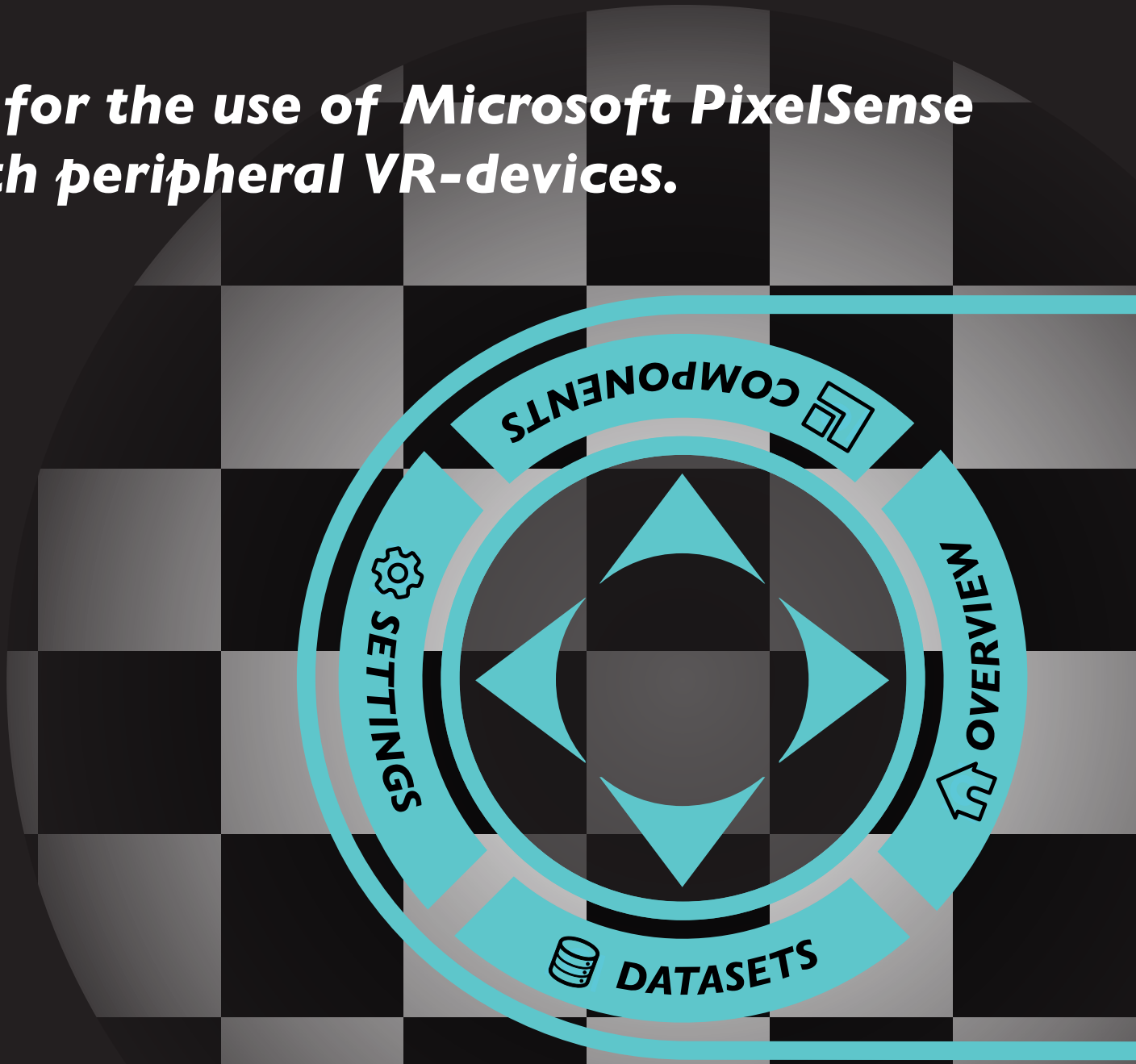
Interface redesign for the use of Microsoft PixelSense in combination with peripheral VR-devices.



Tom Simons

October 2015
BSc graduation project
Industrial Design
University of Twente

Interface redesign for the use of Microsoft PixelSense in combination with peripheral VR-devices.

Bachelor Industrial Design

Author:
T.V. Simons
S1116908

Study:
Industrial Design
Universiteit Twente

Commissioned by:
University of Twente
Drienerlolaan 5
7500 AE Enschede, the Netherlands

Date Bachelor Exam:
29-10-2015

Examination committee:
M.C. van der Voort
R.G.J. Damgrave

Date of publication:
22-10-2015

Preface

Throughout my study Industrial Design, I came into contact with many aspects of the design process. Having done several programming related projects and subjects, I discovered that I found the design aspects relating to the digital domain the most appealing. While searching for an assignment for my Bachelor thesis, I encountered a description of a project to design an interface using haptic virtual reality interaction. Being fascinated by all the possibilities virtual reality offers, and the opportunity of working on a state-of-the-art piece of technology, I applied for an interview.

During the project I had to overcome many obstacles and got a taste of integrating the many aspects of the design process into one product. I feel that I greatly improved my skills, and I am happy to have not only designed a conceptual product, but also to have actually got to create a working prototype.

I would like to thank my supervisor, Roy Damgrave, for all the support and feedback, and all my friends who helped me perform usage tests during several stages of this project.

Summary

In the current day and age the virtual world is becoming more and more a part of our daily routine. Whether you stay in touch with your friends through your smartphone, or watch a video on your laptop, we are surrounded by the virtual world.

In the past few years, significant steps have been made to blur the lines between the real world and the virtual world. Devices like the Oculus Rift allow people to be completely immersed in the virtual world, and that's just the beginning.

The University of Twente's goal is to stay ahead in these technical developments and has commissioned this assignment to make steps in the field of VR. The goal of this assignment was to create a system that allows users to use real world objects in unison with a virtual representation of this object. In the virtual world the objects can be augmented with additional information.

The design process started with making an analysis of the equipment and current system, and the needs of the different stakeholders. Several scenarios were written to further understand the stakeholders. The analysis phase concluded with the formulation of the requirements for the system.

Once the requirements were clear, the design of the system could begin. Possibilities for several peripheral devices to work in the system were explored, the user interface was designed, and a technical design was made.

Once the final concept was chosen, the realisation of this concept could begin. This was done in Unity[1], with some additional coding in Visual Studio[2]. The classes of the system were designed in such a way that easy expansion of the system is possible. During this phase several technical limitations were encountered, due to which slight changes in the concept had to be made.

Finally, the created system was evaluated and several possible improvements were formulated. These improvements were the result of both a usage test and technical limitations which prevented implementation during the realisation phase.

1 *Game engine and programming environment: https://unity3d.com/*
2 *Programming environment: https://www.visualstudio.com/*

Table of contents

Chapter 1: Introduction

1.1 - University of Twente

This project was commissisoned by the University of Twente. The goals of the University of Twente are to promote innovation and entrepreneurship. Besides this the University of Twente is active in areas of social development. Because of this, the University of Twente strives to stay ahead in the field of technical development. This project was commissioned to contribute to this vision.

1.2 - Problem definition

The University of Twente has a research laboratory solely dedicated to research in the areas of virtual reality (VR). This laboratory has state-of-the-art equipment in this field, such as tactile pens that allow users to 'feel' their virtual 3D-model, Oculus Rifts to immerse users in the virtual world, and a Samsung SUR-40 with Microsoft Pixelsense[1]. This is a 'VR-table' that is able to form a link between the real world and the virtual world. By placing real world objects on the table's surface, infrared cameras beneath the screen's surface capture these objects, which in turn allows the VR-Table to 'recognise' them. This device has great potential for creating a close link between the real world and the virtual world. The University of Twente has a prototype of a system for this purpose, but this prototype works on a minimal level, requiring a great deal of expertise to operate, and low flexibility.

The problem addressed in this project was the redesign of this system, in order to utilize it to its full potential. This interface should be able to be used with minimal effort and expertise. It is intended to allow design teams to separate information of their design based on their field of expertise, while maintaining the VR-table as central hub. This way design teams can efficiently communicate with each other, whilst not being overloaded with information that is not relevant for the individual members' field of expertise.

1.3 - Aim of the project

The final goal of the project is de redesign the existing user interface consisting of the VR-table and other peripheral devices. The final design should be realized. The final product should be a useful interface and provide users with a pleasant experience. Besides this, the system should be integrated with the University of Twente's existing workflow, and the look and feel of the interface should integrate with the University's vision.

1.4 - Reading guide

In this report, several terms will be used.
The final product, including everything that is required to make it work will be referred to as 'the system'. The term 'VR-table' will refer to the Samsung SUR-40 with Microsoft Pixelsense. 'Peripheral devices' will refer to all potential electronic devices to be used in the system.

1 See: http://www.microsoft.com/en-us/pixelsense/default.aspx

In the upcoming four chapters, the design process will be explained in detail, starting with the chapter ‘Analysis’. In this chapter a basis is created for further design choices.

The chapter ‘Concept design’, several possibilities for the system will be explored. At the end of this chapter a user interface and the technical structure of the system will be chosen.

In the chapter ‘Concept realisation’ the steps towards realizing the final system will be explained. This chapter also includes a description of the function of all of the classes in the final applications.

In the last chapter, ‘System evaluation’, the final system will be evaluated and recommendations for improvements and expansions will be made.

Chapter 2: Analysis

This chapter will explain the first phase of the design process, the analysis phase. During this phase, a basis is created on which design choices will be based. The current system is analysed, market research is done and several scenarios are created. The software platform is examined and finally requirements are formulated.

2.1 - System analysis

In order to gain an understanding of the goal and the possibilities of the system, an analysis was performed on the current system. This included the setup of the current system, the technical details and a stakeholders analysis to identify all the parties which come in contact with the system.

2.1.1 - Current system

The current system is centred around a 'VR-table'. This is a Samsung SUR40 with Microsoft PixelSense table (formerly Microsoft Surface). The table is an interactive multi-touch computer which can recognize certain real world objects. These objects can be a finger, a special 'tag' that works similar to a barcode, or an unrecognizable 'blob'. Another key feature of the VR-table is that the orientation of these objects can also be recognised, allowing for a full 360-degree experience around the table. For a promotional thesis, Thalen (2013) coded an interface to use the VR-table in combination with one or more computer screens as a design tool for User Centered design. The created interface responded to moving around physical objects on the table, each with a specific tag, which would then move around in a 3D-environment on the computer screen. This environment was created in Blender[1] and included the possibility to display a realistic environment with real time shadows and textures, but also the possibility for a more abstract, less detailed environment.

2.1.2 - Use of the current system

The VR-table and interface with a 3D-environment was used in a demonstration session by Thalen to show the possibilities of implementing virtual reality tools in the design process. This entailed that the interface was mainly set up to be a proof of concept and there was no elaborate user interface built for easy customisation by end users. The current

Image 2.1: Old system
Image source: Thalen (2013)

1 *3D-software: https://www.blender.org/*

system displays a room where objects, such as furniture, can be placed by moving tagged cubes on the VR-Table. One cube is designated as the camera, and can be used to change the point of view in the room.

2.1.3 - Shortcomings

The current system is limited to the current setup, and was used primarily by Thalen as a proof of concept to trigger a discussion about the use of VR in the design process. Changing the setup requires expert knowledge of the used software, such as Blender, and is highly inflexible. The aim of this project is to design a new interface, which is able to implement this type of VR interaction in the design process and expand on the possibilities of the VR-table in combination with other peripheral devices as a design tool.

2.1.4 - Technical details

The current system is capable of recognising special byte-tags. The position, rotation and id of the tag are displayed on the VR-table surface and simultaneously stored in an SQL-database. Another computer reads the information in this database and translates it to the position and rotation of a predefined 3D-model in the Blender environment. This Blender scene is then rendered on a separate screen.

2.1.5 - Stakeholder analysis

The main aim of the new system is to serve as a design tool for design teams during the design process. However, there are more stakeholders than just the specialists in these teams. These teams can be professional teams at companies, but also project teams consisting of university students. Because this is a new technology, companies need to be convinced this system is useful to purchase for their design department, and are thus considered as potential customers. Besides this the system can be used by the University of Twente as an advanced piece of engineering to convey their innovative vision towards potential students and guests.

Design teams

The design teams will be the main end users of the system. These teams consist of different specialists, each with their own field of expertise. The design process usually requires integration of different fields into one system, but not all information is relevant to each team member. One example is the design process of a USB-mouse. This product requires a designer of the ergonomics for the casing of the product, an electrical engineer for all the internal hardware, and a software engineer to make sure the product functions when plugged in. Each of the experts in the team is interested in the specifications of the product relating to their own field. The new system should therefore be able to separate the different sets of specifications, so each specialist will only see the information relevant to them.

Besides this the new system should add ease to the design process. It should provide a fast and intuitive interface between the designers and the computer, more so than with traditional design tools.

Potential customers

Potential customers in this context are companies who need to be convinced that the new system will be an improvement to their design process. This means that the system

should be both functional and easily adjustable for multiple purposes.

Potential students
The new system should also be able to be used as a promotional tool towards potential students. The system should be able to be used in simple demonstrations by people without any technical background. The system should have an advanced look and feel, yet be simple and intuitive to operate.

2.2 - Market research

Although there are many applications which run on the VR-table itself, there aren't many applications which use external devices as an extension of the VR-table. As the VR-Table is in essence a Windows PC, connecting external devices without any specialized software is limited to what a regular PC is able to do, and requires a physical cable to connect the two.

