Graduate School of Natural Sciences
Department of Information
and Computing Sciences

Academic year 2011 — 2012

In collaboration with

3D Manipulations in Handheld Augmented Reality Applications

Robin van Olst

Supervisor: dr. W.O. Hürst

In partial fulfillment of the requirements for the degree of
Master of Science

Thesis ID: ICA-3025985

This thesis was written as part of an internship for Avanade Netherlands, an IT company that is specialized in Microsoft technology. I write this thesis in plural form, because it hasn't only been my personal effort that made it. My supervisors have done a tremendous job at supporting me and providing feedback, which is why I consider them to be an integral part of this thesis. Henceforth, writing this thesis in singular form feels misleading to me, which is why I choose to write in plural form instead.

Abstract

In this master thesis, we investigate and compare the usability of two interaction concepts for 3D augmented reality applications on handheld mobile devices. Augmented reality applications allow a user to look at the live image of a video camera and enrich the scene with 3D virtual objects. Interaction with the virtual objects requires the user to perform manipulations in 3D space, while common interaction on mobile devices is often touch screen based and therefore in 2D space. Mapping 2D space interactions to 3D space manipulations requires a non-trivial solution. We deal with this issue by mapping the interactions along the physical surface on which the virtual objects are augmented. This solution allows the user to move a virtual object along the physical surface, similar to an object in the real world, using only 2D interactions. We developed a touch-based and a crosshair-based interface that act according to this principle. The crosshair-based interface was expected to perform better in performance and usability, since it avoids some issues that the touch-based interface suffers from. We compare the performance and usability of these two interfaces by conducting a user study. From the results, we conclude that the touch-based interface outperforms the crosshair-based interface in both performance and usability for a various number of reasons. During the user study we also observe an important behaviour: users tend to keep the device stationary and use it as a window into the augmented reality environment. We present this observation as part of this thesis' major contributions.

Acknowledgements

First of all I would like to thank Christiaan Veeningen and Gijs Ramakers, my supervisors at Avanade Netherlands, who supported me throughout the course of this thesis. Their extensive support and suggestions for development, writing and data analysis helped me a lot with writing a solid thesis. I would also like to thank dr. Wolfgang Hürst for his support and extensive feedback on my thesis. A special thanks goes out to my family, friends and girlfriend for their endless support and advice in time of need. I also would like to thank all the people who were willing to participate in my user study for their cooperation.

A final word of appreciation goes out to all the interns whom I worked with. Their presence made my internship an especially joyful experience that I won't easily forget.

Contents

Chapter 1

Introduction

Augmented reality (AR) is a live view of the real world that is seamlessly combined with virtual visuals, such as objects or text, in order to enrich the user's experience. Many augmented reality application have already been developed. The most well known application of AR is present in major sports events, such as swimming or rowing. As visualized in Figure 1.1, augmented reality allows the world record line to move along with the swimming, while in rowing the country flags are digitally overlayed on the water in each respective country's lane.

Figure 1.1: Augmented reality in sports.

Since smartphones have become commonly available and their graphical processing power increases with each generation, the field of augmented reality for handheld device — subsequently called *handheld AR* — is becoming more interesting. A very popular handheld AR application is called Layar, which was released in 2009. Layar is an AR browser that displays information about the immediate surrounding of the user, such as building and other locations or points of interest. In June 2012, the creators of Layar released Stiktu, which is a more social variant of Layar, allowing users to annotate the world. It provides users with the ability to annotate any image that can be scanned on the fly, and allows users to view the annotations that other people made. The two application are showen in Figure 1.2.

With regards to future possiblities, augmented reality could provide a lot of interesting additions to various fields of work. For example, in artistic design, artistic impressions that

Figure 1.2: Augmented reality on handheld devices. Left is Layar, right is Stiktu.

would be impossible to realize in the physical world could be augmented to create novel and interactive virtual art accessible for a wide audience. For museums, whole collections of artifacts that are stored in the catacombs due to limited physical space can be virtually displayed to the public next to other physical objects. In entertainment, context-aware movies or games could take advantage of the environment to produce interesting changes in the story or gameplay; imagine a user playing a Real Time Strategy game taking place in a part of his living room, building his base on the table and then waging war upon the enemy's base located on the sofa. In addition, all of the above would be possible while the user is able to look at the scene from any point of view, allowing him to be immersed in the augmented environment. What we especially find interesting, is to think about how our daily lives could benefit from augmented reality.

1.0.1 A future scenario with Augmented Reality

In order to portray the possibilities of augmented reality for everyday life, we'd like to share our idea of how it could be used in the future. Imagine a man called Jonathan, who wants to redecorate a room in his house: *Jonathan is in his house and decides that it is time to redecorate one of the rooms. However, he isn't sure what he would like the room to look like. Using his mobile phone, he browses to a website about designing furniture and downloads a couple of interesting models for each of the following objects: a chair, a side table and an artistic vase. He then launches the augmented reality application and points the camera of his mobile phone at the room. Jonathan selects one of the chair models present on the device and virtually places it in the room. Since the application is aware of the dimensions of the room, it is able to place the chair model on the ground and at the correct scale. Jonathan can now walk around the virtual chair, viewing it from multiple angles, as if it were a physical chair that is actually present in the room. He adjusts the chair to his desired location and orientation, and repeats the same procedure for the side table and the vase. Jonathan tweaks the virtual furniture, and when he is satisfied with the looks of his room, he sends the models* *of the furniture to his 3D-printer. When the construction of his furniture is done, he places it in his room according to the augmented environment he created. When walking around his room, Jonathan now sees the same view in reality as when he was only looking at the augmented reality.*

An augmented reality application as illustrated in our example allows the user to be immersed in an environment where virtual objects appear to be part of the physical environment. The ability to walk around in an augmented reality environment and look at objects from multiple points of view, without limits beyond what is physically possible, creates a lot of potential for interesting augmentations to our daily life. This potential only increases when the user is given the power to interact with the virtual objects.

1.1 Goal of this thesis

Allowing users to interact with virtual content is a major challenge in handheld AR. In order for handheld AR to reach its full potential, users must be able to create, manipulate and edit the virtual content and their properties with respect to the real world in 3D space. However, the common interface on mobile devices is the touch screen, which is limited to interactions in 2D space. Mapping interactions in 2D space to manipulations in 3D space is a non-trivial problem that must be solved. In addition, interaction using the touch screen suffers from other problems as well. The most notable problem is that the finger used for interaction covers the content that the user wants to interact with. In regular applications the interactable content can be scaled up to a size where the user can comfortably interact with it. However, AR doesn't have this luxury, since the size of the content is dictated by its position in the real world. Therefore it is possible that content is too small for the user to comfortable interact with using the touch screen.

At the moment of writing, manipulations in 3D space are uncommon in handheld applications. This is even less the case in handheld augmented reality applications, where the registration of the virtual content to the physical world adds to the complexity of interactions. However, we believe that as research into this topic progresses, such applications will arise in the future. We envisioned such an application in our future scenario described earlier. This idea allows for users to be immersed in an augmented environment of which they are the creator, which requires an intuitive interaction concept to succeed.

In order to determine an intuitive interaction concept, we first select two promising interaction concepts. We limited ourselves to the most common interfaces that can be applied to augmented reality, which are tangible user interfaces (TUIs), gestural interfaces, touch interfaces and magic lens interfaces.

Figure 1.3: Various interfaces possible to be used for augmented reality for mobile devices. In the top left is the TUI, in the top right a touch interface, in the bottom left a gestural interface, and in the bottom right a magic lens interface.

These interface are illustrated in Figure 1.3. In chapter 2, we will determine that TUIs and gestural interfaces are not optimal to be used with handheld AR. For that reason, we choose touch-based and crosshair-based interfaces as the most promising type of interface for interactions in handheld AR.

The aim of this thesis is to contribute to the general goal of interaction in a 3D augmented reality setting for handheld devices. As such, we specify the main goal of this thesis as following:

- Determine which interaction concept is best capable of performing canonical manipulations in 3D space for handheld augmented reality.

To reach this goal, we need to answer the following research questions:

1. Which canonical manipulations must be supported?
2. How can 2D interactions on the device be converted to manipulations in a 3D virtual environment?
3. How does the performance of the touch-based and crosshair-based interaction concepts compare?

4. How does the usability of the touch-based and crosshair-based interaction concepts compare?

In order to answer the last two questions, we will conduct a user study to measure the performance and usability of both interaction concepts.

1.2 Overview

In the remainder of the article, we start by describing our context of our research and discussing related work in chapter 2. In chapter 3, we describe the research we've done in order to create an application that can be used to test interaction concepts. In chapter 4, we describe the interaction concepts that we will test. In chapter 5, we describe the experiment in which we test the interaction concepts. In chapter 6, we analyze the results we've acquired from the experiment, which are discussed in section 6.4. We conclude this thesis in chapter 7.

Chapter 2

Context and related work

The first appearance of augmented reality was in the 1960's, when the first AR interface was developed by Sutherland[38]. Since then, the research in the field of AR was primarily focused on developing the technology to provide a richer visual experience for the user, such as tracking and display devices.[1] The user interaction with the AR environment was either non-existant or very basic. Only more recently has the research shifted to also develop methods to improve the user's interaction.

Many different types of interfaces for AR have been developed over the years, each with its own limitations and best practices. Here we discuss the four types of interfaces that can be used for augmented reality, which we introduced in chapter 1.

2.1 Tangible User Interfaces

Tangible user interfaces (TUIs) allow a user to interact with virtual content using physical objects that are recognized and tracked by the system. TUIs were developed by Ishii and Ullmer in 1997[16] and has since then become one of the most common approaches for interaction with AR. TUIs have familiar properties, are subject to physical constraints and its actionable properties (how the object should be used) are clear to the user. For these reasons, TUIs are intuitive for a user to interact with and they have been proven to work well in practise. TUIs work best when effects are augmented to physical models. For example, the Urp application developed by Underkoffler and Ishii in 1999 allows the user to place and manipulate real building models, which are then augmented with projections of virtual wind and shadows.[40] Another type of TUI, which is currently featured in many AR applications, is the fiduciary marker: a square, black-and-white image that can be printed and attached to a surface. A good example is the VOMAR application developed by Kato et al. in 2000[18], which allows a user to decorate a small-scaled room with furniture using a physical paddle with a fiduciary marker attached to it. However, the use of a TUI also has its downsides: it forces the user to interact with a virtual object by using a physical object, even when interacting with pure virtual objects. This is an unwanted situation; the physical object itself doesn't add value to the augmented reality and also imposes physical limitations which may be undesirable. The interaction is also less suited to be used in combination with with handheld mobile devices than mounted devices (such as the VOMAR application using a head-mounted device), since the user must spread his attention over the mobile device and the TUI, and he only has one hand available for the interaction.

2.2 Gestural interfaces

Gestural interfaces allow a user to interact with virtual content by using body, hand or finger gestures. A lot of research has been done with regards to gestural interfaces and AR, where recent research appears to have shifted towards AR using head-mounted devices (HMDs). Various applications have been developed where the user can press a virtual button by holding his finger on it, or selecting a virtual object by pointing at it[10, 11, 19]. Buchmann et al. developed an urban planning application where the user can draw streets and place buildings using hand gestures[2]. Research into gestural interfaces for non-HMD mobile devices shows that the users find the interactions to be fun and engaging. For example, Caballero et al. developed an application where the user manipulates content in a virtual environment using gestural interactions[3]. The application takes advantage of uncommon hardware to achieve a high performance on gesture recognition. The research of Hürst and van Wezel shows that the performance starts to lack when using commonly available hardware and an AR environment. While users still find the application to be fun and engaging, the issues show that gestural interaction is less suitable for more serious AR applications[15]. Gestural interfaces on mobile devices also suffers from the more limited distance between the camera and the interacting appendage, making this a less than ideal combination.

2.3 Touch-based interfaces

Touch-based interfaces allow users to interact with virtual content through finger presses on the touch screen of the mobile device. This type of interface is commonly available on the current generation of mobile devices and is a familiar way of interacting for a wide audience. It has the advantage that no hardware is required beyond the mobile device itself, and many people are comfortable with using it.

Research into touch-based interfaces for AR has mostly focused on interactions using a stylus[5, 47], which is a pen with a magnetic tip used for interactions on the touch screen. In addition to regular touch interactions, the magnetic tip allows a user to perform interactions by hovering the stylus over the touch, allowing for a wider range of interactions. Generally, the stylus allows the user to perform both moves and drags on a touch screen, as opposed to only drags when using his finger. Another advantage is that the stylus more accurate than finger interactions and it doesn't block the screen. Wilkinson and Calder investigated touch interactions with a stylus on a non-viewable touch screen, which was duplicated to a HMD[45]. In his setup, a user had to select objects using the stylus on the non-viewable touch screen, which worked well in his setup. The downside to using a stylus is that it requires the user to have access this piece of additional hardware, which is what we try to avoid. In addition, a user can only use one stylus at a time. If the user wishes to use more advanced interactions which may require multi-touch functionality, then the stylus may prove to be an impeding factor.

