Computer Science

Johan Häger

An Evaluation of Google Glass

Design, Implementation and Evaluation of an Instruction Application for Google Glass and Smartphones

Bachelor's Project

2015:

An Evaluation of Google Glass

Design, Implementation and Evaluation of an Instruction Application for Google Glass and Smartphones

Johan Häger

This report is submitted in partial fulfillment of the requirements for the Bachelor's degree in Computer Science. All material in this report which is not my own work has been identified and no material is included for which a degree has previously been conferred.

Johan Häger

Approved, 2015-06-05

Advisor: Donald F. Ross

Examiner: Thijs. J. Holleboom

Abstract

Put the text of your abstract here

Contents

List of Figures

List of Tables

Listings

1 Introduction

o Project goal and motivation o Project summary and overview - the ”red thread” o Project results (brief summary) o Dissertation Layout

Nytta med projektet, bakomliggande motivering, hypotes kring resultat (Google Glass kommer vara bttre n smartphone eftersom handsfree and stuff), layout av rapporten.

Prata allmnt om vad det finns fr problem idag, mer specifikt vad kommer vr applikation att lsa, mixa med frgor som kan besvaras bland slutsatserna

1.1 Hypothesis

1.2 Project Results

1.3 Dissertation Layout

Chapter 2 discuss relevant background information regarding Google Glass. The chapter will include an introduction to what Google Glass is, how they came about and what features they have. The background chapter will also discuss similar products to Google Glass, as well as compare Google Glass to smartphones. Finally chapter two will include some discussion on topics relevant to the project, including QR code and ways of presenting information.

The 3rd chapter is about the design of the project. The discussion revolves around how the application is intended to work, and what limitations may apply to the implementation of the application, both on Google Glass and smartphone. The third chapter also discusses the design of the tests done on the application.

Chapter 4 describes the implementation part of the project. The flow of the application is described in detail. Specific aspect of the application is also described in more detail. The layout of the slides as well as the voice commands. The experimental setup and how the tests were performed is also described here.

In chapter 5 the results of the tests are presented. The results are presented along with comments regarding how the results are to be interpreted as well as comments on any potential error factors during testing.

The 6th and final chapter contains conclusions on the project. The conclusions regards the test results, as well as conclusions on the project as a whole. Chapter six also includes comments based on personal user experience from using Google Glass for about three months. Finally future work is discussed and the report is concluded with some concluding remarks.

Attached to the dissertation, after the reference list, are appendixes. In appendix A abbreviations used throughout the dissertation are listed. Appendix B shows all the individual test results. Project code can be found in appendix C. Appendix D is the last appendix which contains the original project specification, written in Swedish.

2 Background

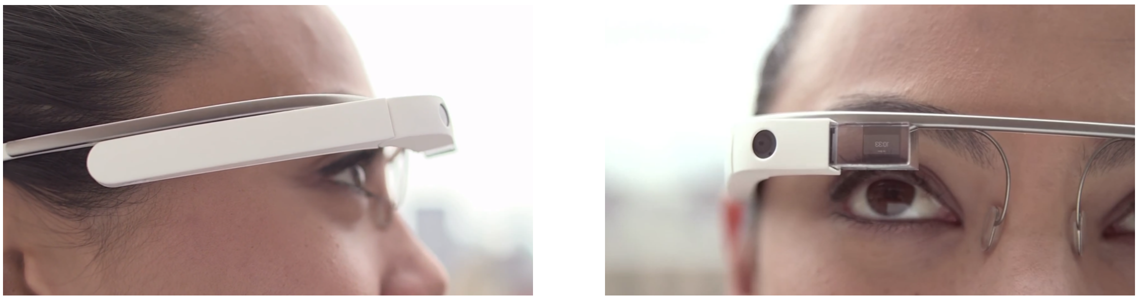
On April 4th, 2012, Google announced "Project Glass" [7]. Google Glass, as the device is now known, was under development for several years at Google's research and development department, Google X. As part of the announcement Google stated: "We think technology should work for you—to be there when you need it and get out of your way when you don't." [9].

Sergey Brin, one of the founders of Google, gave a Ted Talk in February 2013 [23] where he talked about why Google decided to produce the device. His argument was that users stayed on their smartphones for too long. Brin also argued that when users were using their smartphones they were looking down at a screen and were not aware of their surroundings. Instead Google wanted to create a device that would give the user notifications that could quickly be dealt and done with. Google also wanted to make the device hands-free and put the display where the user did not have to look down. Brin stated that the development team at Google X added the camera later on in the development process but in fact the camera had been a great addition to the device and enabled Google Glass to capture the user's surroundings, for instance by taking photographs.

Thad Starner, technical lead/manager (responsible for both the technical direction as well as people management [22]) on Google Glass, claimed that Google Glass was intended to be an extension of the self [82]. He compared Google Glass to a watch. Not in terms of where the user keeps his or her focus (with a watch you must look down, similar to a smartphone), but rather in terms of how a watch is easy to access and how the access is instant. Starner said that with Google Glass, Google wanted to minimise the time between intention and action.

2.1 What is Google Glass?

Google Glass, or simply "Glass" as the device is known within Google, is a head-mounted display (HMD) that can be seen as an augmented reality device (see Section 2.1.1 and Section 2.1.4 respectively) designed to bring notifications to the user more easily than a smartphone does. Google Glass is shown in Figure 2.1. According to Google "Glass is designed to be there when the user needs it and to stay out of the way when the user does not" [49]. Google Glass is meant to give the user relevant information at relevant times.



(a) The user can control Google Glass with the touchpad.

(b) The display sits slightly above the user's line of sight, on the right hand side.

Figure 2.1: Google Glass is equipped with a touchpad and a camera [19].

Google Glass is partially controlled with a touchpad, but can also be controlled through voice commands. The touchpad sits on the right hand side of the user's glass frame and runs from the temple to the ear (see in Figure 2.1 (a)). When the user touches anywhere on the touchpad Google Glass "wakes up" from stand by and displays the start screen (which consists of a clock). The display is mounted above the user's line of sight, on the right hand side (see Figure 2.1 (b)) and can be slightly adjusted so that the user can see all that is currently being displayed.

The display is a projection that goes through an optic lense in the glass piece, seen in Figure (b), which creates a virtual image. A virtual image is an image that, projected through optic lenses, appears to be located at a point where the actual projection is not [78]. In the case of Google Glass the display appears to be located further away from the user than the display actually is. The display is said to be equivalent of a 25 inch high definition screen seen from a distance of approximately 2.5 meters [52].

