/2013/09/final-fantasy-xiv-realm-reborn-adventurers-log-1 26.05.2014
[6] Image from http://news.mmosite.com/content/2011-07-14/ro2_s_2nd_cbt_preview,3.shtml 26.05.2014

2.2.4 Immersion in virtual reality

Total immersion is the goal of virtual reality. However, by using the right stimuli, semi-immersive system can generate the same feeling of immersion as a fully immersive system (Bowman & McMahan, 2007). To create the most immersive experience in virtual reality, actions in the digital world need only to be performed in a similar way to how one would perform them in the real world.

Maps in first-person games have so far been in the HUD. According to Bowman and McMahan, a more realistic design would be to show and interact with the map the way it is read in the real world; held with the hands of the reader, usually at an angle combined with a slight tilt downwards with the head (see Figure 2-5)[7].

Figure 2-5: A typical map reading pose, taken from New Forest Navigation

It is worth noting that screen size and resolution have a huge impact on immersion, allowing the user to see more and reducing information clutter.

2.2.5 Usability of maps

Maps have much in common with normal interfaces. Therefore, the methods applied to create a good interface can be considered when creating methods to show a map. Schneiderman & Plaisant stress that designing a good interface is one of the most important parts when making any kind of application (Schneiderman & Plaisant, 2010). The controls must be intuitive, allowing the user to immediately understand how to perform the actions available. This also goes for hotkeys and shortcuts, which may be difficult for the user to remember if the keys are not related to the action it performs. An alternative to hotkeys is automatic activation when certain criteria are met. A developer must be careful when implementing this, as it may be considered very frustrating for users if an action is performed unintentionally.

As humans mainly rely on their vision when interacting with an interface (Schneiderman & Plaisant, 2010), innovative designs need to be considered when displaying a complex map. The map needs to be organized and intuitive; constantly showing excessive details can create

[7] Image from http://newforestnavigation.co.uk/ 09.06.2014

information clutter for the user. Zhai, Wright, Selker, & Kelin studied the use of different masks in interfaces; a layered effect covering parts of the interface that are not important, making certain elements stand out (Zhai, Wright, Selker, & Kelin, 1997). While the masking technique works, the choice of which mask to use creates very diverse results. The strength of the masking technique was also important, making it difficult to deem one technique the best.

3 Implementation

In addition to the literature study, a testing environment for the different map display methods was created. The main goals of this environment were:

1. Test ways to implement the map display methods.
2. Test performance of the map display methods when using Oculus Rift.
3. Discover unforeseen problems and new possibilities with implementation of the methods with Oculus Rift.

3.1 Concept

The environment to test the methods consists of an outdoor landscape with several landmarks that are easily identifiable for the user. This is for the user to quickly grasp the map, as testing the readability of the map itself is not the goal when using this environment. The perspective is in first-person, and the user is able to walk, jump (to escape eventual cavities he/she gets stuck in) and look around freely. In addition, the user is able to quickly switch between the different map-display methods through hotkeys.

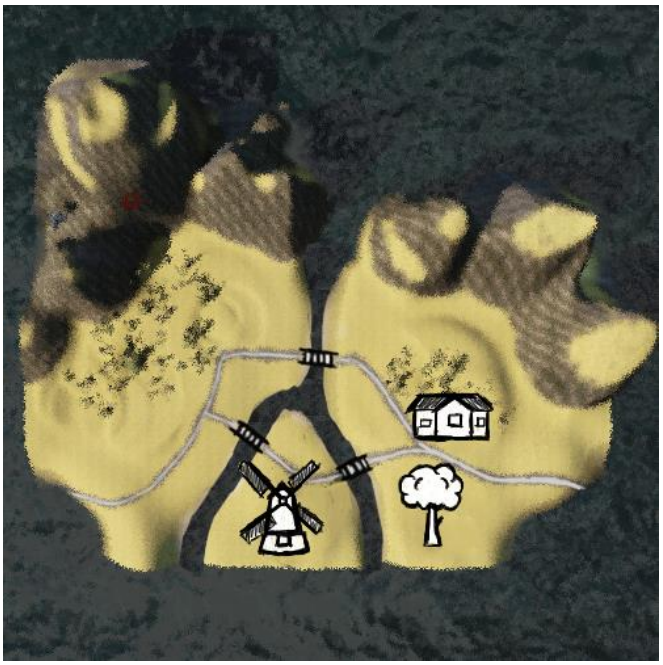


Figure 3-1: The map used in the implementation

The map consists of color coding for the terrain, and drawings for the landmarks (see Figure 3-1). This is an intuitive way to display a landscape with unique landmarks (Steck & Mallot, 2000).

To avoid confusion, no duplicate landmarks will exist.

3.2 Choice of development platform

Unity and Unreal Development Kit (UDK), which both have implemented Oculus Rift support, were the two main choices for development engines. With both these engines, Oculus Rift is plug-and-play ready when the Oculus Rift SDK is installed. Although other engines exist with Oculus Rift support, these are the most commercially available, with many support forums and help desks (Oculus VR, 2014). There is no unified opinion between developers which one is best, so choosing one is a subjective choice.

For the creation of this environment, UDK3 version 2013.09 was chosen. UDK3 is free to use as long as a developer does not have any monetary gain on his/her creations, as opposed to UDK4 which has a monthly fee[8]. It is considered more difficult to use than Unity, although it allows more control (Oculus VR, 2014). The main downside of UDK is that the royalty fee to the creators of the engine is higher when compared to the others. As this project is non-profit, this was not an issue.