The VR-table uses Microsoft Pixelsense, which is an example of a Natural User Interface (NUI). A NUI removes the need for control devices and a Graphical User Interface (GUI) to control the computer's actions. An NUI strives to provide an intuitive control of the computer, as close as possible to real world object manipulation. The new system should make use of this capability, and thus the users should be able to know how the interface works without prior knowledge.

Other examples of Natural User Interfaces include Microsoft's Kinect,[2] Nindento Wii[3], and Elon Musk's new gesture-controlled interface[4].

2.3 - Scenarios

In order to further understand the needs of the stakeholders several scenarios were set up.

2.3.1 - Scenario 1: Design team 1

Tony is an electrical engineer in a design team working on a new type of USB-stick. The design team consists of three other members: An Industrial designer, a programmer and a marketing manager. The design team has to work on the following components: A casing, circuit board and software, packaging and marketing. Tony is only interested in the circuit board and parts of the software. Their company has recently acquired a new design tool to help them work on this project. The tool allows for the entire team to work on the same computer model, but also to move (a part of the) the virtual model to other devices and separate its components into the team members' specific area of expertise. The team members have never used the new design tool before. However, immediately after starting to use it, they know how to move the model around, and after 5 minutes they have figured out how to use the tool in their work. Tony places a mock-up of the USB-stick on the table and immediately a menu appears around the model, which allows the team to choose which part they want to see. Tony drags the tab labelled 'electronics' to the side of the table, which makes the entire circuit board appear on a separate screen. Tony analyses the circuit board while the rest of the team discusses something unrelated.

2.3.2 - Scenario 2: Design team 2

MMaria is using 3D-modelling software to model a new type of electric water boiler. She works in a design team and is in charge of integrating all parts in the 3D-model. She spent several days perfecting the model according

2 *Motion controller: kinect.com*
3 *Gaming console: wii.com*
4 *See: http://www.dailymail.co.uk/sciencetech/article-2414005/Hyperloop-creator-invents-Iron-Man-inspired-3D-software-design-ROCKET-PARTS.html*

to the specifications, and once she is finished presents it to the rest of the team. However, the team members working on the exterior of the product have changed some details in different parts of the model, so Maria has to incorporate those changes to the model as well. Slightly agitated she returns to her computer to change the details, only to discover the changes affect the electronics as well. She contacts the electrical engineers of the team to explain the problem. Frustrated, she waits several days until the electrical engineers have devised a solution. She wishes there would be a more efficient way of communicating between the different disciplines in the team.

2.3.3 - Scenario 3: Customers

FruitBasket is a company specialized in consumer electronics. In order to maintain a competitive edge in their market segment, continuous innovation is required. All products the company produces are carefully designed by teams of specialists. To ensure reliable and efficient communication between the different team members, the company's executive board is researching new design tools to enable their design teams to work fluently. The sought after tool has to remove the obstacles imposed by traditional design tools. Besides this, the new tool has to be cost efficient and must not require training in order to use. The tool has to integrate with the companies established computer aided design (CAD) tools.

2.3.4 - Scenario 4: Promotion

James is an upcoming student. He is fascinated by new technology and wishes to become an 'inventor' when he graduates. During the open days of the University of Twente he is shown the university's virtual reality lab. The lab has a futuristic feel to it and the university's VR-system fits right in this environment.
A demonstration is given where a physical 3D-model, which was printed only minutes ago in the previous stage of the tour, is placed on the table. James feels like he is in a science fiction film, as the table responds to the model, creating a digital version on a computer screen. The demonstrator swipes on the table and a fancy looking exploded view is produced on several screens around the VR-table. James is so impressed, he enrols the next day.

2.4 - Platform

The new system will be realised using the Unity engine. Unity is a development platform able to create both 2D and 3D-applications, and is mostly used for games. It has native support for multiple platforms, such as Windows, Mac OSX and iOS. This multiplatform support allows for easy adaptation of the new software to be able to run on many devices.

Unity is able to natively read 3D-files with the .dae, .3ds, .dxf and .obj extension. These are the most used file formats and most 3D-applications are able to export in these formats. Other formats such as 3DS max, maya and blender files are able to be imported when a copy of the respective software is installed on the computer.

The University of Twente uses Solidworks as main 3D-CAD software. Solidworks is one of the industry standards regarding CAD. Unity is mostly considered a game engine and therefore optimised to be used with polygon-based models, whereas Solidworks creates feature-based models. Therefore, some compatibility problems may arise when trying to integrate the two. Although Solidworks is able to export in one of the four native formats Unity uses, any changes made to the CAD-model will require an additional export to be able to be read by Unity. If possible this step should be automated

so that any changes in a CAD-model will be immediately reflected in the VR-system.

2.5 - Collage

To try to envision the mood the system would have to convey the following collage was made. The collage includes both real and fictional uses of VR.

Image 2.2: Collage

The main thing seen in the collage is that all the people use their entire arms and hands to manipulate the virtual world. This is in line with the idea of a NUI, as real world manipulation also requires physical movement. The collage also conveys a futuristic and complex feel. This contrasts with the simplicity a NUI tries to achieve. However this contrast will allow for a pleasant user experience as the user will feel in control of something complex, without having to go through rigorous training.

2.6 - Requirements

From the analysis above, the following requirements are formulated.

2.6.1 - Use

- The new system should be usable without programming knowledge.
- The new system should be able to separately display information according to relevant field of expertise.
- The new interface should be intuitive to use.
- Limited customisation should be required of the users' CAD-models in order to be usable with the system.
- The system should implement a natural user interface.
- The system should show simple information at first, but this information should expand at the users' will.

2.6.2 - Look and feel

- The new system should have a futuristic look.
- The system should make the user feel highly in control of the information and models.
- The new system should achieve a complex feel while maintaining an intuitive control.

2.6.3 - Features

- The new system should be able to connect with multiple devices.
- The new system should be easily expandable to accommodate for new technologies.
- The new system has to be usable with existing 3D-software and -files.
- The new system has to be programmed in Unity.

2.6.4 - Wishes

Besides these requirements some wishes were derived, to further the natural feel of the user experience.

- The new system should be able to automatically link real world objects to their digital counterpart.
- The system should automate the CAD-file conversion to a Unity native file.

Chapter 2
Analysis

Chapter 3: Concept design

In this chapter the design phase will be discussed. In the first part several ideas are proposed, which could be incorporated in the system. In the second part of this chapter these ideas will be converged into a final concept.

3.1 - Idea generation

Once the analysis was completed, the design of the new system could begin. This began with the generation of ideas for many different aspects of the system. This included features of the system, the graphical design of the user interfaces, and technical structure.

3.1.1 - Devices and features

The VR-table is the main component in the new system, but there will be several other devices that will also be part of the new system. The first component that comes to mind is a monitor, possibly connected to a separate computer. This monitor will be the primary source of output for the system, displaying a virtual version of the models and its environment.

Besides this, head-mounted VR displays, such as the Oculus Rift, are on the rise. These displays allow the user to see the virtual object in 3D-perspective and will therefore enhance the look and experience of the 3D-models and environment.

Displaying content on mobile devices is also a possibility. Mobile devices, such as smartphones and tablets, would allow a great deal of flexibility. In case a tag is placed on the back of a mobile device, not only could this device then be used for the output, but it could also be used to generate input on the VR-table. A phone placed next to a model could enable it to copy the model view to the phone. The tagged phone could also act as a camera model, with the camera's point of view displayed on the phone's

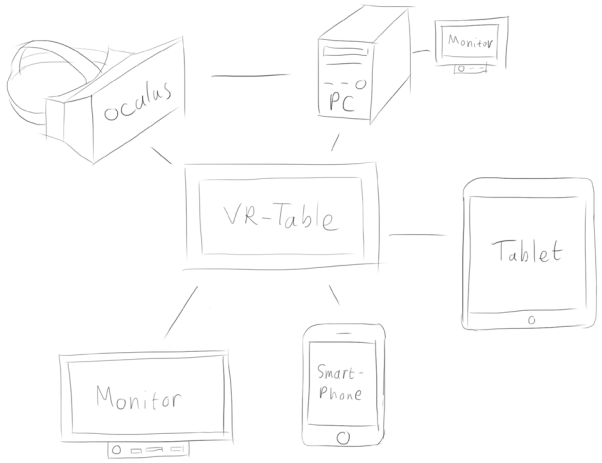


Image 3.1: Possible secondary output devices

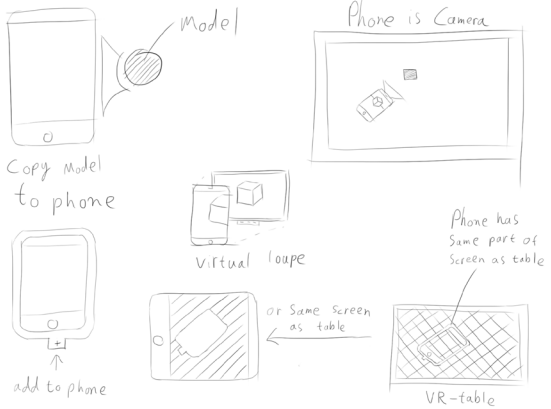


Image 3.2: Mobile device interfaces

screen. Another way to use mobile devices could be in the form of a virtual loupe such as proposed by Weidlich, Cser, Polzin, Cristiano & Zickner. (2009). The user then uses the mobile device to control the model's position orientation and zoom on the screen. This could be just on the mobile device itself, but also on a secondary screen. Several possible interface options are shown in image 3.2.

Aside from the VR-table itself and mobile devices, other input devices are also a possibility. The limitation of the interface between the table and the physical model is that the physical model needs to have a predetermined tag attached underneath it. It is not the model itself that is linked with the virtual model, but the tag on the model that is pre-programmed to correspond to a certain virtual 3D-model. If this process could be automated, it would take away another barrier and shift the system more towards a NUI. It would be possible to achieve this by using several 3D-scanners, such as Microsoft Kinects to scan the objects on the table, finding the corresponding model on the computer, and linking the tag to this model. Kinects have native Unity support, which allows for easy integration with the rest of the software for the system. This is a possibility only because the system is intended to be used in conjunction with a 3D-representation of a virtual model and is of course limited to the resolution of the Kinects and obstruction of the model on the table by other models.

Using a Leap Motion[1] sensor in conjunction with an oculus rift would also enhance the NUI characteristics of the system. The Leap Motion is a sensor that allows users to see and use their hands in virtual reality. This way the control of the virtual models is no longer limited to traditional computer input, such as mouse and keyboards. Instead, users can control their virtual models directly with their hand gestures.

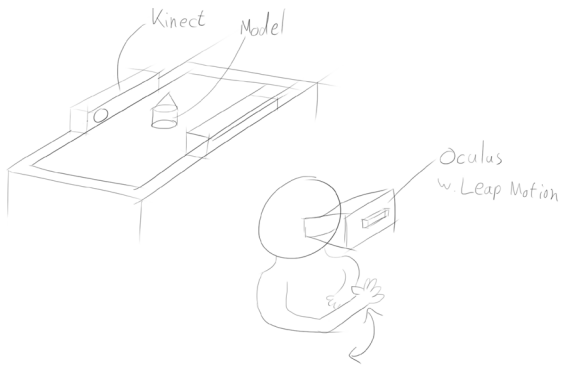


Image 3.3: Possible secondary input devices

3.1.2 - VR-Table user interface

The VR-table is the centre of the system and the primary source of input. Because multiple models are intended to be placed on the VR-table's surface, a compact user interface is required. This allows for more models being placed in each other's proximity. Users should be able to easily expand and collapse the user interface, to show more information on a model, and make room for other interfaces, respectively. To keep the user interface simple and easy to learn and understand, adequate use of submenus should be made. By doing so users are not overwhelmed by a barrage of options, resulting in confusion. Having many submenus inherently requires some options to be hidden from plain sight. The interface should therefore

1 *Motion controller: https://www.leapmotion.com/*

be intuitive to enable users to find all options they want without having to search through all the submenus. To create easy to understand

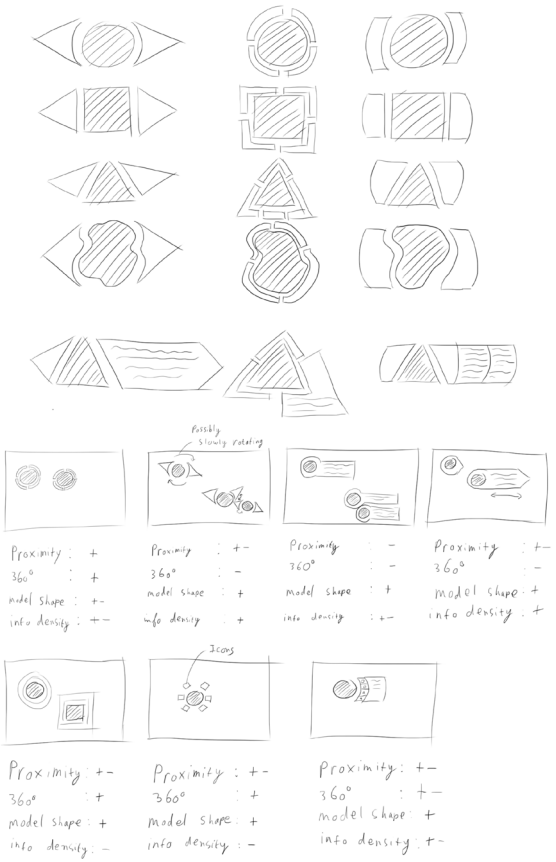


Image 3.4: Scetches of the user interface design of a single model

submenus, categorization is important. One category is the information, or data, that should be displayed. Because the nature of