2.1.1 Head-Mounted Display (HMD)

A head-mounted display (HMD) [71] is a device that is worn on the head and that places a small display in front of one or both of the user's eyes. The device can either be a stand alone device or a part of a helmet. A branch of HMDs are optical head-mounted displays (OHMDs) [74]. A OHMD is a HMD with a see-through display, for instance Google Glass.

2.1.2 Heads-Up Display (HUD)

A heads-up display (HUD) [72] is defined as any transparent display that, when presenting information, does not require userss to look away from their usual viewpoints. In other words, a HUD may be a HMD and a HMD may be a HUD. While a HMD is always worn on the head a HUD can be a stand-alone display. In contrast a HUD must be a transparent display. A requirement a HMD does not have. A OHMD, however, is always a HUD since a OHMD has a transparent display.

2.1.3 Virtual Reality

Virtual reality [67] is defined as a computer generated simulation that enables users to interact with a three-dimensional environment. Virtual realities are common in interactive mediums such as video games. Virtual realities can also be combined with a HMD in order to completely engulf the user in the virtual reality. One such example is the Oculus Rift, seen in Figure 2.2, that completely covers the user's eyes, allowing the user to experience the virtual reality.

Google Glass is able to display a virtual reality but does not work as a virtual reality device. Google Glass only covers a small part of the user's field of vision and as such does not have the capability of simulating a three-dimensional, interactive, environment in

Figure 2.2: The HMD “Oculus Rift” is a virtual reality device [60].

contrast to the Oculus Rift. Oculus Rift, unlike Google Glass, is able to replace the user’s reality with a completely virtual reality since Oculus Rift completely covers the user’s eyes.

2.1.4 Augmented Reality

Augmented reality [34] is defined as the combination of reality (or what is within current context being perceived as reality [1]) with useful, computer generated, data. Augmented reality, unlike virtual reality, is not meant to replace reality, but rather to enhance interaction with the current reality.

A HUD may create an augmented reality. The reason a HUD does not always create an augmented reality is due to the fact that the information being presented might not be useful within the current context. An augmented reality is, as stated above, meant to

[1] Augmented reality is for instance common in video games to give the player environmental and health information.

enhance reality, while a HUD does not have that requirement.

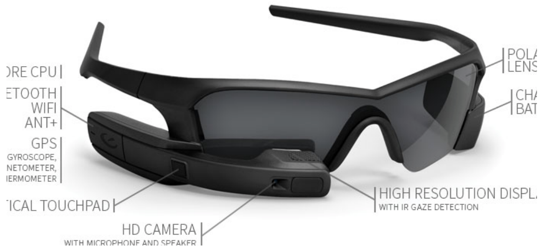
Google Glass is a HUD that has the potential (and intent) to create an augmented reality. Google Glass is intended to present useful information to users while not distracting them from reality. One example of useful information that could enhance users interaction with reality would be a shopping list while users are out shopping, as seen in Figure 2.3.

Figure 2.3: A shopping list while the user is out shopping is useful information [49].

2.2 Similar Products

Today there are several products either already on the market or under development that are more or less similar to Google Glass. Following is a short list (a more extensive list of devices can be found on wikipedia [74]) describing some of the competition Google Glass faces, with each product shown in Figure 2.4.

(a) Microsoft Hololens [58]



(b) Recon Jet [65]

(c) GlassUp [39]

(d) C Wear Interactive Glasses [64]

Figure 2.4: There are many OHMD devices similar to Google Glass [74].

2.2.1 Microsoft Hololens [58]

Microsoft's offer in the augmented reality device space is a HUD that displays information in front of both of the user's eyes, called Microsoft Hololens, seen in Figure 2.4 (a). However, while Google Glass is meant to be worn at all times, Microsoft Hololens is rather a device users only wear when they intend to use Microsoft Hololens. Google Glass is, as Thad Starner stated [82], meant to be an extension of the self and is meant to be worn even though the user might not be actively using Google Glass at the time in order to bring helpful notifications and information to the user. Microsoft Hololens is rather a tool to be used actively for a certain purpose, such as modelling [57], and then put away. Google Glass may be used the same way if the user wants to, but that is not the intent.

The mot striking difference between Microsoft Hololens and Google Glass lies in the interaction with the real world. Google Glass is a two dimensional (2D) display that sits slightly above the users line of sight (see Section 2.1). Microsoft Hololens, on the other hand, is meant to interact with the world even further.

Microsoft intends to give the user tools to work in a three dimensional (3D) space. Microsoft's concept video [59] of Microsoft Hololens shows examples of 3D modelling with the use of kinetic hand-movement detection. Microsoft Hololens will enable users to see what they are working on from different angles simply by walking around the object, just as if the object in question was real and had a physical mass.

2.2.2 Recon Jet [65]

Recon Jet, seen in Figure 2.4 (b) is an HMD developed by Recon Instruments. Recon Jet is suited for athletes. [65] Because of the target audience Recon Jet has been fitted with a display that has high contrast in order to give good readability in high ambient lighting. The display's virtual image appears as a 30 inch wide screen at approximately 2 meters distance [66], to be compared with Google Glass' virtual image which appears as a 25 inch high definition screen seen from a distance of 2.5 meters [52].

Unlike Google Glass, Recon Jet's display is located below the user's line of sight, as seen in Figure 2.4. Recon Jet's target audience, athletes, are used to having their information below line of sight. For instance a bike may have dashboard mounted to the handlebar, or an athlete might be using a watch to check the time. Google Glass is meant to be worn at all times while the location and the brightness of the display indicates that Recon Jet, however, is meant to only be used while the athlete is working out and not more regularly.

2.2.3 GlassUp [37]

GlassUp is an Italian company that received most of its founding for the HMD device, GlassUp (seen in Figure 2.4 (c)), through the crowd-funding site Indiegogo [38]. GlassUp has been accused of being to similar to Google Glass, partially because of the name of the device [13]. GlassUp does however make distinctions between the two products. On GlassUp's Indiegogo page the company made the comparison that looking at Google Glass' display was similar to looking in the back view mirror while GlassUp was similar to looking out the windscreen. The comparison referenced the fact that Google Glass' display is located above the user's line of sight, similar to a rear view mirror.

