Can augmented reality supplement traditional human-machine interfaces?

Gustav Lilja and Fredrik Siemund

DEPARTMENT OF DESIGN SCIENCES
FACULTY OF ENGINEERING LTH | LUND UNIVERSITY
2020

MASTER THESIS

Can augmented reality supplement traditional human-machine interfaces?

Evaluating the role of augmented reality in an industrial environment

Gustav Lilja and Fredrik Siemund

LUND
UNIVERSITY

Can augmented reality supplement traditional human-machine interfaces?

Evaluating the role of augmented reality in an industrial environment

Published by
Department of Design Sciences
Faculty of Engineering LTH, Lund University
P.O. Box 118, SE-221 00 Lund, Sweden

Subject: Interaction Design (MAMM01)
Division: Ergonomics and Aerosol Technology, Department of Design Sciences, Faculty of Engineering LTH, Lund University
Supervisor: Günter Alce (gunter.alce@design.lth.se)
Examiner: Joakim Eriksson (joakim.eriksson@design.lth.se)

Abstract

In this master thesis, the possibilities and limitations of using handheld augmented reality (AR) as a complement to a regular human machine interface (HMI) are explored. This was done by developing a mobile application for Android and connecting it to a real machine, making it possible to perform actions on the machine through the AR application. A user study with 20 test subjects was conducted where the AR prototype was compared to a regular HMI.

The majority of the test subjects preferred the AR interface over the regular HMI. System usability tests showed that it was easier to perform the actions through the AR application. The average time to perform each action was lower with the AR application.

The main advantage of using AR turned out to be the possibility to display the data in a context, i.e. at the right machine component. A drawback with (handheld) AR is that the development tools might not be suitable for large scale applications at this stage.

Keywords: Augmented Reality, Human Machine Interface, Image Recognition, Industry 4.0

Acknowledgements

To our supervisor from LTH, Günter Alce, who guided us through the project and gave us constant feedback.

To our advisor at Beijer Electronics, Tommy Håkansson, who helped us coming in contact with people at Beijer, gave us feedback and supplied us with equipment.

To all the employees at Beijer Electronics who gave us feedback, and a special thanks to the support engineers at Beijer for helping us with the Beijer displays.

Lastly, we would like to thank all the people who participated in our tests and gave us feedback about our work.

Contents

1 Introduction **7**

1.1 Purpose and goal . . . 7
1.2 Research questions . . . 7
1.3 Beijer Electronics . . . 8
1.4 Limitations . . . 8
1.5 Contributions . . . 8
1.6 Related work . . . 8

2 Theory **11**

2.1 Augmented reality . . . 11
2.1.1 Definition of augmented reality . . . 11
2.1.2 A brief history of augmented reality . . . 11
2.1.3 Tracking technologies . . . 12
2.1.4 Form factors . . . 13
2.1.5 Developing for handheld AR . . . 14
2.2 Artificial intelligence . . . 15
2.2.1 Definition of artificial intelligence . . . 15
2.2.2 Machine learning . . . 16
2.2.3 Deep learning and neural networks . . . 17
2.2.4 Convolutional neural networks . . . 19
2.3 Design methods . . . 19
2.3.1 User Centered Design . . . 19
2.3.2 Interviews . . . 19
2.3.3 Prototyping . . . 20
2.4 NASA TLX and SUS . . . 20

3 Design and development **21**

3.1 Concept phase . . . 21
3.1.1 Initial meeting with stakeholders . . . 21
3.1.2 Gathering information on augmented reality technologies . . . 22

3.1.3 Gathering information on deep learning technologies 23
3.1.4 TensorFlow 24
3.1.5 Lo-fi prototypes 24
3.1.6 Testing the ARCore and TensorFlow prototypes 29
3.1.7 Phase summary 31
3.2 Development phase 31
3.2.1 Development of hi-fi prototype 32
3.2.2 Development of Beijer X2 Control interface 34
3.2.3 Combining elevator with hi-fi prototype and Beijer X2 Control . . 36
3.2.4 Phase summary 36
3.3 Testing phase 36
3.3.1 Participants 37
3.3.2 Equipment 37
3.3.3 Roles 37
3.3.4 Location and dates 38
3.3.5 Procedure 38
3.3.6 Test objectives 38
3.3.7 Phase summary 40

4 User study results 41
4.1 Pre-test survey 41
4.2 SUS 44
4.3 NASA TLX 46
4.4 Timekeeping of task completion 48
4.5 Post-test survey 48
4.6 Error sources 52

5 Discussion 53
5.1 Discussion of result in relation to research questions 53
5.2 Discussion on flaws and improvements 55
5.3 The design process 56
5.4 Suggestions on future work 57

6 Conclusions 59

References 61

Chapter 1
Introduction

1.1 Purpose and goal

The fourth industrial revolution, or Industry 4.0 [23], represents the idea in which machines are augmented with wireless connectivity and sensors, creating smart and connected factories. Thanks to recent advances in areas such as communication technologies, automation and artificial intelligence, it has been possible to improve efficiency by, for example, predicting machine maintenance autonomously or react to changes in production with self-organized logistics [29]. One emerging technology, augmented reality (AR), enhances the experience of the real world by adding digital elements to it. This makes it a useful tool in the industry, for example in use cases like expert support through remote guidance or complex assembly.

The purpose of this work is to create an extended human-machine interface through an mobile Android application using AR. By using the application, people working in a factory should be able to easily see relevant and real-time data about a machine and control it when the tablet's camera is aimed towards it. Today, most machines are operated using a built-in display. The goal is that this application should work as a complementary interface to the built-in display interface, showing the user real-time data.

1.2 Research questions

These are the research questions which this thesis is based upon:

- Could an augmented reality interface work as a complementary interface to a traditional touch user interface?
- What are the benefits and drawbacks of using an augmented reality interface in this context?
- What kind of data is suitable for an augmented reality interface?

1.3 Beijer Electronics

Beijer Electronics is a Swedish technology company that develops and manufactures human machine interface (HMI) displays, among others. The company's customers are mainly machine manufacturers which in turn integrate the display into their machines. Beijer Electronics also develops the software iX Developer, which is used to create the interface in the display [9].

1.4 Limitations

In this section the limitations which were set for this project will be presented. The first technical limitation was that the AR application would only be able to control a small machine located inside the office building. This was decided because in order to do it on a large machine in a factory the authors would need full access to the factory, which would not be possible. The other technical limitation was that the application would only be developed for android and not iOS. Because Beijer only uses Windows based computers it became the best option if they would decide to develop the application further in the future, this of course meant that the authors could not use ARKit from Apple.

1.5 Contributions

Fredrik focused more on developing a deep learning lo-fi prototype, capable of recognizing the different coffee machines in the Beijer offices. At a later stage he focused on creating the interface for the Beijer X2 Control and the communication between the AR prototype and the electronic elevator.

At the start of the project, Gustav focused on developing lo-fi AR prototypes using different libraries such as ARCore and Vuforia. In the development phase he focused more on further development of the ARCore prototype and implementing controls for the elevator and a way of handling errors using the application.

Both authors created the two test plans, for the initial and the final test, and played active roles during the test sessions. The authors developed the final prototype together but were responsible for different aspects of the application. The report writing were also done by both with an equal amount of work done.

1.6 Related work

Subakti et al. in [33] developed a mobile markerless augmented reality [32] app very similar to the project in this thesis. They used transfer learning [14] on a MobileNet [17] model and trained it with pictures of three different machines on their campus. They even went a step further by training an object detection model for separate parts of a machine, such as the power cabinet and emergency stop. With the help of the smartphone's depth sensor, they created a distance-aware application where information with various level of detail was displayed depending on the user's distance to the machine. When a machine was recognized, the application fetched machine data from the back-end and displayed it in the application.

Tsai et al. in [34] created an Escape Room game combining indoor Bluetooth positioning, image recognition and augmented reality. Bluetooth positioning and image recognition are used to find clues, the former by revealing a clue when the user is at a specific position in the room and the latter by scanning real world objects. The mobile application was developed using Vuforia [20], an AR platform which makes it easier to add computer vision functionality to mobile devices. With its help, the authors scanned objects that were supposed to contain clues. When a user pointed the camera towards the clue, a clue would pop up. One of the major challenge in this project was to create the coordinate system for the indoor positioning.

Sabarinathan et al. in [31] developed an mobile AR-application for machine maintenance. They used a tool called CraftAR developed by catchoom®for image recognition and augmented reality. Once a machine is recognized, its features are fetched from a local server and displayed in a menu together with other options such as name and manufacturer information, maintenance information and maintenance logs. The user can interact with the different menu options and, for example, add a new entry to the maintenance log. If the user has to perform maintenance on the machine, he or she can get help from the app through its AR-feature, which superimposes AR-elements such as labels and instructions on the machineparts.

Chapter 2

Theory

This chapter contains the theoretical background of the report, including information about augmented reality, machine learning and design methods.

2.1 Augmented reality

In this section, the definition and history of augmented reality is explained. Additionally, different tracking technologies and form factors are evaluated. It ends with a description of how to develop for handheld augmented reality.

2.1.1 Definition of augmented reality

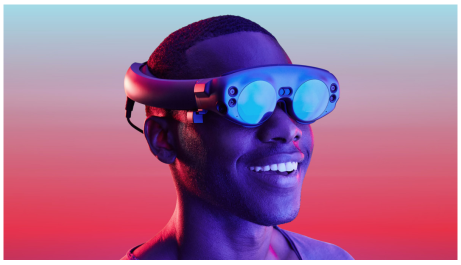
Augmented reality (AR) is a technology which allows the user to view a physical real-world environment that has been enhanced (augmented) by adding virtual information to it. It is meant to enhance the user's interaction with the real world, as opposed to virtual reality (VR), which encloses the user in a virtual world [10]. Augmented reality is also not limited to the use of a particular type of display, such as the head-mounted display (HMD), virtual objects can be displayed in the real world through the use of a camera on a phone or tablet as well. In 1997, Ronald Azuma writes the first survey in AR, providing a widely recognized definition of AR by identifying it as combining the real and virtual world, the real-world being registered in 3D and interactive in real time [4].

2.1.2 A brief history of augmented reality

The first appearance of augmented reality dates back to the 1950s when Morton Heilig thought of the cinema as an activity which could involve all of the viewers senses. Heilig built his first prototype back in 1962 which was called Sensorama. It was able to stimulate

Figure 2.1: Ivan Sutherland's Sensorama in 1966. Image from Pinterest.

(a) Hololens



(b) Magic Leap One

Figure 2.2: In figure (a) the Hololens from Microsoft is displayed. In figure (b) the Magic Leap One from Magic Leap is displayed. Both of which are augmented reality headsets.

several of the users senses while watching a movie. Next up was Ivan Sutherland who invented the first head-mounted display in 1966, which can be seen in figure 2.1. It was further developed in 1968 by creating an AR system using an optical see-through HMD.

Myron Kreuger then invented the Videoplace in 1974 which allowed the user to actually interact with a digital object. The phrase 'augmented reality' was coined in 1990 by a Boeing researcher called Tom Caudell as a means to describe the merging of virtual graphics into physical displays. In the early 2000s the world's first open-source software library, ARToolKit, is created by a man named Hirokazu Kato. With this library the first outdoor AR game, ARQuake, is also released. As of late augmented reality is a widely used technology in phones and tablets. In 2016, the augmented reality game Pokemon Go was launched, it reached a peak of 45 million daily users. Both Apple and Google launched their own respective augmented reality developer kits in 2017, ARKit and ARCore, making it possible for developers to easily develop augmented reality applications. Development of new augmented reality hardware is increasing in parallel with new software, some more notable products are the HoloLens from Microsoft, as can be seen in figure 2.2a and the Magic Leap One from Magic Leap, seen in figure 2.2b.

2.1.3 Tracking technologies

One of the challenges developing augmented reality applications is the accurate tracking and registration between computer-generated object and real world objects [19]. When a user locates to another position, the virtual objects must remain aligned with real world objects, this will depend on how accurate the tracking is. In this project, marker-based tracking and marker-less tracking have been used, both of these are types of vision based tracking. Vision based tracking uses computer vision technologies in order to calculate the camera pose relative to real-world objects [19]. According to Zhou et al, this is the most active area of research in AR [36].

Marker-based tracking

In Marker-based tracking, visual markers are placed in the scene for augmented reality applications [19]. These markers have specific properties that make them easy to track and to identify their position in the real world. Google's augmented reality development kit, ARCore, is able to use 2D images such as posters or product packaging as markers. By providing a set of reference images ARCore can predict the markers position in the real world. However, marker-based tracking comes with a few drawbacks. One restriction is that it can be quite time-consuming and requiring a large number of markers if several objects or locations are to be augmented. Another drawback is that the markers often cannot be attached to the objects which are supposed to be augmented [10].

Markerless tracking

In markerless tracking, natural features of the environment which are to be augmented are used instead. These natural features are sometimes called interest points or key points and must meet several requirements.

- *Fast computational time.* It must be possible to calculate a sufficient number of feature points in order to allow for a pose estimation at an acceptable rate [10].
- *Robustness with respect to changing lightning conditions and image blurring.* The set of feature points must not vary significantly under various lightning conditions or upon image blurring, this can easily happen in outdoor environments [10].

These feature points are then matched against corresponding feature points inside a database to extract a respective six degrees of freedom (6DOF) pose. This task is called feature matching and is one of the most critical tasks of tracking. The feature points inside the database can be used as reference for the real object to be tracked, the real objects dimensions must be known in advance [10]. The database holding these feature points is referred to as a feature map. Often an image or a bitmap is sufficient enough to create a feature map of an object and these are created before the start of the tracking. The tracking process is then initialized by the gathering of visual data from the real world, creating feature maps. The size of the feature maps increase with each new frame and the positions of already found feature points are improved. During this phase the feature points are used to create a 3D plane which is then used as the basis for the virtual objects in the AR application [10].

2.1.4 Form factors

Augmented reality is a diverse field where the technology is implemented on a wide variety of hardware. The main categories can be divided into handhelds and head-worn. The

Figure 2.3: Google's ARCore logo to the left and Apple's ARKit logo to the right.

head-worn includes headsets and helmets while the handhelds include mobile devices (smartphones, tablets, laptops) [25]. Both Apple and Google have developed their own AR developer kits, ARKit and ARCore respectively. ARKit only works for iOS and ARCore works for both Android and iOS. Both of these tool kits have made it possible to more easily develop augmented reality applications for both categories. The market for handheld hardware is however much larger since most smartphones produced today are AR-capable and the head-worn hardware tend to be expensive. It might however be hard to create a truly immersive experience on a smartphone. The reasons being that the user is forced to hold their phone up in front of their face for a long time and that the screen is to small to capture the 3D environment in its entirety. The key capabilities of Google's ARCore are motion tracking, environmental understanding and light estimation. Motion tracking allows the phone to understand it's position relative to the world, this process is, according to Google, called concurrent odometry and mapping, or COM. It allows ARCore to detect visually distinct feature points and use these points to compute its change in location [12]. Environmental understanding refers to allowing the phone to detect the size and location of surfaces in the real world. Lastly, light estimation allows the phone to estimate the environment's current lightning conditions. Apple's ARKit has similar capabilities as ARCore.

2.1.5 Developing for handheld AR

There are a number of software development kits (SDKs) to choose from when developing an augmented reality application. Some of these will be presented in this subsection along with the development environments they can be used in.

- *ARCore*. ARCore is the AR SDK provided by Google. It can be used with multiple developer environments, for example Android Studio and Unity. Unity is a game engine with its own development environment and by using this in combination with ARCore, applications for both iOS and Android can be developed. Google has several tutorials and sample projects available on their website. Each sample makes use of the different features available in the ARCore library and makes it possible to start with some boilerplate code.

- *ARKit*. ARKit [2] is the augmented library provided by Apple and can only be used in Apple's own developer environment, Xcode. It is therefore not possible to develop

Android application using ARKit. Apple has several samples created with ARKit available on their website alongside with design guidelines and session videos.

- *Vuforia.* Vuforia enables developers to add computer vision functionality to application, allowing it to recognize pictures and objects, and augment them by adding virtual elements. Vuforia has several tutorials and samples on their website representing the different features available in the Vuforia library. These samples can be run in both Unity and Android Studio, making it possible to use Vuforia to create both Android and iOS applications. Some of the features available in the Vuforia library are:
  - Model targets, which allows the application to recognize an object by it's shape and then place AR content on it.
  - Object targets are created by scanning an object, this is mostly viable for smaller items with rich detailing.
  - Image target allows the user to place AR content on a flat recognizable surface [35].

2.2 Artificial intelligence

During the last decade, Artificial intelligence (AI) has gained an increasing amount of media attention. Advancements in areas such as self-driving cars and virtual assistants has brought AI closer to the consumer. Despite that, AI has been around since the 1950s, when Christopher Strachey at the University of Oxford developed a program that could play a full game of checkers [7]. As can be seen in figure 2.4, AI encompasses other areas such as machine learning and deep learning. This section will explain what AI, machine learning and deep learning are and how neural networks are used to create a deep learning application.

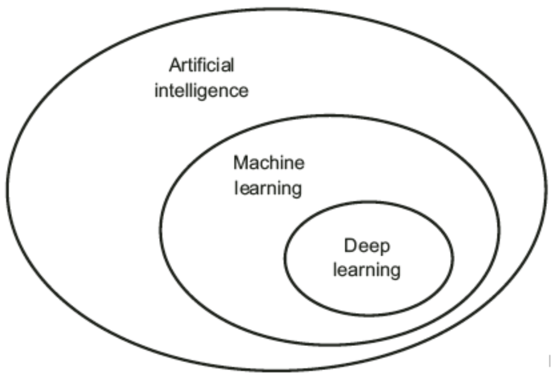


Figure 2.4: Deep learning is a subset of machine learning, which is a subset of artificial intelligence [6].