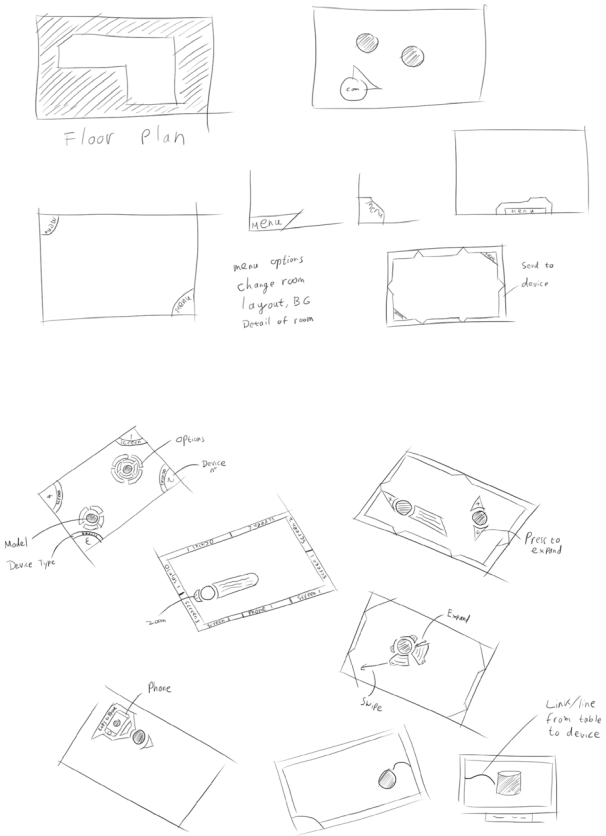


Image 3.5: Scetches of the user interface design of the VR-table

this information differs per product, a further subdivision is hard to predetermine. Another set of information that can be associated with a single product are its components. Components differ from data in the sense that data is an abstract concept associated with the product, such as price, dimensions etc., while components are the physical components of a product, such as circuit board, casing, battery, etc. All these components can in turn have their own datasets associated with them.
Another important aspect of the interface is that it should be usable from any direction. As the intended use of the system is to have users stand on any side of the VR-table's surface, there should be as little directional limited aspects to the interface. Finally, the interface should be able to be used with a variety of model shapes. The tagged object is part of the interface, but the user is the one determining the shape and size of the plain touching the table's surface. This should be taken into account as much as possible with the interface's design.

Besides providing information about the model, the user interface should also be integrated with other connected devices. The VR-Table acts as a central interface with the other devices as secondary interfaces, all these interfaces

should integrate to create one coherent system. This can be achieved by having a similar layout of the interface across all devices, but also by incorporating all connected devices in the table-interface. In image 3.6 some examples on how this could be implemented are shown.

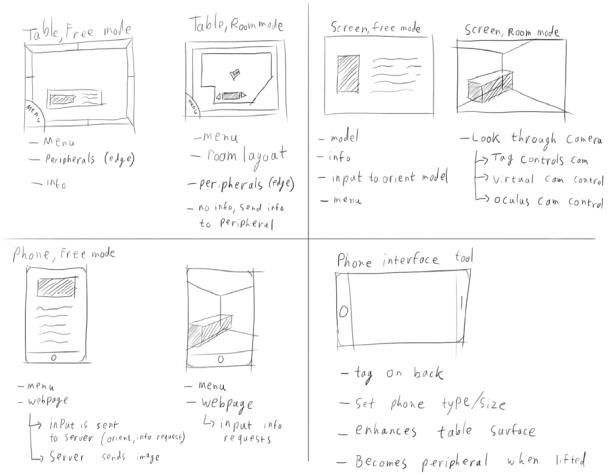


Image 3.6 Scetches of the differences in user interface design for Room mode and Free mode

3.1.3 - Screen User Interface

The second interface to be used is an interface on another screen. This screen can for example be a separate PC with a monitor or a mobile device. The function of these screens is primarily to provide a virtual environment with a virtual 3D-representation of the models and their related information. When displaying the models, the user may want to choose one of two distinct display methods.

Room Mode

The first display method requires the models to maintain the same spatial relation to each other. This method is useful when for example users have a floor plan of their room and wish to create several possible configurations of furnishing. The tagged objects are scale models of pieces of furniture, and their position in relation to each other and on the table matters for the room decor. One of the tagged objects is used to place a camera in the room, as to simulate a person's point of view in the room. The secondary screen then displays a virtual representation of the furnished room from this point of view.

Free mode

The second possibility is to allow users to examine one model up close. This is useful when wanting to examine a detail of a specific model. In this case the camera is not linked to a model on the table, but is controlled by the user on the secondary screen. In this case the spatial relation between the tagged objects on the table is less important.

These two viewing modes not only affect the way the interface on the peripheral device behaves, bus possibly also the layout of the table. In room mode an outline of a floor plan can be added to the table's surface to show the user the bounds of the room. These walls should then also be reflected in the virtual representation of the room on the peripheral devices.

3.2 - Concept generation

With these ideas as a basis, a conceptual system was created. The final system should be able to be expanded with extra functionality easily. Because of this, the choice was made to create a modular concept, instead of several predefined concepts. This means several concepts of different modules were made which integrate to one final system. For this project a difference was made between the essential modules to create the basis for a functional system and extensions, which improve the system. The essential modules are the VR-Table and at least one secondary screen.

3.2.1 - VR-Table user interface

The first module was the user interface of the table. From the sketches created for this interface, three interface concepts were created (images 3.7 3.8 & 3.9). In order to create a basis for a choice for one of these concepts, a simple prototype of the interface was created

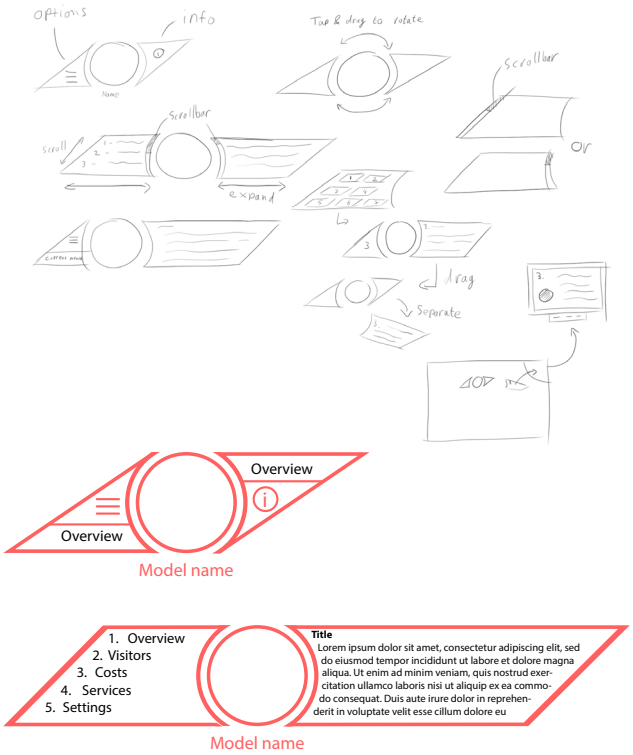


Image 3.7: VR-table interface concept 1

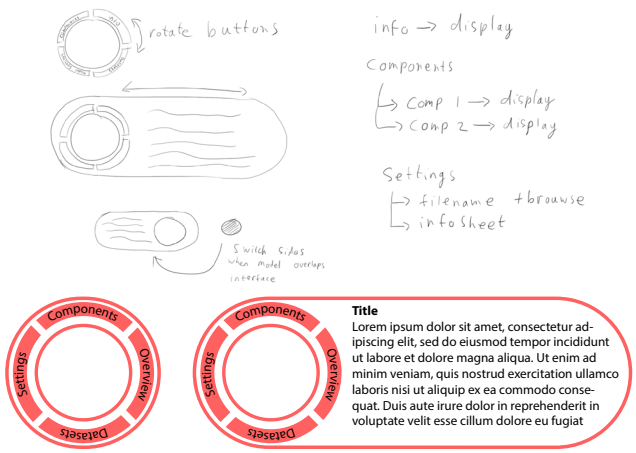


Image 3.8: VR-table interface concept 2

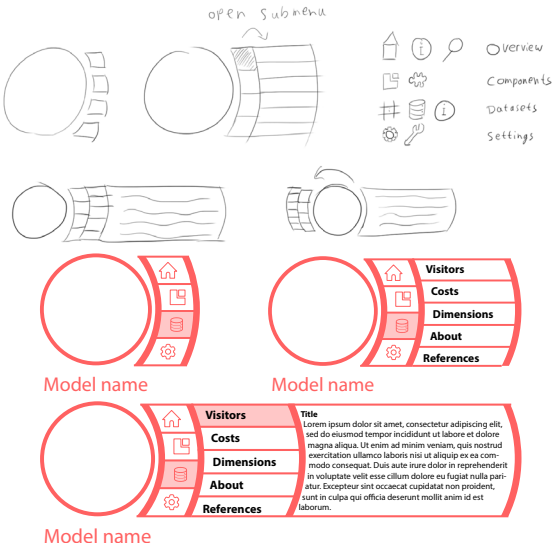


Image 3.9: VR-table interface concept 3

in Intuiface[2]. Intuiface is a program that allows for the creation of simple user interfaces for several platforms, most notably the VR-Table. The Intuiface file and the questions and answers for this usage test can be found in appendix A. One key feature omitted from the usage test was the possibility to send a selected model or information to a peripheral device. This choice was made because of technical limitations. The final concept should include a way to send selected information to a user specified screen, for example by dragging this to an icon at the side of the table. The test subjects mostly showed a preference for the second interface, but also the third interface gathered positive reactions.

After this test the choice was made to use the second interface. This was for two reasons: firstly because most test subjects preferred this interface, and secondly because the third interface takes up quite some space next to the tagged object. As stated earlier, the interface should be as compact as possible. Some improvements were made to the interface design after the usage test, such as inverting the text. This was done because the tagged object obscures the top menu, which was the only

2 *Simple interface creation software: http://www.intuilab.com/*

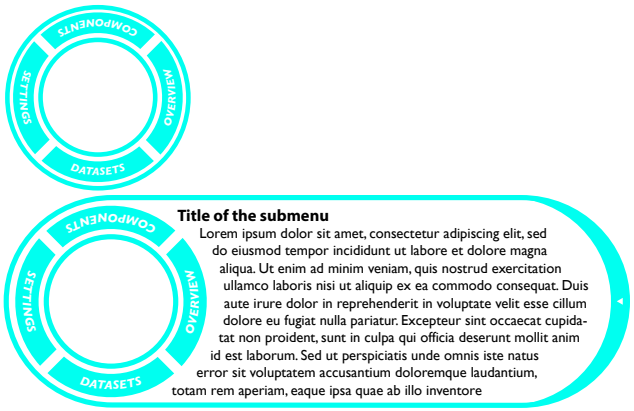


Image 3.10: The chosen user interface.
Top: closed user interface.
Bottom: open user interface.

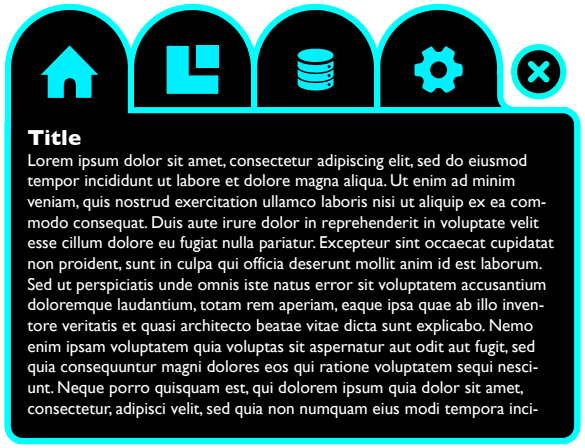


Image 3.11: Final device-UI concept

menu with the text right side up. The colours were changed to give the interface a more futuristic look. The final interface concept can be found in image 3.10.

3.2.2 - Screen user interface

The user interface for the secondary screen was the second module. This interface should display the virtual version of the tagged objects and its information. A usage test was not performed for this interface, because creating a functional prototype would require much time. As this interface should integrate with the interface of the VR-table, the same category division was made. The settings menu was added fields to set the 3D-model's import values, as the scale or orientation may differ per 3D-application. The two display methods explained earlier were included in this interface, allowing the user to enable or disable "Room mode" under settings. When not in room mode, the user gains control over the camera. The main features of this camera control are to orbit and pan by clicking and dragging and to zoom by scrolling. Different 3D-Applications require a different mouse button to be pressed, possibly in conjunction with a modifier key on the keyboard. For this concept mouse-only control was chosen, to allow users to control the camera with one hand and minimal effort. Left clicking and dragging orbits the camera, and right clicking and dragging pans the camera.

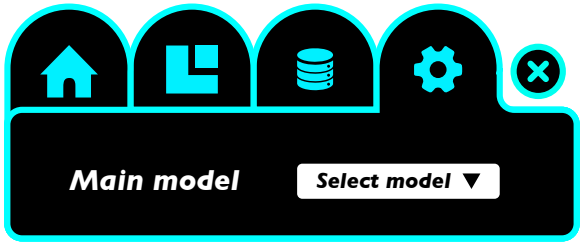


Image 2.12: Model selection menu

Users should also be able to select of which model they want to see extra information. Especially in free mode, the model the user wants to switch to may not be in view. Simply clicking on the model mesh is therefore not always an option. To make all models equally accessible, the user can select the model they wish to focus on from a dropdown menu.