GlassUp instead displays information close to the center of the user's line of sight. GlassUp claimed, on the company's Indiegogo page, that the display was placed closer to the center of the users line of sight so that there would be less strain on the user's eyes. However, the biggest difference from Google Glass is that GlassUp is meant only to act as a second screen. GlassUp is a "receive only" device which displays information from the device currently connected through bluetooth, for instance a smartphone. GlassUp does not do any calculations on its own and must stay connected to a bluetooth device in order to display information [38].

2.2.4 C Wear Interactive Glasses [63]

C Wear Interactive Glasses, seen in Figure 2.4 (d), is an industry focused device developed by Penny in Västerås, Sweden [10]. C Wear Interactive Glasses projects an image onto the actual glass in front of the user's right eye and as such covers a larger area than similar devices such as Google Glass, Recon Jet and GlassUp [62]. The display is said to be perceived as a 75 inch display at a distance of 2.1 meters [64]. The projection is transparent which enables users to still see what is happening in front of them. 2 Being industry focused C Wear Interactive Glasses is also equipped with a hands-free user interface that does not require voice command. C Wear Interactive Glasses uses a jaw sensor which lies against the user's jawbone muscle. The sensor detects tension in the muscle, which registers as a click, to be compared with a toucn on the Google Glass touchpad [64].

C Wear Interactive Glasses, similar to GlassUp, is designed to be connected to an external device [64]. However, where GlassUp is connected through bluetooth C Wear Interactive Glasses is connected through an adapter which can send data and visual information via USB and HDMI. The external device can be a smartphone, a tablet, a PC or even a TV.

2.3 User Interface

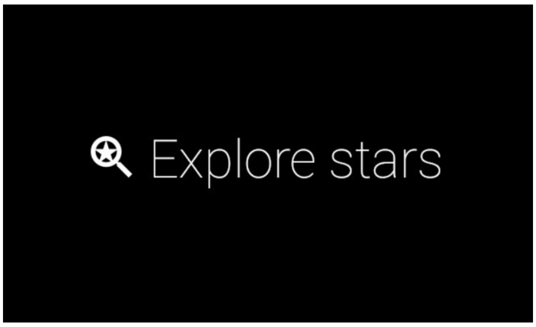
The Google Glass graphical user interface (GUI) is called a timeline [19] (see Figure 2.5). The timeline consists of a row of cards. Cards are basic applications such as a clock (see Figure 2.6 (a)) or information about the weather. Cards can also represent more in-depth applications, on Google Glass called “Immersions” (see Figure 2.6 (b) and (c)). Immersions handle activities such as browsing an image gallery or playing a game.

Figure 2.5: A visualisation of the timeline as the timeline is perceived by the user [19].

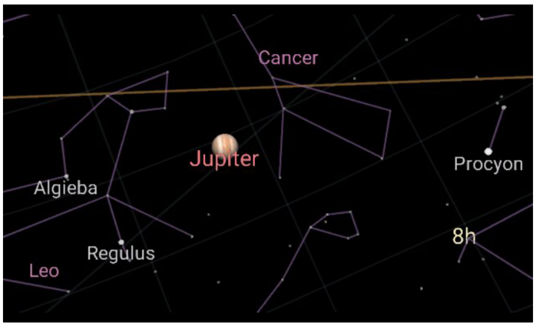
The first screen the user sees when starting up Google Glass is the home screen. The home screen displays a clock and also shows the text ”ok glass”, as seen in Figure 2.6 (a). The home screen is a part of the timeline and acts as the center point. Cards to the left of the home screen are upcoming activities such as an event in the user’s calendar or an upcoming flight. Cards to the right of the home screen are from the past. Cards from the past will for instance show text messages or photos.

In order to move left on the timeline (forward in time) the user must swipe a finger backwards on the touchpad. In order to move right on the timeline (backward in time) the user must swipe a finger forward on the touchpad. The fact that the user must swipe

(a) The Google Glass home screen is a card that displays a clock.



(b) The card “Explore stars” represents an immersion.



(c) The immersion “Explore stars” allows the user to look around at stars using the built-in head motion tracker.

Figure 2.6: Cards can either display basic applications or represent immersions.

backwards when stepping forward in time might not seem especially intuitive. In western culture a timeline is normally represented as going from left to right. One example is books, where the reader not only reads each line from left to right, but also turn pages from the right (the future) to the left (the past). However, on Google Glass, the swiping action could be thought of as swiping cards behind the back. Swiping forward when stepping backwards in time would then in turn mean bringing cards placed behind the back into focus. Cards in the past are behind the user while cards in the future are in front of the user.

When the user wants to turn off Google Glass the user swipes down on the touchpad. Swiping down on the touchpad will put Google Glass in stand-by mode. If the user wants to turn off Google Glass entirely, in other words power down the device, there is a power button on the opposite side of the touchpad. Holding down the power button for a few seconds will turn off Google Glass. For a better visual understanding of how Google Glass works see Figure 2.5 as well as the video referenced in the caption.

Google Glass uses a Bone Conduction Transducer (BCT) to transfer sound to the user [52]. The BCT transfers sound to the inner ear by conducting sound through the bones of the skull [68]. The advantage of this technique is that the sound maintains clarity, even in noisy environments. Also, since the user does not plug any earphone into their ears, external sound is not blocked out.

Google Glass also features a 5 megapixels camera [52]. The camera sits between the touchpad and the display, as seen in Figure 2.1 (b), and is capable of capturing video at a 720p resolution. The camera can be used for video conferencing, as Google showed in 2012 [6], but the camera can for instance also be used when the user wants to scan a QR Code (see Section 2.6).

The user can also interact with Google Glass using voice commands. As seen in Figure 2.6 the home screen consists not only of a clock but also of the words “ok glass”, in quotes. “ok glass” indicates to the user that voice commands are available. The voice command menu is accessed as soon as the user says the words “ok glass”. Doing so brings up a list of voice commands available, as seen in Figure 2.7.