2.2.1 Definition of artificial intelligence

AI can be defined as "the ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings" [7]. This means that AI is a general field, not only including machine learning and deep learning, but also approaches that do not involve any learning. For example, early chess programs included hard coded rules for how the program should react. For a period between the 1950s and late 1980s, this approach, called symbolic AI, was the dominant way of developing AI programs. Experts believed that human-like artificial intelligence could be attained by creating a sufficiently large set of explicit rules for manipulating knowledge. This approach worked well for well-defined, logical problems, but turned out to be unmanageable for problems such as image recognition and speech recognition. Machine learning happened to work much better for these types of problems and arose to take symbolic AI's place.

2.2.2 Machine learning

In machine learning, a program is trained rather than explicitly programmed. As displayed in figure 2.5, a machine learning program is fed with a rather big amount of data and answers (expected output). In this so called training phase, the program "learns" to associate a specific piece of data with an answer. If the program is supposed to classify an image, for example, the data could be an image of a banana and its answer would be the tag "banana".

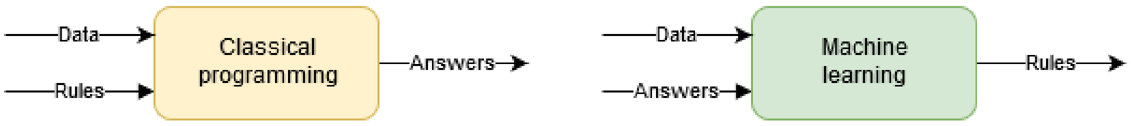


Figure 2.5: Difference between classical programming and machine learning [6].

In machine learning, the most common types of learning are supervised learning, unsupervised learning and reinforced learning [30]:

- In *supervised learning*, the machine learning algorithm is trained with a set of input-output pairs (x_j, y_j), where y_j was generated by an unknown function $y = f(x)$. The task of supervised learning is to find a function h that approximates the true function f. Common supervised learning algorithms are classification and regression. Classification is used when the output y is restricted to a limited set of values. If the output is a number in a range (such as tomorrow's temperature), the learning problem is called regression. Figure 2.5 displays supervised learning.

- *Unsupervised learning* algorithms try to find structures and patterns in input data, even though no expected output is given. Using commonalities found in the input data, the algorithm can react on new data based on the presence or absence of these. The most common task is clustering: dividing a set of data points into groups so that members of a group are similar to each other and dissimilar to data points in other groups.

- In *reinforced learning*, the algorithm learns through "experience". Every time the algorithm does something right, it will be rewarded (positive reinforcement), and when it does something wrong, it will be punished (negative reinforcement). For example, a machine learning program could learn to play chess by using reinforced learning, iterating over and over again until it understands the mechanisms of the game.

In this project, supervised learning was used. When creating a machine learning model using supervised learning, three things are needed:

- Input data. This could, for example, be images, sound files or text files.
- Expected output for each piece of input data. For image recognition, this would be a tag such as ”banana”, ”apple” etc. For speech recognition, this would be a transcript of the sound file.
- A way to measure if the algorithm is doing a good job. In this part (the actual learning), the algorithm's current output is compared to the expected output and the result is used to adjust how the algorithm works.

The machine learning model is supposed to transform the input data into meaningful output. This is done by using known examples of input and output data and learn from them. Therefore, a major challenge in machine learning is to find meaningful representations of the input data and to transform it into a representation which makes it easy for an algorithm to make a decision.

2.2.3 Deep learning and neural networks

Deep learning is a sub-field of machine learning in which the model being trained has more than one layer between its input and output. The number of layers is called the depth of the model. This number often ranges between tens up to hundreds of successive layers. In most cases, neural networks are used to create a deep learning model [16]. The basic structure of how a deep learning model might look like can be seen in figure 2.6. What happens with the input's representation between the different layers is specified in the layer's weight. The weights are basically numbers telling the model how the representation should be modified when going from one layer to the next. To be able to know how far of the model's prediction is from the correct output, a loss function is needed. The loss function computes a distance score which shows how well the model did on this example. This score is in turn used by the optimizer, which uses it to adjust the weights a little. This is called backpropagation, a central algorithm in deep learning [6]. The initial values of the weights are random numbers, thus the distance score will be high in the beginning. For every new input-output pair, the weights will be adjusted and the distance score will decrease. Once a minimal loss has been found, the network is trained.

The basic building blocks of a neural network are:

- A *layer* is the fundamental data structures of a neural network. Many layers are combined into a network. A layer is in turn made up of nodes, each holding a single number. Depending on how the network is designed, the different layers can contain varying numbers of nodes. As can be seen in figure 2.7, each node (also called artificial neuron) receives one or more inputs from other nodes. These are multiplied with the corresponding weight and the products are summed with a bias. Finally, an activation function (also called transfer function) is applied on the result. The activation function is used to limit the minimum and maximum of the output, since it theoretically could range from negative infinity to infinity. Often a sigmoid function is used [28]. The role of the bias is to enable the algorithm to shift the activation function to the

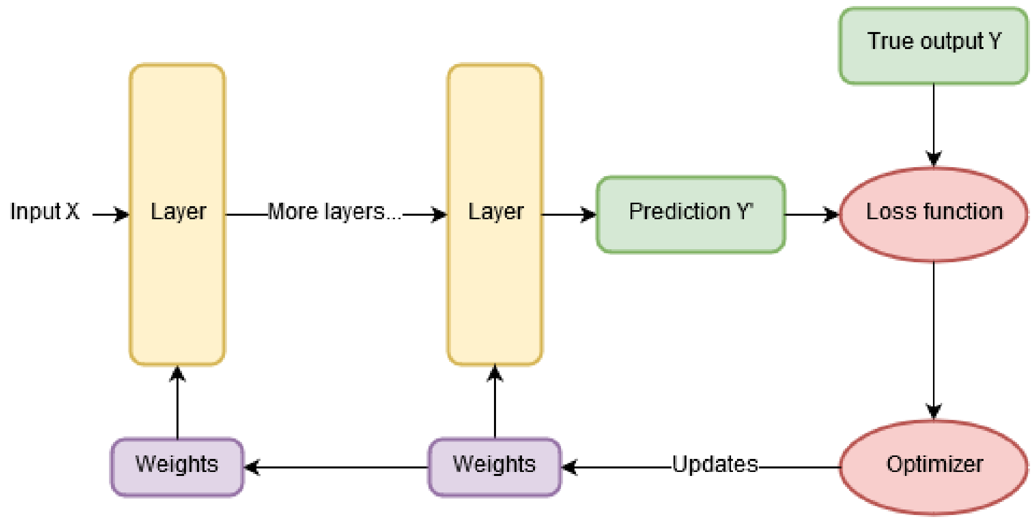


Figure 2.6: The anatomy of a neural network, often used in deep learning [6].

left or right to, for example, output zero for a certain input. The output for the *n*th neuron can be expressed as:

$$y_n = \phi(\sum(w_n x_n) + b)$$

- The *loss function (objective function)* computes the loss of the network by comparing the networks prediction to the actual value. This function is what will be minimized during training.
- An *optimizer*, which updates the network weights in response to the output of the loss function.

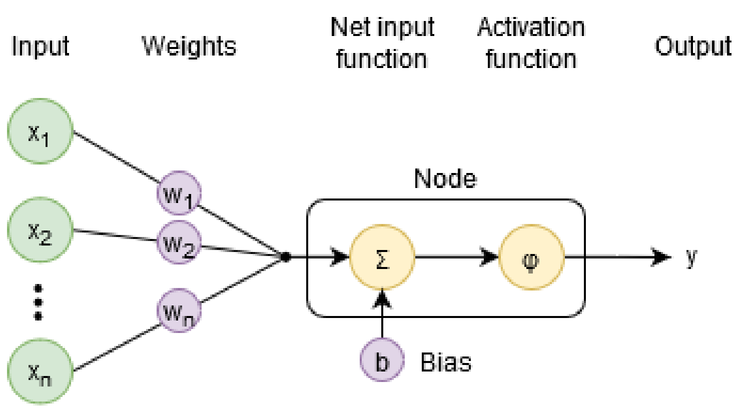


Figure 2.7: How the input of a single node is mapped to its output [28].

2.2.4 Convolutional neural networks

In this project, a special type of deep learning-model, called convolutional neural network (CNN), was used. CNNs are very common in computer vision applications [6]. A convolution is a special type of linear operation and a CNN is a network that uses convolution instead of general matrix multiplication in at least one of its layer [11]. The key advantages of a CNN compared to a regular neural network when used for computer vision are:

- *Translation invariant.* If CNN learns a pattern in one part of an image, it will be able to recognize it in other parts of the image. A regular (densely connected) network would have to learn the pattern again if it appeared in an other location.
- *Spatial hierarchies of patterns.* A first convolution layer learns local patterns such as edges and other shapes, the next convolution layer learns patterns made up of the features from the previous layer, for example an eye or ear, and so on, using the features from one layer as building blocks in the next.

2.3 Design methods

In this section the different design techniques and processes are presented. The user centered design process was used as a basis when structuring how the project would be done. The two techniques interviewing and prototyping were used extensively throughout the process.

2.3.1 User Centered Design

User-centered design (UCD) is a term used to describe design processes in which the end-user influences how the design takes shape [1]. The key principles of UCD are that the designers must first understand who the users will be, the measurements of the designs should be done empirically by letting users use prototypes to carry out real life work and the design process must be iterative and based on feedback from the user testing [15]. UCD have several advantages, one being that the products require less redesign and that they integrate into the environment more quickly since the users motivations and goals have been taken into consideration [1]. The design process can be divided in three phases, The concept phase, the development phase and lastly the testing phase. The purpose of the concept phase is exploring what the users and stakeholders would find desirable in a product of this kind. It consists of evaluations, information gathering, observations and analysis which will result in different concept ideas and lo-fi prototypes. These prototypes are then evaluated by stakeholders and end users. The concept phase ends when one single lo-fi prototypes is chosen. The next phase is the development phase where the lo-fi prototype is further developed into hi-fi prototypes. The hi-fi prototype is then tested in the testing phase with both stakeholder and end-users [3].

2.3.2 Interviews

A user interview is a UX research method during which the researcher asks one user questions about a specific topic, this could include behaviours, goals or the use of a system. User interviews are one-on-one sessions and tend to be an effective method to collect user data. User interviews can be done during multiple times during the life cycle of a project with different purposes. For example, conducting a user interview before the design is ready is done in order to get a better understanding about the user and what their needs are [26]. There are four main types of interviews, unstructured, structured, semi-structured and group interviews. The first three types names corresponds to the level of control the interviewer has on the conversation. If the objective is to evaluate how users react to a new design idea, a more unstructured interview is appropriate. However, if the goal is to evaluate a particular design feature, then a more structured interview might be a better approach [27].

2.3.3 Prototyping

A prototype is an early draft of how a future product might look and feel. It can involve the whole product or just a part of its functionality and it allows designers, stakeholders and users to consider, think about, evaluate and develop an idea [3]. The reason for working with prototypes is that the requirements are almost impossible to determine before the product actually consists in any form. Prototypes can be created using different fidelity's, a low fidelity (lo-fi) prototype could be created with some paper and a pen. Performing usability tests on a lo-fi early in the design process allows the designer to discover what kind of need and requirements different stakeholders have. Furthermore they are easy and cheap to change, already developed products are not [3]. Another form of prototype is a high fidelity (hi-fi) one, these are often interactive and computer generated with a high level of detail.

2.4 NASA TLX and SUS

In this subsection the NASA Task Load Index (TLX) and the System usability scale (SUS) will be briefly explained.

NASA TLX

The NASA Task Load Index is a tool developer by the Human Performance Group at NASA and is used regularly in healthcare and aviation. It is used to assess the perceived physical, mental and temporal demand of performing a specific task or while using a system. It is also a measurement of the perceived performance, effort and frustration [24].

SUS

The system usability scale consists of a 10 item questionnaire with five responses ranging from Strongly agree to Strongly disagree and is used for measuring the usability of a products, systems or services. The participant's scores for each question are converted to a number, added together and multiplied by 2.5 in order to get a point from 0-100. A SUS score above 68 is considered to be above average and anything below is considered to be below average [5].

Chapter 3
Design and development

In this chapter, each design and development phase of this project is presented in detail. An idea is brought to the concept phase and during this phase it evolves into several concepts and lo-fi prototypes. In the development phase these prototypes are further developed and turned into one or more hi-fi prototypes. The testing phase involves testing and evaluating the hi-fi prototype with end-users and stakeholders.

3.1 Concept phase

In the concept phase discussions were held with both stakeholders in order to outline what had to be developed. Relevant information gathering about augmented reality and machine learning was also done in order to determine how these technologies could be utilized to fulfill the visions of the stakeholders and target-users.

3.1.1 Initial meeting with stakeholders

An initial meeting with Beijer Electronics was held in order to figure out what goals they had for this application. Why did they want to develop it and what value could AR give the target-user and Beijer's other customers. Apart from discussing the specific application

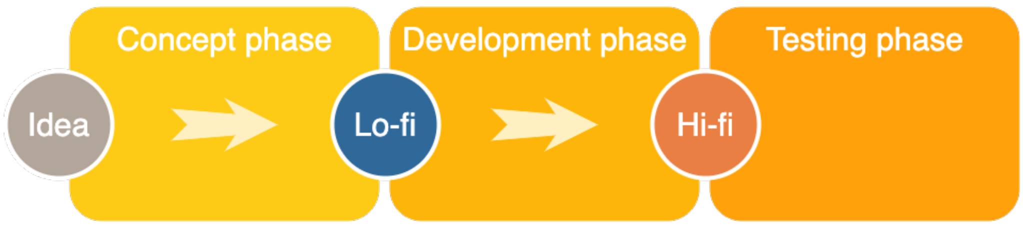


Figure 3.1: Phases through the project.

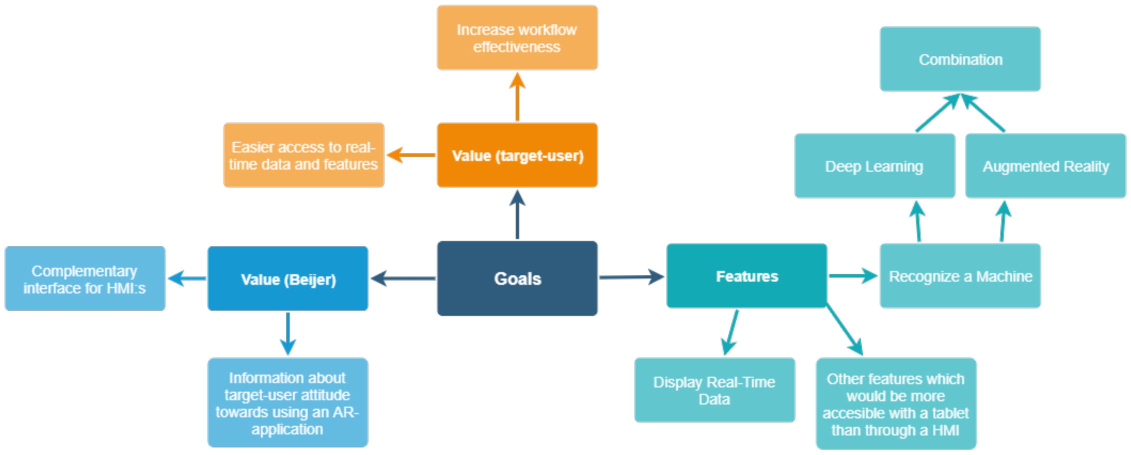

Increase workflow effectiveness
Combination
Easier access to real-time data and features
Value (target-user)
Deep Learning
Augmented Reality
Complementary interface for HMI:s
Value (Beijer)
Goals
Features
Recognize a Machine
Information about target-user attitude towards using an AR-application
Display Real-Time Data
Other features which would be more accesible with a tablet than through a HMI

Figure 3.2: Goals after initial meeting with Beijer Electronics.

and the goals for it, this was also a chance to gather information about how Beijer Electronics business worked, who their customers were and how their own software iX Developer worked with their HMI:s. The outcome of these discussions were that Beijer is currently exploring its role in this new dimension of technology and were value can be added for their customers. The goal for this project is to create a proof of concept and a complementary interface for Beijers' HMI:s. The specific features of this interface were not specified at this point since more domain knowledge had to be gathered and interviews with the target-users had to be held in order to pinpoint what would create value for them. One feature was however clarified and that was that the application would have to be able to recognize a certain machine and visualize real-time data about it. This feature poses several questions since machines on an industrial scale tend to be rather large and if this would be possible with the augmented reality developer kits available today or if this had to be done with deep learning, or a combination of the two. The goals which were outlined after the meeting are represented in figure 3.2.

3.1.2 Gathering information on augmented reality technologies

In order to get a better understanding of what the current state of AR were at and how to develop AR applications several experimental prototypes were created. These prototypes were created because there was a need to examine which environment and which developer kit would suit this project the most and to find out what was possible to achieve with them in a practical way. The different developer environments used were Unity or Android Studio, and the AR SDKs explored were ARCore from Google and Vuforia. In order to get a sense of what features would align most with the project, both ARCore and Vuforia were evaluated in Unity and Android Studio, resulting in several prototypes. ARCore and Vuforia both have a set of varying features. Some of these features which have been tried in this project are presented in the following list.