3.2.3 - Technical structure design

To be able to realize the conceptual system, a technical design had to be made. One of the first technical challenges faced, was the exchange of information to the different devices. Having all devices be directly connected to the VR-Table highly limits the amount of devices

that can be connected to the system. Besides this, a physical cable is required to connect to the VR-Table's computer, which also limits the range of the system. On the flipside, if the secondary monitor would be connected to the VR-Table itself, it would have direct access to the files on the table, and thus the table could act as a central file location. However, this advantage is minimal when taking into account the modern day Internet speeds. Incorporating external devices, which are not directly connected to the VR-Table, would require the user inputted files to be synchronized to all devices, or to be downloaded from a central location. The best option would be to have a central server to manage all the users files and settings. Devices in the system could then download these files to use in their respective app. Position and rotation data of the tagged objects on the table can be stored in a database on this server as well.

In some cases it would be preferable not to have a standalone application on a device. When giving a demonstration for a group of guests for example, it would be preferable if all the guests could follow the demonstration on their own smartphone or tablet, without having to download a dedicated app for this purpose. In this case a webpage would be the ideal solution. The webserver would then have all the necessary files to provide output on the mobile screen, and the client phone would only have to process the users touch input. Because this kind of webpage is not essential to the minimal system, it is considered an extension module.

3.2.4 - N^2-chart

For the realisation phase an overview is needed of all elements in the system and how their relation is to each other. An N2-chart was created in order to determine what information should be exchanged between the different elements. The mobile app and webpage were included here. This was done so their needs can be taken into account when creating the base system. This chart can be found in image 3.13 and 3.14.

N^2-chart Room mode

User	Touch Tagged objects		Info request Camera settings	Info request Camera settings	Info request Camera settings
Visual feedback	**VR-table**	Position Rotation Tag id / finger			
	Model info Connected devices	**Server**	Position Rotation Model info	Position Rotation Model info	Image
Visual feedback		Device id	**PC**		
Visual feedback		Device id		**Mobile App**	
Visual feedback		Device id			**Webpage**

Image 3.13: N^2-chart Room mode

N^2-chart Free mode

User	Touch Tagged objects		Orientation Pan Zoom Info request	Orientation Pan Zoom Info request	Orientation Pan Zoom Info request
Visual feedback	**VR-table**	Position Rotation Tag id / finger			
	Model info Connected devices	**Server**	Position Rotation Model info	Position Rotation Model info	Image
Visual feedback		Device id	**PC**		
Visual feedback		Device id		**Mobile App**	
Visual feedback		Device id			**Webpage**

Image 3.14: N^2-chart Free mode

3.3 - Concept detailing

Before the realisation phase could begin, some detailing was done to improve the interface concepts. The VR-table-interface has some minor change in the colour of some buttons to improve the contrast. The Buttons also have an icon added to further differentiate them from each other. These icons are the same ones used in the interface on the peripheral screens. One major feature added to the VR-table's interface is the ability to duplicate a tag's interface by 'tearing' it from the tagged object. The user can then rotate this standalone version of the interface to make it readable for a user on the other side of the table. This was added to increase the omnidirectional possibilities of the VR-table. Finally, a simple graphic was made to add the possibility of having a tagged object act as the camera in the scene.

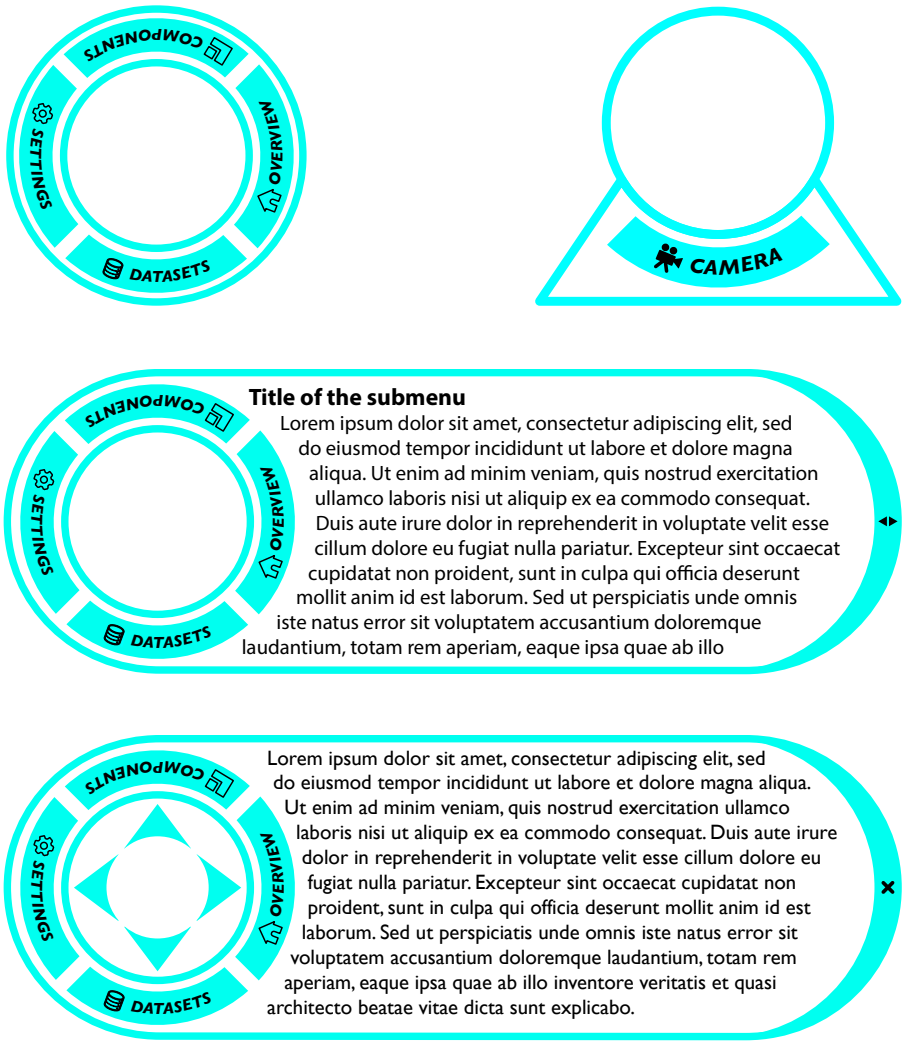


Image 3.15: The final interface designs

Chapter 4: Concept realisation

Once the concept was designed, the realisation of this concept could begin. During this phase some changes had to be made to the concept, due to technical limitations imposed by the hardware, software, and technical expertise on how program certain aspects of the concept.

4.1. - Environment

Most of the final system was created in Unity. The original intention was to create everything in this environment, however this was not possible due to a limitation in Unity. Unity (verson 5.1.1 at the time of writing) works in .NET framework 2.0, while the drivers used to get input from the VR-table's infrared cameras are compiled in .NET 4.0. This means that Unity is unable to compile programs that can directly control the VR-table's infrared cameras. Instead of looking for another environment to program in, a workaround was found to deal with this shortcoming (see 4.2.1 for a technical explanation). This choice was made for two reasons: firstly, Unity is a widely used and comprehensible program, so switching environment would limit the expansion possibilities beyond this project. Secondly, Unity, being a game engine, has a lot of standard functionality to create graphical user interfaces, handle 3D models, virtual cameras, lighting etc. Using a standard programming environment would require that these features be created manually.

4.2 - System overview

The final product consists of three computer programs, an SQL database and a third party file-synchronisation tool. Two of the three programs work in unison on the VR-table to gather user input on the VR-table and provide a graphical user interface. The third program is a standalone executable on other Windows PCs to provide a virtual-reality environment for the models and information. All programs are connected to the SQL database, which stores a unique id, the position and rotation of the input object placed on the VR-table, and whether this input device is a finger, tag or an unrecognisable 'blob'. In case a tag is recognised, the tag id is stored in the database. The possibility to send models or info to devices, directly from the table, and vice versa has been removed. This choice was made because it was technically too difficult to realize. All applications, as well as their source code and project files can be found in appendix B.

4.2.1 - OverlayApp

The functionality of the VR-table consists of two applications working in unison with each other. The first application is a transparent application (OverlayApp), which was created in Visual Studio. Visual studio is a programming environment capable of creating programs for the .NET 4.0 framework. This application is layered on top of the second application (TableApp) and drives the infrared cameras of the VR-table to capture user input. The raw input is processed to determine the type of object placed on the table (tag, finger or blob)

and the position and rotation of the object. This data is then sent to be stored in an SQL-database. Once an object is removed from the table, the corresponding entry is removed from the database. This way the database contains real-time data of all possible information of every object on the table. When this application is loaded, it truncates the database, and removes any entries that may have been there from before. This application was adapted from a demo application supplied by Microsoft for the VR-table[1]. Image 4.1 displays a screenshot of the application before modification. All possible info of an object on the table is also displayed. After modification, all graphical elements were removed.

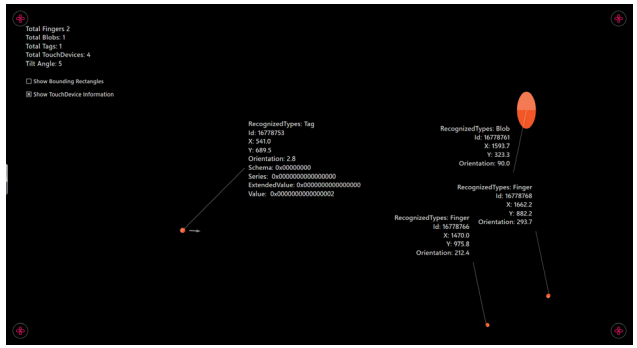


Image 4.1: Microsoft's demo application

4.2.2 - TableApp

The second application on the table (TableApp) is created in Unity and provides the graphical user interface on the VR-table. This application runs underneath the input application. Because the OverlayApp runs on top of this application, the TableApp has no direct access to input devices, such as the keyboard or mouse, however due to the intended touch surface, this was not needed anyway.

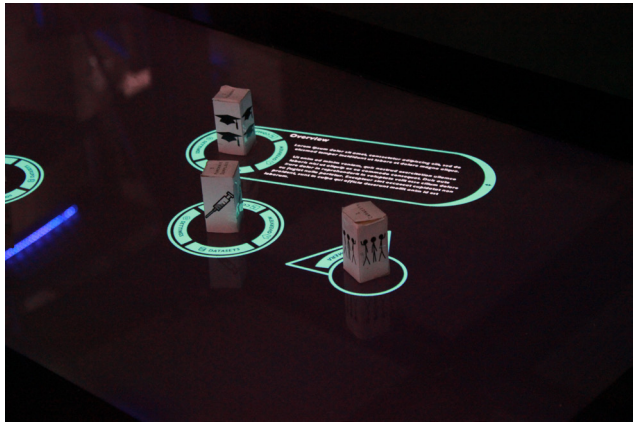


Image 4.2: Both apps running on the VR-table

The TableApp continuously reads all the entries from the database and uses these entries to determine what the user has input. For all read tags, the app will create interfaces related to these tags. Reading the position data of finger entries and calculating the difference from the previous frame simulates touch input. The Table app can load and read text files to provide extra information for the tag placed on its surface. Image 4.2 shows the final system running on the VR-table.

4.2.3 - PC app

The third application (PCApp) works as a standalone program on on one or more separate PCs. This application visualises a virtual reality version of the objects on the table using Collada 3D-files[2]. The PCApp is also connected to the database, and uses its entries to position the 3D-objects at the same position in the virtual world as in the real world. This app is also able to load the additional information provided with each model.

As stated earlier, Unity normally natively

1 *Found in the Surface SDK 2.0: https://www.microsoft.com/en-us/download/details.aspx?displaylang=en&id=26716*

2 *See: https://www.khronos.org/collada/*

supports a variety of 3D-file formats. However when programming the import of external 3D-models a problem was encountered. Unity is not able to load external recourses at runtime, and when compiling the standalone application all resources in the Unity project are compiled with this application. Because of this, the PCApp would be unable to allow the user to load a model at runtime. This problem was solved by using a third party plugin that can load Collada files at runtime[3]. However, the drawback of using this plugin was that it was only able to load the mesh from the file, and not the materials and textures.

4.2.4 - File synchronisation

In order to synchronize the loaded text and model files, the third party application BitTorrent Sync[4] was used. The TableApp also acts as the database for all text and model files, and by syncing the folder containing these files to a PC, the PCApp has access to the same files as the TableApp.

4.3 - Class scheme

In the following paragraphs the technical setup of the code will be explained. Because the OverlayApp is a slightly modified and stripped version of a demo application supplied by Microsoft, and the code to write entries to the database is very similar to the code to read entries from the database, this application will not be described in depth. The exact code and Unity projects can be found in appendix B of this report.