3.3 Creation of landscape

UDK allows the developer to set a template for what kind of application is made. Setting a first-person shooter template, a simple control scheme was automatically applied. This allowed for instant testing of the landscape.

The landscape was created using the terrain editor in UDK. By creating a flat landscape and then manipulating it with convex/concave functions, a diverse terrain could be created. UDK has several terrain textures; however, other textures were needed in the implementation for this project. The predefined textures are from their flagship game; Unreal Tournament 3 [9]. As Unreal Tournament is played in a post-apocalyptic landscape, the predefined textures did not simulate the desired diversity. New textures were collected from UDKResources[10], which lets aspiring developers use pre-rendered textures for free. By using the terrain tool to paint the textures on the landscape in layers, a desired design was achieved (Figure 3-2).

[8] https://www.unrealengine.com/register 10.06.2014
[9] http://planetunreal.gamespy.com/View.php?view=UT3GameInfo.Detail&id=2 30.05.2014
[10] http://udkresources.com/index.php/materials/ 04.06.2014

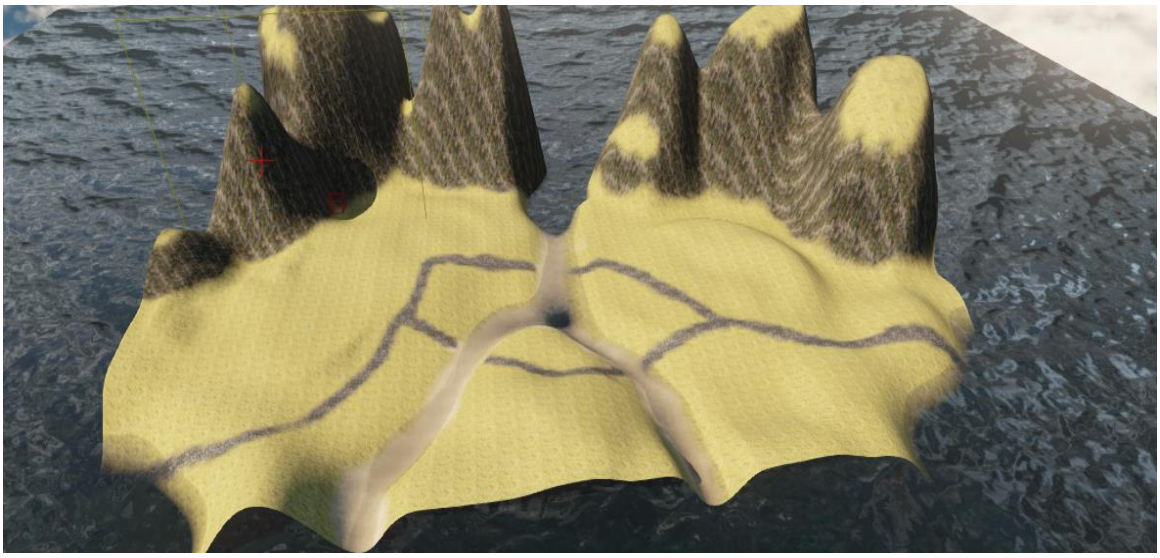


Figure 3-2: 3D landscape with no objects placed

3.4 Creation of 3D objects in the landscape

As the available 3D objects were from Unreal Tournament, their design was unfit to use for landmarks in this project. New landmarks had to be found to be able to use the map for navigation. Several sites allow 3D objects to be downloaded and used for free. For this implementation, free 3D models from UDKResources[11] and Turbosquid[12] were used to create new landmarks.

3D objects can be made in several different file-formats, and many of those which were downloaded were not available to use directly in UDK. A converting tool was needed to transform the 3D objects into meshes of the desired file format: ASE. For this task, 3ds Max was used[13]. 3ds Max is a graphic tool for creating your own 3D objects, but it also allows reading of many different file formats and then saving these as the desired format. Through this, several ASE files were created, ready for import to UDK (Figure 3-3).

In UDK, these ASE files could be given a collision model (to avoid the user running through the object) as seen in Figure 3-4.

[11] http://udkresources.com/index.php/speedtrees/ 04.06.2014
[12] http://www.turbosquid.com/ 04.06.2014
[13] http://www.autodesk.com/products/autodesk-3ds-max/overview 05.06.2014