- Vuforia
  - Image Target: Attach content onto flat images

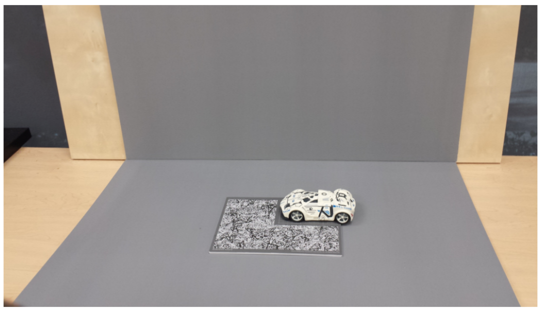


Figure 3.3: Simple scanning stage.

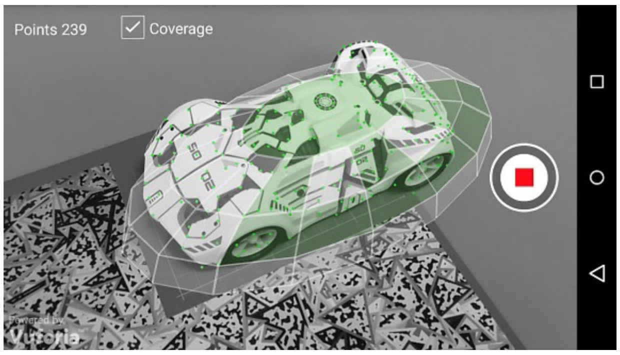


Figure 3.4: The Vuforia Object Scanner application used to scan physical 3D objects.

  - Object Target: Created by scanning an object, good option for small objects with rich surface detail.

- ARCore

  - Augmented Images: Allows ARCore to track and respond to 2D images fixed in a place. By the use of the integrated arcoreimg tool the user can create a database with images which ARCore can track.

Unlike Vuforia, ARCore does not have the ability to track 3D objects, only planes and 2D images. However the object target feature of Vuforia is only applicable on smaller objects such as toys and gadgets. In order to properly scan an object it needs to fit into a Object Scanning Target printout, as presented in figures 3.3 and 3.4.

3.1.3 Gathering information on deep learning technologies

Because of the projects time limits, development of the deep learning prototype had to go fast. Therefore, some type of framework or library had to be used. After some research, the Python library Keras with TensorFlow was chosen.

3.1.4 TensorFlow

TensorFlow is an open source library used for high performance numerical computation which makes creating machine learning applications a lot easier. The library is developed by Google and used for both research and in production [8]. TensorFlow is cross-platform which means that it runs on nearly everything, for example GPUs, CPUs (including mobile and embedded platforms), and so called Tensor Processing Units (TPU) [21]. In this project, the deep learning model ran on a mobile Android device.

TensorFlow can be used with many different programming languages, front-end APIs exists for Python, C, and C++, among others. On top of the Python API, higher-level APIs sit, which makes the training and evaluation of distributed models even easier. One of those is Keras [22], an open-source library built specifically for fast experimentation with deep neural networks.

On mobile devices, special constraints such as lower CPU capacity and a limited amount of power and memory exists. For these type of devices, TensorFlow Lite (TF Lite) [13] was created. TensorFlow Lite deep learning models are smaller in size and stored using a special format. The more optimized and smaller the model is, the faster the model will be, but with a trade-off in accuracy [13].

Transfer learning

Instead of building a new neural network from scratch, it is possible to reuse other models. In this project, a model that was trained to recognize 1000s of different objects was used as a base. The original weights where not adjusted, instead the last layer was replaced with a very small model which uses the base models output as its input. This process is called transfer learning [14] and makes experimenting with neural networks a lot faster. The base model used in this project was Google's MobileNet V2 [18].

3.1.5 Lo-fi prototypes

In this subsection, the different lo-fi prototypes developed in this phase will be presented and why the ARCore and Deep Learning prototypes were chosen to be continued with.

ARCore - Augmented Image

This prototype was developed in Java using Android Studio and was based on the Augmented Images sample provided by Google. The sample allowed the user to scan a picture of the earth and augment a frame around the edges. The reason for starting from this sample was that it allowed the user to scan a picture which could be placed on a machine and augment this machine with controllers and menus to get information about it. The sample could easily be modified to show something else than a frame and by using a tool called the arcoreimg tool pictures of an object could be used to create a database with pictures which the application could then recognize and augment. Each picture in the database was evaluated using the arcoreimg tool and was given a value between 0-100, where a value of over 70 was considered to be a good picture for this application. In order to properly recognize a certain picture it had to include many unique feature points and avoid repetitive features. An idea for this

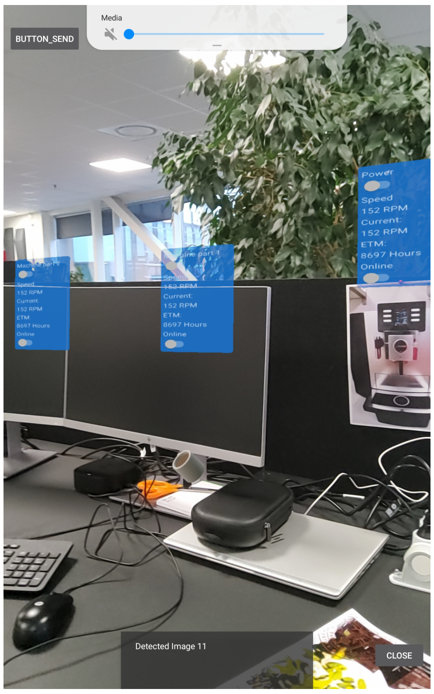


Figure 3.5: Lo-fi prototype based on ARCore Augmented Images, capable of recognizing 2D images of the coffee machine on the right.

prototype was to take pictures of a coffee machine in the office building from many different angles, and by doing so allowing the application to recognize a 3D object by searching through this database of 20 pictures of the coffee machine, the maximum number of images allowed in a arcoreimg database was 20. This did not work well, but the application did recognize a 2D picture of the coffee machine very quickly, the next step was to show menus to the user once the image had been recognized. The menus were placed above the picture so that the user would be able to visualize that the controls in the menu were meant to be used to operate the coffee machine. In order to show how this could be applied to a larger machine in an industrial environment two menus were placed to the left of the original menu with the headers "Machine Part 1" and "Machine Part 2", seen in figure 3.5.

Vuforia - Object Target

As mentioned in 2.1.5, the object target feature allows the user to recognize objects based on a 3D model of it. The 3D model had to be scanned using Vuforia's own tools and is mostly suitable for smaller objects. In order to evaluate this feature a simple prototype was developed using Unity and the Vuforia object recognition sample. The sample included a pre-configured object-recognition scene as a starting point. In order to use this, a physical object had to be scanned with the Vuforia Object Scanner in order to create an Object Data file, a sample of this can be seen in figure 3.4. The file is then uploaded to the Vuforia Target Manager where a Device Database is generated based on those uploaded files. The database is then downloaded and can be used in the sample project provided by Vuforia. In order to try this out a small object was scanned, which can be seen in figure 3.6. The main goal of the project at this point was to find an easy way to recognize a machine or a machine part, therefore the Vuforia Object Target turned out to be a dead end since it is currently designed

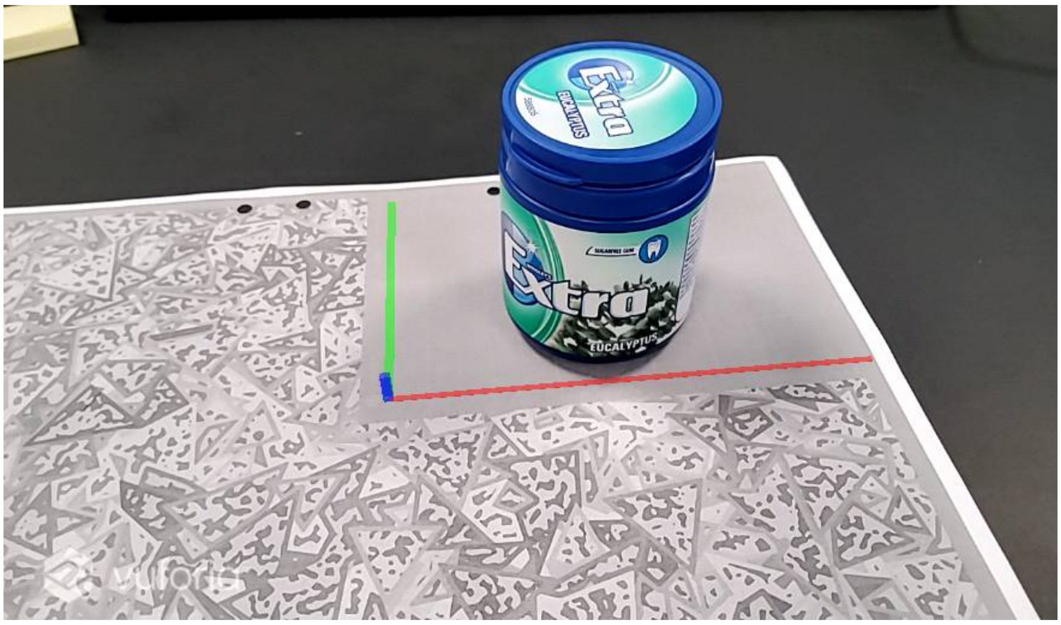


Figure 3.6: Vuforia Object Scanner used on a box of gums.

to work with smaller toys and consumer products and not bigger machinery. There would also be a problem in differentiating between two identical machines in a factory using this method.

Vuforia - Image target

Similar to the ARCore prototype using Augmented Images, Image targets from Vuforia represents images that the Vuforia engine can detect, track and augment 2D printed images. The images had to have many unique feature points and not contain repetitive patterns. The image targets are created using the Vuforia Target Manager and a generated database is then downloaded to be used in a project. In order to evaluate this method and compare it to the ARCore prototype a simple prototype was developed using the Vuforia samples available on their website. The prototype was developed in Unity since it was more difficult getting Vuforia to work as intended in Android Studio. It was capable of recognizing a picture of a book and display a menu to the user with features that could be interesting for an operator or a service technician in a factory. The idea was to have a unique picture for each machine, therefore overcoming the problem with having two identical machines. However when the camera faced away from the picture the augmented menu disappeared, this would make it hard to augment a larger machine since complementary interfaces would not be able to be placed on different parts of the machine, as is done in the ARCore prototype.

TensorFlow - Image classification with deep learning

Beijer Electronic's vision of this project was an app that would be able to recognize a machine solely on how it looks. Therefore, one of the first prototypes experimented with was an Android application based on a TF Lite example by Google. The example app used a MobileNet deep learning model that was able to recognize 1000s of different objects.

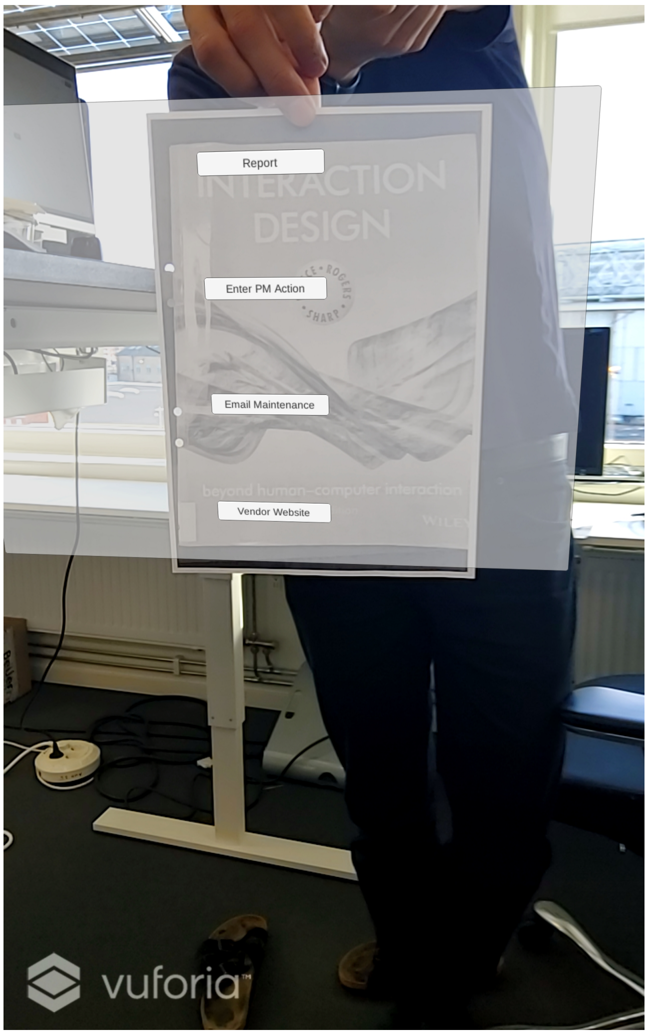


Figure 3.7: Vuforia Image Target prototype used on the picture of the book.

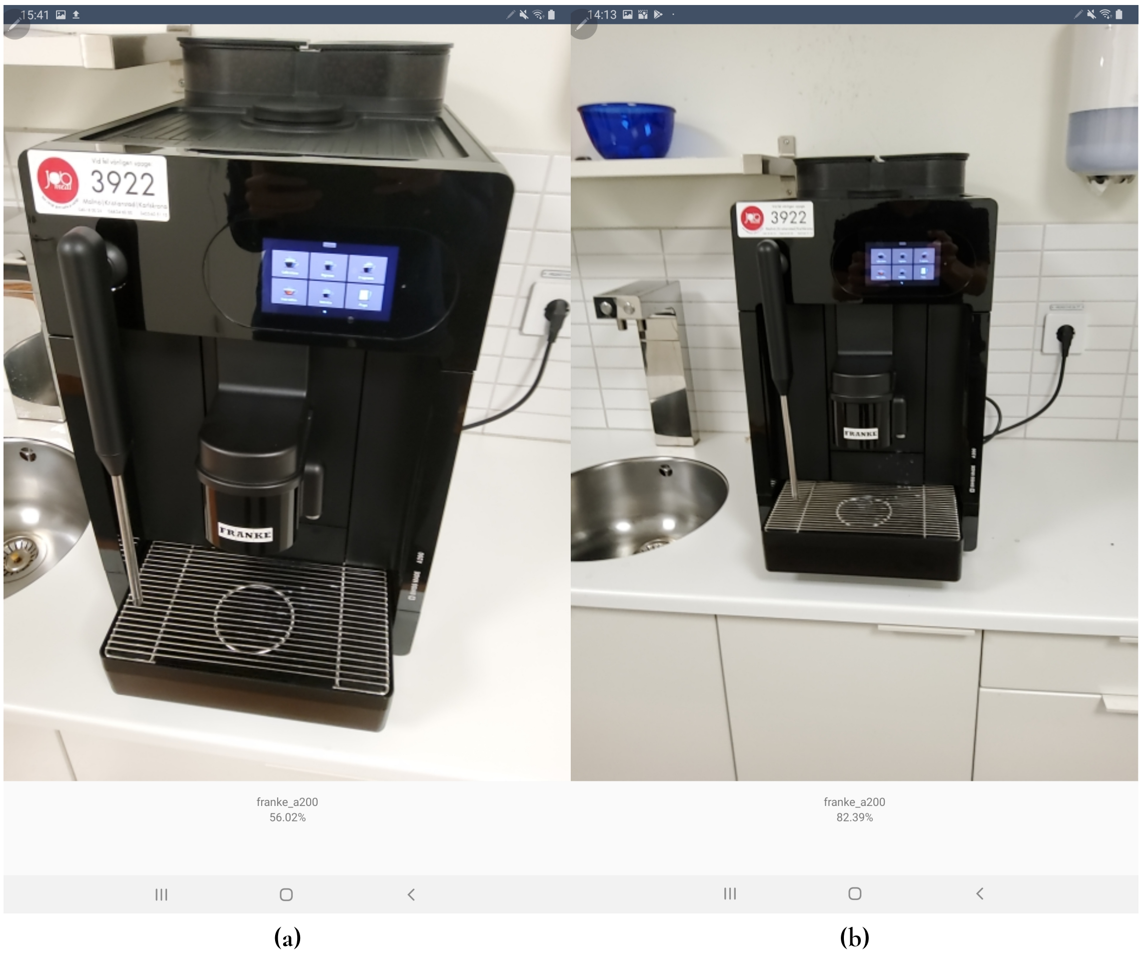


Figure 3.8: In figure (a) the TensorFlow app uses a model trained with 5 epochs and the probability score is 56%, in figure (b) the model is trained with 2000 epochs and the probability score is 82%.

The first step was to retrain the model so it would recognize a machine. Since no machines where close at hand at Beijer Electronic's HQ in Malmö, three different coffee machines where used instead. Using a smart phone camera, the coffee machines were filmed from many different angles. These video clips were split into around 600 images each using a tool called ffmpeg. A Python script written by Google was then used to retrain a MobileNet on the coffee machine images. After some initial testing, the number of images of each coffee machine was reduced to 100. In hindsight, this number could probably have been even lower, since these images where only used to retrain the last layer of the model.

The first prototype was unstable and the detected object's probability score fluctuated between 40% and 90%. The probability score indicates how sure the model is that the current image represents one of the coffee machines it was trained on. After some more experimenting, the number of epochs was increased from 5 to 2000 which resulted in a much more stable probability score, as can be seen in figure 3.8.