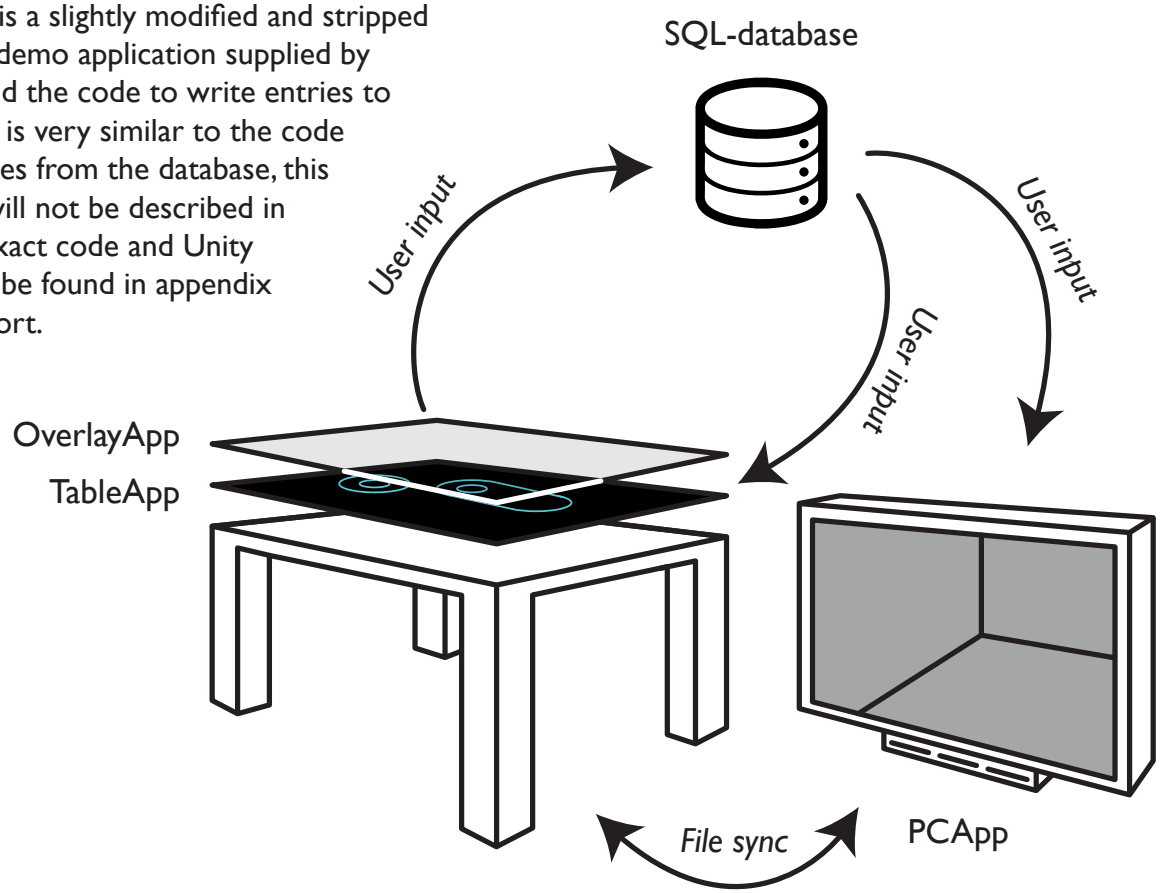


Image 4.3: System overview

3 *SimpleCollada: https://www.assetstore.unity3d.com/en/#!/content/19579*

4 *File Synchronization tool: https://www.getsync.com/*

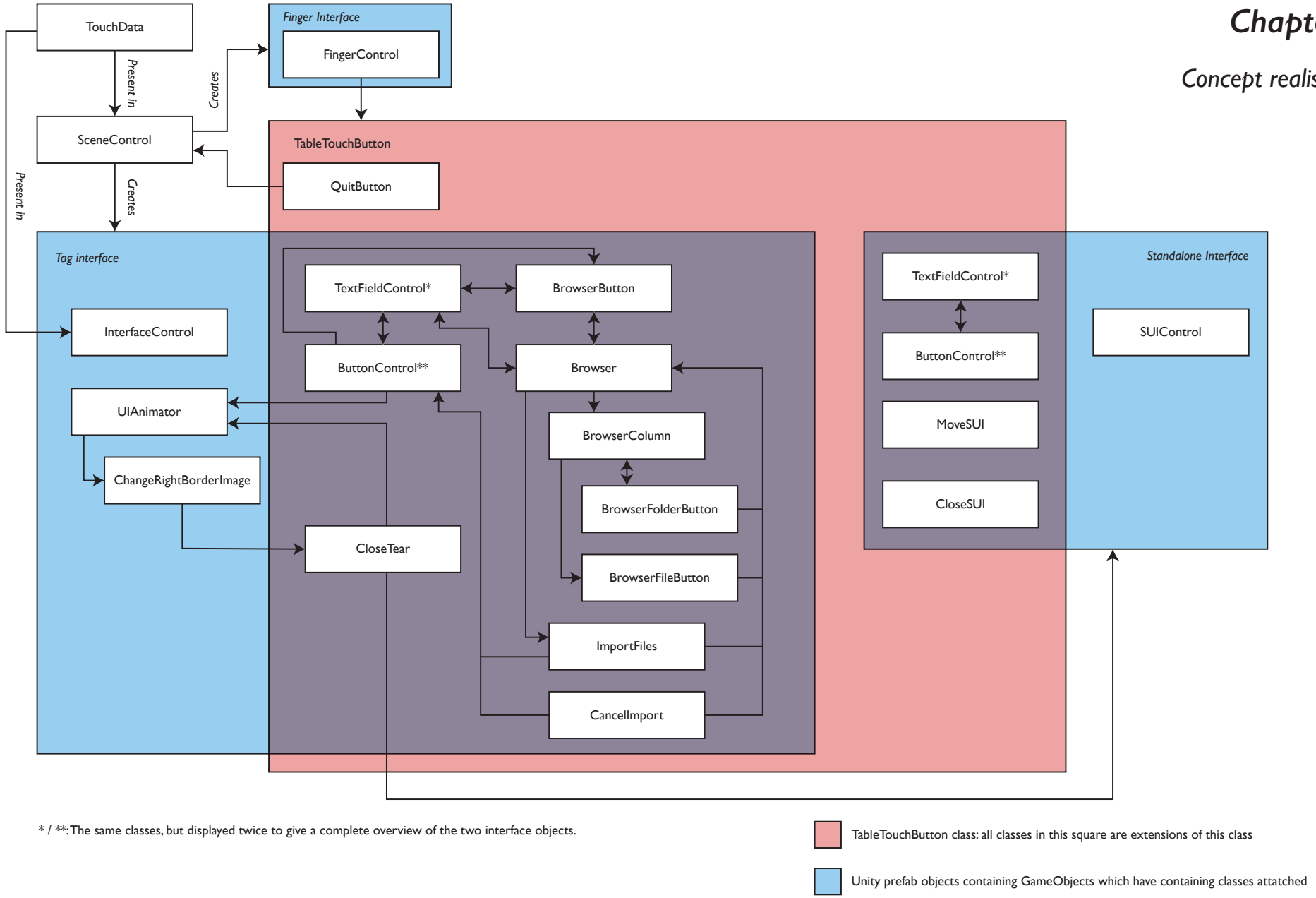


Image 4.4: Class Scheme TableApp

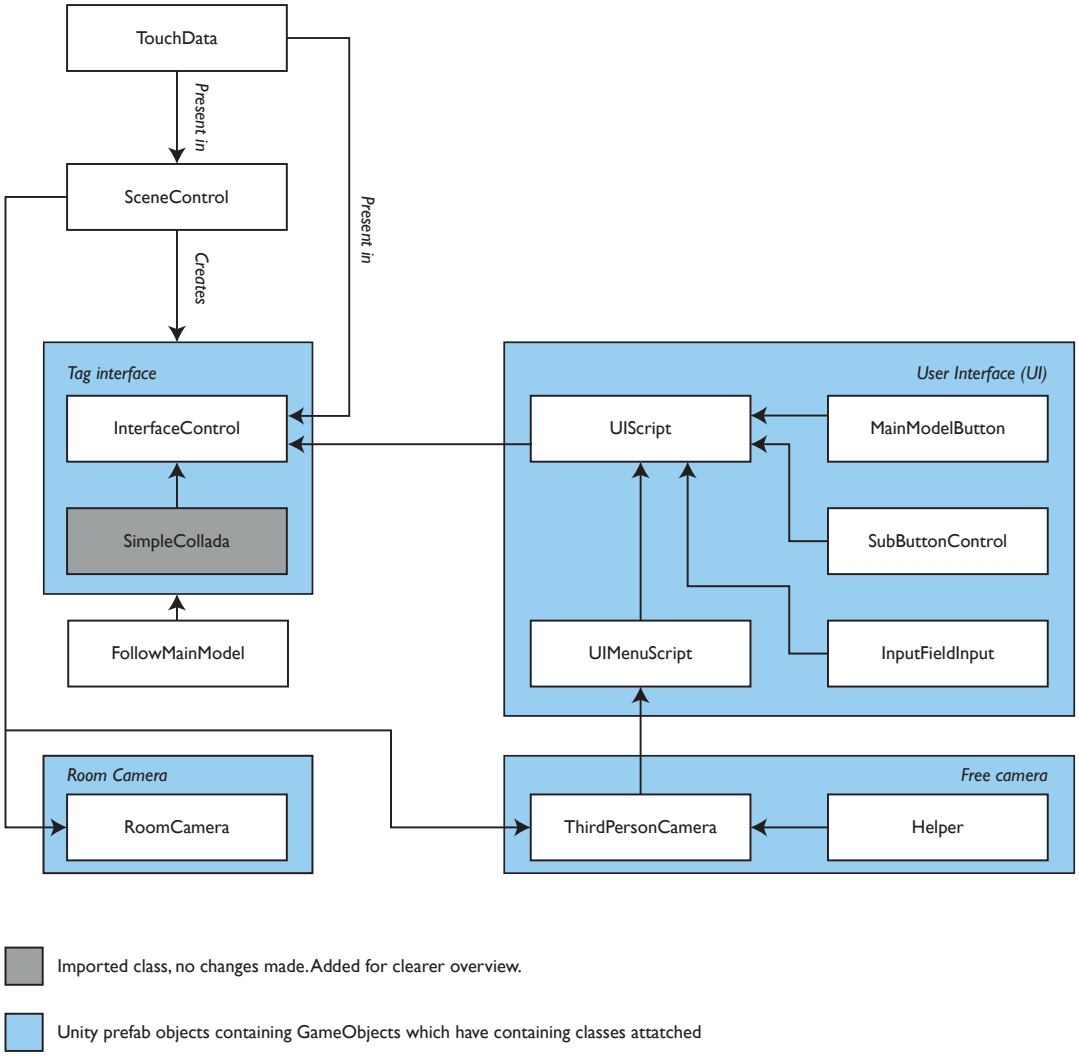


Image 4.5: Class Scheme PCApp

4.4 - TableApp

First the classes of the TableApp will be discussed. Because the OverlayApp was created out of necessity, minimal functionality was added. The TableApp processes all input data and creates output for the user.

4.4.1 - Touchdata

An instance of the TouchData class stores the information associated with one entry from the database. This class has some minor methods such as one to check whether or not the entry is a tag and one to transform the position in table space to a position in Unity's camera space.

4.4.2 - SceneControl

The Scenecontrol class handles connecting to and reading the database, and creating the appropriate objects associated with the database entries. It also starts an instance of the OverlayApp so users won't have to do this themselves. There are two lists maintained by this class: one that contains the raw entries from the database, and one that contains all objects associated with the database entries. By comparing these, the application is able

to determine if an object has been added or removed from the VR-table's surface. When the app is closed the database is truncated.
The database is cleared at two times: when the application is started and when the application is exited. The first is done as to ensure the input app starts without residual entries from a last session, or manually inserted into the database between sessions. If clearing on exiting the app would not be done, and there would be an entry at the position of the close button of the application, the application would close immediately after being opened a second time.

4.4.3 - InterfaceControl

For every entry that is read by the SceneControl a premade TagInterface-object is loaded. There are three different types of interface-objects: one for tags, one for the tag that is assigned as the camera, and one for fingers. This object contains images, buttons, text fields and other components necessary to make the interface function. At the top of this hierarchy is the InterfaceControl. This class contains an instance of the TouchData class, where, among others, its position, rotation and tag id are stored. Upon initialization, it creates the folder structure on the hard drive for this tag where the info and model files will be held. Some code was added to create the functionality to automatically rotate the interfaces when they overlap each other. In the end this was too difficult to realize in a reliable manner, and thus this functionality was removed.

4.4.4 - TableTouchButton

Because the table app doesn't use Unity's default input functionality, but derives input from the database entries, Unity's default touch input modules could not be used. Instead, the TableTouchButton class was created as a base class to handle touch input. All classes that require handling touch input extend from this class.
Because it is a base class, it only has methods that are called when the user taps on the table, drags on the table or a drag on the table ends. In child classes these methods can be overridden to add functionality specific to each button. Although this class acts as an event handler for the fingers, the class FingerControl handles the triggering of these events. Upon the creation of an instance of the FingerControl class, it checks if its position is inside the bounds of any of the TableTouchButtons present. If it is, it will trigger the TableTouchButton to track the fingers position. If the position stays within a certain value and is then removed, this will trigger a Tap event. If the position exceeds this value, a Drag event is triggered. Classes of the TableApp that are extensions of this class will be referred to as buttons.

4.4.5 - FingerControl

Entries with their tag id set to "Finger" will trigger the creation of a FingerInterface-object. This object has the InterfaceControl component attached to regulate its position and rotation, and it also has the FingerControl class attached. One of the functions of this class is to calculate the difference in position and rotation from the previous frame, in order to determine the amount of movement of the finger. Besides this it is the event trigger for the TableTouchButtons. When this class is initialized, it will check whether it is in bounds of any TableTouchButton. If it is, it will trigger the affected TableTouchButton to handle the finger's input action.