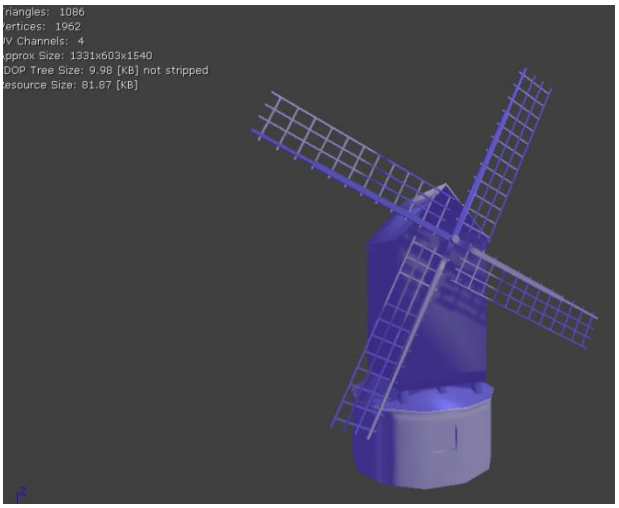


Figure 3-3: Mesh from ASE-file imported into UDK

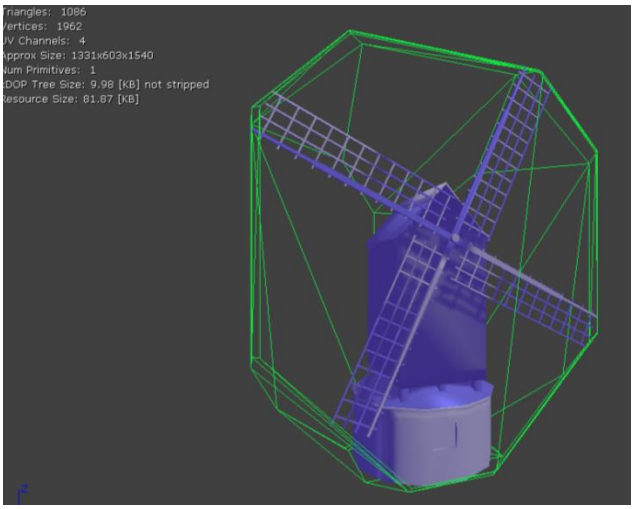


Figure 3-4: Mesh with collision model added

All the models that had been converted to ASE needed a texture. This was downloaded together with the model, and could easily be applied in UDK (Figure 3-5).

When all the objects had been created, and after having applied textures to the meshes, they could be placed out in the landscape as seen in Figure 3-6.

Figure 3-5: Completed 3D object with texture

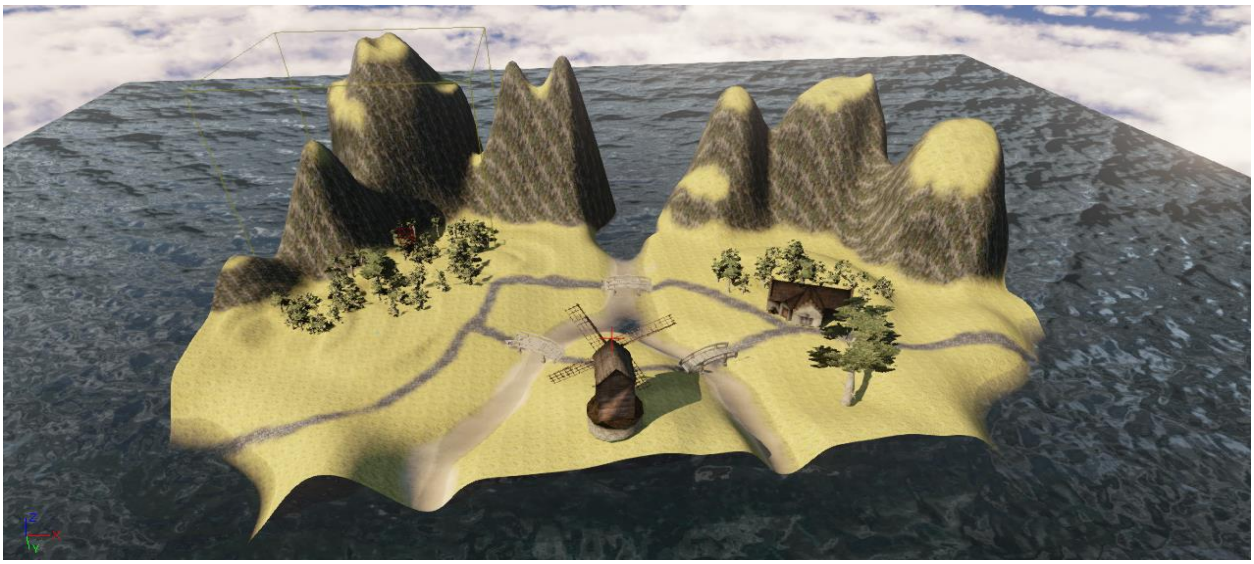


Figure 3-6: Completed 3D landscape with objects placed

3.5 Creation of map display methods

To create the map display methods, two tasks had to be done:

1. Create unique 3D-models to use in the methods.
2. Create and edit unrealscript-code (uc-code) to implement the methods.

As 3ds Max was already installed, it could now be used to create unique 3D models. This was needed to create the different maps to be used in the methods. First, the mesh was created in 3ds Max using the modelling tool and then converted to an FBX-file (**F**ilm**b**o**x**), which can be read by UDK. After creating the mesh, Photoshop was used to draw the different textures and save them as a TGA-file (**T**ruevision **G**raphics **A**dapter). Using the same steps as earlier, the mesh was given a texture, and was then ready to be used. The resulting maps are seen in Figure 3-7.