Research into non-stylus touch-based interfaces focus mostly on techniques to increase the accuracy of the interactions. The work of Lee and Billinghurst introduces a technique called Snap-To-Feature, which aids the user in drawing the outline of physical objects[21]. Vogel and Baudisch developed a technique to aid selection by providing the user with a pointer above the touch location of his finger[43]. The work of Olwal et al. introduced several techniques to aid a user with zooming, allowing him to perform fluent zooming with a single finger by using a rubbing technique.

Research involving touch interaction and 3D manipulation has mostly limited itself to tabletops[34, 42, 46], and not into mobile devices. Research into touch interaction and AR has mostly limited itself to 2D manipulations, such as annotation[21, 22]. To the best of our knowledge, there has been no research looking into touch interactions for handheld AR which explores possibilities to perform 3D manipulations.

There are some downsides to the touch interface. Touch is a 2D interface, which means that translating touch interactions to 3D manipulations requires a non-trivial solution. The user's interacting finger will partially occlude the screen, limiting vision of the AR environment from him. This is moreover problematic, since the size of a virtual object in an AR environment depends on its location in the real world. It could well be that an object will appear small and is hard to manipulate. This is further complicated by the "fat finger problem" touch interfaces suffer from[14].

2.4 Crosshair-based interfaces

Crosshair-based interfaces are technically magic lens interfaces[47] with the addition of a crosshair. A magic lens interface is defined as: an interface for handheld devices that allows the user to see additional information when pointed at a physical surface or object[13]. The use of a crosshair for magic lens interfaces allows it to avoid some of the problems that touch-based interfaces have to deal with, such as the previously mentioned fat finger problem.

Some research into magic lens interfaces only looked into providing visual cues, thus not featuring any interaction with the virtual objects. For example, the work of Morrison et al. introduced a techniques in which a map is augmented with visual cues, providing users with information on how to find their way around the city[25]. The work of Henze and Boll provided a similar technique[13].

Research that did provide interaction often featured a crosshair, but generally didn't explore beyond selection. For example, the work of Rohs and Oulasvirta uses the crosshair to select a marker on a big screen[31]. The research performed by Henze and Boll explored the possibility of using a crosshair to create a virtual box around physical image by selecting two corner points[14]. The work of Liao et al. explored the use of a crosshair for initial selection of a 2D object, and touch gestures in order to fine-tune the selection[23]. The application that Rohs et al. developed allowed a user to select real-world building by pointing at them with the crosshair[32].

One of the few interfaces that does provide interaction with a crosshair beyond selection, is the work of Henrysson et al.. The interface they developed includes translation and rotation of virtual object in order to assemble a 3D AR environment[12]. The interface allows the user to target a virtual object and select it by pressing a button on the device. Object translation and rotation was achieved by the use of a menu and button interface, where the user was able to select the canonical manipulation they wanted to use. Our crosshair-based interface differs at this point, since our manipulations are all possible with touch interacions.

Chapter 3

Research

In order to test interactions in a 3D augmented reality environment, we need to have an application that provides us with the functionality that we require. Since such an application isn't commonly available, we built it ourselves. In this chapter, we motivate our choice of hardware, software and augmented reality toolkit and discuss their limitations. In subsection 3.2.4, we motivate which manipulations we require for our application, which answers our first research question. In subsection 3.2.5, we describe how we convert interactions on the device to manipulations in the virtual environment, which answers our second research question.

An important note we'd like to make is that this thesis was written as part of an internship at Avanade, an IT company who specializes in Microsoft technology. The internship had the purpose to investigate the potential of augmented reality on Windows Phone. While the result of this particular goal is uninteresting to this thesis, their involvement motivates some choices that were made in the development process.

3.1 Hardware

The handheld device used to develop the application is a HTC Radar smartphone, which was provided by Avanade and chosen for pragmatic reasons. It has a Qualcomm MSM8255 Snapdragon processing unit running at 1GHz, 512 MB RAM and a 5 megapixel camera[6]. The device is capable of recording video of 2560x1920 pixels at 30 fps. It has a 3.8 inch screen, which has 480x800 pixels. The capacitive touch screen is capable of detecting at least four touch locations.

3.2 Software

The platform for the mobile phone was an easy choice. Since this thesis was also part of an internship for a Microsoft specialist, the logical choice was to use a version of Windows

Mobile or Windows Phone. At the time of writing, we have a choice between several operating systems: Windows Mobile 6.5.3 (the latest version of Windows Mobile 6) and Windows Phone 7.5 (the latest version of Windows Phone 7).

Windows Mobile, which is the predecessor of Windows Phone 7, was not considered to be a valid option. This operating system received its last update in Februari 2010[39] and its respective Marketplace was discontinued in May 2012[30]. Since our work focuses on current and future technology, we found that using an operating system that was practically discontinued to be unopportunistic.

Windows Phone 7.5 proved to be the mature mobile operating system we were looking for. It features the necessary functionality required to run graphically advanced application, which augmented reality applications are. This functionality is backed up with an expansive documentation, as well as many examples of applications, both from the Microsoft Software Developer Network (MSDN) and other sources on the internet. As a result, Windows Phone 7.5 became our choice of the mobile operating system to use.

3.2.1 AR toolkit

We investigated the possibility to use an advanced AR toolkit, but found that nearly all were incompatible with WP7. Two recent AR toolkits for handheld devices that are gaining popularity are the metaio SDK[24] by metaio GmbH and Vuforia SDK[29] by Qualcomm, but neither has made a version for WP7. Computer vision software for .NET capable of AR are AForge[20] and Emgu CV[4] (a .NET wrapper for OpenCV[17]), however these .NET libraries are not compiled against the WP7 assembly. WP7 will only run applications developed with the Silverlight SDK for Windows Phone or XNA 4.0 Framework and there are only two AR toolkits that are compatible with these: Goblin XNA[27] and SLARToolkit[33]. At the time of development, Goblin XNA didn't support WP7.

Henceforth, our only option to have augmented reality on the WP7 is by using the SLARToolkit. This toolkit is a freely available AR library created by Schulte, based on NyARToolkit[26]. It can only perform marker-based tracking, unlike more advanced AR toolkits.

To use the SLARToolkit, we first had to acquire the image data. The WP7 SDK provides three formats to read the camera feed: RGBA, YCbCr and Y (luminance). Reading from the camera feed requires a significant amount of time, depending on which format was chosen to read. For the formats RGBA, YCbCr and Y, the timings were respectively $40ms$, $15ms$ and $10ms$. We chose the last method for two reasons. Since there isn't an abundance of processing power avaiable on our smartphone, the application should be as efficient as possible if the interaction with the virtual environment is to be responsive. An additional advantage is that the user's focus is attracted to the virtual environment, since its colorful appearance stands out against the greyscale physical environment.

Silverlight vs. XNA

The SLARToolkit was made to be used with Silverlight. The downside of the standard implementation is that it uses a Silverlight video element to display the camera's video feed, and fetches a separate video feed to register the virtual environment. The latter feed updates slower than the former, which results in a misregistration when the device is moved. Since accuracy is part of our test, we had to come up with another solution.

Displaying the separate video feed, so that the physical and virtual world actually align, turned out to be too slow for real-time augmented reality. In Silverlight, Bitmaps were used to capture the frames of the separate video feed, which SLARToolkit then uses for registration. However, displaying these bitmaps on the screen caused a significant delay per frame, which made real-time processing impossible.

We also attempted to use a combination of Silverlight and XNA, which is natively supported by WP7 by sharing the graphical device. With this combination, we used XNA's Textures to capture the video feed's frames and draw them on the screen, while the virtual environment existed of Silverlight objects. This solution caused the framerate to jitter; the screen would often freeze for a second every couple of seconds. We suspect that the XNA part wasn't given enough processing time by the threading system, causing this kind of unstable behaviour.

Our next attempt was to use a pure XNA application, which turned out to be the solution we were looking for. Apparently, Silverlight generates a lot of overhead, which was avoided by using a pure XNA solution. The use of XNA's Texture class allowed us to efficiently render video frames to the screen. A custom class was created to create objects and XNA's BasicEffect class was used to register these objects to the video frame. XNA also allowed us to specify that the processing and rendering loop must alternate, which resulted in a more responsive interactions. In comparison with Silverlight, the framerate was easily quadrupled. In other words, XNA provided us with the performance, stability and graphical capabilities that we required.

In hindsight, this solution seems like a rather obvious one. However, to our surprise, it turned out that we developed a practically novel technique to use SLARToolkit. During the time of developing, a search for specific keywords that connect SLARToolkit and XNA produced exactly one hit: the code file of a Google Code user who was hosting his personal project there. No documentation that relates SLARToolkit and XNA could be found.

Behind the scenes, SLARToolkit does still use Silverlight's Bitmaps to calculate the matrices needed for registration. We didn't modify the toolkit to support XNA's Textures, since that would have been a project on its own.

3.2.2 Registration

The SLARToolkit can only perform marker-based registration, which means it can only register virtual content to a physical world by detecting a *marker*. A marker is a black-white image with a thick, black border around it. The SLARToolkit is able to determine from which perspective the device is looking at the marker by determining the deformation of the square border. The image inside the border is used to determine the orientation of the marker.

There are various downsides to using a marker compared to markerless registration. Marker-based registration requires you to attach a marker to a surface, which isn't required for markerless registration. For example, markerless registration could use the side of a building to register virtual objects, which would be impossible to do with marker-based registration. Other downsides are that a marker must be printed and placed on a surface, while markerless registration could use the surface itself.

However, since SLARToolkit is only able to perform marker-based registration, we are required to use a marker. The marker we used and its application is displayed in Figure 3.1.

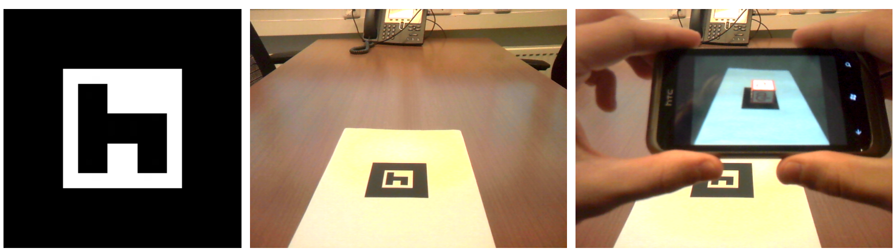


Figure 3.1: The marker used for registration, the marker printed out and placed on the table, and the registration with a virtual object.

3.2.3 Virtual environment

Ideally, we'd like to test our interaction concepts in a room-like setting, where the physical and virtual walls and ceiling are correctly registered. In this setting, the user would be able to construct an entire room; he would be able to interact with life-sized furniture that is placed on the ground, but also coathangers attached to the wall or a chandelier hanging from the ceiling. Such a setting would be interesting to test our interaction concepts in, but it is not feasible to produce due to our hardware and software limitation mentioned earlier.

Instead, we use a simplified version of this setting for our purposes. This setting is a small-scaled floor, properly size to fit on a part of a table, and doesn't feature virtual walls or a ceiling. In this environment, furniture must always be situated on a surface and correctly orientated. This means that we do not consider more advanced positioning (such as stacking or other alternative surfaces) and uncommon orientations (such as a chair on its side). Therefore, in our virtual environment, the virtual objects are always situated on the ground plane. Translation of a virtual object can only occur along this ground plane and rotation of a virtual object is only possible along the Z-axis.

3.2.4 Canonical manipulations

Canonical manipulations are the modifications a user applies to a virtual object in order to change its position, orientation or size in the virtual environment. There exist four canonical manipulations: *selection*, *translation*, *rotation* and *scaling*. We will motivate their use in the context of our virtual room, which contains furniture that the user can interact with to create an appealing room decoration.

Selection is an important manipulation to support. It allows the user to pick a single object from a group of objects, and apply other manipulations to it while leaving the other object untouched. Without selection, only one object could ever be present in the virtual environment. Since we wish to create a virtual room with multiple pieces of furniture in it, we require selection to be available. Therefore, selection is included in our interaction concepts.

Translation is the ability for a user to move an object from one location to another. This manipulation is integral to creating a virtual room with furniture. Without translation, the user would only be able to manipulate stationary furniture, which doesn't allow a user to make a decoration. Therefore, this manipulation is required, but it is not required to allow translation in all three dimensions. As mentioned earlier, we only consider furniture that is placed on the ground. Since all the interactable objects are on one surface, we don't have to consider translation upwards or downwards, reducing the interaction space to two-dimensional. Therefore, the translations that must be supported at least are the translations along the ground plane. This two-dimensional translation is included in our interaction concepts.