4.4.6 - ButtonControl

This one class primarily controls the four buttons on the outside of the interface. This class is an extension of the TableTouchButton class, and by overriding the OnDrag and OnTap methods, the four buttons were made responsive to touch input by the user. By

dragging, the four buttons rotate around the centre, making them easily accessible for the user. By tapping, the class first checks which of the four buttons is pressed, and then processes the appropriate action.
Besides these four buttons, it also handles the buttons of the submenus for components and datasets. This is because these buttons behave in a similar fashion to the overview button, by calling the TextfieldControl to read and display the corresponding text file.

4.4.7 - UIAnimator & ChangeRightBorderImage

Once a button is pressed, the interface will expand to allocate space for more information. The animation itself is handled by Unity's built-in animation system, but the change of animation states is handled by the class UIAnimator. ChangeRightBorderImage handles the (de) activation of the image and button on the right side of the interface (see 4.4.15 CloseTear).

4.4.8 - TextFieldControl

This class handles the text that is displayed on the screen. It has methods to read text files from the hard drive, and process this file to produce the text visible on the table surface. In order to make the text wrap around the circular image, the text is divided into lines. Each line is offset in the x-direction according to its y-position using a cosine function. Besides the creation of the text fields, this class also scrolls the text when the user drags on the text. This scrolling is limited to ensure the text remains in the bounds of the interface. Because of this response to touch, the TextFieldControl is also an extension of the class TableTouchButton.

4.4.9 - BrowserButton

The TableApp has a built-in browser to load models and text files into the system. To activate this browser, the user taps on the browse file button, found in the settings menu, or in the respective menus of Overview, Components, or Datasets, if they have no files loaded. Tapping this button will call the OpenBrowser method in the Browser class, and pass it the information needed to process importing file(s), such as the destination, file type and whether or not multiple files are allowed to be imported.

4.4.10 - Browser

The main controller of the browser is the class Browser. This class has methods that can be called to open and close the browser. It also has a method that draws the browser. This method is called when the browser opens, but also when the user taps on a folder (see 4.4.12 BrowserFolderButton). It also keeps track of which files the user has selected.

4.4.11 - BrowserColumn

A browser column displays the content of one folder. This includes all subfolders and all files of a specific file type (.txt or .dae). The current folder is the rightmost column, with all parent folders up to the root of the drive on its left. The BrowserColumn class is an extension of the TableTouchButton class. When a user drags on the column, it will scroll either horizontally by moving the columns, or vertically, by moving the subfolders and files in this column up or down. Scrolling is limited to ensure the subfolders and -files stays within the bounds of the interface.

4.4.12 - BrowserFolderButton

When tapping on this button, the browser is redrawn with the path associated with this specific button. This works similar to opening a folder in a regular file explorer.

4.4.13 - BrowserFileButton

Tapping on this button is similar to clicking on a file in a regular file explorer. This will highlight

the file to mark it as selected and add it to the file list in the Browser class. In case the selection of multiple files is not possible, a new selection will clear this list and deselect the other buttons.

4.4.14 - ImportFiles & CancelImport

Once the file or files are selected and the button ImportFiles is tapped, all files in Browser's import list will be copied to the destination set when the browser was opened, overwriting possible existing files. In case the browser was opened by mistake, tapping CancelImport will close it again.

4.4.15 - CloseTear

When the interface is expanded, a button on the right side will appear. This button can either be tapped on or dragged on. Tapping on this button will cause the interface to be closed. When dragging here, a standalone interface will be created. This is a clone of the normal interface, but is able to exist independent whether or not the tag it is related to is placed on the table.

4.4.16 - SUIControl

Because a standalone interface is not directly linked to a tag on the table's surface, the class InterfaceControl cannot control it. It is instead controlled by the class SUIControl. This class handles the position, and stores which tag it is an instance of. It also controls the interface's rotation. Because this rotation cannot be driven by the tag's rotation, the user can control this rotation by using two fingers. One finger is the rotation centre, and dragging the second finger inputs the amount of rotation.

4.4.17 - MoveSUI & CloseSUI

To move the standalone interface, a button was added in the area normally obscured by the tag. Dragging on this button will cause the interface to follow the active finger. To close a standalone interface, the CloseTear button on the right side of the interface was replaced with a close button. The CloseSUI class removes the interface when this button is tapped.

4.4.18 - Quitbutton

The Table app operates on the entire surface of the table. This means the regular close button of a windows application is not visible, and even if it were, it would not be accessible due to the overlay app. Because of this, the TableApp has a quit button, which closes both the OverlayApp, and the TableApp.

4.5 - PCApp

The PC App uses the same classes as the TableApp on the base level. In this case they will not be mentioned here again. Some clasess have been slightly modified, but are mostly the same as in the TableApp. In this case only the modifications will be discussed.

4.5.1 - SceneControl

This class maintains its function of reading and processing the database entries. Loading the overlay-app was removed and the of management of two cameras was added: one for room mode, where the camera is a tag on the table, and one for free mode, where the user has control over a third-person camera. These two cameras have their own classes to control their behaviour.

4.5.2 - InterfaceControl

This class has added functionality to load Collada 3D-files. To achieve the loading of 3D-files at runtime, a third party plugin was used. This plugin came with a demo file, and

most of the code used to load the model from the hard drive was copied from this demo. Only slight modifications were made to add a default material. Functionality to read import settings from a separate text file was also added. This may be needed to adjust for the difference in scale factor or axes between 3D-applications (see 4.5.10 InputFieldInput).

4.5.3 - RoomCamera

Because the RoomCamera is linked to a tag, this class is an adaptation of the InterfaceControl class. It no longer needs to load a model, so this functionality is removed. Instead, it sets the camera's default position if no camera tag is detected. If the camera tag is removed from the table, the SceneController resets the camera to this position. The class loads settings for the camera from a file on the disk, most notably the vertical offset from the floor.

4.5.4 - ThirdPersonCamera & Helper

Besides the possibility of having a model on the table act as the camera, there is also a free mode, in which the user has control over a third person camera system. The control for this class is centred in the ThirdPersonCamera class, with some additional methods in the Helper class. These classes were taken from Simons (2012). An adjustment was made to enable the user to pan the camera besides orbiting and zooming.

4.5.5 - FollowMainModel

This class is attached to the free camera's target. It ensures that when the model is moved on the table, the target, and thus the free camera, moves along with it. This way the free camera will always maintain the same relative position to the main model.

4.5.6 - UIScript

This class is the main control of the user interface of the PCApp. It handles the model selection, menu display, text file loading and display, and controls most buttons. Part of the user interface event handling is managed by Unity's user interface event system. Where this event system doesn't have enough functionality, this class provides additional event handling.

4.5.7 - UIMenuScripts

This script manages the creation of the dropdown menu used for the model selection in combination with Unity's user interface event system. It creates a button for each interface, except the camera, sorts this list and resizes the panel containing all the buttons.

4.5.8 - SubButtonControl

When selecting Components or Datasets, the user has to select a file to load. This class contains the code to load the file associated with the button that has been pressed.

4.5.9 - MainModelButton

This class is attached to the buttons in the dropdown menu for model selection. It sets the corresponding main model when the button is pressed.

4.5.10 - InputFieldInput

Carrying over 3D-models from one program to another can cause some problems. Common problems are 90 degree rotation due to one program using the z-axis for up instead of the y-axis, or a different scale factor due to different units being used. To solve this and possible other positioning problems when importing the Collada file, fields were added in the settings menu to enable the user to translate, rotate and scale the model. This class processes the input and writes this to a file to save the settings. This way removing and replacing the tagged object from the table's surface doesn't require the user to re-input the offset values. Unity's user

interface event system handles the reloading of the model when a value is changed.

4.6 - Final system

In the following paragraphs a short explanation will be given, to allow users to quickly get started using the system.

4.6.1 - Setup

The final TableApp folder contains the following files and folders:

- Start.bat: use this to launch the Table app.
- TableApp.exe: the TableApp.
- TableApp_Data: folder containing the data files for the Unity app
- Several folders containing plugins and other app data.
    - Resources: the folder containing the resources for the app
    - TableInput: the folder containing the OverlayApp, its plugins and the database configuration file.
        - User: the folder containing the subfolders and data specific to each tag.

The PCApp folder contains the following files and folders:

- PCApp.exe: The PCApp, use this to launch the PCApp.
- PC_App_Data: the folder containing the data files for the PCApp
    - Several folders containing plugins and other app data.
    - Resources: the folder containing the resources for the app
        - DatabaseConfig.txt: the file containing the login data for the database.
        - User: the folder containing the subfolders and data specific to each tag.

4.6.2 - Getting started

Use BitTorrent Sync to sync the folder "User" in "TableApp_Data\Resources" to the corresponding folder in "PCApp_Data\Resources". This has to be done for every computer running an instance of the PCApp.

To start the TableApp run the program Start.bat. This program runs the Unity application without the window borders. When the TableApp starts, it automatically loads the OverlayApp, so the user won't have to start two programs.

Wait a few moments to ensure the OverlayApp is loaded correctly. Place a tagged object on the table's surface, an interface should appear.

If the TableApp is not responding, this could be due to the OverlayApp being displaced. Check this by pressing alt+tab on the keyboard to make sure TableToSQL.exe is the most left in the row, and TableApp.exe is the second most left.

Use the built in browser to load text and Collada files for tags under settings or the respective menus. This can also be done manually in Windows explorer, but make sure to only have one text and Collada file in the tag's root folder. These files have to be synced with the PCApps, which may take a moment. When loading a model, this model has to be refreshed by temporarily removing and replacing the tagged object on the table's surface.

4.6.3 - Tag usage

The application can recognise Byte tags[5]. Tag 0x01 is reserved for the Camera. An exception may occur when the OverlayApp insufficiently reads a tag and marks the tag id as 0. For this reason tag 0x00 should not be used.

5 *A sheet containing all byte tags can be downloaded here: http://www.microsoft.com/en-us/download/details.aspx?id=11029*

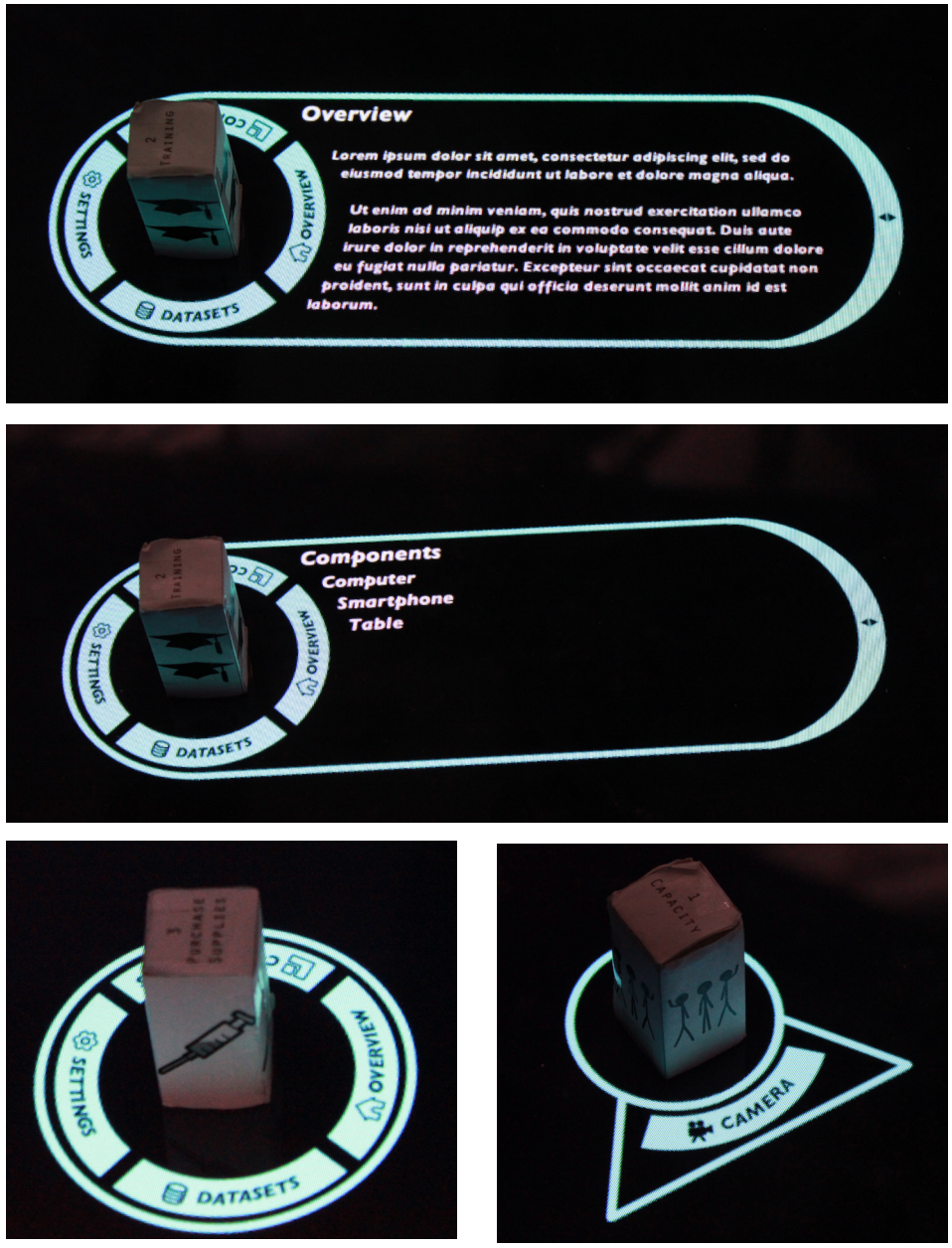


Image 4.6: Several pictures of the final system running on the VR-table.