The app was configured so that a pop up would be shown if the probability score was above 80% for more than 1 second. The popup contained specific information about the machine such as if the power was on or off and and for how long the machine had been online. At this stage, this data was hard-coded into the app, but the goal was that the data would be fetched from a machine in the future. From the menu, the user was able to close the pop up or get more possible actions by going to an other view. From there it was possible to contact maintenance or order spare parts.

Combination of ARCore and TensorFlow

After testing the possibilities of developing an AR application using both Vuforia and ARCore and in both Unity and Android Studio, the most viable option seemed to be the ARCore and Android Studio combination. This was mainly because the rendered content would hold a fix position in the world space even if the camera wasn't pointed at the tracked image, and it was easier to implement buttons, menus and other interface components natively in Android when using Android Studio.

To be able to recognize a real world object and not just a printed 2D image, the deep learning prototype had to be combined with the ARCore prototype. The idea was to use deep learning for recognizing the machine and ARCore to augment the recognized machine. Combining these two proved to be hard since both ARCore and TensorFlow each requires to run a camera session on Android in order to augment and recognize the machine respectively. This resulted in a prototype where the screen was divided into two camera sessions where one was blacked out, while the other one was working. The application was capable of either recognizing a machine or augment an image, but not augment that which had been recognized by the TensorFlow model. There was also little to no information to be found regarding combining ARCore and a TensorFlow model. Because of this project's time limits, the process of combining TensorFlow with ARCore was not further pursued.

3.1.6 Testing the ARCore and TensorFlow prototypes

The two most promising prototypes were the ARCore prototype and the TensorFlow one. Because these could not be combined both of them would be tested in a small comparative user study. This was done in order to get feedback from Beijer employees about what they considered to be important. The TensorFlow prototype was able to recognize machines effectively and display static information about the machine, just like a regular application. The ARCore prototype was able to present data and controls in a context-based way, which hopefully would allow the users to more intuitively understand what information was related to which machine or machine part. The purpose of this test was to get information about who the actual end-users were, what features might suit better in a AR application than in a Beijer display and how important did the test subjects think it was to be able to actually scan a real machine versus scanning a 2D image of it.

Participants

The test subjects from this user study was employees from Beijer Electronics in Malmö and the number of participants were six. The age range was between 34-58, the gender distribution can be seen in the figure 3.9. The test subjects roles varied from Business developer to support engineer, so their technical competence was varied. The majority of the test subjects had used a Beijer display before in different contexts.

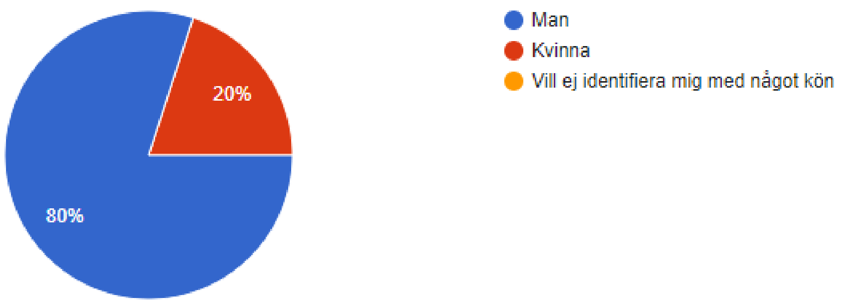


Figure 3.9: Gender distribution from first comparative test.

Test setup

The test subjects were first asked to fill out a survey about their age, gender, role at the company and their past experiences with a Beijer display. Even though they would not be testing a Beijer display it was important to know how the applications they were about to test could function as a complementary interface to the Beijer display and how. After that they got to try out both the ARCore prototype and the deep learning prototype, the order in which the different test persons tried the prototypes varied. When testing the deep learning prototype the test subjects were asked to scan the coffee machine just outside the room and then send an email to maintenance using the application. When testing the AR prototype, the test subjects were asked to scan a picture of the same coffee machine, interact with the menus for the different machine parts and then use the application to send an email to maintenance, as seen in figure 3.10.

After the tests were done the test subjects were asked to fill out a post-test survey about their initial thoughts on the prototypes, if they could be useful and if not why, how easy were they to use, is there any special feature they would like to see in the application and which one they preferred. The scheduled time for one test session was 40 minutes, but the time to finish a session varied between approximately 20-40 minutes, depending on the test subject.

Test results

The test subjects initial thoughts on the prototypes were that they were rudimentary at this stage but that they could see the potential in both of them. Some test participants could not find the menus which were hovering over machine part 1 and 2 while using the ARCore prototype since there were no indicators to show the user that they were actually there. However a majority of the test subjects agreed that the ARCore prototype was both easy to use and that it would be useful to have context-based information instead of having to look through the entire system which you would have to do in a Beijer display. Several test subjects mentioned that it would be valuable to get information and indications if an error had occurred with a machine with the help of an arrow or a sign placed in the area of the error. A majority of the test subjects also thought the deep learning application was useful and that they saw the potential in it. The advantage being that no identifiers such as QR codes or images would be required in order to recognize a machine and this saves the user time.

When asked which prototype the test subjects preferred, the majority answered that the ARCore prototype was more interesting and that there was more to explore with context-

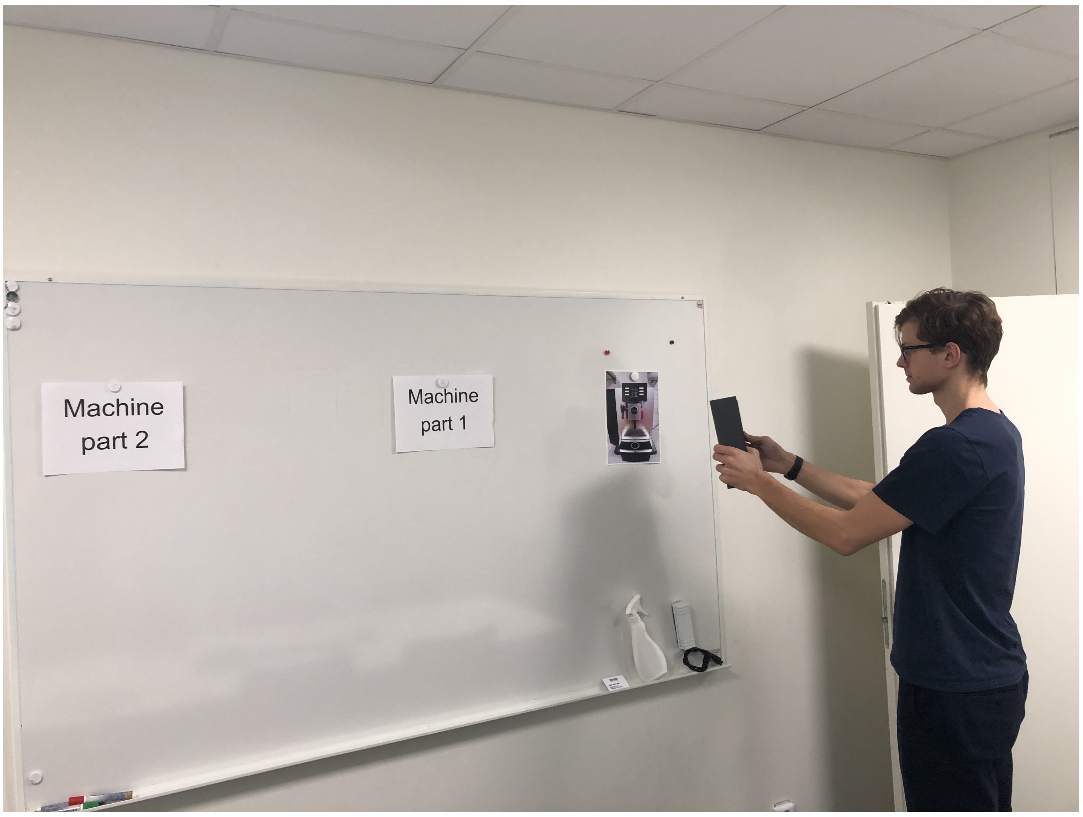


Figure 3.10: Testing the AR prototype.

based information. A clear majority of the test subjects also agreed that the end-users who could benefit from such an application would be service technicians who tend to a machine maybe once a month or when a problem has occurred with it or operators who tend to the machine on a daily basis. Then the context-based information provided by an AR application could guide them towards the error very quickly or in an easy way provide the operators with relevant reading of a machine without having to navigate through the Beijer display.

3.1.7 Phase summary

After examining the results from the user study in this phase it was decided to only move forward with one prototype and that was the ARCore prototype. According to the test subjects the most important and relevant thing was to examine how AR could provide value to the end-user and how the application could function as a complementary interface to the Beijer display.

3.2 Development phase

The goal of this phase is to improve the simple ARCore prototype into a more sophisticated hi-fi prototype which works as a complementary interface to a Beijer HMI called X2 Control. It should be able to control certain parts of a real world machine.

The machine used in the final test is the small electronic elevator shown in figure 3.11.

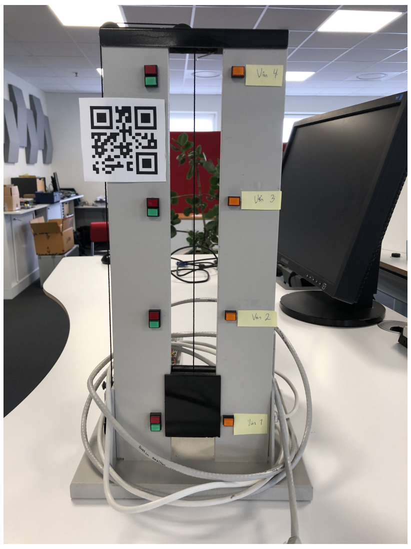


Figure 3.11: Picture of the electronic elevator used in the final test

The elevator is controlled by a programmable logic controller (PLC) which handles the four orange lamps, four red/green lamps and the motor moving a black square (the so called ”elevator”) up and down. By changing certain bits in the PLC, external devices such as a Beijer display can control the elevator. In this section, the development process of the ARCore prototype will be explained and how the communication with the Beijer display and the elevator works.

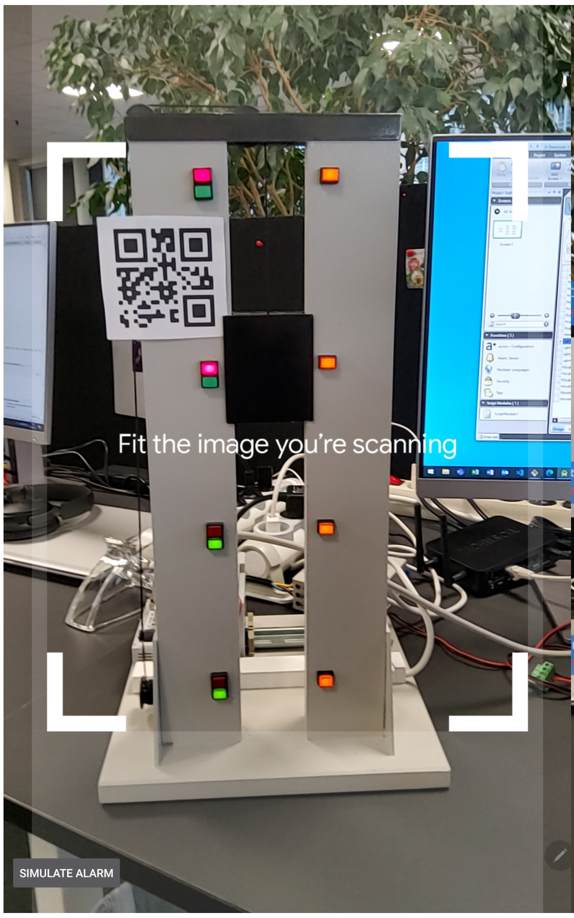
3.2.1 Development of hi-fi prototype

In order to create a hifi prototype which would satisfy the needs for this project there were several things that had to be implemented, these are presented in this list.

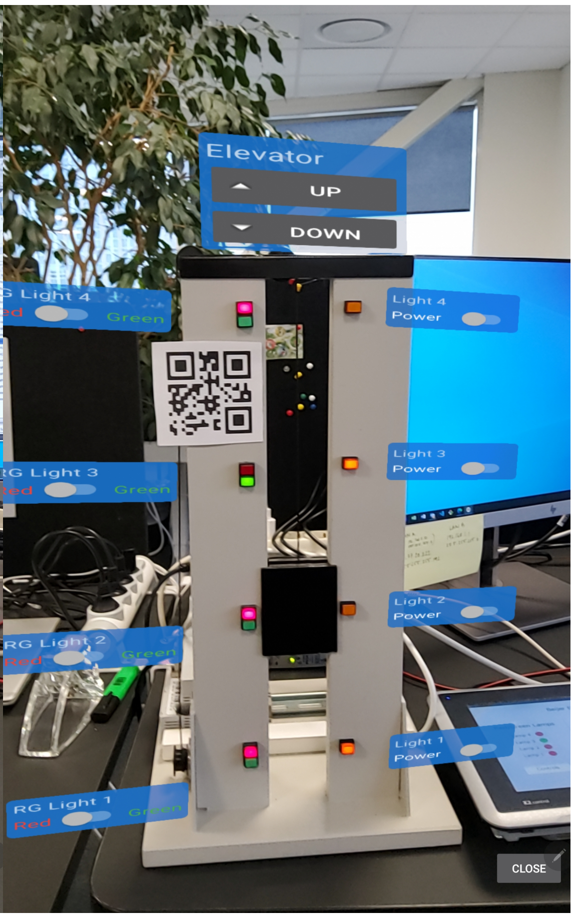
- Remove the old menus and create controls to operate the lights and the elevator.
- Simulate an error with the elevator and provide the user with enough information and guidance to solve the problem
- Allowing the application to communicate with the elevator and get updates about its current state.

To start with the image of the coffee machine was replaced with an image of a qr-code, this was done because the previous image might confuse users. It made no sense to scan a picture of a coffee machine when the user should operate an elevator. Then the old menus were replaced with button switches for each light and each switch was named after the floor it represented. In order to control the elevator two buttons were placed hovering above the elevator with the texts "Up" and "Down", this can be seen in figure 3.12 b.

After the controllers for the elevator had been implemented the focus shifted towards the error handling part of the application. The first screen the user sees when the application is opened up can be seen in figure 3.12 a.



(a)



(b)

Figure 3.12: In figure (a) the first screen the users sees when the application is opened, in figure (b) the controllers for the elevator is shown.

From the first view, the user either scans the qr-code and the controllers for the elevator is shown, or the user presses the "Simulate Alarm"-button which toggles a boolean. If the boolean is true and the user scans the qr-code, a 3D model of an error is rendered along with instructions on how to approach the error, an example of this can be seen in figure 3.13 a.

The instructions given to the user is the distance to the error and an instruction to move the tablet close to error in order to see more information. In order to calculate the distance between the error and the camera, both objects poses had to be found. A pose from the ARCore API is an object's position and orientation to the world around it. From the pose, the x, y and z values can be retrieved. Since the camera was constantly moving, the position had to be updated in every frame. The position of the error was however static. By using the equation seen below, the distance between the two objects could be calculated.

$$x = cameraXPosition - errorXPosition \tag{3.1}$$

$$y = cameraYPosition - errorYPosition \tag{3.2}$$

$$z = cameraZPosition - errorZPosition \tag{3.3}$$

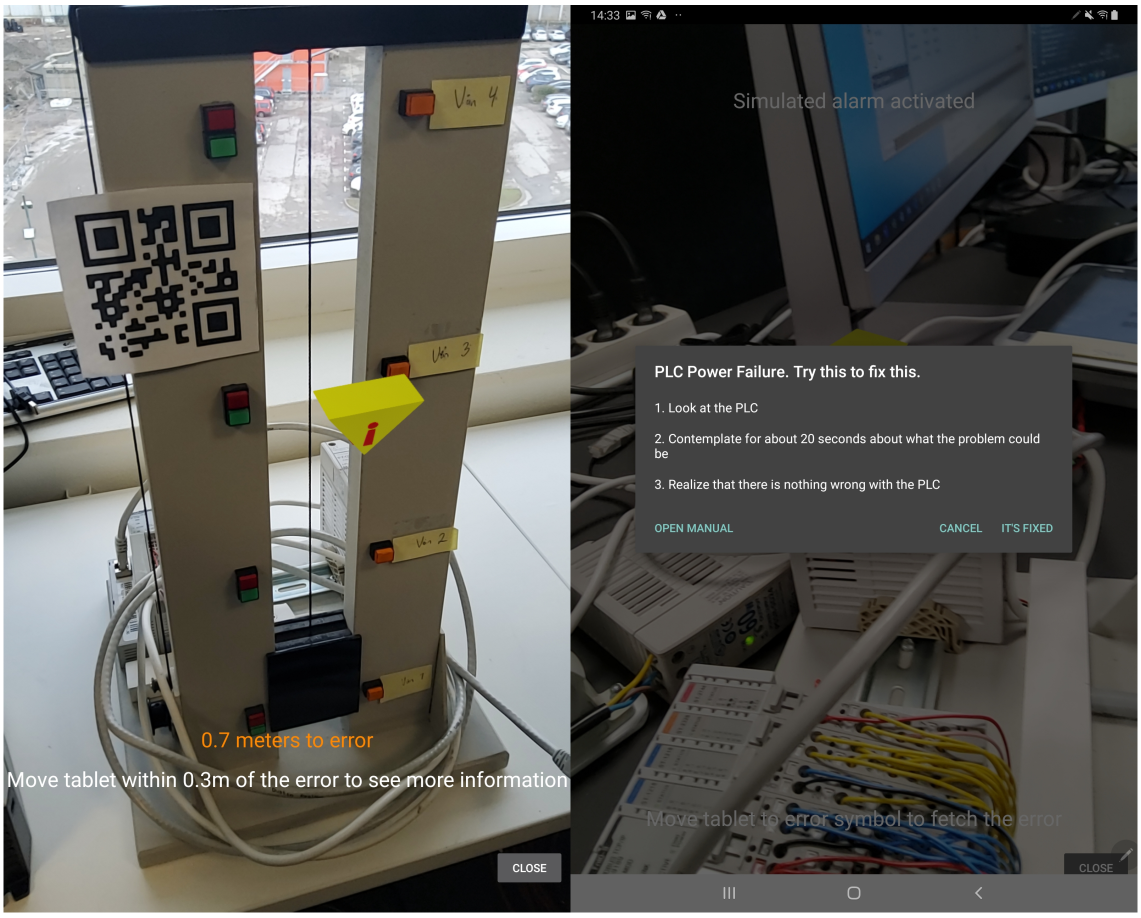
$$distance = \sqrt{x^2 + y^2 + z^2} \tag{3.4}$$

A 3D model of a warning sign and the distance from the error were supposed to guide the user to where the error is located on the machine. The text "Move tablet to error symbol to fetch the error" provides information about how to get more information about the error. The reason for not showing more information directly and letting the user go and "pick up" the error is that there could be many errors occurring simultaneously. The user can then choose to go and pick up any of those errors. If the user moves the tablet within 0.3 meters of the error, a popup window with instructions on how to solve the problem will appear, along with a button to open the manual for the component causing the problem, this can be seen in figure 3.13.