Rotation is the ability for a user to change the orientation of an object. Whilst this manipulation is not as integral to room decoration as translation, it is an important part of it. For example, without rotation, placing chairs at each side of a square table would result in the chairs all pointing in the same direction, instead of at the table. In order for the chairs to be correctly orientated, the user must be able to rotate the chairs. Therefore, rotation is required in order to make a room decoration, but similar to translation it is not necessary to support rotation in all three dimensions. A chair or a table doesn't make much sense if it is placed on its side, which means that we don't have to consider those rotations. Only one rotation makes sense in this context, which is a sideway rotation (also called *yaw*).

This rotation allows a user to correctly orientate the four chairs in the example given above. Effectively, this reduces the interaction space of rotation to one-dimensional, which is the least that we require. This one-dimensional rotation is included in our interaction concepts.

Scaling is the ability for a user to change the size of an object, making it larger or smaller. In our context, this manipulation doesn't make a lot of sense, since furniture is generally one size. While different sizes of a piece of furniture may exist, it is usually built according to a principle of appropriate size. For example, reducing the size of a comfortable chair by 25% may cause it to not be a comfortable chair any more. While scaling does have its place in room decoration, such as by making a dinner table twice as long, it is not required to feature it. For this reason, we choose not to include this manipulation in our interaction concepts.

In summary, three canonical manipulations are required to in order to create an appealing room decoration: selection, translation and rotation. Translation is only necessary along the ground plane, and for rotation we only require a sideway rotation. This answers our first research question.

3.2.5 Pointing system

In order to apply the above manipulations to objects in the virtual environment, we need to be able to convert touch screen interactions to manipulations in the virtual environment. To achieve this conversion, we use a method that we call the *pointing system*.

The pointing system is able to trace a certain position on the screen to a location in the virtual environment, depending on how the device is 'looking' at the virtual environment. The traced location is generally a location on a virtual object or on the ground plane, whichever is closer. The pointig system allows a user to 'touch' a virtual object, which in turn allow him to interact with it. This system is integral to allowing users to perform translation interactions, which we will explain in the next chapter.

In summary, the pointing system allows us to convert 2D interactions on the device to manipulations in the 3D virtual environment, since the third dimension (upwards, the Z-axis) is not taken into account. Translation is only necessary along the ground plane, and for rotation we only require a sideway rotation. This answers our second research question.

Chapter 4

Interaction concepts

As mentioned in the introduction, the touch-based and the crosshair-based interaction concepts are interesting for our purpose. For this experiment, we created an interface from both concepts. These interfaces are able to perform the manipulations that we specified in subsection 3.2.4.

4.1 Touch-based interface

The touch-based interface is designed to be similar to the touch interface that smartphone users are already used to. Touch interfaces for handheld devices across all operating systems are used in a similar way. Villamor et al. made touch interactions insightful in their Touch Gesture Reference Guide, which is an extensive guide on the use and support of touch gestures used for many platforms[41]. Based on this guide, we determined the interactions we want to use to apply the manipulations. We included two pages from this guide in Appendix A on which the relevant interactions are present.

4.1.1 Selection

For the *select* action, the guide only specifies a tap gesture, which is defined as ”briefly touch surface with fingertip”, as a way to select an object. This method does have a downside: when a user wants to perform a one manipulation, he is required to explicitly select the object beforehand and unselect afterwards. While it is possible for the user to perform these interactions, we reckon it is more convenient for the user if selection is done implicitly. This requires that the selection is incorporated in the other manipulations.

For example, a user wants to perform a translation to an object. In the former case, he is required to explicitly select the object, perform the translation, and then unselect the object again. In the latter case, he is only required to perform the translation, with the added requirement that the translation specifies which object is selected.

We consider the latter case to be more user friendly, since it removes the unnecessary interactions of selection and unselection, which can be specified implicitly. For this reason, we chose no gesture for our select action, and instead incorporate the implicit selection in translation and rotation.

4.1.2 Translation

The *move* action in the guide correlates to our translation manipulation. The guide provides four gestures to perform a move: drag-and-drop, multi-finger drag, flick and press-and-tap. The drag-and-drop gesture is a logical choice for translating an object, since it allows you to accurately grab and place the object. Translation using the multi-finger drag gesture requires an extra finger, while providing no additional benefit over the drag-and-drop gesture. Since the extra finger can only introduce inaccuracies in object placement, we consider this gesture suboptimal. The flick gesture requires the user to perform an erratic gesture. This gesture doesn't allow a user to place an object exactly where he wants it to be and is therefore unsuitable for our purposes. The press-and-tap gesture uses a second finger to denote the place of where the item should go. This works well in the case of predefined locations, but not in our case where an item can be place anywhere in a region.

Based on this analysis, we chose to use the drag-and-drop gesture for our translation action. The advantage of this gesture is that the implicit selection, which we require to be present, can be performed when the translation is initiated.

The user initiates translation by touching the object that he wants to move. This selects the object and makes the object follow the moving finger on the touch screen. To stop translation and selection, the user releases his finger from the touch screen.

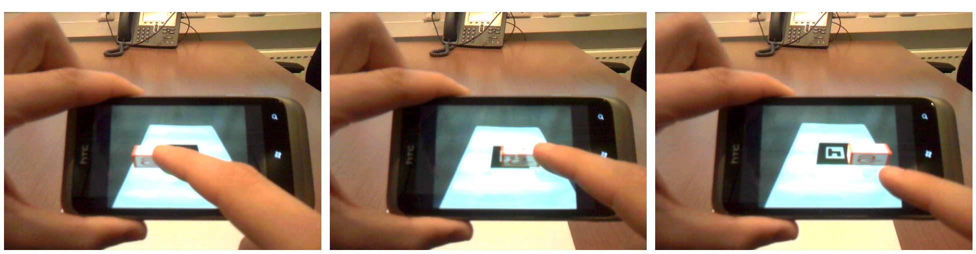


Figure 4.1: Translation for the touch-based interface.

Use of pointing system

The goal to implementing translation was to mimic translation in the real world, where a user picks up an object and drops it in a different location. This requires that translation of an object in the virtual environment also follows the interacting finger on the touch screen. In order to achieve this, we used the pointing system, which we described in subsection 3.2.5

to make the connection between the touch screen and the virtual environment. For example, the pointing system allows a user to 'grab' an object and drag it into the (virtual) distance, without the object moving away from the interacting finger. This behaviour is consistent with how a user would drag and drop an object in the real world, which meets our requirements.

An important feature of the pointing system is that a translation is also possible by moving the entire device. Moving the device can be considered a move of the virtual environment with respect to the device. The pointing system functions as the connection between the touch screen and the virtual environment. Logically, a change of state on one side causes a change of state on the other side, meaning that the moved virtual environment causes the object to move on the touch screen. However, we require that the object does not move away from the interacting finger. Therefore, the object must change its position in the virtual environment in order to fulfill its requirements.

In other words, the user has two ways of translating in object in this case. He can translate the object by touching the object and moving his finger across the touch screen, and by touching the object and move the whole device. Both methods complement each other, meaning the user can choose to apply both methods at the same time.

4.1.3 Rotation

The *rotation* action in the guide correlates to our rotation manipulation. The guide provides three (nameless) gestures to perform the rotation, of which all use two fingers. The first two gestures are essentially the same: the rotation is the angular difference between the line of both starting points and the line of both end points. The second gesture, where one finger remains stationary, is a substate of the first gesture. For this reason, we consider these gestures to represent a single 'separate fingers'-gesture. This gesture is useful for our purposes, since one finger will already be used for selection and translation, as explained earlier. This gesture only requires the addition of a second finger in order to apply the rotation action. The last gesture uses two fingers kept next to each other to perform a rotation. This gesture isn't very useful for our purposes, since it lacks a selection context similar to our chosen translation gesture and the former rotation gesture.

For this reason, we chose to use the separate fingers gesture for our rotation action. The advantage of this gesture is that it can be performed while the object is being translated and/or implicitly selected.

In order to initiate rotation, the user only has to move a second finger over the touch screen while he is selecting or translating the object. Rotation stops when the user removes his second finger from the touch screen.

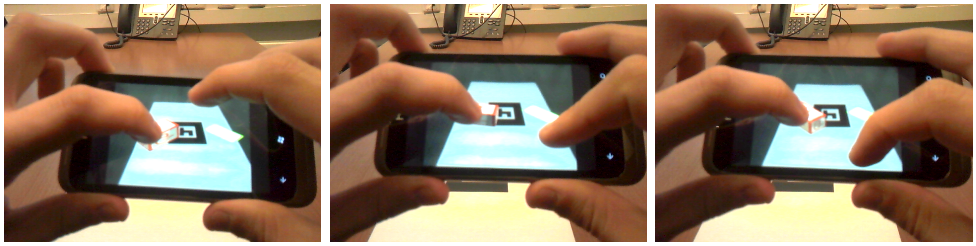


Figure 4.2: Rotation for the touch-based interface.

Use of pointing system

The rotation is applied by a circular movement of the second finger around the first finger that is used for selection and/or interaction. Due to the latter, the object is already subject to pointing system. Therefore, if the second finger performs a drag on the touch screen, the angle between the start and end point on the touch screen is the same as the start and end point in the virtual world. In other words, we don't need to explicitly use the pointing system for rotation, since we already use it implicitly. By calculating the angle on the touch screen only gives us a better performance than in the case where we have to convert the touch screen positions to locations in the virtual environment. For this reason, we chose to apply rotation using the positions on touch screen only.

4.2 Crosshair-based interface

The crosshair-based interface is designed to take advantage of the crosshair present in the center of the screen. The crosshair is used to aim it at an object and select it, which causes the object to 'stick' to the crosshair. While selected, the user is able to interact with the object. Compared to the touch-based interface, the crosshair can be seen as a stationary finger in the center of the screen, of which the user can control the pressing and releasing. The most important difference with the touch-based interface is that translation is done purely by moving the device.

4.2.1 Selection

Selection for the crosshair-based interface is different than for the touch-based interface. Compared to the touch-based interface, the crosshair acts as a stationary finger. The user is no longer required to specify a location on the screen in his interaction; he only has to specify whether he wants to select the object he is aiming at. This makes this version of selection one-dimensional, in contrast to the touch-based interface, which has two-dimensional selection. This means that select action can be done by a simple button.

For the select action, we initially intended to use the camera button present on the right top on the device when held sideways. Unfortunately, the use of this button was deemed infeasible. The camera button on the device has two active states: half-pressed and full-pressed. The half-pressed state of the button invokes a delay of 0.5 seconds. This delay makes it more difficult for the user to accurately select an object. The full-pressed state of the button doesn't provide a better alternative. It requires that the user presses through the half-pressed state, which we found to be overly physically demanding and unintuitive for a frequently occuring interaction such as selection. For these reasons, we chose not to use the camera button as our select action.

Since the use of other buttons on the device is restricted by the operating system, our only other option for interaction is to use the touch screen. One-dimensional interaction be possible by a touch interaction anywhere on the touch screen. This method provides us with two options, which we reviewed in the previous section for the touch-based interface: using a tap gesture as suggested by the Touch Gesture Reference Guide[41], or incorporate the select action into the other interactions. We found the latter method to be unopportunistic for our purposes. Incorporation of the select action would mean that selection and rotation would be applied by the same touch interaction. For illustration, the user would use one finger to select an object by aiming at it and placing his finger on the touch screen, rotate the object by moving the finger across the touch screen, and unselect the object by releasing his finger. In other words, it is not possible for the user to initiate selection without also initiating rotation, nor to cancel rotation without also cancelling selection. Since these are separate manipulations, the user should be able to perform them separately, and not be forced to use both whenever he wishes to apply only one. For this reason, we decided to use the tap gesture for selection.

The tap gesture allows the user to explicitly select the object at which he is pointing the crosshair. A tap at any location on the touch screen will initiate selection. The same interaction will also cancel the selection.

4.2.2 Translation

Translation for the crosshair-based interface is different compared to the touch-based interface. Where the touch-based interface provides the ability to translate using touch and/or the pointing system, the crosshair-based variant can only translate using the pointing system.

No action is necessary from the user in order to initiate translation, other than having to select an object. Upon selection, the object follows the crosshair, similar to how it follows a finger in the touch-based interface, as explained in Figure 4.1.2. To cancel translation, the user must unselect the object.