Top left: Displaying text.
Top Right: Settings menu.
Middle Left: Submenu.
Middle Right: File browser.
Bottom Left: Closed Interface.
Bottom Right: Camera interface.

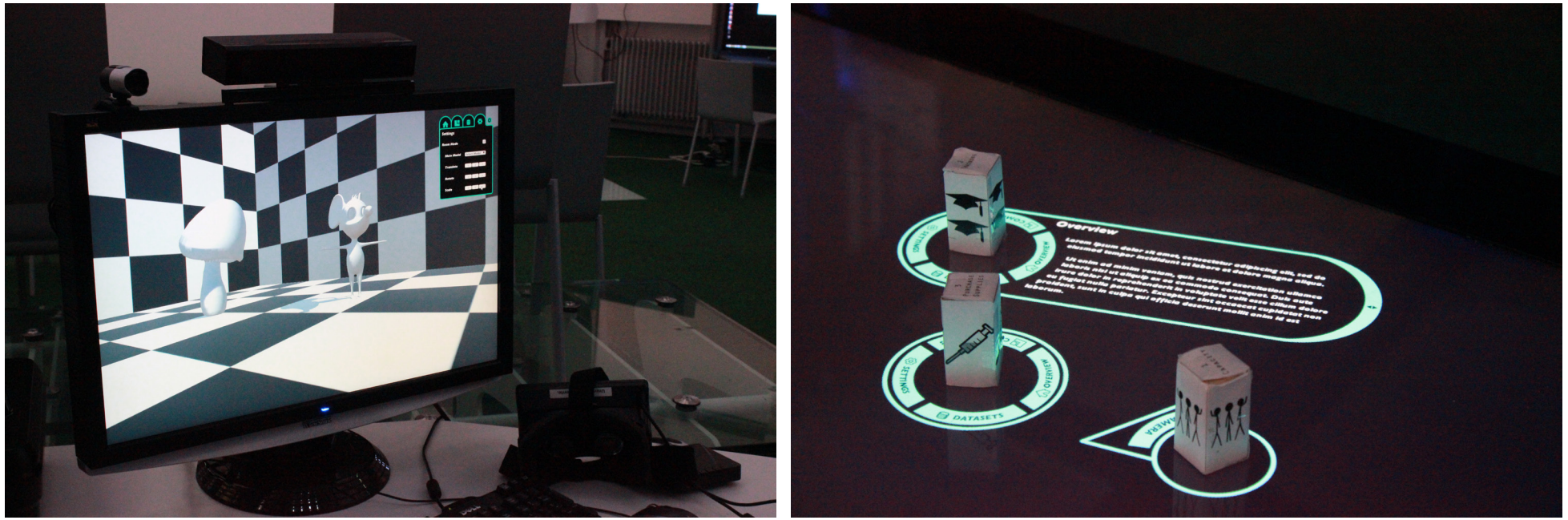


Image 4.7: Picture of the final system running on a PC.

Left: PC output.
Right: Configuration of tagged objects on the table.

Note the configuration of the tagged objects is the same as the configuration of the virtual objects in the PC output.

Chapter 5: System evaluation

After the programming and testing of the system was completed, it was evaluated. Several recommendations were formulated based on problems that were encountered during the realisation phase, and the final usage test.

5.1 - Usage test

To test if the user interfaces of the two applications work in an intuitive manner, a usage test was performed. In this test several users were asked to perform certain actions. The results of this test can be found in appendix C. Several improvements to the interfaces were formulated based on this user test. These improvements were incorporated in chapter 5.2 - Recommendations. This test was also used to evaluate the recommendations specified in chapter 2.6 - Requirements.

5.2 - Requirements

After the usage test was completed, a test was performed to check if the system met all requirements set in the Analysis phase.

5.2.1 - Use

- *The new system should be usable without programming knowledge.*
  This requirement has been fulfilled; no programming knowledge is required when operating the system.
- *The new system should be able to separately display information according to relevant field of expertise.*
  This requirement is fulfilled, the system allows users to display different text and models on different devices.
- *The new interface should be intuitive to use.*
  Based on the results of the usage test, this requirement is considered fulfilled.
- *Limited customisation should be required of the users' CAD-models in order to be usable with the system.*
  The system is able to read Collada files. Most CAD-software is able to export in this format, and thus this requirement is considered fulfilled.
- *The system should implement a natural user interface.*
  Use of the VR-table requires no mouse or keyboard, which is in line with a NUI. Only the PCApp requires the use of a mouse. An improvement would be to replace this mouse with a form of NUI control (see 5.3.3 Motion Controller).
- *The system should show simple information at first, but this information should expand if the user wants to.*
  The system makes use of several sub menus to access information. This requirement is therefore considered fulfilled.

5.2.2 - Look and feel

- *The new system should have a futuristic look.*
  Compared to several futuristic looking images such as those seen in the collage in chapter 2.5 - Collage, this requirement is considered fulfilled.
- *The system should make the user feel highly in*

control of the information and models.
Based on the usage test this requirement is considered fulfilled.

- *The new system should achieve a complex feel while maintaining an intuitive control.*
  Based on the usage test this requirement is considered fulfilled.

5.2.3 - Features

- *The new system should be able to connect with multiple computer screens.*
  The system stores user input in a central database. This database can be read by multiple computers, and therefore this requirement is fulfilled.
- *The new system should be easily expandable to accommodate for new technologies.*
  The modular build-up of the system allows for easy expansion in the future. This requirement is therefore considered fulfilled.
- *The new system has to be usable with existing 3D-software and -files.*
  The system makes use of Collada files. This is an existing file format, and many 3D-applications are able to export these files.
- *The new system has to be programmed in Unity.*
  This requirement is partially fulfilled. Due to Unity's limitations a part of the system was programmed in visual studio (see 4.1 - Environment).

5.2.4 - Wishes

- *The new system should be able to automatically link real world objects to their digital counterpart.*
  This wish was not implemented in the final system.
- *The system should automate the CAD-file conversion to a Unity native file.*
  This wish was not implemented in the final system.

5.3 - Recommendations

Although the final product allows the user to work with the interface without many problems, there are still some improvements that can be made. These improvements were not possible in the currens system, due to technical reasons, lack of programming knowledge or lack of time.

5.3.1 - One TableApp

One of the biggest drawbacks of the system has already been addressed in this report. This is the necessity of having two applications running over each other on the table. Not having the correct order of the applications, or popups interfering with this order is the biggest cause for unwanted behaviour. Merging these applications would be the most ideal solution. Another solution would be to automatically switch the window to the overlay-app when the Unity app is set as the main window. The program also has to be started through a .bat file to remove the borders of the TableApp. This is necessary because two full screen applications cannot run over each other. Merging the two apps or disabling the window borders in the TableApp's programming can solve this problem.

5.3.2 - Central file database

Another big drawback is the need to synchronize the "User" folder to all PC's with the use of a third party application, in this case BitTorrent Sync. A better solution would be to

have a central file database where all text and model files are stored, and have each instance of both the TableApp and the PCApp download the content of these files directly from this server. Collada stores its data in a plain text format, so streaming this data should be no problem.

5.3.3 - Refresh when file is changed

When a Model or text file is loaded on the VR-Table, it takes a while to synchronize to the PCs with the PCApp. Even when the sync is complete, the user has to manually reload the model or text file. Detecting when the sync is complete, or when a file stream from a central database is complete could be used to trigger the reloading of models and text.

5.3.4 - Enhancing Collada import

The current PCApp uses an external plugin to load Collada files at runtime. This plugin is however unable to load materials and textures. The plugin is provided with the source code, so adapting to include this is possible. This would however require a more in depth analysis of the code of the plugin and the Collada file format, and was thus omitted for this project.

5.3.5 - Embedded info in 3D-file

Separate text files are used to provide information on the model. This requires the user to keep track of which text files belong to which model. If the information would be added as metadata to the 3D-file, the info would be linked to the model through the file. The system should then be adapted to enable it to extract the information from the 3D-file, instead of reading separate text files.

5.3.6 - Overwriting files on import

When importing files using the TableApp, files with the same name will be overwritten or deleted without warning. The same will happen when importing a 3D-model or overview file, if there is already one present. Adding a check and either moving these files to the recycle bin or asking the user if they want to overwrite the existing file would be a useful extra feature.

5.3.7 - Integrating devices

Currently the user has to select a main model in the PCApp itself. Allowing the user to select information and the main model directly on the VR-Table for a PCApp would further integrate the two apps. To enable this, the two apps would have to communicate with each other directly or through a separate database, and require distinguishing each instance of the PCApp. For this reason this feature was omitted from the current system.

5.3.8 - Standalone models

On the TableApp it is possible to create an interface that is able to exist, without the tag it is linked to being placed on the table. Adding similar functionality to the PCApp, would be an improvement to the system. Especially in free mode it would be useful to be able to view the model, independent of the tagged object.

5.3.9 - Overlapping interfaces

On the TableApp, interfaces are not aware of each other. This entails that when interfaces overlap, their text may become unreadable. A possible solution could be to rotate the interfaces once they overlap. This was tried in this project, but it soon became apparent that the amount of coding and calculations required enabling rotation in the correct direction and correct amount was too complicated for this project.

5.3.10 - Environment loading

The current PCApp has a default checkerboard pattern for the floor and walls. Allowing the user to load custom images for the environment

will improve the possibilities. The same goes for the background image on the TableApp, which is now a solid black colour. The table's infrared cameras can pick up reflections from bright pixels on the screen, which can impede the tag and finger detection. This should be taken into account when enabling users to load their own background image.

5.3.11 - Room camera settings

The current room camera imports certain settings, such as its height from the ground and focal length, from a text file. This file is however not editable from within the application. Adding functionality to access these settings through the user interface of the PCApp would be a convenient improvement.

5.3.12 - Improving the browser

The test subjects performing the usage test had some problems with using the system's built in browser. Several improvements can be made to the browser to make it more similar to a browser on a computer. This can possibly be done by highlighting the current folder in the hierarchy and adding icons to differentiate between a file and folder.

5.4 - Possible expansions

The current system was designed with the possibility to expand its functionality. Currently the only peripheral device being used is a PC, which runs the PCApp.

5.4.1 - Other Peripheral devices

Besides using a PC as a peripheral device, a mobile version of the PCApp could be created for use on smartphones or tablets. This would require a different file management and syncing system to accommodate for the data structure of smartphones.
Besides a dedicated app for mobile users, a web version could also be created, so users wouldn't have to install an app on their phone. This is useful for demonstration purposes, so guests can see the functionality on their own device, without having to install an app.

5.4.2 - Oculus rift

Adding oculus rift support to the PCApp will add the possibility to experience the 3D-models in 3D-perspective. This would immerse the users even more in the experience, and shift the system more towards a NUI.

5.4.3 - Motion Controller

When a leap motion sensor is used to track the users hands, the now mouse-controlled third person camera could be removed, and orientation and position of the model can be controlled by the users hand gestures instead. This would shift the system more towards a NUI.
Besides this, a Motion controller such as a Kinect could scan all real world objects on the table and automatically link them to their virtual counterpart.

5.4.4 - Tagged objects

The current system allows the use of simple static tagged objects. Adding microelectronics to the models themselves could expand the possibilities of the system. This would add a tactile interface to the objects, so model specific input can be provided.

References

The following References were used for the realisation of this project.

Barnes, M., Levy Finch, E., Sony Computer Entertainment Inc. (2008) COLLADA – Digital Asset Schema Release1.5.0 Specification. Retrieved from: https://www.khronos.org/collada/

Buckit Media (2015). Will Intel's RealSense finally change the user interface? *Buckit Media: Retreived from: http://buckitmedia.com/will-intels-realsense-will-finally-change-the-user-interface/*

Griffiths, S. (2013). Hyperloop creator invents incredible Iron Man-inspired 3D software system to design and print ROCKET PARTS. *Daily Mail: Retreived from: http://www.dailymail.co.uk/sciencetech/article-2414005/Hyperloop-creator-invents-Iron-Man-inspired-3D-software-design-ROCKET-PARTS.html*

Microsoft Coorporation (2012). Microsoft Surface 2.0 SDK. *Retreived from: https://www.microsoft.com/en-us/download/details.aspx?displaylang=en&id=26716*

Orbcreation (2014). SimpleCollada. *Orbcreation: Retreived from: https://www.assetstore.unity3d.com/en/#!/content/19579*

Several Authors (2015). Natural user interface. *Wikipedia: Retreived from: https://en.wikipedia.org/wiki/Natural_user_interface*

Simons, T.V. (2012). Vrije Opdracht BitBot. *Project Report, University of Twente, Enschede.*

Thalen, J.P. (2013). Facilitating user centred design through virtual reality. *PhD thesis, University of Twente, Enschede.*

Weidlich, D., Cser L., Polzin, T., Cristiano D., Zickner, H. (2009, March 26). Virtual reality approaches for immersive design. *International Journal on Interactive Design and Manufacturing 3 (2009), No.2, pp.103-108.*

Wood, J. (2010), Unity and MySQL. *Retreived from http://forum.unity3d.com/threads/unity-with-mysql.63364/*

Appendix A

In this appendix the first usage test can be found. It includes the questions and answers, as well as the Intuiface project used to perform the test.