3.2.2 Development of Beijer X2 Control interface

To be able to compare the AR Prototype with something, the elevator (fig. 3.11) had be controlled with something else. Since Beijer Electronics develops HMI terminals, the natural choice was to use one of their displays. The software iX Developer was used to program the display's user interface. With its help, a program consisting of 5 different screens was created, two of which can be seen in figure 3.14.

In the real world, Beijer displays are used on machines with 100s or more different settings, often requiring a rather complex user interface. To simulate the complexity in this project, the controls for the the lamps and elevator were placed on different screens. The home screen, shown in 3.14a, has three big buttons, one to control the red/green lamps, one to control the orange lamps and one to control the elevator. In the top right corner, it has a small button to open the alarm list. When the user presses one of the control buttons, such as the one to control the red/green lamps, an other view is displayed, as shown in figure 3.14b. From here, the user can press buttons to perform actions on the elevator, such as turning a



0.7 meters to error

Move tablet within 0.3m of the error to see more information

Simulated alarm activated

PLC Power Failure. Try this to fix this.

1. Look at the PLC

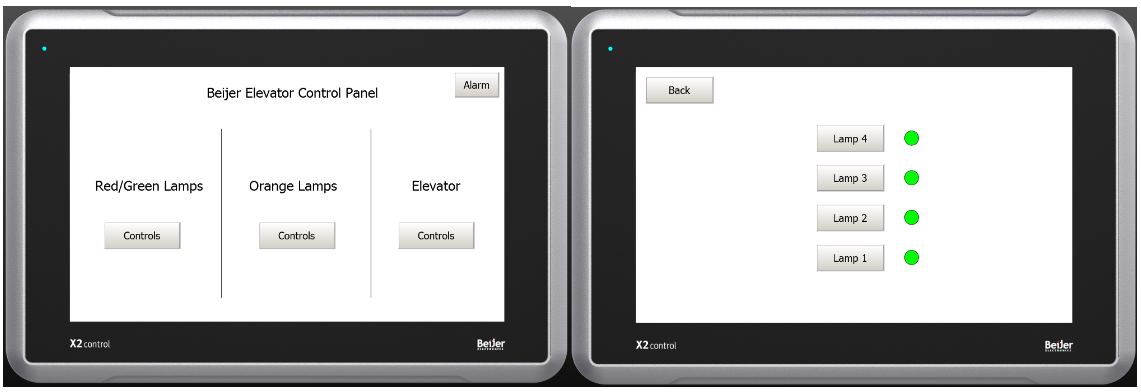
2. Contemplate for about 20 seconds about what the problem could be

3. Realize that there is nothing wrong with the PLC

Move tablet to error symbol to fetch the error

(a) (b)

Figure 3.13: In figure (a) the user is in the simulated alarm sequence of the application, in figure (b) the user has fetched the error and is shown more information about how to solve it.



(a) (b)

Figure 3.14: The interface on the Beijer display. (a) shows the home screen, (b) shows the view for controlling the red and green lamps.

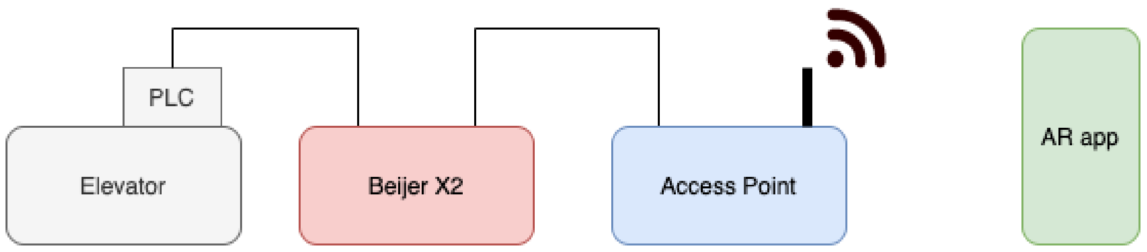


Figure 3.15: The setup of the different devices used during the final test.

lamp on or off. Next to the lamp control buttons, an indicator shows whether the lamp is turned on/red or off/green. The alarm button opens a new view which shows information about a (simulated) alarm, including location (name of the module) and description of the error.

3.2.3 Combining elevator with hi-fi prototype and Beijer X2 Control

To be able to perform actions on the electric elevator, a device would have to toggle bits in the elevator's PLC. Since that is more or less what Beijer's HMIs are built for, it is fairly easy to do with them. In iX Developer, the communication with a PLC is handled with so called tags. A tag includes information such as the bit address and what type of action, such as read or write, should be performed. The Beijer display was connected to the PLC with a ethernet cable and after configuring the IP addresses it was possible to perform actions on the elevator.

Since direct communication between the Android app and the PLC was not possible, the app would have to go via the Beijer terminal. Fortunately, it was possible to set up a web server on the Beijer X2 Pro which made it possible to change tags through HTTP-requests. Since the app only could perform wireless communication, a small wireless network had to be set up. This was done with a Tosibox access point, which was connected to the Beijer display with an ethernet cable. The full setup can be seen in figure 3.15. Now the app was able to read and write tags over WiFi. Methods were added to read the tags on startup and write tags when buttons were pressed in the app.

3.2.4 Phase summary

After developing the hifi prototype to be able to scan the qr-code, control the elevator, handle an alarm simulation and implementing the interface for the Beijer X2 Control using iX Developer the prototype was ready to be tested. By having these two interfaces a comparative test could be conducted between the ARCore prototype and the Beijer X2 Control interface.

3.3 Testing phase

In this section, the procedure for the final user study will be presented along with what equipment was used, the test setup, which participants were involved and what were their objectives. This user study was a comparative test between the ARCore prototype and the interface for the Beijer X2 Control. The results from the test will be presented in the next chapter.

The main purpose of this test was to examine the usability of the ARCore prototype as well as the physical and mental demand required by the test subjects to perform various tasks. Another goal was to compare it to the interface of the Beijer X2 Control and find out if some tasks could be performed better with an augmented reality interface.

3.3.1 Participants

The number of test subjects in this user study was 20, half of which were Beijer employees and the other half was non-employees with no previous experience of using a Beijer display. The participants had varied technological competences. The reason for having half of the test subjects being non-employees was to examine whether any previous experience of operating a Beijer display would have any effect on solving the test objectives.

3.3.2 Equipment

The equipment used for this test are presented in the following list.

- Electronic elevator
- 2 laptops
- Beijer X2 Control
- TOSIBOX Lock 150 - Remote access and networking device
- 2 Smartphones
- Samsung Galaxy Tab S6

3.3.3 Roles

Two persons are involved in the testing, a test leader and an observator.

- Test leader had the following responsibilities
  - Explaining how the test will be conducted, what is to be tested and why.
  - Guiding the test subject through the test session
  - Recording the whole test session
- Observator had the following responsibilities
  - Timekeeping of each subtask done during the test session
  - Providing the test subjects with all the necessary surveys.
  - Notify the test leader when the test subject finishes a task.

3.3.4 Location and dates

Beijer Electronics, Stora Varvsgatan 13, 211 24 Malmö. The tests were done between the dates of 17/2-2020-21/2-2020.

3.3.5 Procedure

The test was done in a closed room with only the test leader, observator and test subject present. In order to be able to use the results from the user study later the participants were asked to sign an informed consent, they were also given an unique id. The pre-test survey contained questions such as the gender, age, if they were a Beijer employee or not, and previous experience with either AR or a Beijer display. The test subjects would start to test either the ARCore prototype or the Beijer X2 Control, this was done in order to counterbalance the effect that a specific order might have on the outcome of the test. By varying the test order the chances of the order influencing the result are reduced. After each prototype was tested the test subjects filled out both a NASA TLX survey and a SUS. At the end of the test they filled out a post-test survey, the reason for not conducting an interview was that the test subjects might give more honest answers if they were not asked directly about which interface they preferred. The post-test survey consisted of questions about which interface the test subjects preferred when performing each test objective and why they preferred one over the other.

- Informed consent
- Pre-test survey
- Test AR Prototype
- Fill out SUS about ARCore prototype
- Fill out NASA TLX about ARCore prototype
- Test Beijer X2 Control
- Fill out SUS about Beijer X2 Control
- Fill out NASA TLX about Beijer X2 Control
- Post-test survey

3.3.6 Test objectives

In this sub section the different task objectives for testing the ARCore prototype and the Beijer X2 Control will be presented along with the introductions given to the test subjects by the test leader for each objective. To start with all the lights on the elevator were turned off, the elevator started at the first floor or third floor depending on which interface the test subject was to start with. In order to avoid transfer of learning during the test session both the component which caused the alarm and the lights which were to be turned on or off were different for the ARCore prototype and the Beijer X2 Control.

ARCore prototype

1. Change the colors of the lamps - Scan the QR-code and let us start with the lamps on the left side.
    - Left side
        - The lamp on floor 4 should be green
        - The lamp on floor 3 should be red
        - The lamp on floor 2 should be green
        - The lamp on floor 1 should be red
    - Right side
        - The lamp on floor 4 should be turned off
        - The lamp on floor 3 should be turned on
        - The lamp on floor 2 should be turned off
        - The lamp on floor 1 should be turned on
2. Control the elevator - move the elevator to the third floor.
3. Error handling - We will now test to solve a error that has occurred with the elevator. First, I want you to press close in the right corner. Then press simulate a alarm and scan the image again.
    - Find which component on the elevator triggers the alarm.
    - Find the manual for the faulty component.

Beijer X2 Control

When the test subject is supposed to find the malfunctioning component in this test, the participant was handed a computer with the Google page open. This was done to simulate how operators in actual factories search for the manual for different components in real-life situations.

1. Change the colors of the lamps.
    - Left Side
        - The lamp on floor 4 should be red
        - The lamp on floor 3 should be green
        - The lamp on floor 2 should be green
        - The lamp on floor 1 should be red
    - Höger sida
        - The lamp on floor 4 should be turned on
        - The lamp on floor 3 should be turned off
        - The lamp on floor 2 should be turned off
        - The lamp on floor 1 should be turned on

2. Control the elevator - move the elevator to the first floor.
3. Error handling - We will now try to solve an error that has occurred with the elevator, press the button where it says Alarm.
    - Find which component on the elevator triggers the alarm.
    - Find the manual for the faulty component by using Google.

3.3.7 Phase summary

After testing and comparing the hifi prototype with the Beijer X2 Control interface all of the test results had to be summarized. By using many surveys and letting the test subjects perform the same kind of tasks in two different interfaces and in different ways many interesting facts were found which are presented in the next chapter.

Chapter 4
User study results

In this chapter, results from the user study are presented. It is divided into the following sections: pre-test survey, SUS, NASA-TLX, timekeeping of task completion, post-test survey and error sources.

4.1 Pre-test survey

The pre-test survey consisted of a number of general questions about the test subject, such as gender, age, occupation, AR experience and Beijer display experience. The results can be seen in figure 4.1, 4.2, 4.3, 4.4, 4.5 as well as table 4.1.

Table 4.1: Occupation of test subjects.

Occupation	Count
Student	10
Support (technical/customer)	4
Business development	2
Software development	1
CTO	1
UX design	1
Product management	1

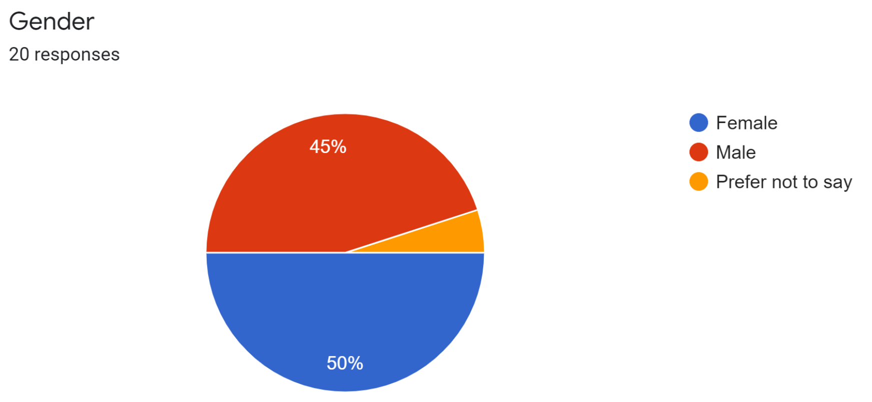


Figure 4.1: Gender distribution of test subjects.

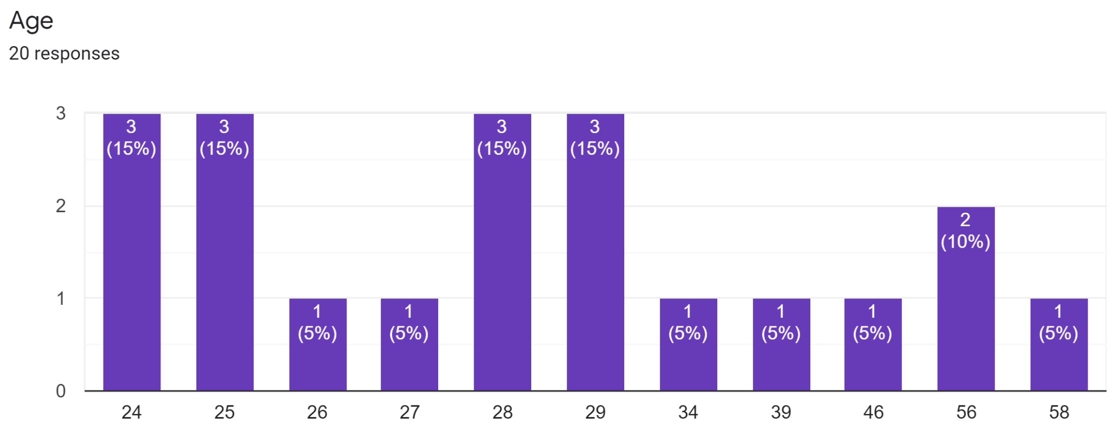


Figure 4.2: Age distribution of test subjects.

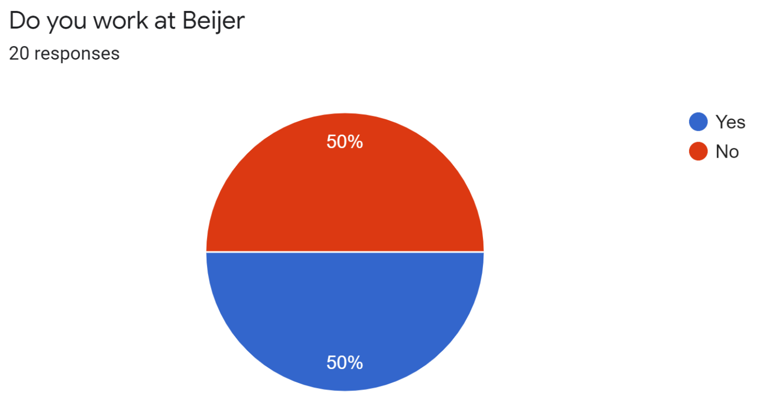


Figure 4.3: Distribution of Beijer and non-Beijer employees.

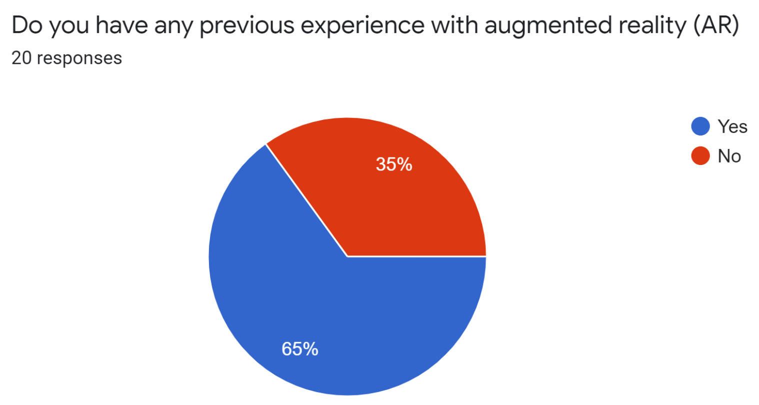


Figure 4.4: Distribution of AR and no AR experience.