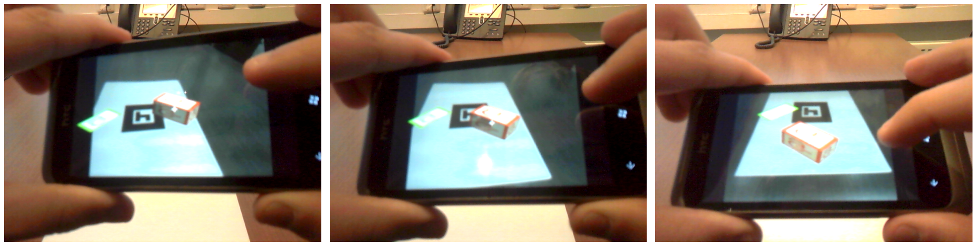


Figure 4.3: Translation for the crosshair-based interface.

4.2.3 Rotation

Rotation is applied in the same way as it is applied in the touch-based interface, as explained in subsection 4.1.3. The only difference is that the user doesn't use a finger for selection, but instead uses the crosshair. This means that the user only needs to use one finger to perform rotation, in contrast to two fingers using the touch-based interface.

Rotation is initiated when the user places a finger on the touch screen and performs a drag while an object is selected. Rotation is cancelled when the user removes his finger from the touch screen.

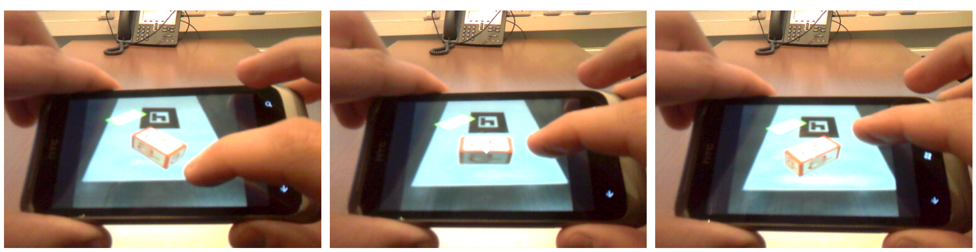


Figure 4.4: Rotation for the crosshair-based interface.

Chapter 5

Experiment: comparative user study

In this experiment we compare the two interfaces we described in chapter 4 in an augmented reality test environment in order to determine which one performs better at performance and usability. In section 5.1, we describe the experiment setup. In section 5.2, we describe our expectations. The results of the experiment are presented in the next chapter.

5.1 Experiment setup

The experiment is a user study, which was set up as a within subject approach, meaning that every subject tested both interfaces. The experiment starts with a demo scene, used to explain to the subject what AR is and to allow him to get acquainted with it. After this introductory scene, the subject had to perform a sequence of actions for each interface, which we call the interface routine.

5.1.1 Interface routine

The routine exists of three parts. In the training session, the subject is familiarized with the interface by completing objectives in a level. In the test session, the subject completes the same objectives in the same level, but now aims to complete as many objectives as possible within a certain time frame. In the questionnaire, the subject is asked to rate his task load at different scales and is asked for his opinion regarding the interface.

In order to eliminate bias and learning effect, the order in which the subjects performed the routine was counterbalanced between subjects.

Training session

The training session was meant to allow the subject to become familiar with the interface and comfortable using it. Before the training started, the subject was informed about how the interface works, after which the subject was given the time to practise. This session didn't have a time limit and the subject was encouraged to take all the time he needed to get used to the interface. This course of action was chosen to make sure that the subject fully understood the concept behind the interface, which in turn increases the reliability of the results gathered during the test session. When the subject felt he was comfortable enough with controlling the interface, the test session was loaded.

Test session

For this test, we chose to apply a time limit of two minutes. The choice for a time limit and its duration was a result from an informal user study. We noticed that the amount of completed objectives could heavily fluctuate between subjects, which indicates that a limit on the amount of objectives to complete is a suboptimal choice. For this reason we chose to use a time limit instead. The time frame of two minutes was determined to be optimal for our purpose. Time frames longer than this amount showed that subjects started to lose attenion, which could negatively impact our results. Time frames shorter than this amount are more prone for subjects not completing a single objective, which would also negatively impact our results. Henceforth, the time frame of two minutes was chosen to be optimal our setup.

The subject performs the same actions during the test session as he did during the training session. The same objectives must be completed in the same level. The order in which the objectives appear is randomized, in order to prevent learnability.

During the test, the application recorded important data to the device's internal storage. The data includes:

Score The amount of objectives completed.

Device movement How much the device was physically moved relative to the marker, including angular movement.

In addition, the subject's behaviour was observed.

After the test was finished, the subject was asked to fill out a questionnaire.

Questionnaire

The questionnaire aims to determine the task load experienced by the subject. It is based on the NASA task load index[7, 8], which we included in Appendix B. Its ability in providing researchers with reliable task load data has been confirmed by the work of Hart[9], and it has been commonly used in literature whenever researchers wish to determine the task load for a task. However, we aren't interested in the task load index itself, since we are not dealing with well refined interfaces. For less refined interfaces, like the ones we are using, the task load scales are much more informative. The task load scales we are interested in, are: mental demand, physical demand, performance, effort and frustration. We chose these scales because they give us information regarding the mental and physical state of the subject. We left out temporal demand, since we deemed this factor irrelevant due to the time constraint that is present in the test session.

In addition to the task load scales, the subjects were about about their age, their personal interface preference, the reasons for that preference and other remarks concerning the task.

5.1.2 Level setup

We created a level with the virtual environment, designed to consistently test the three manipulations that the interfaces feature: selection, translation and rotation. The level consists of one interactable object (the white/orange rectangular box) and three goals (the green rectangles), as shown in Figure 5.1.

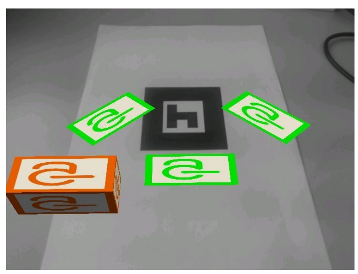


Figure 5.1: The level, showing all goals.

The objective is to correctly place the object on a goal. Of the three goals, only one goal is ever visible to the subject, which is his current objective. When the object is correctly placed on the goal, the objective is completed and the subject's score is incremented. The goal then disappears and the next goal, which is chosen randomly from the pool of hidden goals, appears and becomes the new objective. This routine is visualized in Figure 5.2.

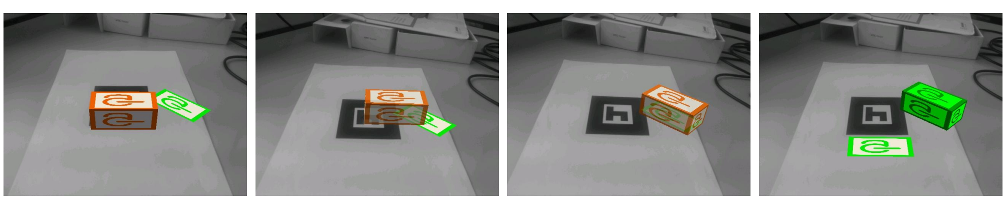


Figure 5.2: Object placement on goal.

The three goals are all placed at equal distance from the center of the level and from each other, meaning that translation will always require the same distance to be covered. The two top goals are respectively -45 and 45 degrees rotated. We acknowledge that these values should have been respectively -60 and 60 degrees in order for all rotations to consistently be 60 degrees. However, since the goals are randomly chosen, the average rotation will still be 60 degrees.

The subject is assisted with placing the object by allowing him to place it within a certain margin of the goal. The values for the margin are chosen to require the subject to be precise with placement, without requiring an overly high amount of accuracy. Further assistance is provided by turning the object translucent when it is selected, improving the subject's ability to align the object and the goal. When it is correctly placed, the object turns green as visual feedback, which lasts until the next time it is selected.

5.2 Expectations

The touch-based and crosshair-based interfaces differ from each other at a few points. We motivated our choices for the differences in chapter 4. Based on these differences, we can draw expectations with regards to the performance, and the task load scales described in subsection 5.1.1.

Performance A major difference between the touch-based and crosshair-based interface is that the interactions of the latter are all performed with one finger. The touch-based interface requires rotation to be performed with two fingers, which takes a bit more time than doing it with one finger. It also doesn't easily allow translation and rotation to be done at the same time, which is easier in the case of the crosshair-based interface. For this reason, we expect that the crosshair-based interface will outperform the touch-based interface.

Mental load Mental load is the amount of attention a task requires for the subject to perform it. The touch-based interface provides two ways to manipulate virtual objects: using touch interactions that are commonly featured on touch screen devices, and by using the pointing system. With this interface, the subject can choose their preferred method, or even combine them. The crosshair-based interfaces only provides the latter method for object manipulation. In the case that a user finds that manipulation using purely the pointing system to be difficult, he doesn't have the choice that the touch-based interface provides. For this reason, we expect the touch-based interface to result in a lower mental load.

Physical load Physical load is the difficulty that the subject physically experiences when performing the task. The most difficult interaction that a subject will perform is rotation in the case of the touch-based interface. This interaction is the only one that requires two fingers to be used at the same time, which is a major difference with all the other interactions that only require one finger. The crosshair-based interaction doesn't contain a two-fingered interaction, which means it is easier to physically perform. For this reason, we expect the crosshair-based interface to results in a lower physical load.

Effort and frustration We join these two task load scales together here, since they both determine how difficult it is for a subject to control an interface. We are unsure which interface the users find to require more effort or spark more frustration. We suspect that the touch-based interface will require more effort when performing rotation than the crosshair-based interface. On the other hand, the crosshair-based interface may prove more difficult to use, possibly overshadowing the disadvantage of the touch-based interface. Depending on what the subject finds to be more demanding or irritating, the scales may tip in the direction of either interface.

Chapter 6

Results

The results described below are from the within-group user study described in section 5.1. A total of twenty-six subjects, of which twenty-three males and three females, participated in this experiment. The duration of one experiment was around thirty minutes, which includes the introduction, the two interface routines and the questionnaire. The ages of the subjects ranged from 20 to 45, of which ten at ages 20 — 24, nine at ages 25 — 29, four at ages 30 — 35 and three at ages 40 — 45. The visual representation of the ages can be found in Appendix C, Figure C.1. All subjects were employees or interns at Avanade at the time of the experiment. All twenty-six subjects had an affection with technology and had experience with using a smartphone. We limited the number of subjects to twenty-six for pragmatic reasons.

The experiment was performed within the controlled environment of multiple identical, $9m^2$ sized, closed rooms. All rooms featured a table, at which the subjects were seated at identical locations. The AR marker used for registration was placed on the table on identical locations as well. Light conditions within the room could be regulated to allow for optimal marker recognition.

We analyze the results of the quantitative data in section 6.1. In section 6.2, we evaluate the qualitative data that we've collected. In section 6.4, we discuss the results with respect to our research questions.

6.1 Performance and usability analysis

We gathered quantitative data from our device and questionnaire and formulate them using the following fifteen variables:

1. Gender
2. Age
3. Interface preference
4. Score (Touch)

5. Mental demand (Touch)
6. Physical demand (Touch)
7. Performance (Touch)
8. Effort (Touch)
9. Frustration (Touch)
10. Score (Crosshair)
11. Mental demand (Crosshair)
12. Physical demand (Crosshair)
13. Performance (Crosshair)
14. Effort (Crosshair)
15. Frustration (Crosshair)

As mentioned in section 1.1, our goal is to determine whether the touch-based or crosshair-based interface by determining which one performs better with regards to performance and usability. We determine this by taking the performance and usability variables that are specific to the touch-based interface and compare them to their crosshair-based interface counterpart. However, before we can draw conclusions from this comparison, we need to determine if their values are significant. If a value is not significant, it can be the result of chance, which means that the variable can't be used as proof to support either interfaces.

Of the fifteen variables, the last twelve form six variable pairs, which we call *paired samples*. In order to determine significance between paired samples, we need to determine whether we need to use a parametric or non-parametric test. The parametric significance test for paired samples, which is the paired Student's t-test[37], assumes that the entered data follows a normal distribution. The non-parametric alternative to this test is the Wilcoxon signed-rank test[44], which doesn't assume a normal distribution. To determine whether a variable follows a normal distribution, we performed a normality test.

This and all the other statistical tests present in this thesis have been performed with SPSS, version 19. We use the conventional significance levels of 0.1, 0.05 and 0.01 (which respectively stand for 10%, 5%, 1%) to determine statistical significance.

The results of the normality test can be seen in Figure 6.1. In this diagram, we can see two major columns, which stand for two seperate tests that have been run: the Kolmogorov-Smirnov test and the Shapiro-Wilk test. The former test is appropriate for larger data sets and the latter test is appropriate for smaller data sets. Since our sample size of twenty-six subjects fits a small data set, we use the significance values determined by Shapiro-Wilk's test. The variables in our diagram that are statistically significicant are shaded in blue. The pairs for Score and Frustration are the only pairs of which both variables are normally distributed, meaning we can use the t-test to determine their significance. For the other pairs, we use the Wilcoxon signed-rank test instead.