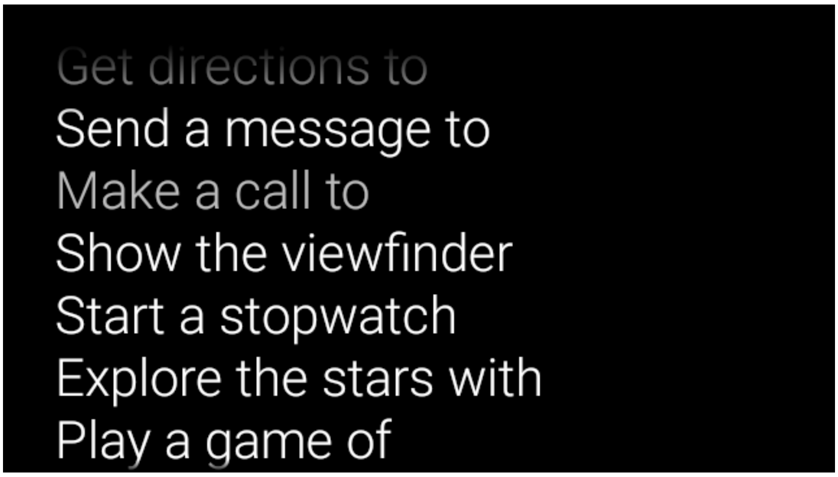


Figure 2.7: Saying “ok glass” will bring up the voice command menu [54].

In order to progress further the user must say one of the options being displayed out loud. Doing so will either make Google Glass perform the task spoken or give the user the option to add an input option to the task chosen. For instance, if the user where to say “ok glass, Start a stopwatch”, Google Glass would start a stopwatch.

Google Glass also supports head motions as a form of input from the user. Head motions are not enabled in the timeline as a way of input but tilting the head may wake up Google Glass from stand by mode, if the user has enabled the head wake up feature [48]. The head motion interface may also be used in certain immersions, such as “Explore stars” seen in Figure 2.6 (c).

2.4 A Comparison with Smartphones

Compared to smartphones one of the biggest advantages of Google Glass is the fact that Google Glass is a HMD. With a smartphone the user needs to either hold the smartphone in either one or both hands, or alternatively put the smartphone on a table. In other words can Google Glass offer a hands-free experience that smartphones cannot.

Another advantage of Google Glass compared to smartphones also comes from the fact that Google Glass is an HMD. The user does not need to look away in order to see what is currently being displayed. Google Glass does not distract from what the user is currently doing as much as a smartphone where the user needs to either look away or hold up the smartphone in from of their eyes.

However, smartphones does give the user a bit more control. The control comes from the fact that smartphones supports multi-touch, which Google Glass does not. On a smartphone users may also touch directly on the screen, in contrast to Google Glass where the touchpad sits on the right hand side of the user. Smartphones also have a larger touch area than Google Glass.

The smartphone screen size has been increasing ever since the iPhone first launched in 2007 [73], as seen in Figure 2.8. Looking at currently available smartphones, in Figure 2.9, the increase in screen size does is set to continue as the average screen size is approaching five inches. In terms of comparison with Google Glass the increase in screen size entails that more information could be displayed on a smartphone than on Google Glass.

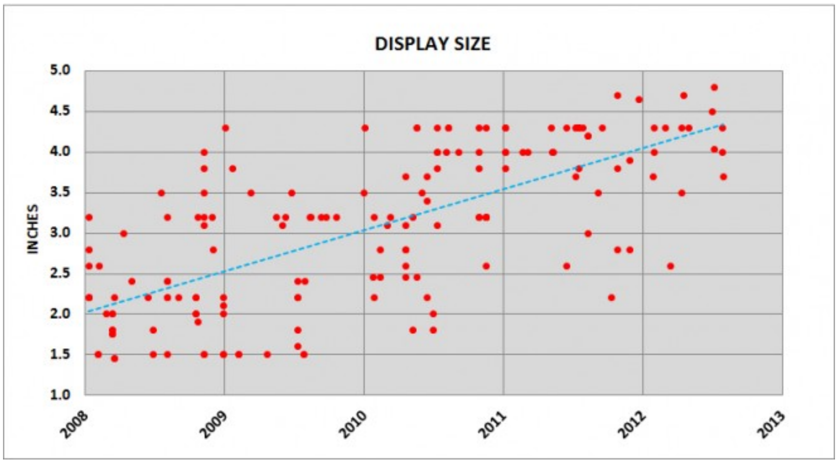


Figure 2.8: Smartphone screens have been increasing in size for several years [12].

However, one of the biggest differences between smartphones and Google Glass is the plural, smartphone**s**. There are several smartphone brands competing on the market, each offering several models. Google Glass is simply Google Glass, one product. As seen in Section 2.2 Google Glass does face competitors that have approached HMDs differently,

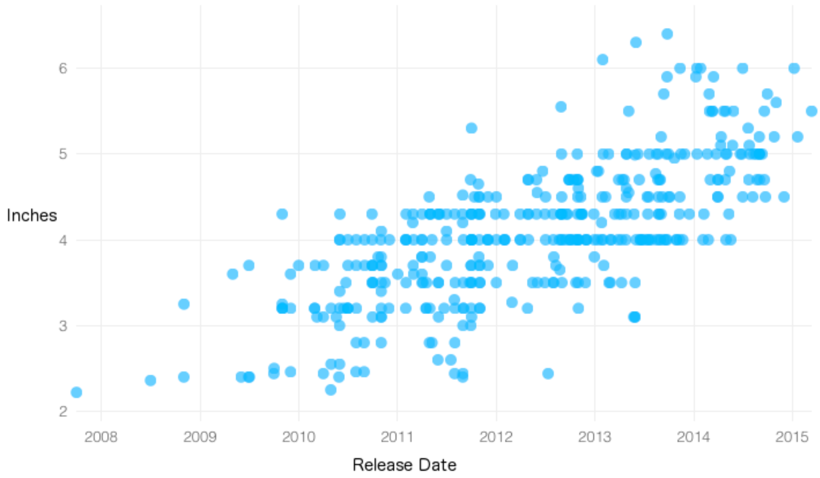


Figure 2.9: Screen sizes of the most popular, currently available smartphones [26].

and as HMDs increase in popularity there is potential for an even wider offering of models and screen sizes.

2.5 Limitations of Google Glass

One early concern with Google Glass came from people who wore regular glasses every day, as Google Glass seemed to require separate frames. Isabelle Olsson at Google responded on the issue on April 12th 2012 with the following: “We ideally want Project Glass to work for everyone, and we’re experimenting with designs that are meant to be extendable to different types of frames.” [8].

Today many eyecare providers have been trained for Google Glass and Glass frames. These trained eyecare providers are however mostly located in the United States [28], but Google points out that many eyecare providers should be able to help replace the lenses on Google Glass’ frames [29].

As described in an article posted on forbes in 2013 [18], a more alarming concern has been the health of the user’s eyes. Concerns were raised regarding eye strain and misalignment of the user’s eyes, as Google Glass placed a screen above one eye and not both.