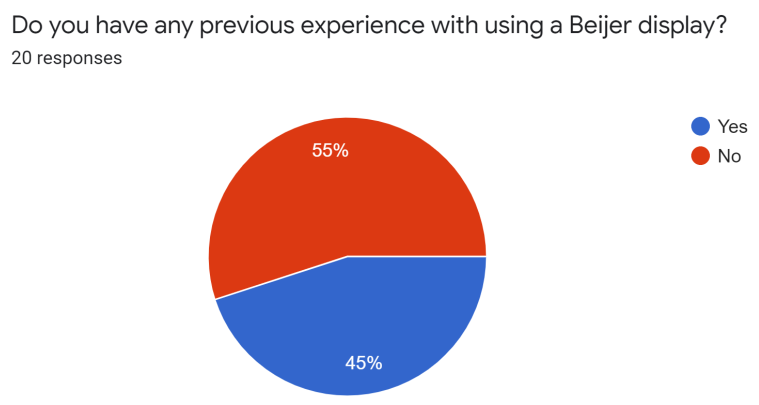


Figure 4.5: Distribution of Beijer display and no Beijer display experience.

4.2 SUS

The scale consists of 10 predefined questions with five responses ranging from Strongly agree to Strongly disagree. An open question was added to the end of the questionnare which asked the participants to provide a comments about the system, which was optional. This was done for both the ARCore prototype and the Beijer X2 Control.

The average score for the ARCore prototype and Beijer X2 Control are presented in figure 4.6. A SUS score above 80.3 is considered to be excellent and a score between 68-80.3 is considered to be good. In figure 4.7, the average score for the five positive statements are presented. The higher the score, the better. In figure 4.8, the average score for the five negative statements are presented. The lower the score, the better.

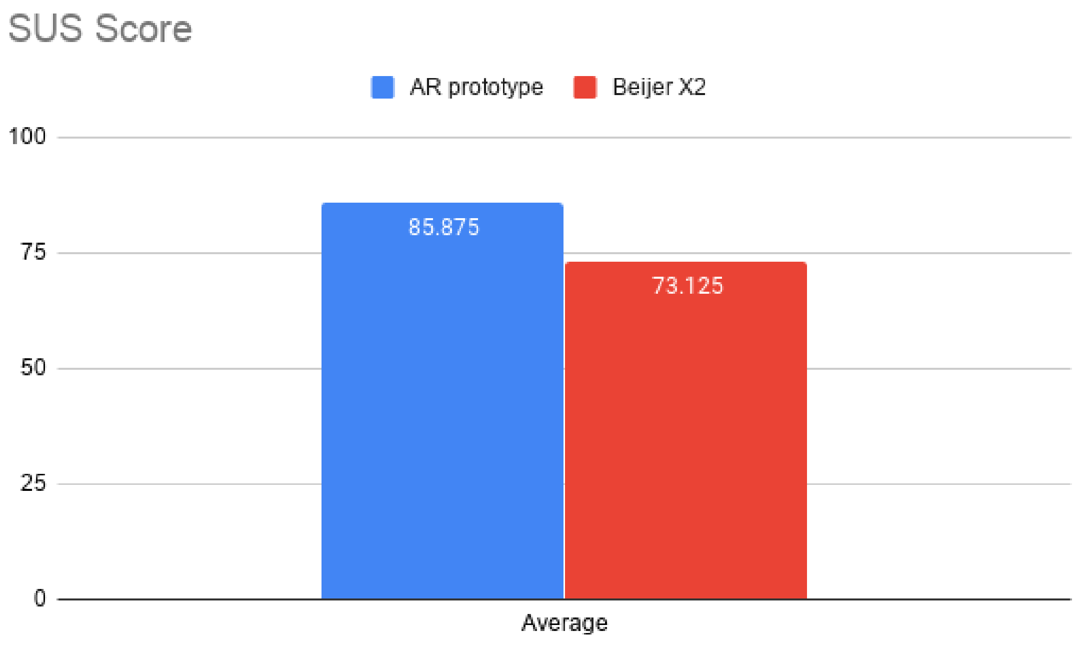


Figure 4.6: Average SUS score for both AR prototype and Beijer X2 Control.

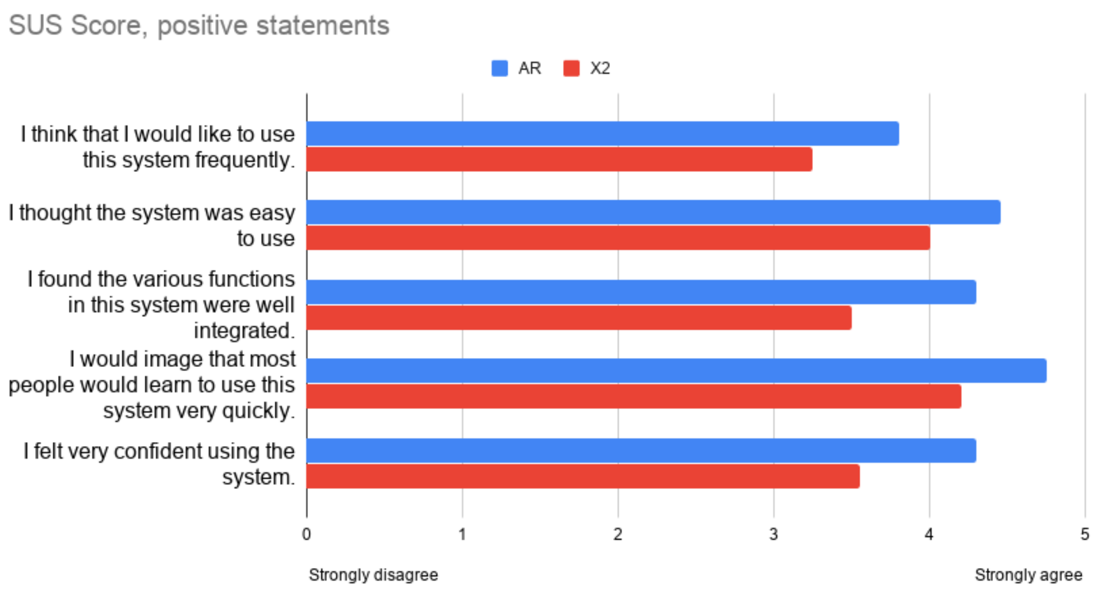


Figure 4.7: Average SUS score for the five positive statements.

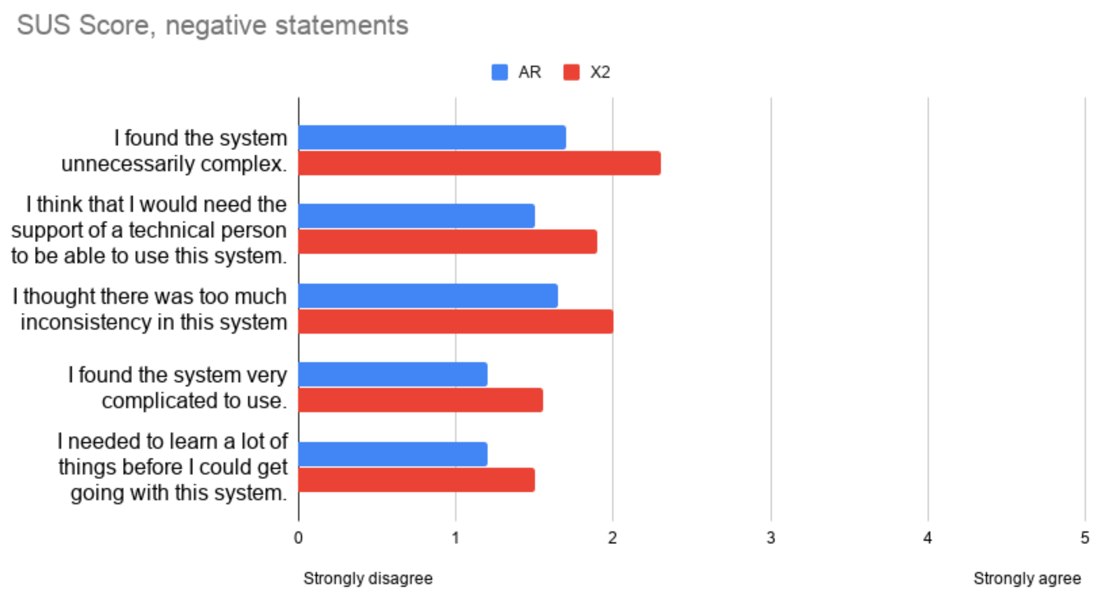


Figure 4.8: Average SUS score for the five negative statements.

Comments about the ARCore prototype

One participant thought that it was very useful to see everything in AR and felt that it was easier to use than the Beijer X2 Control, however two participants mentioned that the QR-code recognition was troublesome. Another test subject mentioned that the "AR objects" got a bit disoriented and got stuck in the wrong place, which made the system feel a bit unstable, however the same participant also thought that it was easy to find the faulty component and the correct manual. Three participants thought that the application was nice, intuitive and easy to use.

Comments about the Beijer X2 Control

One participant mentioned that it was cool to control the elevator, but that it was a bit unclear how to toggle the lights since they only had to press a button, the participant would have preferred a switch instead. Another participant thought that it was unnecessarily hard to find the faulty component during the test and that this could be made more obvious. One participant thought that the buttons on the X2 Control were not as intuitive as in the ARCore prototype and that the troubleshooting was more complex.

4.3 NASA TLX

NASA TLX is a measure of the perceived workload based on six dimensions, mental, physical and temporal demand, as well as perceived performance, effort and frustration. A lower average score is better than a higher. For the perceived performance dimension, the scale ranges from "perfect" (a low score) to "failure" (a high score). For the other dimensions, the scale ranges from "very low" to "very high". In figures 4.9 and 4.10 total average score and the scores for each of the dimensions for each prototype are presented respectively.

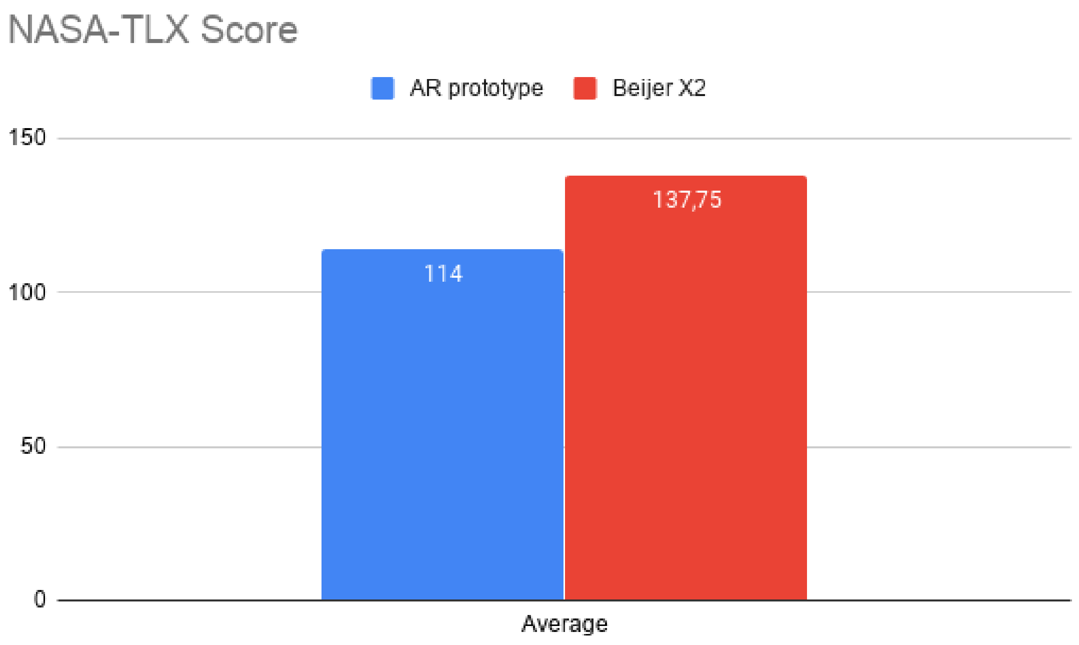


Figure 4.9: Average NASA TLX score for both AR prototype and Beijer X2 Control.

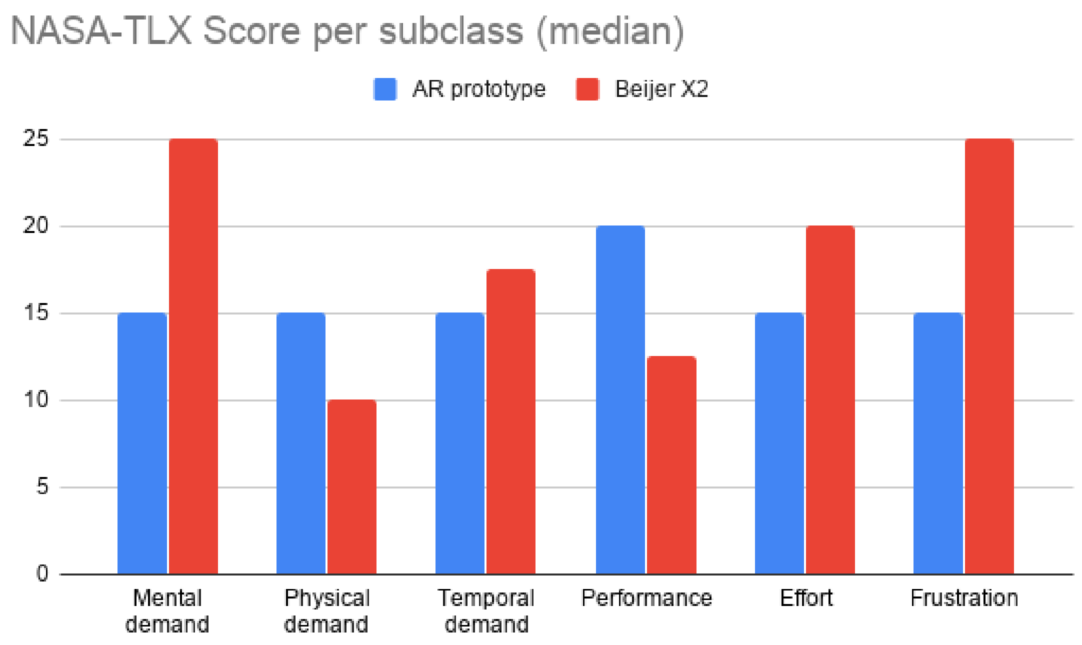


Figure 4.10: NASA TLX score for both AR prototype and Beijer X2 Control for each dimension.

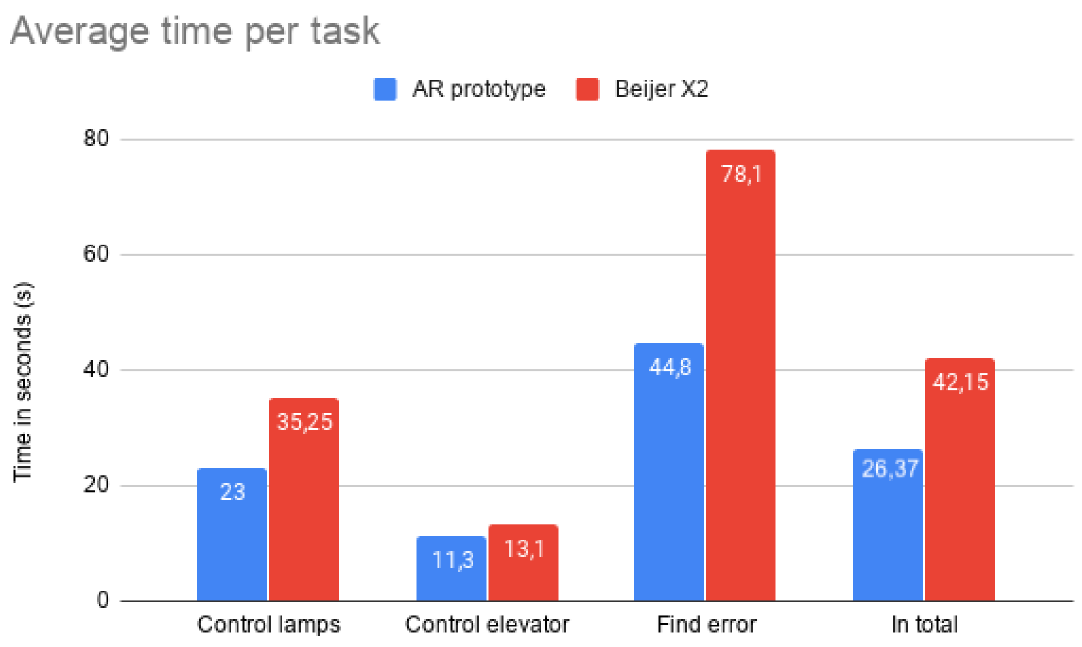


Figure 4.11: Average time per task for both AR prototype and Beijer X2 Control.

4.4 Timekeeping of task completion

In this section the average times for each task will be presented. During the test each sub-tasked was timed in order to see if there were any differences between the Beijer X2 Control and the ARCore prototype interface. The data is presented in figure 4.11.

- Task 1 - Toggle lights on both the left and right sides.
- Task 2 - Move the elevator to a specific floor.
- Task 3 - Find faulty component and manual to the component.