We ran the appropriate test for the variable pairs, which produced two diagrams; one for the t-test, and one for the Wilcoxon signed-rank test. We made the relevant information

Tests of Normality						
	Kolmogorov-Smirnov			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
Gender	,523	26	,000	,376	26	**,000**
Age	,207	26	,006	,828	26	**,001**
InterfacePreference	,376	26	,000	,630	26	**,000**
ScoreTouch	,097	26	,200	,974	26	,718
ScoreCrosshair	,089	26	,200	,962	26	,439
MentalDemandTouch	,148	26	,147	,948	26	,207
PhysicalDemandTouch	,198	26	,010	,930	26	**,078**
PerformanceTouch	,240	26	,001	,850	26	**,001**
EffortTouch	,145	26	,167	,952	26	,257
FrustrationTouch	,110	26	,200	,947	26	,198
MentalDemandCrosshair	,180	26	,029	,873	26	**,004**
PhysicalDemandCrosshair	,161	26	,081	,902	26	**,017**
PerformanceCrosshair	,166	26	,063	,927	26	,068
EffortCrosshair	,195	26	,012	,897	26	**,013**
FrustrationCrosshair	,149	26	,141	,942	26	,151

Figure 6.1: Normality test. Shaded variables are significant for $p < 0.5$, indicating non-normality.

Paired significance tests		
	T-test Sig. (2-tailed)	Wilcoxon Asymp. Sig. (2-tailed)
Score Touch - Score Crosshair	0,031	
MentalDemandTouch - MentalDemandCrosshair		0,020
PhysicalDemandTouch - PhysicalDemandCrosshair		0,485
PerformanceTouch - PerformanceCrosshair		0,218
EffortTouch - EffortCrosshair		0,063
FrustrationTouch - FrustrationCrosshair	0,079	

Figure 6.2: T-test and Wilcoxon signed-rank test. Significant variables are shaded; the darker shade indicates a significance level of 5%, the light shade a level of 10%.

insightful by merging them into a single diagram, shown in Figure 6.2. The original diagrams can be found in in Appendix C, Figure C.2 and Figure C.3.

Using the above results, we analyze the performance in subsection 6.1.1 and the usability in subsection 6.1.2. In order to interpret these results better, we calculate the correlation between all of the above variables. This allows us to determine if there are any unexpected relationships between variables that have an impact on the results that we collected. We perform the correlation test in subsection 6.1.3.

6.1.1 Performance

Which interface performs better in terms of performance was the third research question we defined. In order to analyze performance, we solely look at the score that the subjects were able to get during their tests. We don't look at the performance variable acquired from the questionnaire. Since that variable is a measure of how the subjects judged their own performance, it is not an objective measure on how the subjects actually performed at using the interface.

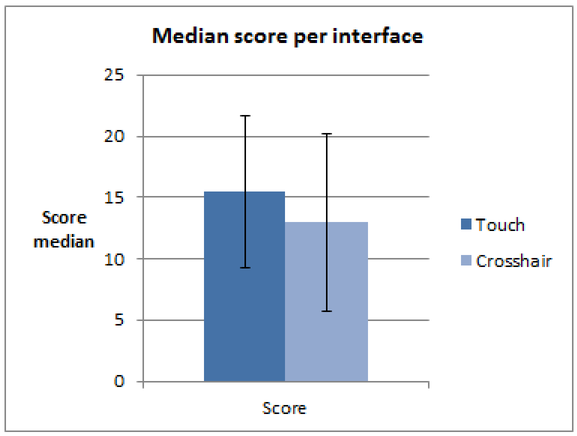


Figure 6.3: Median score per interface.

We illustrated the score results in Figure 6.3. In Figure 6.2, we see that the score results are significant ($p = 0.031$). Therefore, we can conclude that subjects perform significantly better with the touch-based interface than with the crosshair-based interface. This goes against our expectation that the crosshair-based interface should perform better, since it should be more efficient in use. In summary, we can conclude that the answer to our third research question is that the touch-based interface performs better.

6.1.2 Usability

The fourth research question we defined, was which interface performs better in terms of usability. We analyze usability by analyzing the results of the five task load scales that we gathered from the questionnaire. In Figure 6.2, we can see that physical demand ($p = 0.485$) and performance ($p = 0.218$) don't significantly differ between each interface. For this reason, we will not review their respective results, since they could have occured by chance. In contrast, the task load scales mental demand ($p = 0.020$), effort ($p = 0.063$) and frustration ($p = 0.079$) are statistically significant. We illustrated the results of these three task load scales in Figure 6.4.

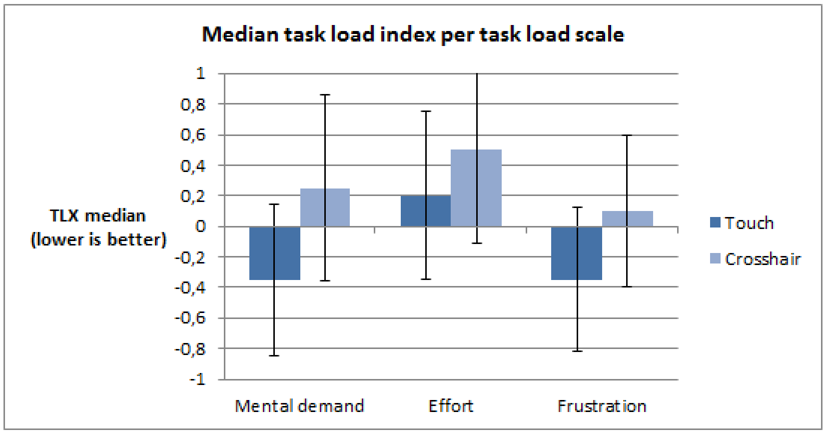


Figure 6.4: Median task load per interface. A lower task load is better.

From Figure 6.4, it is proven that the subjects find the touch-based interface to be significantly easier to use than the crosshair-based interface. We can conclude that the mental demand required to use the touch-based interface is significantly lower than the touch-based interface. This confirms our expectation expressed in section 5.2. The same counts for effort and frustration. We can conclude that users find it requires more effort to control the crosshair-based interface, and also experience more frustration using it. As such, we can conclude that, of all task load scales of which the difference between interfaces was deemed significant, the touch-based interface comes out on top, which answers our fourth research question.

6.1.3 Correlation of variables

In order to determine the relationships among the variables, the correlation between them must be calculated. The standard method to calculate correlation between two variables is by calculating the Pearson product-moment correlation coefficient[36]. However, this calculation is based on the values of the variables, which makes it susceptible to outliers and skewed data. Since this is the case for our data, we are required to use the non-parametric version of Pearson's corralation method, which is *Spearman's rank correlation*[35]. We used this method to calculate a corralation matrix, in which the corralation between all variables is calculated, as well as the corralation's significance.

The correlation matrix is shown in Figure 6.5. The method used to calculate the correlation between variables depends on the normality of both variables. If the variables are both normally distributed, then the Pearson method is used to calculate the correlation. In all other cases, the Spearman's rank method is used instead. The original correlation matrices for Pearson and Spearman's rank can be found in Appendix C, respectively Figure C.4 and

Merged correlation matrix																	
Pearson and Spearman rank correlation			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	Gender	Correlation Coefficient	1,000														
		Sig. (2-tailed)	.														
2	Age	Correlation Coefficient	-,121	1,000													
		Sig. (2-tailed)	,556	.													
3	Interface Preference	Correlation Coefficient	-,066	-,261	1,000												
		Sig. (2-tailed)	,750	,198	.												
4	Score Touch	Correlation Coefficient	-,442	-,609	,151	1,000											
		Sig. (2-tailed)	,024	,001	,462												
5	Score Crosshair	Correlation Coefficient	-,297	-,611	,473	,821	1,000										
		Sig. (2-tailed)	,140	,001	,015	,000											
6	Mental Demand Touch	Correlation Coefficient	,402	-,248	,323	-,073	,086	1,000									
		Sig. (2-tailed)	,042	,222	,108	,723	,674										
7	Physical Demand Touch	Correlation Coefficient	,016	-,144	,329	-,214	,020	,203	1,000								
		Sig. (2-tailed)	,938	,483	,101	,294	,922	,321	.								
8	Performance Touch	Correlation Coefficient	,178	,151	,288	-,415	-,296	,121	,184	1,000							
		Sig. (2-tailed)	,385	,462	,154	,035	,142	,555	,368	.							
9	Effort Touch	Correlation Coefficient	,145	-,253	,354	-,137	,151	,453	,507	,265	1,000						
		Sig. (2-tailed)	,480	,213	,076	,504	,463	,020	,008	,190							
10	Frustration Touch	Correlation Coefficient	,105	,046	,369	-,372	-,167	,228	,361	,440	,495	1,000					
		Sig. (2-tailed)	,611	,823	,063	,061	,414	,263	,070	,025	,010						
11	Mental Demand Crosshair	Correlation Coefficient	,347	,258	-,141	-,561	-,388	,477	,342	,198	,314	,262	1,000				
		Sig. (2-tailed)	,083	,203	,493	,003	,050	,014	,087	,333	,119	,195	.				
12	Physical Demand Crosshair	Correlation Coefficient	,016	-,123	-,156	-,181	-,020	-,023	,643	-,142	,195	-,126	,369	1,000			
		Sig. (2-tailed)	,938	,550	,445	,377	,922	,912	,000	,490	,341	,539	,063	.			
13	Performance Crosshair	Correlation Coefficient	,331	,328	-,323	-,462	-,614	-,041	-,350	,373	-,026	,087	,065	-,315	1,000		
		Sig. (2-tailed)	,099	,102	,107	,017	,001	,841	,080	,061	,900	,671	,753	,117	.		
14	Effort Crosshair	Correlation Coefficient	,226	-,177	-,073	-,178	-,163	,077	,365	,084	,542	,157	,369	,431	,134	1,000	
		Sig. (2-tailed)	,267	,386	,723	,384	,427	,708	,067	,684	,004	,443	,063	,028	,515	.	
15	Frustration Crosshair	Correlation Coefficient	,169	,048	-,552	-,130	-,269	-,296	-,080	,032	,151	,269	,175	,044	,478	,318	1,000
		Sig. (2-tailed)	,409	,817	,003	,527	,184	,143	,696	,878	,463	,184	,392	,830	,014	,113	

Figure 6.5: Merged correlation matrix. Significant variables are shaded orange; the darker shade indicates a significance level of 1%, the light shade a level of 5%.

Figure C.5.

In this correlation matrix, there are a total of twenty correlations that are significant for $p < 0.05$, of which nine are significant for $p < 0.01$. All of the related correlation values are above $Abs(c) > 0.4$, which indicates at least moderate correlation. One case is above $Abs(c) > 0.8$, which indicates a strong correlation. We didn't consider correlations that are only significant for $p < 0.1$, since all of their correlation values are lower than the ones we've already highlighted. In other words, the relationship between those variables is weaker than the ones already highlighted, which makes their presence less important. We also won't consider the correlations of a variable with itself, which are logically perfect correlations. These values form the diagonal of ones in the correlations matrix.

We explain the relationship between correlated variables per variable below:

1. **Gender** Gender correlates moderately negative to Score Touch. From this, we can conclude that female score significantly lower using the touch-based interface than the males. Gender also correlates moderately positive to Mental Demand Touch. This means that we can conclude that males need to pay a significant lesser amount of attention to use the touch-based interface as opposed to females. Apparently, males have a significantly easier time using the touch-based interface compared to females. While we can't conclude that the mental demand is the reason for this (correlation does not prove causation), it does seem to point in that direction.

2. **Age** Age correlates moderately negative to both Score Touch and Score Crosshair. We can conclude that older users score worse on both interfaces in general, which is conform to the common notion that a user's performance goes down as he grows older.

3. **Interface Preference** Interface preference correlates positively to Score Crosshair. From this, we can conclude that a good score for the crosshair-based interface relates to a preference for that interface, which is a logical conclusion. Interface preference also correlates negatively to Frustration Crosshair. From this, we can conclude that a high frustration for the crosshair interface relates to a preference for the touch-based interface. Based on these two correlations, we can conclude that a both variables have a significant impact on the user's interface preference, where Frustration Crosshair is the major contributor due to its high significance ($p = 0.003$).