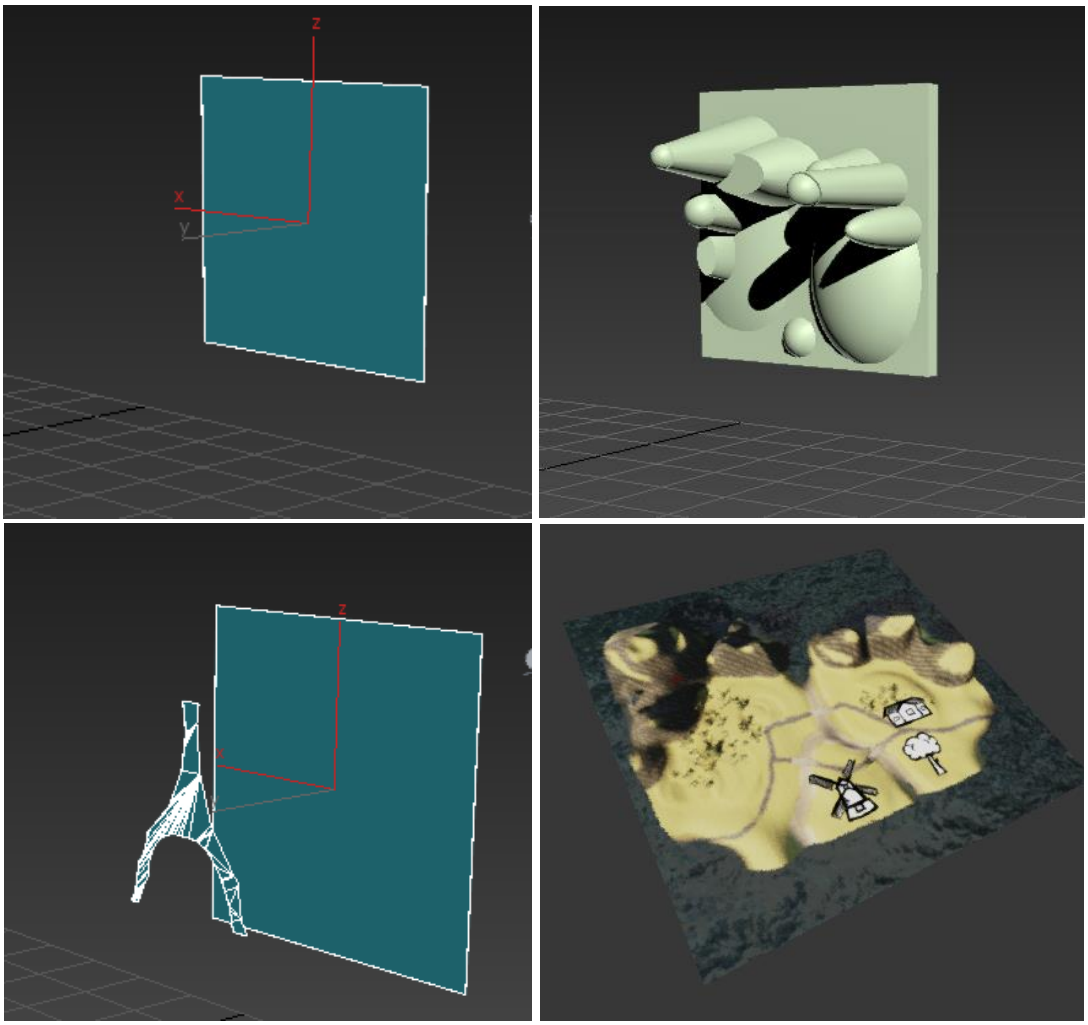


Figure 3-7: Meshes of the maps to be used in the different display methods. Top left: standard map. Top right: 3D map. Bottom left: map with elevated element. Bottom right: standard map mesh with texture.

To create uc-code, a text-editing program was needed. For this project, Notepad++ with a uc-code language plugin was used. To quickly test the code for bugs, a simplified build/run-tool was downloaded from "Magic Stone Studios"[14].

UDK automatically applies the code for Unreal Tournament to a landscape if nothing else is stated. This means all code for a working game is already written. The developer only needs to create classes which extend the old classes and overwrite/add/disable methods, or create new classes which add new functions.

To display the different map methods, the maps were handled the same way a UDK-based first-person-shooter handles weapons. Setting each method as a weapon-class, each map could be given unique 3D-models, animations and behavior. The maps could then be "equipped" by editing an equipment-manager-class already implemented in Unreal Tournament.

[14] http://magicstonestudios.com 04.06.2014

4 Resulting map-display methods and discussion

Based on information from the literature study, different map display methods were developed during the earlier phases of the thesis work. These map display methods have taken into account several theories from: stereoscopy, head tracking, immersion in virtual reality, Oculus Rift technology, and other factors. Using UDK, these methods were implemented in a virtual reality environment

The resulting map display methods come in two categories; map emergence and map manipulation. Map emergence involves methods to make the map appear and disappear for the user, while map manipulation involves methods on how the map can convey additional information. When using the Oculus Rift's internal tracking, three new input methods are provided for the user:

1. Rotation tracking from head motion, i.e. rotating the head simulates actions performed by a mouse (Barfield, Hendrix, & Bystrom, 1999).
2. Accelerometer threshold, i.e. quick head motions in certain directions work as pushing buttons.
3. Angle threshold, i.e. moving the head more than a given angle in certain directions work as pushing buttons.

For the threshold methods, the possibilities are many. They can replace any button functions, and these functions are as many as the currently used map functions toggled by hotkeys.

4.1 Map emergence methods

4.1.1 Stationary map viewed through rotation

When utilizing the Oculus Rift's head tracking, a map can be placed at a desired angle and slightly below the neutral field of view, requiring the user only to tilt the head downwards to see the map (see Figure 4-1 and Figure 4-2).

Figure 4-1: The view when looking straight forward in the program

Figure 4-2: The view when looking down in the program

Another function available combined with this is pulling up the map slightly when looking down (see Figure 4-3 and Figure 4-4). This reduces the amount of head-tilt required to view the whole map, and simulates the action of pulling up the map towards the face to see it more clearly:

Figure 4-3: The view when looking straight forward in the program, map held down.

Figure 4-4: The view when looking slightly down in the program; the map gets pulled up, thereby reducing tilt needed.