4.5 Post-test survey

The final part of the test session was the post-test survey in which the participants were asked to fill out their preferred interface for each of the tasks they had performed and give a reason as to why they preferred it. The reason for using a survey was that it was more likely that the test subjects were to give their honest opinion about the interfaces since the questions were straightforward and not so open. Figures 4.12, 4.13, 4.14, 4.15, 4.16, 4.17 and 4.18 show which interface the test subjects preferred to use for each task and they are based on test subjects with different gender, previous AR experience and previous Beijer display experience.

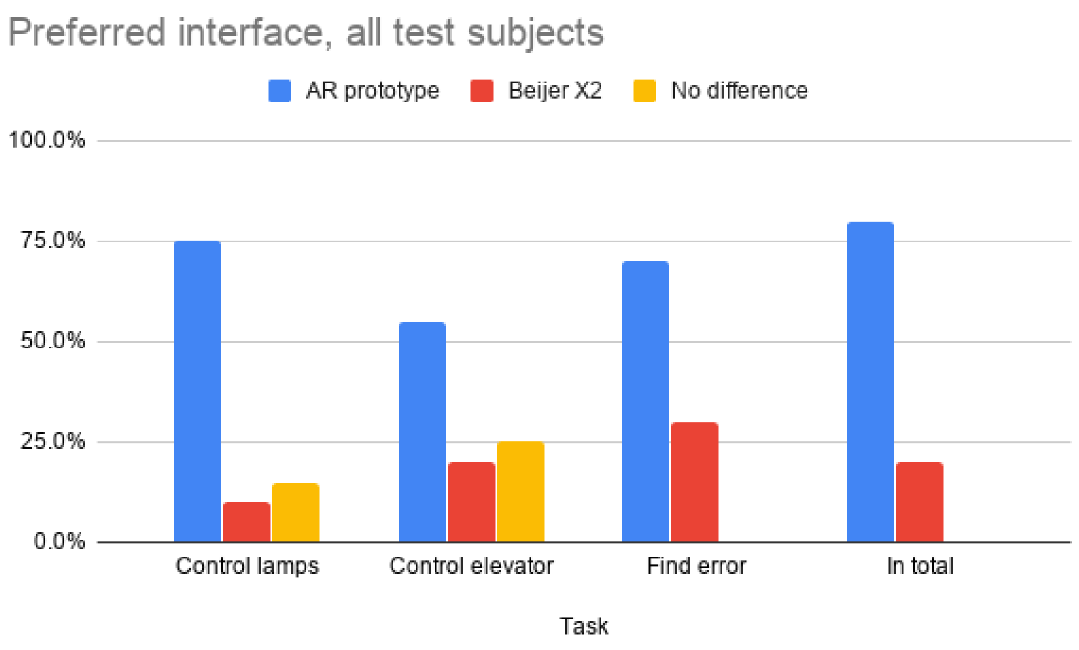


Figure 4.12: Preferred interface for each task, overall

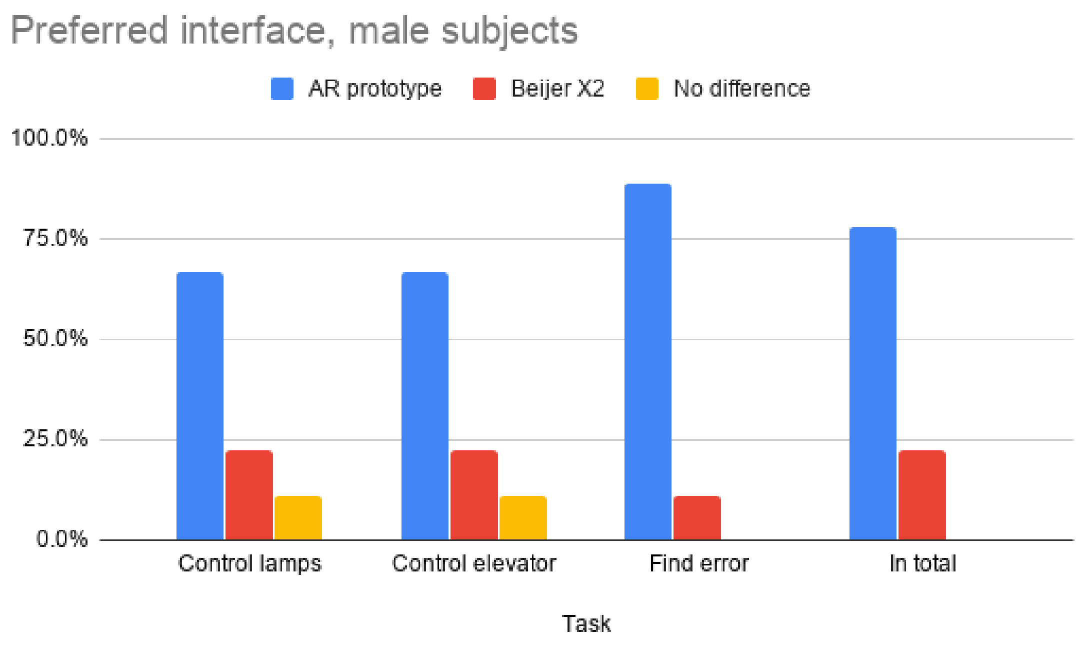


Figure 4.13: Preferred interface for each task, test subjects which were males.

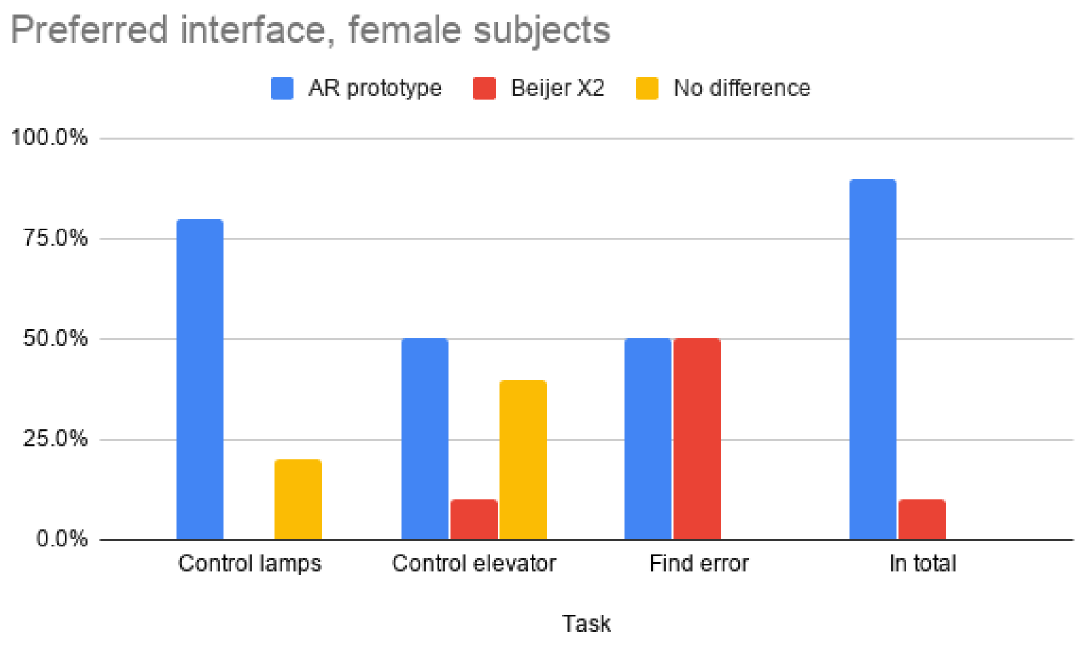


Figure 4.14: Preferred interface for each task, test subjects which were females.

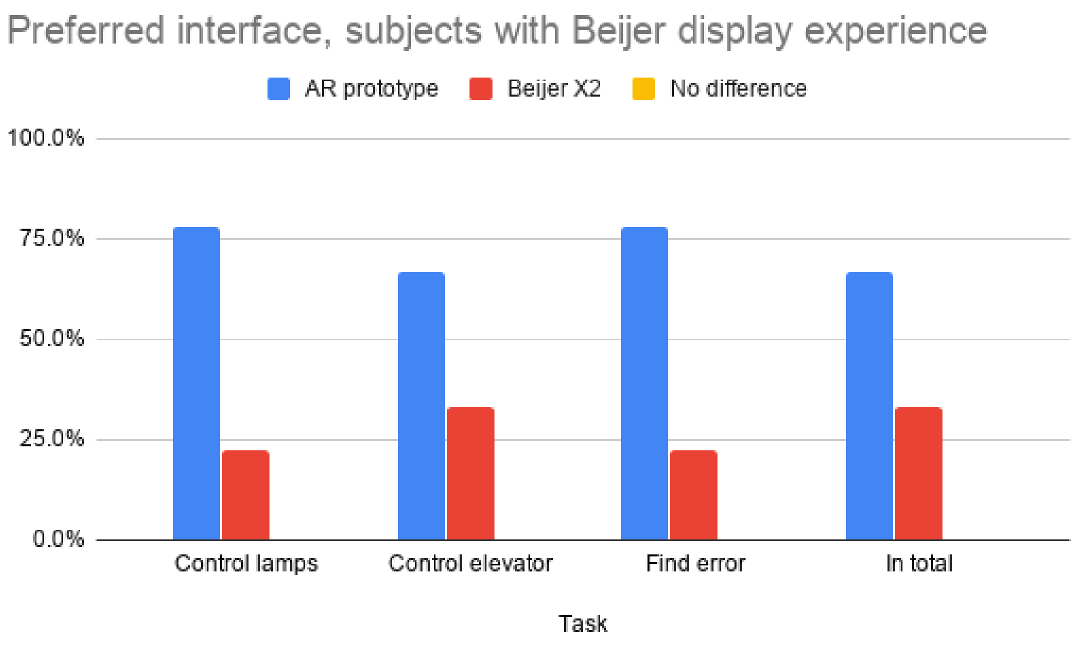


Figure 4.15: Preferred interface for each task, test subjects which had previous Beijer display experience.

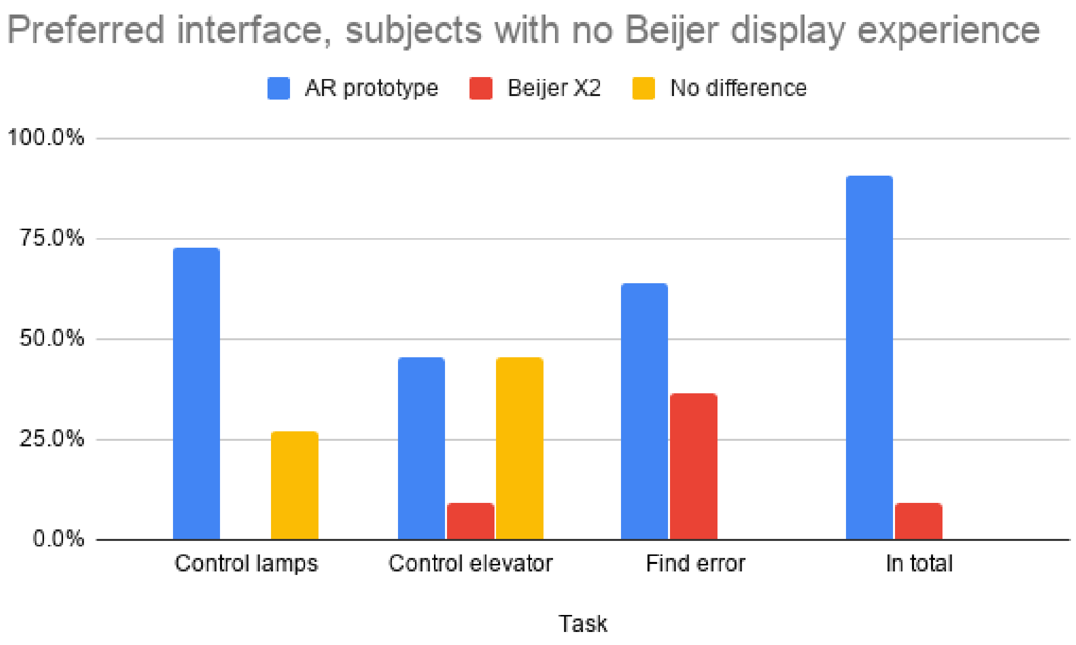


Figure 4.16: Preferred interface for each task, test subjects which had no previous Beijer display experience.

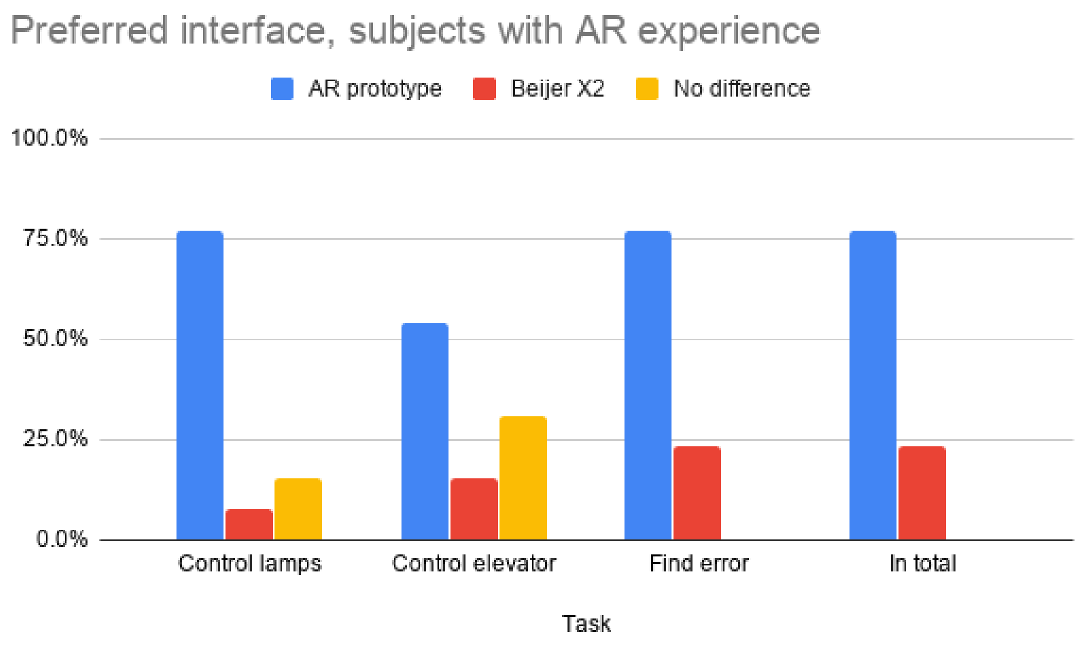


Figure 4.17: Preferred interface for each task, test subjects which had previous AR experience.

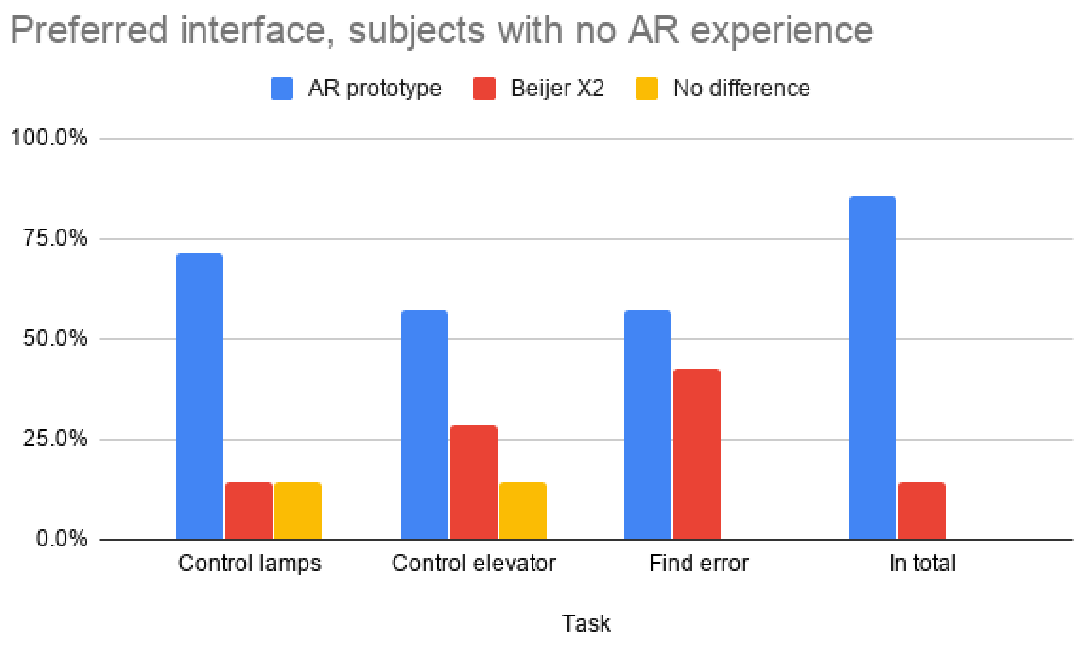


Figure 4.18: Preferred interface for each task, test subjects which had no previous AR experience.

4.6 Error sources

The part that was most prone to error caused by human failure was the timekeeping of each task. The role of observer was changed every day during the test sessions and it was the observers role to keep the time during the test. Because of this, the observers might have started the stopwatch earlier or later than the other observer for different test subjects.

The task which the test subjects were to search for the manual for the faulty component using Google was also prone to error since the search history from previous test sessions remained for a few tests, which resulted in a faster time for some participants.