A.1 - Intuiface project

The Intuiface project used to preform the first usage test can be found in appendix A.1.

A.2 - Usage test 1

This usage test aims to find the best user interface out of three concept interfaces. Three Models are used in this usage test: one of a church, one of a ghost, and one of a farm. The test was performed by asking the users to find certain (placeholder) information about these objects. The following tasks and questions were given to the test subjects:

Interface 1

- Where do you look up an overview of churches?
- Where do you look up how many people visit the church each year.
- You notice that instead of information about churches, information about farms is displayed. Where do you fix this problem.

Interface 2

- Where do you look up an overview of farms?
- Where do you look up how many buildings the farm has?
- You notice that instead of information about farms, information about ghosts is displayed. Where do you fix this problem.

Interface 3

- Where do you look up an overview of ghosts?
- Where do you look up information about the history of ghosts?
- You notice that instead of information about ghosts, information about churches is displayed. Where do you fix this problem.

Questions:

- Which Interface was the most intuitive?
- Were there any problems using one or more interfaces?
- Did you miss any functionality in any of the interfaces?
- Did you have any trouble understanding the icons?
- Any other comments or problems you encountered?

A.2.1 - Test Subject 1

Interface 1

- *Where do you look up an overview of churches?*
  The left side, this would show options on a smartphone.
- *Where do you look up how many people visit the church each year?*
  Also on the left side.
- *You notice that instead of information about churches, information about farms is displayed. Where do you fix this problem.*

Under settings, also in the left menu.

Interface 2

- *Where do you look up an overview of farms?*
  This is done by pressing 'Overview'.
- *Where do you look up how many buildings the farm has?*
  This is done by pressing components and counting them, or under datasets.
- *You notice that instead of information about farms, information about ghosts is displayed. Where do you fix this problem.*
  This is done in the settings menu.

Interface 3

- *Where do you look up an overview of ghosts?*
  This is done by pressing the 'Home' icon.
- *Where do you look up information about the history of ghosts?*
  This is done in either the second or third menu, this is not clear right away.
- *You notice that instead of information about ghosts, information about churches is displayed. Where do you fix this problem.*
  Under settings (the bottom option).

Questions:

- *Which Interface was the most intuitive?*
  The second interface is the easiest to understand.
- *Were there any problems using one or more interfaces?*
  The first interface shows the least options, this is the hardest to understand. The icons on the third interface were not immediately clear.
- *Did you miss any functionality in any of the interfaces?*
  The
- *Did you have any trouble understanding the icons?*
  In the first interface I would say that the left menu is similar to a smartphone, the information icon is not really clear, because to me it seems like a help menu.
  In the third interface the second and third icon are not very clear.
- *Any other comments or problems you encountered?*
  Not anything that has not yet been said.

A.2.2 - Test Subject 2

Interface 1

- *Where do you look up an overview of churches?*
  The right side, this would show informaton.
- *Where do you look up how many people visit the church each year?*
  Also on the right side.
- *You notice that instead of information about churches, information about farms is displayed. Where do you fix this problem.*
  Under settings, the left menu.

Interface 2

- *Where do you look up an overview of farms?*
  This is done by pressing 'Overview'.
- *Where do you look up how many buildings the farm has?*
  This is done by pressing Components.
- *You notice that instead of information about farms, information about ghosts is displayed. Where do you fix this problem.*
  This is done in the settings menu.

Interface 3

- *Where do you look up an overview of ghosts?*
  This is done by pressing the 'Home' icon, or

the second icon.

- *Where do you look up information about the history of ghosts?*
  This is done in either the first or second menu, this is not clear right away.
- *You notice that instead of information about ghosts, information about churches is displayed. Where do you fix this problem.*
  Under settings (the bottom option).

Questions:

- *Which Interface was the most intuitive?*
  The second interface.
- *Were there any problems using one or more interfaces?*
  I would prefer to have all options on one side and the info on the right. This is not the case with the first interface. The first interface also has too many options in the left menu.
- *Did you miss any functionality in any of the interfaces?*
  I would like to see the tree structure of the third interface in the second interface.
- *Did you have any trouble understanding the icons?*
  I wouldn't have expected components as the second icon at first. After I learned this it made sense. Datasets and components are difficult to grasp as only an icon. By pressing and noticing a mistake, this is quickly learned, though.
- *Any other comments or problems you encountered?*
  I like the tree structure in the third interface, but it takes up a lot of space.
  The current interfaces are for right handed people.

A.2.3 - Test Subject 3

Interface 1

- *Where do you look up an overview of churches?*
  Probably the right side, this would show informaton, or let me select information.
- *Where do you look up how many people visit the church each year?*
  Also on the right side.
- *You notice that instead of information about churches, information about farms is displayed. Where do you fix this problem.*
  Under settings, or by pressing the name of the model at the bottom.

Interface 2

- *Where do you look up an overview of farms?*
  This is done by pressing 'Overview'.
- *Where do you look up how many buildings the farm has?*
  This is done either by pressing Components or Datasets.
- *You notice that instead of information about farms, information about ghosts is displayed. Where do you fix this problem.*
  This is done in the settings menu.

Interface 3

- *Where do you look up an overview of ghosts?*
  This is probably done by pressing the 'Home' icon.
- *Where do you look up information about the history of ghosts?*
  I guess the third menu. otherwhise the second menu.
- *You notice that instead of information about ghosts, information about churches is displayed. Where do you fix this problem.*
  Under settings (the bottom option).

Questions:

- *Which Interface was the most intuitive?*
  The second interface, although the third interface is also quite intuitive once you know the options.
- *Were there any problems using one or more interfaces?*
  The first interface is not clear,

Appendix A

Usage test 1

- *Did you miss any functionality in any of the interfaces?*
  It could be possible to use the model as a sort of dial: when holding the interface with one or two fingers, rotating the model would scroll through the menus.
- *Did you have any trouble understanding the icons?*
  The difference between the second and third icon on the second interface is not clear. The icons on the first interface were also confusing.
- *Any other comments or problems you encountered?*
  The text on the second interface should be flipped. The top text is right side up, but is obscured by the object.

Appendix B

In this appendix the final software of this project can be found. It includes the compiled applications as well as the source code.

B.1 - OverlayApp

The OverlayApp, as well as its source code, can be found on the DVD accompanying this project.

B.2 - TableApp

The TableApp, as well as its source code, can be found on the DVD accompanying this project.

B.3 - PCApp

The PCApp, as well as its source code, can be found on the DVD accompanying this project.

Appendix C

In this appendix the second usage test can be found. It includes the questions and answers. The software used to perform this test can be found in appendix B.

C.1 - Usage test 2

After a brief explanation of the goal of the system, the participants were asked to do the following tasks. No explanation of the user interface was given.

Table, model 1: already set up
- Show a summary of the product.
- Show more information concerning the costs of the product.
- Look up how many components the product has.
- Close the expanded interface.
- Show the same information you are reading to someone on the other side of the table.

Table, model 2: not set up
- Load a model in the system.
- Load a folder of text files in the system.

PC interface
- Your model is too small. Change its scale.
- Look up the information from a text file you just loaded in the previous task.
- Zoom in on the top of the model.

Questions:
- Is the interface on the table clear and intuitive?
- Is the interface on the PC clear and intuitive?
- Are the interfaces two parts of the same system or two different interfaces?
- Any other comments or problems you encountered?

C.1.1 - Test Subject 1

Table, model 1: already set up
- *Show a summary of the product.*
  Components is tapped first, afterwards Overview is tapped.
- *Show more information concerning the costs of the product.*
  Components is tapped.
- *Look up how many components the product has.*
  Components is tapped and the number of lines are counted.
- *Close the expanded interface.*
  The same button as when opening the interface is tapped.
- *Show the same information you are reading to someone on the other side of the table.*
  The edge of the interface is dragged to rotate it. This did not work as expected. The expectation was that the interface would rotate. Instead it created a new interface. This interface was attempted to be rotated with one finger at first, when this had no effect, two fingers were used. This had the desired effect.

Table, model 2: not set up

- *Load a model in the system.*
  This was attempted in the components tab first. When this did not work, the correct button in the settings menu was found.
- *Load a folder of text files in the system.*
  Firstly the menu was tried where the text should be loaded. If this did not work the components menu would be tried. Lastly the settings menu would be attempted if neither of these options worked.

PC interface

- *Your model is too small. Change its scale.*
  Right click on the model. This had no effect. Afterwards the model was selected in the menu and the scale was adjusted in the settings menu.
- *Look up the information from a text file you just loaded in the previous task.*
  The menu is selected; here the select model menu is displayed. The wrong model is selected by accident. The select menu is now gone and could not be found anymore.
- *Zoom in on the top of the model.*
  First scrolling in room mode is tried. This did not work. Fairly quickly after this the option to disable room mode is found. Afterwards zooming and moving the camera is no problem.

Questions:

- *Is the interface on the table clear and intuitive?*
  Not always. Especially the difference between datasets and components is not really clear.
- *Is the interface on the PC clear and intuitive?*
  Only the disappearing 'select model' menu is confusing, the rest is clear.
- *Are the interfaces two parts of the same system or two different interfaces?*
  To me these are two separate interfaces.
- *Any other comments or problems you encountered?*
  The term datasets is unclear to me. This seems like a menu for the developer. I'm also not quite sure what the system can be used for; this may be because there are now abstract blocks instead of models on the table.

C.1.2 - Test Subject 2

Table, model 1: already set up

- *Show a summary of the product.*
  Overview is tapped.
- *Show more information concerning the costs of the product.*
  Datasets is tapped, and then costs.
- *Look up how many components the product has.*
  Components is tapped, the number of lines are counted.
- *Close the expanded interface.*
  The right side of the interface is dragged to the left. This did not have the desired effect. The second attempt was to tap the same button as when opening the interface. This had the desired effect.
- *Show the same information you are reading to someone on the other side of the table.*
  The creation of a second interface occurs without problems. The right close button is dragged to move the interface further. When this has no effect, the arrows on the left are found. The interface is rotated with two fingers in the arrow area.

Table, model 2: not set up

- *Load a model in the system.*
  This is done under Settings. The browser interface gives some problems. It is not clear in which directory the user is. Scrolling in the browser is not clear.
- *Load a folder of text files in the system.*
  A folder is selected and then the import button is pressed. This has no result. Afterwards the individual files are selected before pressing the import button.

PC interface

- *Your model is too small. Change its scale.*
  This is done under settings. A dropdown was expected in the value area similar to the model select menu. When this doesn't work, values are entered manually.
- *Look up the information from a text file you just loaded in the previous task.*
  The model is selected and the required menu is selected.
- *Zoom in on the top of the model.*
  Scrolling and clicking is tried. The user has no success in performing this task until the room mode checkbox is pointed out. After this the same actions as before are performed, which give the desired result.

Questions:

- *Is the interface on the table clear and intuitive?*
  Yes, the only thing is that the browser is not really clear. This could be because it is not my computer and I am not familiar with this folder structure. It is inconvenient that the browser doesn't remember at what folder you stopped browsing.
- *Is the interface on the PC clear and intuitive?*
  Yes, the only hiccup was the room mode switch.
- *Are the interfaces two parts of the same system or two different interfaces?*
  They are one whole.
- *Any other comments or problems you encountered?*
  Changing the import settings of a model in the PCApp with sliders, rotating knobs etc. would be an improvement. Show which axis is which for the translation. The placement of the monitor in relation to the table should be considered.

C.1.3 - Test Subject 3

Table, model 1: already set up

- *Show a summary of the product.*
  Overview is tapped.
- *Show more information concerning the costs of the product.*
  First Components is tapped, the required option is not here. Then Datasets is tapped and the required option is found.
- *Look up how many components the product has.*
  Components is tapped.
- *Close the expanded interface.*
  The close button on the right is pressed. If this didn't work the same button as when opening the interface would have been pressed.
- *Show the same information you are reading to someone on the other side of the table.*
  A SUI is created with no problem. Rotating the SUI is attempted by moving one finger in a circular motion in the arrow section. When this has no effect, two fingers are used (not only in the arrow section), which has the desired effect.

Table, model 2: not set up

- *Load a model in the system.*
  The Components and Datasets menus are searched without result.
- *Load a folder of text files in the system.*
  The empty menu is selected. Using the browser gives some problems: Blue buttons (files) are interpreted for selected files. The

selected folder is not highlighted, and thus after selecting a folder it is unclear in which directory the user is.

PC interface

- *Your model is too small. Change its scale.*
  An attempt is made to select the model by left clicking on it. When this has no effect, the model is selected from the dropdown menu, and the scale is adjusted in the settings menu.
- *Look up the information from a text file you just loaded in the previous task.*
  The model is selected from the dropdown. After this the user navigates to the relevant info menu.
- *Zoom in on the top of the model.*
  Room mode is disabled under settings, then the camera is moved without problems.

Questions:

- *Is the interface on the table clear and intuitive?*
  Yes, but I would have expected to be able to rotate an interface by dragging at its edge. As for the browser: it is not very clear in which directory you currently are. The selected folder should be highlighted. Also the blue files seem selected items, rather than files.
- *Is the interface on the PC clear and intuitive?*
  Yes, but the selected model does not stay in the closed dropdown menu. Adding this would be an improvement
- *Are the interfaces two parts of the same system or two different interfaces?*
  The same. The same icons and categories are used.
- *Any other comments or problems you encountered?*
  Everything has been mentioned before