Method discussion:

With this method, the map is constantly available for the user. This creates dynamic map reading, allowing the user to view the map while performing the other actions possible at the same time. This method can be used on any part of the screen. One example is using it inside a vehicle cockpit as a map on the wall, placed to the left or right of the user's frontal direction.

Considering immersion, this method imitates the action of bringing a map closer to the user's face. Combined with the pull-up function, the action becomes less static and more natural, further simulating the real world action. A person holding a map will change the angle the map is held depending on if he/she is currently viewing the map, and will not hold the map close to his/her face when not viewing it.

A third function that can be implemented is changing the angle the map is held relative to the direction of view, thereby always aligning it to the user's point of view.

4.1.2 Semi-hidden map viewed through hotkey

The previous method is immersive, but it presents an issue when the user looks down; if the user just wants to look at the ground, the map will be in the way. To counter this, a toggle hotkey can be implemented to simulate pulling up the map. This avoids unintentional activation, and several hotkeys can be implemented to control the degree of map pull-up. As an alternative to several hotkeys, one can instead implement a cycling function to a hotkey, making the map more and more visible for each push. This is illustrated in Figure 4-5.

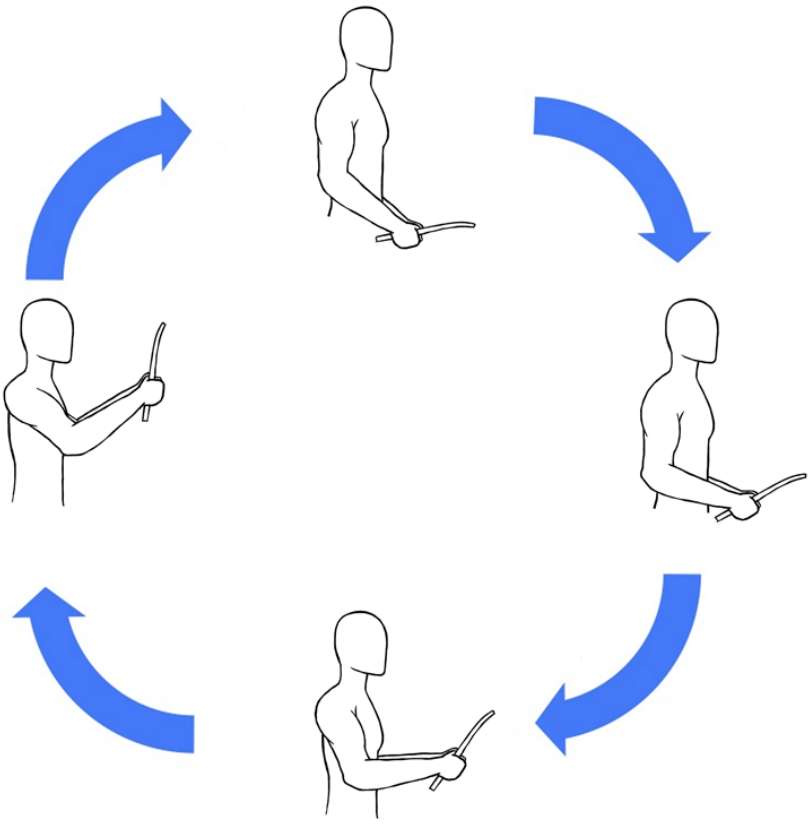


Figure 4-5: Cycling through map visibility for semi-hidden map

Method discussion:
Hotkeys are easy to use and helpful, but not very immersive. A hotkey only has two definite states, while a real world action usually consists of a smooth motion between states; i.e. different stages between start and finish. Using hotkeys can solve many problems with a virtual reality application, but can just as well break the immersion.

An alternative to hotkeys are triggers; keys that can register the amount of pressure exerted on it by the user. These kinds of hotkeys will simulate a real world action better, and create a bridge between the two states a normal hotkey has.

4.1.3 Semi-hidden map viewed through quick-tilt

To create a middle ground between the two previous methods, the accelerometer in the Oculus Rift can be used for activating the pull-up action. With a quick nod the user can activate the pull-up action, and also avoid increase in the number of hotkeys. This utilizes a function available to the Oculus Rift that is not used for anything in particular, as the accelerometer is mainly designed for keeping track of rotational position (Oculus VR, 2014).

Method discussion:
This method uses the Oculus Rift head tracking, but it does not simulate a real world action when reading a map. Using head tilting to trigger actions can be useful, but very few actions are actually triggered by a quick tilt of the head in the real world. Although it may be immersive for certain applications, e.g. using a quick tilt as an accepting nod when asked a question, using it for immersion in map reading might not be a good design. It might be considered a better alternative to hotkeys.

4.1.4 Transparent map

As discussed in chapter 2.2.3, many games utilize a transparent map that covers a part of or the full screen for task effectiveness. Even though it is a helpful tool, users might perceive it as either distracting or difficult to read.

The stereoscopic vision of the Oculus Rift can help solve this issue. With it, the user can more easily discern what is close (the transparent map) and what is far away (the landscape). This adds a new effect in addition to the transparency, clearly separating the map and the landscape and reducing the blending effect of the two. The difference between a transparent map in stereoscopic view and Monoscopic view is illustrated in Figure 4-6 and Figure 4-7.