During the test with the AR prototype, the indicator showing where the error is located on the machine was not always working as expected. For some test subjects, the indicator was not at the exact right position, causing some confusion around which component it pointed towards.

Finally, there were some participants from Beijer Electronics who had participated in the initial test with the former prototype, and they might have done the test faster since they knew what kind of application it was.

Chapter 5
Discussion

In this chapter discussions will be held about whether the results have answered the research questions, what flaws does the ARCore application have and what improvements could be made. There will also be an evaluation of the design process used in this project as well as suggestions on future work with this sort of application.

5.1 Discussion of result in relation to research questions

In this section the results from the test sessions will be thoroughly analyzed and the data will be used to answer the research questions for this project. The research questions are presented again in this list:

- Could an augmented reality interface work as a complementary interface to a traditional touch user interface?
- What are the benefits and drawbacks of using an augmented reality interface in this context?
- What kind of data is suitable for an augmented reality interface?

What are the benefits and drawbacks of using an augmented reality interface in this context?

Based on the findings from the user study one of the main drawbacks of the AR prototype is the physical demand and the perceived performance of success in comparison to the Beijer X2 Control interface, which can be seen in 4.10. To start with, the QR-code scanning process performance in the application was varied, sometimes it took a long time before the test subjects managed this successfully. This might have led to test subjects feeling that they were less successful in performing this task. It also resulted in more physical demand since the test subjects had to hold the tablet upright and point it towards the QR-code. Furthermore, the test subjects had to keep the camera pointed towards the machine for the duration of the test, and when they were to find the faulty component, the test subjects had to stand up and look around the elevator. This resulted in a higher perceived physical demand when using the ARCore prototype compared to the Beijer X2 Control. The physical demand might have been lesser if the form factor had been different for the ARCore prototype. By using a Microsoft Hololens or a Magic Leap, the test subject would only be required to turn their head towards the elevator, therefore reducing the physical demand.

Another factor which might have caused the perceived performance to be less when using the ARCore prototype was when the test subjects were asked to find out more information about the faulty component. In order to get more information, the test subject had to bring the tablet within a distance of 0.3 meters, several test subjects missed this information, causing the perceived performance to drop. Eight test persons mentioned that they would rather press on the faulty component instead of moving closer and fetching the error. One test person mentioned that going to close to a faulty component might even be dangerous in a real-time situation and that some machines have security zones around them, if the user steps inside of that zone the whole machine stops resulting in a stop in the production.

The benefits of using an AR interface was that the data was presented in a contextual manner, meaning that the test subjects clearly saw what data and which controls were related to which machine part in real-time. The AR interface raised the user's situational awareness, meaning it helped them to more easily create a mental model of the environment and understand the situation. The data is not presented in this way in the Beijer X2 Control, the test subjects had to press a button and turn their head to see if the light had turned on or off, whereas they could see this directly in the ARCore prototype. Ten test subjects mentioned in the post-test interview that they thought that it was easier to use the ARCore prototype to control the lights because they could see the results of their interactions in the same interface and in the real world. Eleven test subjects mentioned the same thing regarding the second task of the test session, when they were asked to move the elevator to a specific floor.

A major benefit of using AR in this context can be seen when the test subjects were asked to find the faulty component during the test session. According to figure 4.12, almost 70% of all the test subjects preferred the ARCore prototype interface to find the error with, even though 50% of the test subjects works at Beijer. Thirteen test subjects mentioned in the post-test survey that it was easier to find the faulty component with AR since the error symbol was visible and clearly marked the component. In comparison, the Beijer X2 Control alarm only gave the user text information about which component had caused the error, this proved to be a problem for the test subjects which had no previous experience with working with these kind of components.

What kind of data is suitable for an augmented reality interface?

When analyzing the results in figure 4.12 from the post-test interview, one interface stood out. In both task 1, controlling the lights, and task 2, controlling the elevator, there were subjects which thought that there was no difference between the Beijer X2 Control and the ARCore prototype. But every test subject chose either the ARCore prototype or the Beijer

X2 Control when they were asked which interface they preferred to do the third task, finding the error. According to the results, 70% of the test subjects preferred the ARCore prototype, mostly because the prototype provided them with contextual data about where the problem was situated, as opposed to just being told in text. And when looking at figure 4.11 the biggest time difference between the two interfaces was task 3, the difference being on average 33,3 seconds. The test subjects performed all of the tasks faster on the AR prototype, but the difference between the interfaces was largest when performing task 3.

The most suitable data for an augmented reality interface in this context seems to be information regarding the location of errors. For users who are not used to these kind of components and error searching in a Beijer display, the time efficiency was greatly increased when the data was presented to the test subjects in a contextual manner in relation to the actual faulty component.

Could an augmented reality interface work as a complementary interface to a traditional touch user interface?

An AR user interface could possibly work as a complementary interface to a traditional touch user interface in some aspects. As mentioned before in this discussion, the most obvious aspect in which an AR interface makes sense is the error handling. To be able to provide the user with quick, relevant and contextual information as soon as an error occurs might decrease the time until it is fixed. And when looking on results from both the NASA TLX and the SUS, the ARCore prototype scored better than the Beijer X2 Control, apart from the physical demand and the perceived performance measured in the NASA TLX. However, the mental and temporal demand were lower when using the ARCore prototype as well as the perceived effort and frustration. The average time per task was also less when using the ARCore prototype. Even though the ARCore prototype generally got better scores than the Beijer X2 Control interface it does not mean that it should be completely replaced with an AR interface at this stage. There are areas in which AR makes sense to use. To answer this research questions: yes it could be used as a complementary interface to the Beijer interface and would most likely make a big difference in reduction of time when it comes to handling errors.

5.2 Discussion on flaws and improvements

To develop the AR prototype on Android, the SDK AR Core was used. It worked very well most of the time, especially the tracking functionality (keeping track of the surroundings by using sensors and remembering where it is) worked well - once the elevator was recognized the AR elements stayed at their positions even if the user turned away or walked around in the room. However, since the structure of the elevator was somewhat irregular and contained a big hole, it felt like ARCore sometimes lost track of its surface resulting in the AR elements moving unexpectedly.

An other thing that did not work especially good with ARCore was the recognition of the QR code. To recognize an image, ARCore's Augmented Image API was used. To scan the QR code, users sometimes had to move the tablet back and forth for a couple of seconds before the app would recognize it. Compared to conventional QR scanners - which often recognize a QR code in under one second - this behaviour was a bit annoying and probably affected the overall impression of the AR prototype.

An issue that was raised by some test subjects was that the two interfaces used in the user test were not completely equal in complexity. As mentioned earlier, the user interface of the Beijer display was designed to imitate how it might look like in a display used on a big machine with tens or hundreds of different settings. In addition, the author's iX Developer experience and knowledge of how the displays are used in real life are very limited. This might have impacted the end result to the Beijer display's disadvantage.

Finally, some of the test subjects that did not work at Beijer had a problem with seeing the purpose of controlling the elevator and turning the lamps on and off. The goal was to simulate how a Beijer display is used together with a real machine. A bigger or more "real" machine might have made the purpose more clear.

5.3 The design process

The design process used in this project was divided in three phases, the concept phase, the development phase and testing phase. This has worked very well during the course of the project because each step in the process has been associated with a particular phase, this made it easy to differentiate between what things needed to be done right away, what things needed to planned for and what could be expected to be done in the later stages of the project. Something specific for this project was that the programming started almost right away, because the techniques which were available had to be tested as soon as possible in order to figure out what could be done with this project. So there was never paper prototyping done, instead several simple prototypes were developed using different techniques.

The idea of combining machine learning with augmented reality came early in the process because the machines had to be recognized in some way. So the work was split between developing one machine learning prototype and several AR prototypes using various libraries and tools. The best AR prototype would then be combined with the trained deep learning model. By the end of the concept phase it was made clear that the combination of machine learning and AR would not be easy and would take a lot of time, so its importance had to be reconsidered. And when evaluating the research questions, a smooth way of recognizing a machine was actually not important to be able to answer the questions which were being asked. Instead the focus shifted towards how could an augmented reality interface serve as a complement to the already existing interface in the Beijer X2 Control. And what data could the user benefit most from when being presented with an AR interface. By performing a test and asking Beijer employees what their thoughts were several ideas were brought into the development phase which would be implemented in the hi-fi prototype. If the choice of abandoning the machine learning part of the project had been considered sooner the first tests could have been done sooner, but it took some time to evaluate if it was relevant to the project or not.

During the development phase continuous communication was held with Beijer employees which was very helpful because they both provided ideas and help when the prototype had to be able to communicate with the elevator.

Finally, the test sessions was planned with the help of the authors advisor from the university to establish what kind of measurements and questions would be suitable to test this prototype and at the same time get enough information to answer the research questions. Half of the test subjects were Beijer employees and the other half were not, this was done in order to get some feedback from people with no previous knowledge of a Beijer display or connection to the company. The test sessions were held at Beijer HQ in Malmö during five consecutive days. After which the results were analyzed and summarized into graphs.

5.4 Suggestions on future work

A major improvement of the AR prototype would be if it was able to recognize a machine solely based on how it looks. The issue with combining a image recognition model with ARCore has been thoroughly discussed in this text and requires high technical expertise and time exceeding the scope of this project. However, having this functionality would drastically improve the user experience. There exists SDKs, such as Vuforia, that have this functionality. However, Vuforia's object recognition only works on small objects sized not much bigger than a banana.

Evaluating the advantages as disadvantages of the AR Prototype and Beijer display on a much bigger machine would be interesting to see. Having a small machine such as the elevator used in this project did not leverage AR's full potential. With a big machine it would probably become much more apparent what a difference contextual data might make.

Chapter 6

Conclusions

Augmented reality is an emerging technology and its potential is being evaluated in many sectors. This project focused on how to create an AR application which could serve as a complementary interface to a Beijer display and in what capacity would it be able to serve as such an interface.

An AR prototype was created which was able to visualize real-time data and controllers to a small electronic elevator. It was capable of moving the elevator up and down, turning on lights and providing the user with relevant information when an error occurred with the elevator. An interface with the same capabilities was developed for the Beijer X2 Control in order to test which features worked best in an AR interface and what worked best in a more traditional touch user interface.

By testing the interfaces against each other it was clear that the strength of augmented reality in this scenario was to give the user context based information. This was especially obvious when presenting the user with a visual representation of an error, as opposed to giving the user text based information about the error as was given in the Beijer X2 Control interface. In general most of the users felt that it was easier to control the elevator and handle errors with it, using an AR interface. The application could however be improved, for instance the controllers and the visualisation of the error sometimes rendered in different places, which caused the users to be confused about which controller controlled what on the elevator. And it must be taken into consideration that in comparison to the Beijer X2 interface, the AR application demands more of user physically, since it demands that the user is pointing the device against the elevator for while using the application.

In conclusion, the solution worked well as a complementary interface and it showed the possibility of using augmented reality in an industrial environment as a time saver when solving errors with a machine. But the tools to develop a more general solution capable of being used on a large scale might not be possible yet because the technology is not mature enough.

References

[1] C. Abras, D. Maloney-Krichmar, and J. Preece. "User-Centered Design". http://citeseerx.ist.psu.edu/viewdoc/downloaddoi=10.1.1.94.381&rep=rep1&type=pdf, SAGE Publishing, 2004. Accessed on: November 12, 2019.

[2] Apple. "Get Ready for ARKit 3". Available: https://developer.apple.com/augmented-reality/arkit/. Accessed on 2020-01-15.

[3] M. Arvola. *"Interaktionsdesign och UX - om att skapa en god användarupplevelse"*. 1th ed. Lund, Sverige: Studentlitteratur AB, 2016.

[4] R.T Azuma. A survey of augmented reality. *Presence: Teleoperators & Virtual Environments*, 6(4):355–385, 1997.

[5] J. Brooke. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[6] F. Chollet. *"Deep learning with Python"*. Manning Publications, Shelter Island, NY, 2017.

[7] B.J. Copeland. "Artificial intelligence". https://www.britannica.com/technology/artificial-intelligence, Nov 2019. Accessed on 2019-12-16.

[8] J. Dean, R. Monga, et al. "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems". *CoRR*, abs/1603.04467, 2016.

[9] Beijer Electronics. "Our Company". https://www.beijerelectronics.com/en/About___us/Our___company. Accessed on 2019-03-08.

[10] B. Furht. *"Handbook of Augmented Reality"*. 1th ed. New York, NY: Springer New York, 2011.

[11] I. Goodfellow, Y. Bengio, and A. Courville. *"Deep Learning"*. MIT Press, 2016. http://www.deeplearningbook.org.

[12] Google. "Fundamental concepts". https://developers.google.com/ar/discover/concepts. Accessed on 2020-01-14.

[13] Google. "TensorFlow Lite". https://www.tensorflow.org/lite. Accessed on 2020-02-26.

[14] Google. "What is transfer learning?". https://www.tensorflow.org/js/tutorials/transfer/what_is_transfer_learning. Accessed on 2019-11-22.

[15] J.D. Gould and C. Lewis. "Designing for Usability: Key Principles and What Designers Think.". *Communications of the ACM*, vol. 28, no. 3: pp. 300–311, Mar 1985.

[16] M. Heller. Deep learning vs. machine learning: Understand the differences. https://www.infoworld.com/article/3512245/deep-learning-vs-machine-learning-understand-the-differences.html, Jan 2020. Accessed on 2020-01-09.

[17] A.G. Howard and M. Zhu. "MobileNets: Open-Source Models for Efficient On-Device Vision". https://ai.googleblog.com/2017/06/mobilenets-open-source-models-for.html, Jun 2017. Accessed on 2019-11-29.

[18] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv e-prints*, page arXiv:1704.04861, April 2017.

[19] R. Ihsan and U. Sehat. A Survey on Augmented Reality Challenges and Tracking. *Acta Graphica*, vol. 24:pp 29–46, Feb 2013. [Online]. Accessed on 2020-01-13.

[20] PTC Inc. "Vuforia: Market-Leading Enterprise AR". https://www.ptc.com/en/products/augmented-reality/vuforia. Accessed on 2019-11-22.

[21] N. Jouppi. "Google supercharges machine learning tasks with TPU custom chip". https://cloud.google.com/blog/products/gcp/google-supercharges-machine-learning-tasks-with-custom-chip, May 2016. Accessed on 2020-02-25.

[22] Keras. "Keras: The Python Deep Learning library". https://keras.io/. Accessed on 2020-02-25.

[23] B. Marr. "What is Industry 4.0? Here's A Super Easy Explanation For Anyone". https://www.forbes.com/sites/bernardmarr/2018/09/02/what-is-industry-4-0-heres-a-super-easy-explanation-for-anyone/, Jul 2019. Accessed on 2019-11-26.

[24] NASA. "NASA TLX: Task Load Index". https://humansystems.arc.nasa.gov/groups/TLX/, Aug 2019. Accessed on 2019-03-08.

[25] J. Peddie. *"Augmented Reality: Where We Will All Live"*. California, Tiburon: Springer, Cham, 2017.

[26] K. Pernice. "User Interviews: How, When, and Why to Conduct Them". https://www.nngroup.com/articles/user-interviews/, 2018, 7 Oct. Accessed on 2019-12-05.

[27] J. Preece, Y. Rogers, and H. Sharp. *"Interaction Design: Beyond human-computer interaction"*. 4th ed. Chichester, West Sussex, United Kingdom: John Wiley & Sons Ltd, 2015.

[28] M. Roos. Deep learning versus biological neurons: floating-point numbers, spikes, and neurotransmitters. `https://towardsdatascience.com/deep-learning-versus-biological-neurons-floating-point-numbers-spikes-and-ne` Aug 2019. Accessed on 2020-01-13.

[29] J. Roubaud. How predictive maintenance fits into industry 4.0. `https://www.engineering.com/AdvancedManufacturing/ArticleID/15798/How-Predictive-Maintenance-Fits-into-Industry-40.aspx`, Oct 2017. Accessed on 2019-11-26.

[30] S.J. Russell and P. Norvig. *"Artificial intelligence : a modern approach"*. Prentice Hall series in artificial intelligence. Pearson Education, Upper Saddle River, New Jersey, 2010.

[31] K. Sabarinathan, N. Kanagasabapathy, V. D. Ambeth Kumar, P. K. Rishikesh, R. V. Priyadharshan, and A. Abirami. "Machine Maintenance Using Augmented Reality". In *2018 3rd International Conference on Communication and Electronics Systems (ICCES)*, pages 613–618, Oct 2018.

[32] S. Schechter. "What is markerless Augmented Reality?". `https://www.marxentlabs.com/what-is-markerless-augmented-reality-dead-reckoning/`, Apr 2019. Accessed on 2019-11-22.

[33] H. Subakti and J. Jiang. "Indoor Augmented Reality Using Deep Learning for Industry 4.0 Smart Factories". In *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, volume 02, pages 63–68, July 2018.

[34] C. Tsai and K. Hsu. "An Application of Using Bluetooth Indoor Positioning, Image Recognition and Augmented Reality". In *2016 IEEE 13th International Conference on e-Business Engineering (ICEBE)*, pages 276–281, Nov 2016.

[35] Vuforia. "Vuforia Engine Features". Available: `https://library.vuforia.com/content/vuforia-library/en/features/overview.html`. Accessed on 2020-01-15.

[36] F. Zhou, D. H.B.L, and M. Billinghurst. "Trends in augmented reality tracking, interaction and display: A review of ten years of ISMAR". In *2008 7th IEEE/ACM International Symposium on Mixed and Augmented Reality Mixed and Augmented Reality*, pages 193–202, Sep 2008.