Google also saw these potential issues and approached Eli Peli, professor of ophthalmology who had been studying HMDs for two decades, as the development och Google Glass started.

Peli claimed that Google Glass has been designed with more safety and comfort in mind than previous, similar products. Peli pointed out that Google Glass is see-through and only covers a small part of the user's field of vision. As such Google Glass does not require a potentially poorly adjusted camera to capture the environment and display the environment to the user, which could cause eye strain.

Peli also pointed out that Google Glass is meant only to be used for short periods. Google Glass is meant to give the user notifications that can be quickly dealt with. The user should not be looking at the display for long periods of time, which would have the potential to lead to eye strain. While Peli stated that the risks are zero, he still claimed that the likelihood of Google Glass causing any damage is minimal.

Even though, according to Google's expert, there might not be any health risks involved, there is still a question of how much help Google Glass may be to users. A study performed in 2002 [84], regarding the effects of OHMDs, showed that OHMDs may only be of help to users under controlled forms. Whenever the surrounding evnironment becomes too distracting, for instance within a moving crowd, performance decreases. The study however noted that pilots had been able to successfully turn HMD into a tool they could use to their advantage. Since the study was not carried out over a long period of time the participants were potentially not given enough time to get used to wearing and using their HMDs, explaining the poor performance when using a distracting background.

2.6 QR Code

As mentioned in Section 2.3 Google Glass is equipped with a camera that could be used to take photos from the user's perspective. One potential use of the camera would be to scan Quick Response (QR) codes. The QR Code was announced in 1994. Having been under development for several years at Denso Wave [15] the goal was to create a new form of barcode that could carry more information than a linear barcode and be easily read.

A conventional barcode is capable of storing approximately 20 digits while a QR code can store several thousand digits [16]. Information is encoded using standardised encoding modes and displayed as a 2D barcode. A QR code has several standardised fields, as seen in Figure 2.10. Using position fields a QR code can be read from any direction, compared to a conventional barcode which can only be read horizontally [17].

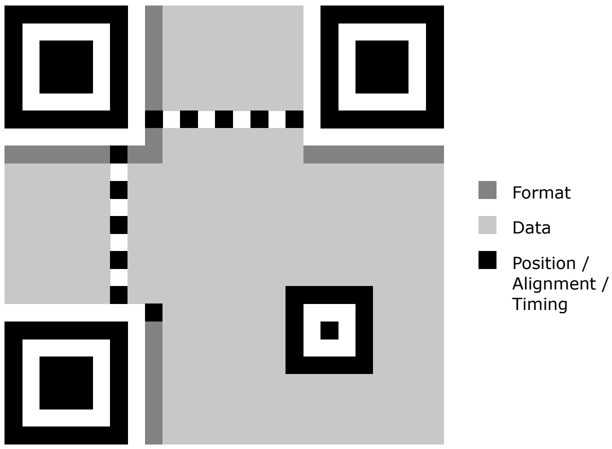


Figure 2.10: The standardised fields of a QR code [75].

A QR code can be used to encode information, originally written with alphabetic characters, Japanese symbols (Kanji, Katakana and Hiragana) or numeric characters [14]. With the help of a QR code information which would otherwise have taken up a large space can now be easily fitted in smaller areas.

2.6.1 Decoding

Decoding a QR code is a fairly straight forward process which, although time consuming, can be done by hand, as described in several guides around the Internet [11, 24, 80]. A QR code may be divided in to three standardised fields, which help QR code scanners to identify and decode QR codes. The three fields can be seen in Figue 2.10. The position, alignment and timing fields are used to identify and position the QR code correctly as a QR code may be scanned from any direction. In order to decode a QR code the three main position fields residing in three of the QR code's corners must be positioned as seen in Figure 2.11.

The next step is to identify the mask used on the data field in the QR code. The mask is used to remove any large, empty or filled, areas within the data field which might make the decoding process more difficult for QR code scanners. The mask field is a part of the format field, seen in Figure 2.10. The format field holds information about error correction, as well as which mask has been used on the data field. The mask is found among the first five bits in the format field. The first two bits of the first five bits of the format field contain information regarding the amount of error correction data the QR code contains, and the other three contain information on which mask has been used.

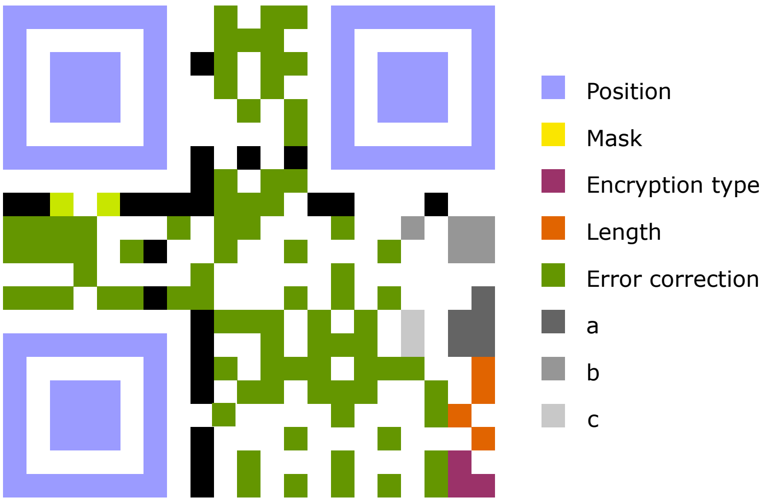


Figure 2.11: A QR code example, encoded with the string "abc".

In Figure 2.11 the mask field has the value of 101, seen more clearly in Figure 2.12. Filled blocks should be interpreted as a 1 and empty blocks should be interpreted as a 0. However, the mask field is always XOR:d with 101 prior to being printed in the QR code and must as such be XOR:d back to the original mask value. In the case of Figure 2.11 the original mask value is calculated as follows:

$$101 \text{ XOR } 101 = 000$$

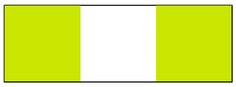


Figure 2.12: Mask pattern encoded as 101.

There are eight different mask patterns in total, each represented by a unique bit string. The mask pattern represented by 000 can be seen in Figure 2.13, while the rest of the mask patterns may be found here [5]. The mask used in Figure 2.11 means that all bits should be flipped if the following formula is true:

$$(i + j) \ mod \ 2 = 0,$$

where i and j represent the indexes of the specific block, horizontally and vertically respectively [5].