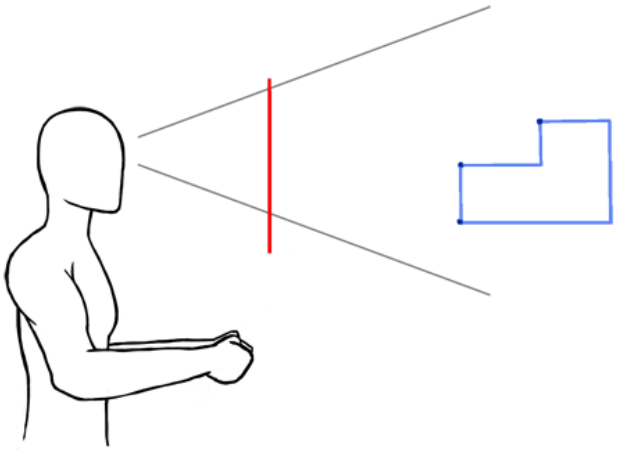


Figure 4-6: Viewing an object behind a transparent surface in stereoscopic view. The object and the items on the transparent surface are more easily distinguished from one another.

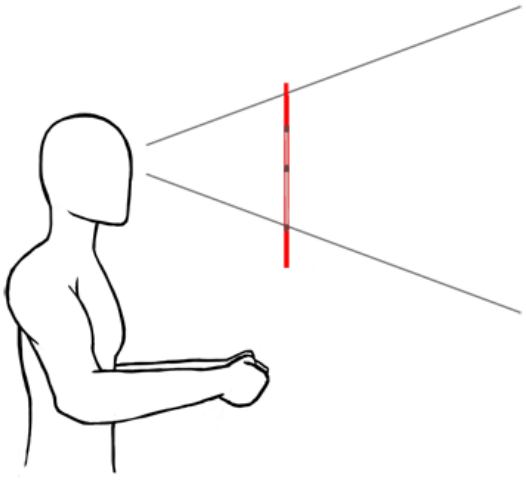


Figure 4-7: Viewing an object behind a transparent surface in monoscopic view. The object and surface blend together, making it difficult to distinguish between items on the transparent surface and objects in the landscape.

Method discussion:

A transparent map may not simulate normal map reading, but when considering today's new technology it is a realistic design. Headgear, goggles and glasses with interfaces displayed on them have been in production for some time in the military. An example is Google Glass[15], making this kind of technology available to the public. It is a user-friendly design when multitasking, and should be considered in virtual reality applications that focus on usability.

[15] http://www.google.com/glass/start/ 07.06.2014

A big limitation when using this method is that the best level of transparency is very subjective. There is a small step between the map being unreadable and/or the objects behind the map being obscured. If a developer wishes to implement this method, it is important that the user can control the level of transparency.

4.2 Map manipulation methods

4.2.1 Viewing a larger map

Once a user navigates through a large landscape with many landmarks, a small map with many details will be hard to read and give little to no information. It could therefore be required to have a large map where only a part of it can be displayed on the screen. The user will then have to scroll on the map to see other parts. As scrolling on the map will require input from the mouse, an input device is lost while interacting with the map. However, by using the Oculus Rift's head tracking for scrolling, the developer can avoid the user disabling one of his other input devices to use for map navigation. The user can then utilize the mouse for gameplay or map interaction instead.

Method discussion:
Increasing task efficiency is always a good thing, but this method requires the user to be accustomed to use the head tracking while using a mouse and a keyboard. This might make this method inefficient, as the more common scrolling methods might be more intuitive. Examples are auto scroll when the mouse reaches the border of the displayed part of the map, or toggle scrolling by pressing/holding a hotkey while moving the mouse.

Using the head tracking to control other aspects might be a better design. Some examples are controlling map rotation, zoom in/out or nodding up/down to change between maps (e.g. floors of a building).

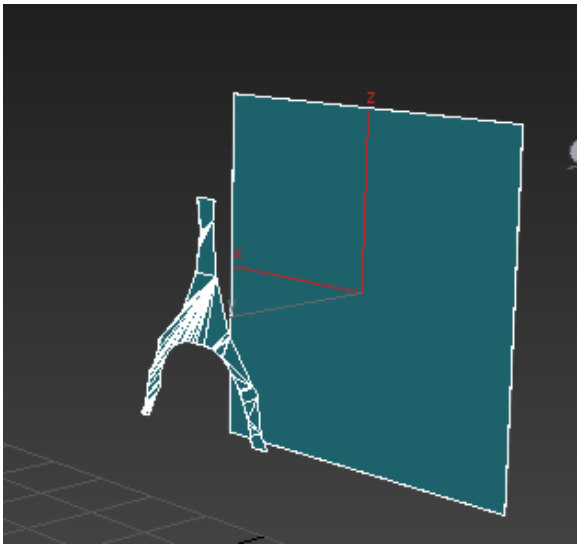
4.2.2 Extra information through stereoscopic vision

Maps convey information through colors, points, lines and figures. With the addition of stereoscopic vision, one can use depth as a method to give the user additional information. A flat map can be replaced with a 3D model of the landscape with varying levels of depth to e.g. tell the user the height differences. With monoscopic vision, the user would have to rotate the 3D model to interpret the additional information. Stereoscopic vision allows the user to read this information from a top-down view, thereby eliminating the need for map rotation.

Method discussion:
When relaying information this way, one must consider how important the accuracy of the information is. If the intent is to give a general idea of the landscapes topology, it is an intuitive way to display it. However, if the intent is to show accurate height levels, it might be difficult for the user to get any exact information.

If the developer chooses to let the 3D model of the map be rotatable, stereoscopic vision will still increase readability. Depth perception will make it easier to grasp the design of the model, and the developer may omit other visual cues (as lighting and textures) without making the model difficult to read.