4. **Score Touch** Score Touch has a strong positive correlation with Score Crosshair. This correlation makes sense, since there are both scores and are therefore expected to be correlated. Score Touch correlates moderately with five other variables. Gender and Mental Demand Crosshair are correlated negatively with a significance of $p < 0.01$ to this variable, and Age, Performance Touch and Performance Crosshair are correlated negatively with a significance of $p < 0,05$. From these correlations, we can conclude that older users score significantly worse at the touch-based interface, and females to a lesser extend. We suspect that users rate their own performance lower when they score lower with the touch-based interface. The mental demand and performance for the crosshair-based interface are also correlated with the score for the touch-based interface. From this, we can conclude that users that perform worse in these aspects of the crosshair-based interface score significantly higher with the touch-based interface. We can't explain why this is the case, but it is evident that it is the case.

5. **Score Crosshair** As mentioned earlier, this variable has a strong positive correlation with Score Touch, for the reason that they are both scores. Age and Performance

Crosshair both correlate moderately negative to the score of the crosshair-based interface. From these correlations, we can conclude that older users and users that rate their own performance worse to score significantly worse at the crosshair-based interface. The correlation of this variable to Interface Preference has been described in the paragraph of the latter, where it is more appropriate.

Variables 6 — 15 We chose not to review the correlations of task load scales among each other. We are mainly interested in how the first five variables influence and are influenced by the task load scales. While there are some interesting relationships among the task load scales, they don't aid us in explaining why users reached a certain score or chose their preferred interface. Therefore, we decided not to include this information due its lack of relevance.

6.2 Qualitative data

In addition to the quantitative data we've collected, we also gathered feedback from the subjects and made observations during the tests. Here we review the feedback and observations with respect to the task load scales that we tested in section 6.3. In addition, we made two observations that weren't reflected in the feedback. We review these in subsection 6.3.4.

6.3 Feedback

The user's feedback was gathered using the questionnaire. The subjects motivated their interface preference with various reasons, which we categorize here according to the task load scales. We also review our own observations with respect to the feedback.

6.3.1 Mental demand

Observations showed that some subjects had trouble understanding that the object would follow the position where the crosshair pointed on the ground plane. These subjects had great difficulty properly positioning the object on the goal, and later expressed that they found the interface to be counterintuitive to them. Nine of the twenty-six subjects stated that the touch-based interface felt more intuitive, natural, logical or comfortable to use, opposed to none for the crosshair-based interface. Two subjects specifically expressed that they find that the crosshair-based interface requires you to also consider the crosshair or 'yourself' when manipulating an object, which puts an extra burden on the interaction.

6.3.2 Physical demand

While there was no significant difference between the touch-based and crosshair-based interface with regards to physical demand, we did make observations that indicate that subjects found the touch-based interface physically more difficult to use. Some subjects experienced difficultied with performing the touch-based interactions; they had a hard time selecting the object or performing the two-fingered rotation. Interestingly, ten out of eleven subjects who prefer the crosshair-based interface motivate their choice with finger related arguments. Some subjects expressed this was due to their bigger fingers, while other subjects found that too little of the screen is visible when interacting with two fingers in the touch-based interface.

6.3.3 Effort and frustration

Effort and frustration was proven to be significantly higher for the crosshair-based interface. This was especially noticable for some subjects, who didn't have steady hands. If a subject has steady hands, the crosshair-based interface can allow for more precise object placement than the touch-based interface. However, the subjects who didn't have steady hands suffered from the inability to compensate for the erratic hand movement, and as such had trouble placing objects correctly. The touch-based interaction is more robust against non-steady hands when the hand holding the device and the hand interacting with the objects are the same one, or when they are held together.

An interesting note to place here is that some users like the fact that the crosshair-based interface required more effort. They viewed the experiment as a game and found that the interface was more challenging than the touch-based interface.

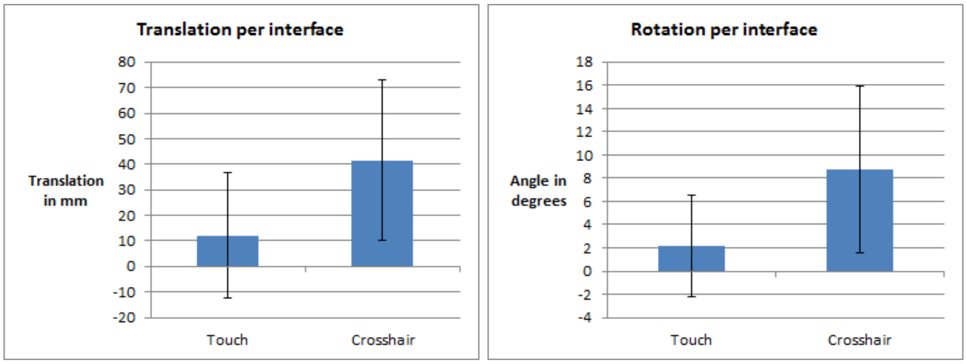
6.3.4 Additional observations

In this section, we review two observations that we consistently observed during tests among many subjects. The first is that subjects seem to prefer to use the device in a stationary position. The second is that subjects have the tendency to interact near or at the object's location. These events were observed often enough that we believe they have a place in this thesis.

Subjects prefer a stationary device

One important observation we made is that subjects prefer to hold the device stationary when performing interactions with the touch-based interface. With the crosshair-based interface, subjects were forced to physically move the whole device in order to translate objects. With the touch-based interface, it was observed that the subjects moved the device less and often keeping it quite stationary.

We were able to find evidence of this observation in the data we recorded on the device. For each interface, we measured how much the device was moved and turned during the test. We measured the amount of translation in millimeters and the amount of rotation in angles.



(a) Translation per interface in millimeters. (b) Rotation per interface in degrees.

Figure 6.6: Device movement per interface.

We used completed objectives as marking points for the measurements in order to remove noise, potentially caused by inaccurate registration and unsteady hands. The results are illustrated in Figure 6.6a and Figure 6.6b, which supports our observation.

From these graphs, it is clear that subjects prefer to hold their device stationary when it is not required to move it. For both translation and rotation, the amount the device is moved and turned is four times higher for the crosshair-based interface than it is for the touch-based interface. We suspect the reason for this behaviour is that subjects find the unnecessary change in viewpoint bothersome. Another reason could be that it costs more energy to move a whole device back and forth than it is for a finger to be moved across a surface.

Subjects interact at or near the object

Another important observation we made was that *subjects have the tendency to perform interactions at or near the object* they're interacting with, even when it is not necessary. In the case of the crosshair-based interface, some subjects performed interactions at or near the crosshair. In the case of the touch-based interface, some subjects performed rotations by holding both fingers very closely together, as if they were trying to hold the object. In both cases, it caused the subjects to block their view when perform interactions, which is suboptimal. This happened even though the subjects were informed beforehand on how the interface could be used in an optimal manner, which was part of the explanation of the interfaces. We suspect the reason for this behaviour is that is the subjects didn't have a visual cue (such as a bent arrow for rotation) to perform the above-mentioned interaction. Instead, their visual cue is the object itself, which causes the suboptimal interactions.

We didn't look for evidence of this observation in the data we had at our disposal. The reason for this is that it isn't possible to accurately quantify how the optimal interaction should be performed, which makes it impossible to see how often this situation actually occured.

6.4 Discussion

In this experiment, we investigated the touch-based and crosshair-based interfaces and compared them to each other. We can conclude that the touch-based interface trumphs the crosshair-based interface in both performance and usability. We reckon that this interface can be used in the future scenario that we described earlier in the thesis, when the computational power allows for such environments to registered to the world.

When we look at all the data, the feedback and the observations, we can conclude the following:

The touch-based interface performs better than the crosshair-based interface. With the touch-based interface, users are able to perform a task quicker and easier. The interface is relatively simple and it requires little effort to use it, even though it isn't the most efficient way. Since the interface is one that the users are already familiar with, they are easily used to it and they aren't easily frustrated with it. The touch-based interface also allows them to use the device as a window into the augmented reality environment, without being forced to be a part of it when they want to interact with it. However, in order for this interface to be accessible to a large audience, the fat finger problem must be addressed. We expect the usability of this interface to rise significantly when an appropriate solution to this problem is applied.

The crosshair-based interface lacks the performance and usability that the touch-based interface has. A possible benefit that this interface has, is that it could be applied to a game, where the use of the interface is part of the challenge. However, for an interface that is accessible to a large audience, this one is not suitable.

Chapter 7

Conclusion and future work

7.1 Conclusion

In this thesis we have investigated the potential of a touch-based and a crosshair-based interfaces to be used for object manipulation in a 3D augmented reality environment on the mobile platform. The main goal of this thesis was to determine which interfaces is more appropriate for an application with such specifications. At the start of this thesis, in section 1.1, we defined four research questions that needed answering in order to reach the main goal of this thesis. These research questions are:

1. Which canonical manipulations must be supported?
2. How can 2D interactions on the device be converted to manipulations in a 3D virtual environment?
3. How does the performance of the touch-based and crosshair-based interaction concepts compare?
4. How does the usability of the touch-based and crosshair-based interaction concepts compare?

Research question 1 was answered in subsection 3.2.4, where we determined that we selection, translation and rotation are required for our purpose. Research question 2 was answered in subsection 3.2.5, where we explained how our pointing system can be used to facilitate the conversion from interaction to manipulation. Research question 3 was answered in section 6.1, where we determined that the touch-based interface outperforms the crosshair-based interface in terms of performance. Research question 4 was answered in section 6.2, where we determined that the touch-based interface outperforms the crosshair-based interface in terms of usability.

With these research questions answered, we can conclude that the main goal of our thesis has been reached: the touch-based interface is best capable of performing canonical manipulations in 3D space for handheld augmented reality.

In the experiment that we conducted, we performed a comparative user study in order to determine which interface would perform best in an AR environment, as described in a future scenario we used as a guideline. From this experiment, we can conclude that the crosshair-based interface suffers from a couple of problems that prevent it from reaching the performance and usability that the touch-based interface has. Since crosshair-based interface is built to be more efficient than the touch-based interface, it falls short to providing the usability that we require of the interface. Interestingly, it doesn't even provide the effiency that it was meant to provide, since the users turned out be a lot more proficient with the touch-based interface. However, this doesn't render the crosshair-based interface useless. It may be applied in a game setting, where the higher difficulty of the interface provides a challenge to the player.

The experiment also provided an unexpected insight, which we'd like to present as a major contribution of this thesis. We were able to prove that users prefer to keep their mobile device stationary when it is not necessary to move it. The shift in viewpoint caused by the moving of the device makes interactions more difficult to perform and causes the user to frustrated. We would advice other researchers to take this contribution into account when developing their own interaction for 3D environments in augmented reality, or at least perform a test to check if their interaction doesn't suffer from it.

7.2 Future work

For our future work, we'd like to explore some options to improve the touch-based interface. A logical next step is to solve the fat finger problem that touch-based interfaces generally suffer from, in order to allow the interface to target an even wider audience.

Another interesting addition to this interface would be the support of a physics engine. Since we are focusing on interaction with objects that can be present in the real world, having them act as if that actually exist in the real world would be the next step in immersion in augmented reality. It could also introduce a more playful version of the interface, where the user would be able to slide objects across the room, stack objects that can fall over and even create simple games using the objects that he has at his disposal. The use of a physics engine is definitely an interesting way to go, although the processing power of the mobile device must be able to support this. Registering an augmented environment to the real world already requires a lot of resources with analyzing the video feed. Until a more lightweight version of registration becomes commonly available or the processing power of mobile devices significantly increases, the use of a physics engine in a room-like environment will be a tough goal to achieve.

Regardless of such limitations, the research area of interactions in augmented reality proved to be very interesting. We feel that, when it comes to the possibilities this research area provides, we are still only scratching the surface of it. Undoubtably many research projects for this area will arise, and we hope that this thesis may serve as an inspiration to those who wish to enter this field of research. We wish all future researchers the best of luck with their own augmented reality projects.

Bibliography

[1] Mark Billinghurst, Hirokazu Kato, and Seiko Myojin. Advanced interaction techniques for augmented reality applications. In *Proceedings of the 3rd International Conference on Virtual and Mixed Reality: Held as Part of HCI International 2009*, VMR '09, pages 13–22, Berlin, Heidelberg, 2009. Springer-Verlag. ISBN 978-3-642-02770-3. doi: http://dx.doi.org/10.1007/978-3-642-02771-0_2. URL http://dx.doi.org/10.1007/978-3-642-02771-0_2.

[2] Volkert Buchmann, Stephen Violich, Mark Billinghurst, and Andy Cockburn. Fingartips: gesture based direct manipulation in augmented reality. In *Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, GRAPHITE '04, pages 212–221, New York, NY, USA, 2004. ACM. ISBN 1-58113-883-0. doi: 10.1145/988834.988871. URL http://doi.acm.org/10.1145/988834.988871.