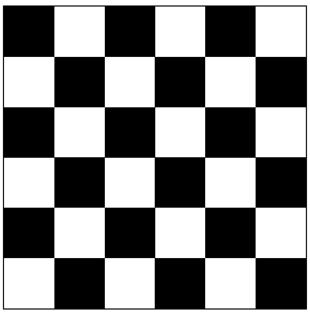


Figure 2.13: Mask pattern represented by the bit string 000 [5].

Having decoded the mask pattern the next step is to decode the data area. The data area always contains a header, containing information on the encryption type as well as data length. The blocks containing the encryption type are always the size of four blocks, and always located in the lower right corner, as seen in Figure 2.11. The number of blocks used for the data length may vary between eight and ten blocks depending on the encryption type.

As such the first step in decoding the data area is to decode the encryption type. The information in the data area is to be read from the lower right and upwards, in a zig-zag pattern, with a width of two blocks. When reaching the top the zig-zag pattern continues, although downwards. See Figure 2.14 for a better understanding of the zig-zag pattern. The start point is always in the lower right corner since the encryption type must be decoded first.

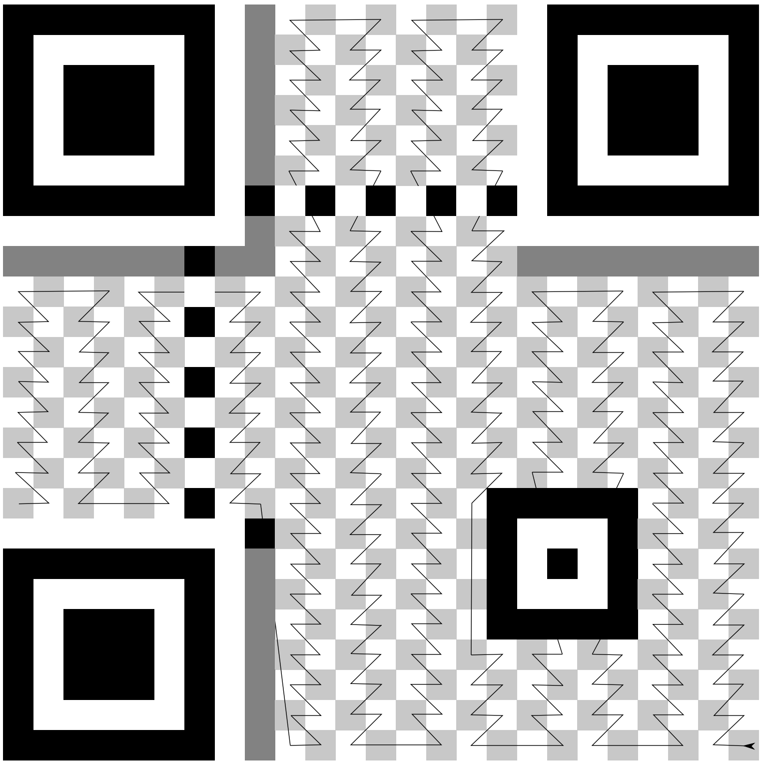


Figure 2.14: The zig-zag pattern used when decoding a QR code.

Using the zig-zag pattern the encryption type bit string in Figure 2.11 can be found to be 1101, seen more clearly in Figure 2.15. However, the encryption type is a part of the data section and as such must be unmasked. Using the mask formula gives the following results:

$$(20 + 20)\ mod\ 2 = 40\ mod\ 2 = 0$$

$$(20 + 19)\ mod\ 2 = 39\ mod\ 2 = 1$$

$$(19 + 20)\ mod\ 2 = 39\ mod\ 2 = 1$$
$$(19 + 19)\ mod\ 2 = 38\ mod\ 2 = 0$$

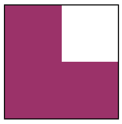


Figure 2.15: Encryption type encoded as 1101.

As such the blocks at position (i, j) = (19, 19), and (i, j) = (20, 20) must be flipped. The “flipping” process may be done by XOR:ing the massked encryption type bit string with a bi string representing the bits that must be “flipped”, putting ones at the positions where the corresponding bit in the masked encryption type bit string must be flipped, and zeroes at the position where the corresponding bit does not need to be flipped. The masked encryption type bit string, 1101, must as such be XOR:d with 1001, as follows:

$$1101\ XOR\ 1001 = 0100$$

There are several encryption types used in QR codes, however the most common ones are the following (represented by the following bit strings):

- 0001 Numeric
- 0010 Alphanumeric
- 0100 8-Bit Byte

The encryption type used in Figure 2.11 is as such of encryption type 8-bit Byte.

After having decoded the encryption type the next step is to decode the length. The encoded length is the length of the message encoded in the QR code. Since the message encoded in the QR code may not cover the entire data section of the QR code (the rest is made up of error correction information) it is necessary to know the length of the message in order to tell when the message ends. Since the encryption type used in Figure 2.11 is an 8-bit Byte the size of each field that follows is eight bits in size.

The length field in Figure 2.11, seen also in FIgure 2.16, is the following bit string: 10011010. However, the length field is a part of the data section and must as such be unmasked in order for the original length value to be obtained.

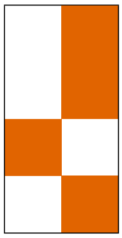


Figure 2.16: The message length encoded as 10011010.

$$(18 + 20)\ mod\ 2 = 38\ mod\ 2 = 0$$
$$(18 + 19)\ mod\ 2 = 37\ mod\ 2 = 1$$
$$(17 + 20)\ mod\ 2 = 37\ mod\ 2 = 1$$
$$(17 + 19)\ mod\ 2 = 36\ mod\ 2 = 0$$
$$(16 + 20)\ mod\ 2 = 36\ mod\ 2 = 0$$
$$(16 + 19)\ mod\ 2 = 35\ mod\ 2 = 1$$
$$(15 + 20)\ mod\ 2 = 35\ mod\ 2 = 1$$
$$(15 + 19)\ mod\ 2 = 34\ mod\ 2 = 0$$

The encryption type bit string must as such be XOR:d with 10011001, as follows:

$$10011010\ XOR\ 10011001 = 00000011 = 3$$

The message in Figure 2.11 is as such of length 3.