4.2.3 Extra information through layer focus



Building on the previous method, a new method to emphasize on different aspects on the map can be implemented. Both maps and interfaces can use different methods to make certain parts of the map stand out. As established, stereoscopic vision can be used to convey additional information on a map. The developer can then make the whole map flat except one focused element, e.g. rivers, buildings or forests. The focused element of the map is elevated towards the user, with the rest of the map behind it.

Figure 4-8: Map with elevated river layer

Method discussion:
This method builds heavily on the concept of element-focusing from chapter 2.2.5. Unlike the masking techniques discussed, using stereoscopic vision combined with elevation of elements does not reduce the readability of the rest of the map. The user will perceive it as additional information, without omitting other information. What must be taken into consideration is not elevating the element too much, as it might start to cover the user's field of view.

The level of elevation giving the best results depends on the quality of the sterescopic images, as well as the user's ability to grasp stereoscopy. An alternative design might be to elevate the whole map except the focused element. This avoids the issue of having the focused element covering other parts of the map, and instead draw attention to the parts that are not elevated.

4.3 Summary

The Oculus Rift is a relatively new device; therefore, most of the previous research did not take the Oculus Rift into account directly. However, because the Oculus Rift uses similar technology, it is safe to assume that the findings from previous research apply to it. In the same way, computer games are a form of virtual reality. Map-display methods shown in games can therefore be used as inspiration for new map-display methods in virtual reality.

Although the methods were designed without considering position tracking, it will be fully possible to implement it into each method. As each map can be imported as an animation into UDK, having a map that updates visuals based on position should be fully possible. However, how to do this is not part of this thesis' objectives.

When considering what the best design choice is, a concrete answer is difficult to achieve. A map-display method that simulates a real world action might be immersive, but it might not have the best usability. Real-life actions are often more cumbersome compared to a simple push of a button. A good design would be to let the user customize the level of immersion and usability, and not assume to know what the user prefers. Since each method has several ways to be implemented, adding some sort of customization to each one should be considered.

4.3.1 Credibility of Implementation

The implemented methods work and act the way they are supposed to, but they rely heavily on UDK. Implementing them in other engines might be much more difficult, if not impossible without extensive modifications.

With the Oculus Rift support in UDK, creating a game or application for the Oculus Rift is done the exact same way as creating any other game or application. The simplification of implementation is user friendly, but clearly limiting when it comes to advanced control. Certain map-display methods seemed restricted to implement, as there were no predefined methods in UDK to access important functions.

The first issue was accessing the accelerometer. In theory, it is possible to use the accelerometer for methods, as it is used to trigger motion blur when rotating at different speeds. However, it does not seem that unrealscript can access this function directly. With extensive knowledge regarding the Oculus Rift, it should be possible to read directly from the sensors through an unrealscript method. Acquiring this kind of knowledge was too great of a task for this thesis, and the use of this function is at the moment purely theoretical.

The second issue was implementing a larger map to look around. Switching over to a map interface is possible, but assigning special commands to the Oculus Rift's head tracking seemed difficult. It falls under the same kind of problem as the first issue; theoretically possible, but requires detailed understanding of the Oculus Rift to implement.

The implementation does however prove that it is possible to create the most of map-display methods with today's technology. If UDK can achieve it, then all other engines and programming languages should also be able to do so (unless they have severe restrictions regarding editing code). However, one major restriction enforced by the implementation is the maps occupying the weapon inventory slots. This will deny the use of weapons in an UDK created game, meaning a workaround is needed for the map-display methods to be implemented in such a game.

5 Conclusion

The objective of this thesis was to investigate the different opportunities for a developer to utilize the Oculus Rift's functions, like head tracking and stereoscopic vision, to improve map reading in virtual reality.

The literature study gave insight into different technologies to improve usability and immersion. The most important features that can be used with the Oculus Rift technology were concluded to be stereoscopic vision and head tracking. Stereoscopic vision gives a better overview of a 3D environment. It also simulates the feeling of actually seeing the environment with your own eyes, instead of looking at a regular screen. Head tracking adds a new input device, allowing multitasking not possible with just a keyboard and mouse. By tracking head movements, real world actions can also be simulated, which increases immersion. Previous studies also showed that maps are an important feature in computer games, the most common types being a mini-map on the HUD and a toggled map interface.

The resulting map-display methods were based on previous studies, and formed by considering creative ways to utilize this research. The methods were divided into two main types:

1. Map emergence methods: how to make the map appear and disappear for the user, e.g. stationary map viewed through rotation
2. Map manipulation methods: how the map can convey additional information e.g. extra information through layer focus

To improve immersion and/or usability, all of the resulting map-display methods have potential to be useful tools. Each method can be implemented in many different ways, allowing customization of each method for the user, which again can increase usability.

All of the resulting map-display methods can theoretically be implemented. UDK was used in this thesis to test implementation of some of the methods. Although UDK has a large number of functions, it lacks the ability to directly access and modify the head tracking. This restricted testing some of the map-display methods relying on head tracking, as extensive understanding of the Oculus Rift's sensors and output was needed.

The methods using stereoscopic vision, e.g. transparent map, provided decent results. Exactly how great the effect could be is not concluded in this thesis, as it relies heavily on the quality of the displays. Without further testing, it is not possible to say if the Oculus Rift Development Kit 1 has a high enough resolution to utilize these methods fully.