[3] María Luz Caballero, Ting-Ray Chang, María Menéndez, and Valentina Occhialini. Behand: augmented virtuality gestural interaction for mobile phones. In *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services*, MobileHCI '10, pages 451–454, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-835-3. doi: http://doi.acm.org/10.1145/1851600.1851704. URL http://doi.acm.org/10.1145/1851600.1851704.

[4] canming. Emgu cv. http://sourceforge.net/projects/emgucv/.

[5] Nguyen Thong Dang, Monica Tavanti, Ivan Rankin, and Matthew Cooper. A comparison of different input devices for a 3d environment. In *Proceedings of the 14th European conference on Cognitive ergonomics: invent! explore!*, ECCE '07, pages 153–160, New York, NY, USA, 2007. ACM. ISBN 978-1-84799-849-1. doi: 10.1145/1362550.1362581. URL http://doi.acm.org/10.1145/1362550.1362581.

[6] GSM Arena. Htc radar — full phone specifications. http://www.gsmarena.com/htc_radar-4131.php.

[7] S. G. Hart and L. E. Stavenland. Nasa-tlx form. http://humansystems.arc.nasa.gov/groups/TLX/downloads/TLXScale.pdf.

[8] S. G. Hart and L. E. Stavenland. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. In P. A. Hancock and N. Meshkati, editors, *Human Mental Workload*, chapter 7, pages 139–183. Elsevier, 1988. URL http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20000004342_1999205624.pdf.

[9] Sandra G. Hart. Nasa-task load index (nasa-tlx); 20 years later. In *Human Factors and Ergonomics Society Annual Meeting*, volume 50, 2006. URL http://www.ingentaconnect.com/content/hfes/hfproc/2006/00000050/00000009/art00009.

[10] Gunther Heidemann, Ingo Bax, and Holger Bekel. Multimodal interaction in an augmented reality scenario. In *Proceedings of the 6th international conference on Multimodal interfaces*, ICMI '04, pages 53–60, New York, NY, USA, 2004. ACM. ISBN 1-58113-995-0. doi: 10.1145/1027933.1027944. URL http://doi.acm.org/10.1145/1027933.1027944.

[11] Steven J. Henderson and Steven Feiner. Opportunistic controls: leveraging natural affordances as tangible user interfaces for augmented reality. In *Proceedings of the 2008 ACM symposium on Virtual reality software and technology*, VRST '08, pages 211–218, New York, NY, USA, 2008. ACM. ISBN 978-1-59593-951-7. doi: 10.1145/1450579.1450625. URL http://doi.acm.org/10.1145/1450579.1450625.

[12] Anders Henrysson, Mark Ollila, and Mark Billinghurst. Mobile phone based ar scene assembly. In *Proceedings of the 4th international conference on Mobile and ubiquitous multimedia*, MUM '05, pages 95–102, New York, NY, USA, 2005. ACM. ISBN 0-473-10658-2. doi: 10.1145/1149488.1149504. URL http://doi.acm.org/10.1145/1149488.1149504.

[13] Niels Henze and Susanne Boll. Evaluation of an off-screen visualization for magic lens and dynamic peephole interfaces. In *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services*, MobileHCI '10, pages 191–194, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-835-3. doi: 10.1145/1851600.1851632. URL http://doi.acm.org/10.1145/1851600.1851632.

[14] Niels Henze and Susanne Boll. Who's that girl? handheld augmented reality for printed photo books. In *Proceedings of the 13th IFIP TC 13 international conference on Human-computer interaction - Volume Part III*, INTERACT'11, pages 134–151, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 978-3-642-23764-5. URL http://dl.acm.org/citation.cfm?id=2042182.2042195.

[15] Wolfgang Hürst and Casper van Wezel. Multimodal interaction concepts for mobile augmented reality applications. In *Proceedings of the 17th international conference on Advances in multimedia modeling - Volume Part II*, MMM'11, pages 157–167, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 3-642-17828-6, 978-3-642-17828-3. URL http://dl.acm.org/citation.cfm?id=1950054.1950073.

[16] Hiroshi Ishii and Brygg Ullmer. Tangible bits: towards seamless interfaces between people, bits and atoms. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '97, pages 234–241, New York, NY, USA, 1997. ACM. ISBN 0-89791-802-9. doi: 10.1145/258549.258715. URL http://doi.acm.org/10.1145/258549.258715.

[17] Itseez. Opencv. http://opencv.org/.

[18] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana. Virtual object manipulation on a table-top ar environment. In *IEEE and ACM International Symposium on Augmented Reality, 2000.(ISAR 2000). Proceedings*, pages 111–119, 2000.

[19] Steven J. Kerr, Mark D. Rice, Yinquan Teo, Marcus Wan, Yian Ling Cheong, Jamie Ng, Lillian Ng-Thamrin, Thant Thura-Myo, and Dominic Wren. Wearable mobile augmented reality: evaluating outdoor user experience. In *Proceedings of the 10th International Conference on Virtual Reality Continuum and Its Applications in Industry*, VRCAI '11, pages 209–216, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-1060-4. doi: 10.1145/2087756.2087786. URL http://doi.acm.org/10.1145/2087756.2087786.

[20] Andrew Kirillov. Aforge. http://code.google.com/p/aforge/.

[21] Gun A. Lee and Mark Billinghurst. A user study on the snap-to-feature interaction method. In *Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality*, ISMAR '11, pages 245–246, Washington, DC, USA, 2011. IEEE Computer Society. ISBN 978-1-4577-2183-0. doi: 10.1109/ISMAR.2011.6092398. URL http://dx.doi.org/10.1109/ISMAR.2011.6092398.

[22] Gun A. Lee, Ungyeon Yang, Yongwan Kim, Dongsik Jo, Ki-Hong Kim, Jae Ha Kim, and Jin Sung Choi. Freeze-set-go interaction method for handheld mobile augmented reality environments. In *Proceedings of the 16th ACM Symposium on Virtual Reality Software and Technology*, VRST '09, pages 143–146, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-869-8. doi: http://doi.acm.org/10.1145/1643928.1643961. URL http://doi.acm.org/10.1145/1643928.1643961.

[23] Chunyuan Liao, Qiong Liu, Bee Liew, and Lynn Wilcox. Pacer: fine-grained interactive paper via camera-touch hybrid gestures on a cell phone. In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 2441–2450, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-929-9. doi: 10.1145/1753326.1753696. URL http://doi.acm.org/10.1145/1753326.1753696.

[24] metaio GmbH. metaio mobile sdk. http://www.metaio.com/software/mobile-sdk/.

[25] Ann Morrison, Antti Oulasvirta, Peter Peltonen, Saija Lemmela, Giulio Jacucci, Gerhard Reitmayr, Jaana Näsänen, and Antti Juustila. Like bees around the hive: a comparative study of a mobile augmented reality map. In *Proceedings of the 27th international conference on Human factors in computing systems*, CHI '09, pages 1889–1898, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-246-7. doi: 10.1145/1518701.1518991. URL http://doi.acm.org/10.1145/1518701.1518991.

[26] Nyatla. Nyartoolkit. http://sourceforge.jp/projects/nyartoolkit/.

[27] Ohan Oda. Goblin xna. http://goblinxna.codeplex.com/.

[28] Alex Olwal, Steven Feiner, and Susanna Heyman. Rubbing and tapping for precise and rapid selection on touch-screen displays. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 295–304, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-011-1. doi: 10.1145/1357054.1357105. URL http://doi.acm.org/10.1145/1357054.1357105.

[29] Qualcomm. Vuforia sdk. https://ar.qualcomm.at/sdk.

[30] Redmond Channel Partner. Microsoft shuttering windows mobile marketplace in may. http://rcpmag.com/articles/2012/03/09/windows-mobile-marketplace-to-close-in-two-months.aspx.

[31] Michael Rohs and Antti Oulasvirta. Target acquisition with camera phones when used as magic lenses. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 1409–1418, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-011-1. doi: 10.1145/1357054.1357275. URL http://doi.acm.org/10.1145/1357054.1357275.

[32] Michael Rohs, Antti Oulasvirta, and Tiia Suomalainen. Interaction with magic lenses: real-world validation of a fitts' law model. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 2725–2728, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0228-9. doi: 10.1145/1978942.1979343. URL http://doi.acm.org/10.1145/1978942.1979343.

[33] René Schulte. Slartoolkit - silverlight and windows phone augmented reality toolkit. http://slartoolkit.codeplex.com/, May 2010.

[34] Shamus P. Smith, Elizabeth L. Burd, Linxiao Ma, Iyad Alagha, and Andrew Hatch. Relative and absolute mappings for rotating remote 3d objects on multi-touch tabletops. In *Proceedings of the 25th BCS Conference on Human-Computer Interaction*, BCS-HCI '11, pages 299–308, Swinton, UK, UK, 2011. British Computer Society. URL http://dl.acm.org/citation.cfm?id=2305316.2305369.

[35] C. Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):pp. 72–101, 1904. ISSN 00029556. URL http://www.jstor.org/stable/1412159.

[36] Stephen M. Stigler. Francis galton's account of the invention of correlation. *Statistical Science*, 4(2):73–79, 1989. ISSN 0883-4237. doi: 10.1214/ss/1177012580.

[37] Student. The probable error of a mean. *Biometrika*, 6(1):1–25, March 1908. doi: 10.1093/biomet/6.1.1. URL http://biomet.oxfordjournals.org.

[38] Ivan E. Sutherland. The ultimate display. In *Proceedings of the IFIP Congress*, pages 506–508, 1965.

[39] the::unwired. Release: Microsoft introduces windows mobile 6.5.3 professional for windows phones. http://www.theunwired.net/?item=release-microsoft-introduces-windows-mobile-6-5-3-professional-for-windows-phones.

[40] John Underkoffler and Hiroshi Ishii. Urp: a luminous-tangible workbench for urban planning and design. In *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, CHI '99, pages 386–393, New York, NY, USA, 1999. ACM. ISBN 0-201-48559-1. doi: 10.1145/302979.303114. URL http://doi.acm.org/10.1145/302979.303114.

[41] Craig Villamor, Dan Willis, and Luke Wroblewski. Touch gesture reference guide. urlhttp://static.lukew.com/TouchGestureGuide.pdf.

[42] Luc Vlaming, Christopher Collins, Mark Hancock, Miguel Nacenta, Tobias Isenberg, and Sheelagh Carpendale. Integrating 2d mouse emulation with 3d manipulation for visualizations on a multi-touch table. In *ACM International Conference on Interactive Tabletops and Surfaces*, ITS '10, pages 221–230, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0399-6. doi: 10.1145/1936652.1936693. URL http://doi.acm.org/10.1145/1936652.1936693.

[43] Daniel Vogel and Patrick Baudisch. Shift: a technique for operating pen-based interfaces using touch. In *Proceedings of the SIGCHI conference on Human factors*

in computing systems, CHI '07, pages 657–666, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-593-9. doi: 10.1145/1240624.1240727. URL http://doi.acm.org/10.1145/1240624.1240727.

[44] Frank Wilcoxon. Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 1(6):80–83, 1945. ISSN 00994987. doi: 10.2307/3001968. URL http://dx.doi.org/10.2307/3001968.

[45] Brett Wilkinson and Paul Calder. Investigating touch interactions for an augmented world. In *Proceedings of the 20th Australasian Conference on Computer-Human Interaction: Designing for Habitus and Habitat*, OZCHI '08, pages 25–32, New York, NY, USA, 2008. ACM. ISBN 0-9803063-4-5. doi: 10.1145/1517744.1517752. URL http://doi.acm.org/10.1145/1517744.1517752.

[46] Cary Williams, Xing Dong Yang, Grant Partridge, Joshua Millar-Usiskin, Arkady Major, and Pourang Irani. Tzee: exploiting the lighting properties of multi-touch tabletops for tangible 3d interactions. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 1363–1372, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0228-9. doi: 10.1145/1978942.1979143. URL http://doi.acm.org/10.1145/1978942.1979143.

[47] Ka-Ping Yee. Peephole displays: pen interaction on spatially aware handheld computers. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '03, pages 1–8, New York, NY, USA, 2003. ACM. ISBN 1-58113-630-7. doi: 10.1145/642611.642613. URL http://doi.acm.org/10.1145/642611.642613.

Appendix A

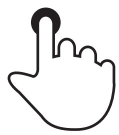
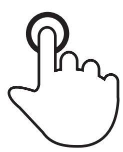
Touch gesture reference guide

Below are two pages of the Touch gesture reference guide, which was used to determine appropriate interactions for the manipulations a user could perform. The original can be found at http://static.lukew.com/TouchGestureGuide.pdf.