Finally, the data is decoded. The first 8-bit Byte, seen in Figure 2.17, gives the following bit string: 11111000. However, being a part of the data section, the bit string must be unmasked, as follows:

$$(14 + 20)\ mod\ 2 = 34\ mod\ 2 = 0$$
$$(14 + 19)\ mod\ 2 = 33\ mod\ 2 = 1$$
$$(13 + 20)\ mod\ 2 = 33\ mod\ 2 = 1$$
$$(13 + 19)\ mod\ 2 = 32\ mod\ 2 = 0$$
$$(12 + 20)\ mod\ 2 = 32\ mod\ 2 = 0$$
$$(12 + 19)\ mod\ 2 = 31\ mod\ 2 = 1$$
$$(11 + 20)\ mod\ 2 = 31\ mod\ 2 = 1$$
$$(11 + 19)\ mod\ 2 = 30\ mod\ 2 = 0$$

The first 8-bit Byte bit string must as such be XOR:d with 10011001.

$$11111000\ XOR\ 10011001 = 01100001 = 64 + 32 + 1 = 97$$

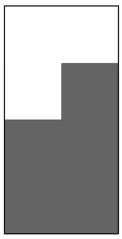


Figure 2.17: The first 8-bit Byte encoded as 11111000.

The second 8-bit Byte, seen in Figure 2.18, reaches the top of the data section and as such the last four bits must be read horizontally to the left, as described in Figure 2.14. Doing so gives the following bit string: 11110100. Again, the bit string must be unmasked.

$$(10 + 20)\ mod\ 2 = 30\ mod\ 2 = 0$$
$$(10 + 19)\ mod\ 2 = 29\ mod\ 2 = 1$$
$$(9 + 20)\ mod\ 2 = 29\ mod\ 2 = 1$$
$$(9 + 19)\ mod\ 2 = 28\ mod\ 2 = 0$$
$$(9 + 18)\ mod\ 2 = 27\ mod\ 2 = 1$$
$$(9 + 17)\ mod\ 2 = 26\ mod\ 2 = 0$$

$$(10 + 18)\ mod\ 2 = 28\ mod\ 2 = 0$$
$$(10 + 17)\ mod\ 2 = 27\ mod\ 2 = 1$$

The second 8-bit Byte bit string must as such be XOR:d with 10010110.

$$11110100\ XOR\ 10010110 = 01100010 = 64 + 32 + 2 = 98$$

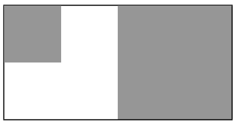


Figure 2.18: The second 8-bit Byte encoded as 11110100.

The third, and final 8-bit Byte, seen in Figure 2.19, to be decoded is read downwards as described in Figure 2.14, giving the following bit string: 00000101. The bit string must be unmasked:

$$(11 + 18)\ mod\ 2 = 29\ mod\ 2 = 1$$
$$(11 + 17)\ mod\ 2 = 28\ mod\ 2 = 0$$
$$(12 + 18)\ mod\ 2 = 30\ mod\ 2 = 0$$
$$(12 + 17)\ mod\ 2 = 29\ mod\ 2 = 1$$
$$(13 + 18)\ mod\ 2 = 31\ mod\ 2 = 1$$
$$(13 + 17)\ mod\ 2 = 30\ mod\ 2 = 0$$
$$(14 + 18)\ mod\ 2 = 32\ mod\ 2 = 0$$
$$(14 + 17)\ mod\ 2 = 31\ mod\ 2 = 1$$

The third 8-bit Byte bit string must as such be XOR:d with 01100110.

$$00000101\ XOR\ 01100110 = 01100011 = 64 + 32 + 2 + 1 = 99$$

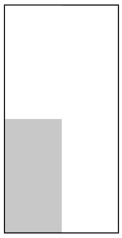


Figure 2.19: The third 8-bit Byte encoded as 00000101.

Since the message length was found to be of size 3 all parts of the message in Figure 2.11 have been found. The rest of the data section contains information regarding error correction, used in case the QR code was somehow damaged.

The message in Figure 2.11 has been decoded as 97, 98 and 99. Converting there numbers using an ASCII table gives the following result: a, b and c [2]. The encoded message in Figure 2.11 has as such been decoded to “abc”, which is correct and may be checked by scanning Figure 2.11 using a QR code scanner.

Although the decoding of a QR code may seem like an extensive process, the process may be divided in to the following five parts:

1. Aligning the position

2. Finding the mask used on the data section

3. Unmasking the encryption type

4. Unmasking the length of the encoded message

5. Unmasking the message

Although time consuming, decoding information from a QR code by hand is possible and follows the same steps as a QR code scanner.

2.7 Information (and Ways of Presenting Information)

In 1985 Sture Allén, professor of computational linguistics, and Einar Selander, honorary doctor at Umeå University, in their book—Information on Information—defined information after having gone through "a large number of examples from texts of different kinds" [83]. Allén and Selander defined information as "a certain amount of facts or ideas". While defining the concept of information can be done, there are several ways of presenting information.

2.7.1 Text

Text is one of the oldest forms of presenting information, with written text dating back to 4000 years B.C. [69]. Text is also a simple form of presentation that does not require much high end hardware. Other forms of presenting information require more memory, more computational power and more graphical power. Text also has the advantage that users can read through text at their own pace. Text may be viewed independently of time in contrast to sound and video.

Text does however have the disadvantage of requiring attention. The person reading the text must focus their attention on the text throughout and cannot look away in order to receive all the information being presented. Text is also restricted to the language the text has been written in. Several texts must be written in order to make the text available to people who might not have the native language in the specified country as their first language. For instance, in 2010, 44.7% of the Spanish speaking population in the United States spoke English less than "very well" [3].

2.7.2 Images

The advantage of using images as the form of presenting information is that one can show the viewer the information rather than telling the viewer the information. Showing the viewer could potentially mean that more information could be presented within a smaller space than text could achieve. Images also give the same advantage as text in terms of at what pace the viewer could perceive the information. Images, similar to text, may be viewed independently of time in contrast to sound and video.

In a similar way to text, images require the viewers attention in order to understand the information. The viewer cannot look away from an image and still receive the information. Another disadvantage with images is the fact that images can be interpreted in different ways. The saying “a picture is worth a thousand words” goes both ways. On one hand images may present much information with one single image. On the other hand the information may not be crystal clear and not as clear cut as a describing text might be.

Images may also present information in two different ways. One way is with photographs. Photographs may present abstract and/or concrete information and visualise what might be difficult to describe only using words. Another way of using images is by presenting information using graphs. Graphs are usually more clear cut with regard to information they present. However, graphs may in some cases be an insufficient way of presenting information. Graphs are used to present statistical information and are usually easier than photographs to translate into words. Statistical information, and as such also graphs, can summarise a period in time where as photographs captures a moment.