The methods are not final, and surely there are additional ways to utilize the Oculus Rift to improve immersion and/or usability. Until new information becomes available or the Oculus Rift is improved, the methods discussed in this thesis can be considered current solutions.

6 Recommendations for further work

- The map-display methods in this thesis are not final. As more research is done and the Oculus Rift improves, it will be interesting to investigate new methods.
- With the restriction placed on the weapon inventory by the current implementation, other ways to implement the map-display methods in unrealscript should be investigated.
- Evaluate if other development platforms are better than UDK to create Oculus Rift applications and games.
- Perform user tests on the different map display methods, discovering which one has the greatest usability and/or immersion.
- Perform user tests on each method's different design choices, finding the best way to design each method.

7 References

Andersson, K. (2014). *RiftEnabled™ – Oculus Rift enabled list of games & demos*. Retrieved May 21, 2014, from http://www.riftenabled.com/admin/apps/

Arthur, K. W., Booth, K. S., & Ware, C. (1993, July). Evaluating 3D task performance for fish tank virtual worlds. *ACM Transactions on Information Systems, 11*(3), pp. 239-265.

Barfield, W., & Rosenburg, C. (1995, March). Judgements of Azimuth and Elevation as a Function of Monoscopic and Binocular Depth Cues Using a Perspective Display. *Human Factors: The Journal of the Human Factors and Ergonomics, 37*(1), pp. 173-181.

Barfield, W., Hendrix, C., & Bjorneseth, O. (1995, October). Spatial performance with perspective displays as a function of computer graphics eyepoint elevation and geometric field of view. *Applied Ergonomics, 26*(5), pp. 307-314.

Barfield, W., Hendrix, C., & Bystrom, K.-E. (1999, April). Effects of Stereopsis and Head Tracking on Performance Using Desktop Virtual Environment Displays. *Presence: Teleoperators and Virtual Environments, 8*(2), pp. 237-240.

Bowman, D. A., & McMahan, R. P. (2007, July). Virtual Reality: How Much Immersion Is Enough? *Computer, 40*(7), pp. 36-43.

Chacos, B. (2013, July 10). *PCWorld*. Retrieved June 9, 2014, from http://www.pcworld.com/article/2043946/this-is-not-a-toy-oculus-rifts-virtual-talents-could-transform-real-lives.html

Darken, R. P., & Cevik, H. (1999, March 13-17). Map usage in virtual environments: orientation issues. *Virtual Reality, 1999. Proceedings., IEEE*, pp. 133-140.

Darken, R. P., & Sibert, J. L. (1996). Navigating large virtual spaces. *International Journal of Human-Computer Interaction, 8*(1), pp. 49-71.

Davis, E. T., & Hodges, L. F. (1995). Human stereopsis, fusion, and stereoscopic virtual environments. In W. Barfield, & T. A. Furness III, *Virtual Environments and Advanced Interface Design* (pp. 145-174). New York: Oxford University Press, Inc. New York, NY, USA ©1995. ISBN 0-19-507555-2

Fauster, L. (2007). *Stereoscopic Techniques in Computer Graphics.* Technische Universität Wien, Institut für Computer Graphik und Algothmen. Wien: Technische Universität Wien.

Kalawsky, R. S. (1996). *Exploiting Virtual Reality Techniques in Education and Training: Technological Issues.* Loughborough University of Technology, Advanced VR Research Centre. Loughborough: Sima.

Kickstarter. (2014). *Oculus Rift: Step Into the Game by Oculus — Kickstarter*. Retrieved May 22, 2014, from https://www.kickstarter.com/projects/1523379957/oculus-rift-step-into-the-game

Matthew, H. C., & Harrison, B. (2004). *Oxford Dictionary of National Biography*. Oxford University Press. ISBN 978-0-19-861411-1

Oculus VR. (2014). *Oculus Rift - Virtual Reality Headset for 3D Gaming*. Retrieved May 1, 2014, from http://www.oculusvr.com/

Pyedog Productions. (2009). *Nonny de la Peña*. Retrieved May 22, 2014, from http://www.nonnydlp.com/

Rheingold, H. (1992). *Virtual Reality*. New York: Simon & Schuster. ISBN 978-0671778972

Schneiderman, B., & Plaisant, C. (2010). *Designing the User Interface.* Boston: Pearson. ISBN 978-0-321-60148-3

Steck, S. D., & Mallot, H. A. (2000). The Role of Global and Local Landmarks in Virtual Environment Navigation. *Presence, 9*(1), pp. 69-83.

Virtual Reality Blog. (2009). *Virtual Reality*. Retrieved May 1, 2014, from http://www.vrs.org.uk/virtual-reality/invention.html

Zhai, S., Wright, J., Selker, T., & Kelin, S.-A. (1997). *INTERACT '97 Proceedings of the IFIP TC13 International Conference on Human-Computer Interaction* (pp. 59-66). London: Chapman & Hall.

General web pages:

www.mechdyne.com 30.05.2014

https://www.unrealengine.com/register 10.06.2014

http://planetunreal.gamespy.com/View.php?view=UT3GameInfo.Detail&id=2 30.05.2014

http://udkresources.com/index.php/materials/ 04.06.2014

http://udkresources.com/index.php/speedtrees/ 04.06.2014

http://www.turbosquid.com/ 04.06.2014

http://www.autodesk.com/products/autodesk-3ds-max/overview 05.06.2014

http://magicstonestudios.com 04.06.2014

http://www.google.com/glass/start/ 07.06.2014