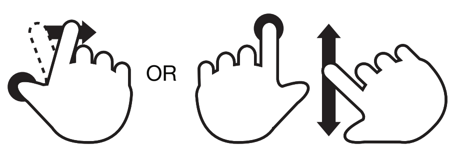
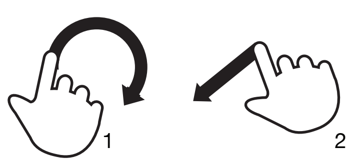
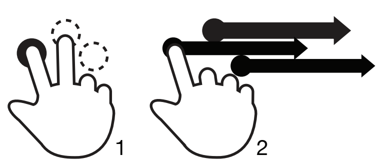
Major User Actions

Currently supported by touch gesture systems

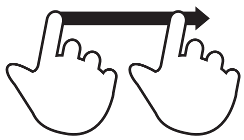
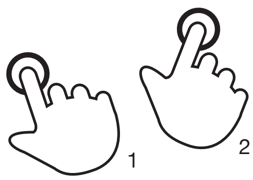
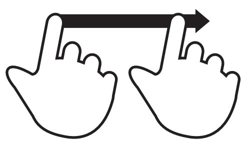
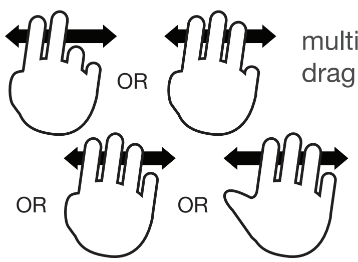
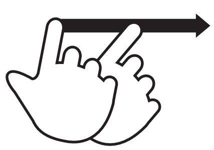
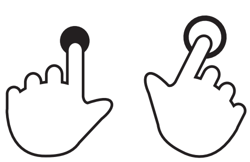
BASIC ACTIONS

user action	gesture	description
Change mode	 press	Touch surface for extended period of time
Open	 double tap	Rapidly touch surface twice with fingertip
Select	 tap	Briefly touch surface with fingertip

OBJECT-RELATED ACTIONS

user action	gesture	description
Adjust	 press and drag	Press surface with one finger and move second finger over surface without losing contact
	 lasso and cross	Make circular motion with finger, then cross over selected object
Bundle	 press and tap, then drag	Touch first object while second finger taps other objects, the move selected objects by dragging first finger

OBJECT-RELATED ACTIONS (continued)

user action	gesture		description
Delete		drag (across item or off-screen)	Move fingertip over surface without losing contact
Duplicate		tap (source and destination)	Touch object, then touch elsewhere on surface
Move		drag (and drop)	Move fingertip over surface without losing contact
		multi-finger drag	Move two to five fingertips over surface without losing contact
		flick	Quickly brush surface with fingertip
		press and tap	With one finger on object, touch elsewhere on surface with second finger
Rotate		rotate	

Appendix B

NASA-TLX

The Task Load Index method developed by NASA is a method to measure difficulty that is experienced by the user. It is developed by Hart and Stavenland and is commonly used in literature to measure the task load for a task. The NASA-TLX measures the task load of six task scales:

- Mental load
- Physical load
- Temporal load
- Performance
- Effort
- Frustration

For all the scales, a lower value means a lower task load, which is better. This is the reason why the Performance scale is reversed.

Figure 8.6

NASA Task Load Index

Hart and Staveland's NASA Task Load Index (TLX) method assesses work load on five 7-point scales. Increments of high, medium and low estimates for each point result in 21 gradations on the scales.

Name	Task	Date

Mental Demand — How mentally demanding was the task?

Very Low — Very High

Physical Demand — How physically demanding was the task?

Very Low — Very High

Temporal Demand — How hurried or rushed was the pace of the task?

Very Low — Very High

Performance — How successful were you in accomplishing what you were asked to do?

Perfect — Failure

Effort — How hard did you have to work to accomplish your level of performance?

Very Low — Very High

Frustration — How insecure, discouraged, irritated, stressed, and annoyed wereyou?

Very Low — Very High

Appendix C

Results related tables and images

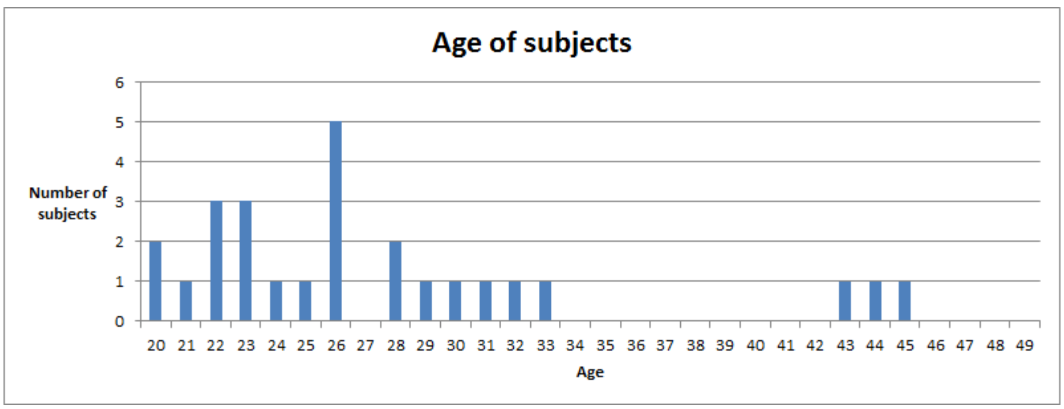


Figure C.1: The number of participants that are of a certain age.

Paired Samples Test - Student's t-test								
	Paired Differences							
				Interval of the				
	Mean	Std. Deviation	Std. Error Mean	Lower	Upper	t	df	**Sig. (2-tailed)**
ScoreTouch - ScoreCrosshair	1,88462	4,21736	,82709	,18119	3,58805	2,279	25	**,031**
FrustrationTouch - FrustrationCrosshair	-,21923	,60993	,11962	-,46559	,02713	-1,833	25	**,079**

Figure C.2: Student's t-test.

Test Statistics - Wilcoxon signed-rank test				
	MentalDemandCrosshair - MentalDemandTouch	PhysicalDemandCrosshair - PhysicalDemandTouch	PerformanceCrosshair - PerformanceTouch	EffortCrosshair - EffortTouch
Z	-2,331	-,698	-1,232	-1,858
Asymp. Sig. (2-tailed)	,020	,485	,218	,063

Figure C.3: Wilcoxon signed-rank test.

Correlations																	
Pearson correlation			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	Gender	Correlation Coefficient	1,000														
		Sig. (2-tailed)															
2	Age	Correlation Coefficient	-,155	1,000													
		Sig. (2-tailed)	,449														
3	Interface Preference	Correlation Coefficient	-,066	-,201	1,000												
		Sig. (2-tailed)	,750	,324													
4	Score Touch	Correlation Coefficient	-,407	-,559	,143	1,000											
		Sig. (2-tailed)	,039	,003	,487												
5	Score Crosshair	Correlation Coefficient	-,261	-,558	,488	,821	1,000										
		Sig. (2-tailed)	,198	,003	,011	,000											
6	Mental Demand Touch	Correlation Coefficient	,412	-,119	,308	-,073	,086	1,000									
		Sig. (2-tailed)	,036	,563	,125	,723	,674										
7	Physical Demand Touch	Correlation Coefficient	,002	-,191	,326	-,191	,093	,191	1,000								
		Sig. (2-tailed)	,993	,350	,104	,349	,652	,351									
8	Performance Touch	Correlation Coefficient	,011	,188	,251	-,418	-,235	,121	,211	1,000							
		Sig. (2-tailed)	,957	,358	,216	,034	,248	,556	,302								
9	Effort Touch	Correlation Coefficient	,223	-,102	,384	-,137	,151	,453	,465	,301	1,000						
		Sig. (2-tailed)	,274	,620	,053	,504	,463	,020	,017	,135							
10	Frustration Touch	Correlation Coefficient	,118	,080	,375	-,372	-,167	,228	,311	,456	,495	1,000					
		Sig. (2-tailed)	,565	,696	,059	,061	,414	,263	,123	,019	,010						
11	Mental Demand Crosshair	Correlation Coefficient	,294	,269	-,118	-,519	-,389	,502	,350	,169	,283	,338	1,000				
		Sig. (2-tailed)	,145	,184	,566	,007	,049	,009	,079	,408	,161	,091					
12	Physical Demand Crosshair	Correlation Coefficient	-,019	-,116	-,145	-,151	,002	,006	,694	-,107	,204	-,049	,383	1,000			
		Sig. (2-tailed)	,925	,572	,479	,463	,993	,976	,000	,602	,317	,813	,053				
13	Performance Crosshair	Correlation Coefficient	,324	,255	-,292	-,458	-,614	-,031	-,452	,321	-,011	,084	,027	-,336	1,000		
		Sig. (2-tailed)	,106	,209	,147	,019	,001	,880	,020	,110	,959	,682	,897	,093			
14	Effort Crosshair	Correlation Coefficient	,149	-,182	-,055	-,119	-,105	,118	,343	,047	,531	,190	,390	,407	,070	1,000	
		Sig. (2-tailed)	,469	,373	,789	,562	,610	,566	,086	,818	,005	,352	,049	,039	,734		
15	Frustration Crosshair	Correlation Coefficient	,176	-,054	-,492	-,130	-,269	-,296	-,086	-,042	,151	,269	,141	,105	,386	,362	1,000
		Sig. (2-tailed)	,389	,792	,011	,527	,184	,143	,678	,839	,463	,184	,492	,610	,051	,069	

Figure C.4: Pearson correlation matrix.

Correlations

Spearman's rho			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	Gender	Correlation Coefficient	1,000														
		Sig. (2-tailed)	.														
2	Age	Correlation Coefficient	-,121	1,000													
		Sig. (2-tailed)	,556	.													
3	Interface Preference	Correlation Coefficient	-,066	-,261	1,000												
		Sig. (2-tailed)	,750	,198	.												
4	Score Touch	Correlation Coefficient	-,442	-,609	,151	1,000											
		Sig. (2-tailed)	,024	,001	,462	.											
5	Score Crosshair	Correlation Coefficient	-,297	-,611	,473	,813	1,000										
		Sig. (2-tailed)	,140	,001	,015	,000	.										
6	Mental Demand Touch	Correlation Coefficient	,402	-,248	,323	-,110	,078	1,000									
		Sig. (2-tailed)	,042	,222	,108	,594	,705	.									
7	Physical Demand Touch	Correlation Coefficient	,016	-,144	,329	-,214	,020	,203	1,000								
		Sig. (2-tailed)	,938	,483	,101	,294	,922	,321	.								
8	Performance Touch	Correlation Coefficient	,178	,151	,288	-,415	-,296	,121	,184	1,000							
		Sig. (2-tailed)	,385	,462	,154	,035	,142	,555	,368	.							
9	Effort Touch	Correlation Coefficient	,145	-,253	,354	-,115	,177	,439	,507	,265	1,000						
		Sig. (2-tailed)	,480	,213	,076	,575	,386	,025	,008	,190	.						
10	Frustration Touch	Correlation Coefficient	,105	,046	,369	-,322	-,153	,250	,361	,440	,486	1,000					
		Sig. (2-tailed)	,611	,823	,063	,108	,457	,218	,070	,025	,012	.					
11	Mental Demand Crosshair	Correlation Coefficient	,347	,258	-,141	-,561	-,388	,477	,342	,198	,314	,262	1,000				
		Sig. (2-tailed)	,083	,203	,493	,003	,050	,014	,087	,333	,119	,195	.				
12	Physical Demand Crosshair	Correlation Coefficient	,016	-,123	-,156	-,181	-,020	-,023	,643	-,142	,195	-,126	,369	1,000			
		Sig. (2-tailed)	,938	,550	,445	,377	,922	,912	,000	,490	,341	,539	,063	.			
13	Performance Crosshair	Correlation Coefficient	,331	,328	-,323	-,462	-,614	-,041	-,350	,373	-,026	,087	,065	-,315	1,000		
		Sig. (2-tailed)	,099	,102	,107	,017	,001	,841	,080	,061	,900	,671	,753	,117	.		
14	Effort Crosshair	Correlation Coefficient	,226	-,177	-,073	-,178	-,163	,077	,365	,084	,542	,157	,369	,431	,134	1,000	
		Sig. (2-tailed)	,267	,386	,723	,384	,427	,708	,067	,684	,004	,443	,063	,028	,515	.	
15	Frustration Crosshair	Correlation Coefficient	,169	,048	-,552	-,160	-,290	-,289	-,080	,032	,123	,268	,175	,044	,478	,318	1,000
		Sig. (2-tailed)	,409	,817	,003	,435	,151	,153	,696	,878	,551	,185	,392	,830	,014	,113	.

Figure C.5: Spearman's rank correlation matrix.