2.7.3 Audio

Images and text both share the disadvantage of requiring the user’s vision in order for the information to be perceived. Audio solves this problem. While audio does require the user’s attention audio uses a different sense than text and images. With audio as the form of presenting information the listener can look away and yet still receive the information that is being presented. In other words audio is well suited for multitasking as long as the other task the listener is performing does not involve listening to audio as well.

Audio does however have the disadvantage of having to be understood in real-time. The listener does not possess the same amount of control as he or she does with either text or images. Audio may be paused and rewound but the fact that audio is still tied to a timeline is a disadvantage. Another disadvantage with audio is that, similar to text, audio is dependent on the language. If information were to be used on a world-wide basis several audio files would be required (given that the audio contained spoken words) translated into different languages.

2.7.4 Video

Since video consists of many images bundled together video gives the same advantages as images in terms of showing the viewer the information instead of telling them. Video presents the viewer with images at such speed that the images give the impression of movement. Video may also include audio. The inclusion of audio potentially gives video all the same advantages as audio. In other words video could potentially give the advantages of two other forms of information presentation.

However, similar to audio, video is presented in real-time. The viewer is bound to the playback speed of the video. Even though a video may be paused or even rewound the viewer does not possess the same amount of control as with images or text. With text and images the reader (or viewer) can deceide the pace at which the information should be perceived for themselves. If the video does not include audio video, similar to images or text requires full attention in order for the information to be perceived.

2.8 Summary

Google Glass was announced in 2012 along with the statement "We think technology should work for you—to be there when you need it and get out of your way when you don't." [9]. Google wanted to create a device where the user did not have to look down [23] as well as a device where the time between intention and action was minimised [82].

Google Glass (see Figure 2.1 (a) and (b)) is an small HMD that is partially controlled with a touchpad mounted on the right hand side of the frames. The display sits slightly above the user's line of sight, on the right hand side. Google Glass' display is a projection that goes through an optic lense, creating a virtual image which makes the perception of the display to be equivalent of a 25 inch high definition screen seen from a distance of approximately 2.5 meters [52].

Today there are many products similar to Google Glass either already on the market or in development. Microsoft Hololens is an HMD focused on letting the user work in a 3D space (with for instance 3D modelling [57]) by covering both of the user's eyes. Recon Jet, GlassUp and C Wear Interactive Glasses are three products more similar to Google Glass in the sense that they only display information in front of one eye. However, Recon Jet is aimed at athletes, to be used while athletes are working out. GlassUp and C Wear Interactive Glasses are meant to be connected to an external device, such as a smartphone or a PC. Google Glass is a stand-alone device meant to be worn at all times.

The GUI of Google Glass is called a timeline and consists of a row of cards [19]. Cards are basic activities, such as a clock, but may also represent more in-depth applications, such as a game, on Google Glass called "Immersions". The center point of the timeline is the home screen and the first screen the user sees when turning on Google Glass. Cards to the left of the home screen are upcoming events, such as a flight, and cards to the right of the home screen are from the past, such as text messages. The user moves left on the timeline by swiping a finger backwards on the touchpad and in order to move right the user must swipe a finger forwards on the touchpad.

In order to play sounds Google Glass uses a BCT which transfers sound through the bones of the skull [52]. The advantage of this technique is that external sound is not blocked out. In order to capture the environment Google Glass is also equipped with a 5 megapixles camera [52] and a microphone. Using the camera and the microphone the user may give input to Google Glass, for instance when using voice commands in order to control Google Glass hands-free.

The hands-free experience is also what sets Google Glass appart from regular smartphones. Smartphones must be held by the user or put on a table. The user must also look down at the screen of a smartphone, in contrast to Google Glass which puts the display slightly above the user's line of sight. Smartphones do however give the user a bit more control with multi-touch and touchscreen. Another advantage of smartphones is the larger screens. As seen in Figure 2.8 and Figure smartphoneSizeChart, Smartphone screens have been increasing in size ever since the iPhone launched in 2007 [12]. However, as HMDs increase in popularity, there is potential for a wider offering of models and screen sizes.

Smartphones and Google Glass may in many cases be used for similar applications. For instance QR code scanning, since both smartphones and Google Glass are equipped with a camera. The QR code was announced in 1994 by Denso Wave [15]. The goal was to create a new form af barcode that could carry more information than a linear barcode and be easily read. While a conventional bardoce is capable of storing approximately 10 digits a QR code can store several thousand digits [16]. A QR code also uses position fields which make the QR code readable from any direction, compared to a conventional barcode which can only be read horizontally [17]. A QR code can be used to encode information, originally written with alphabetic characters, Japanese symbols or numeric characters [14].

Information has been defined as "a large number of examples from texts of different kinds" [83] and may be presented in a number of different ways. The four main ways of presenting information are text, images, sound and video. Text and images both have the advantage of being independent of time. Readers/viewers may perceive the information at their own pace. Images may also be divided into photographs and graphs. The difference of the two lies in the fact that graphs are used to present statistical information. Text and images does however both share the disadvantage of requiring readers/viewers vision.

Audio solves the problem of requiring the user's vision since audio uses a different sense. The listener may as a consequence multitask in terms of listening to the information being presented through audio while performing other tasks. For instance the listener may be listening to the radio while driving a car. In contrast to text and images audio is not independent of time. Video has the same disadvantage as audio of not being independent of time. Video is, however, unique as video is the only form of presenting information which may combine images and as such show movement. Adding audio to video gives viewers the advantage of choice as they may choose to either watch or only listen.

3 Design

The application designed and implemented in parallel with this report is a proof-of-concept of an application used for assembling components. The application is intended to function as a substitute for instruction manuals, where Google Glass will allow users to scroll through the instructions hands-free. A proof-of-concept of a similar application for smartphones has been designed and implemented at the same time, in order to provide a point of reference as well as help evaluate the pros and cons of using Google Glass. The application for Google Glass and the application for smartphones will function and look the same, unless otherwise mentioned, in order to help the evaluation and comparison.

The application works as seen in Figure 3.1. First the user must use the application to scan a QR code. The QR code is then decoded. The decoded information from the QR code is an ID used to download information regarding the product connected to the QR code the user just scanned. The downloaded information contains the product name, as well as the necessary components and instructions needed to build the specific product. The product information is sorted into classes representing the product information, where attributes of the product class contains information on components and instructions. The product information is then sent to the display.

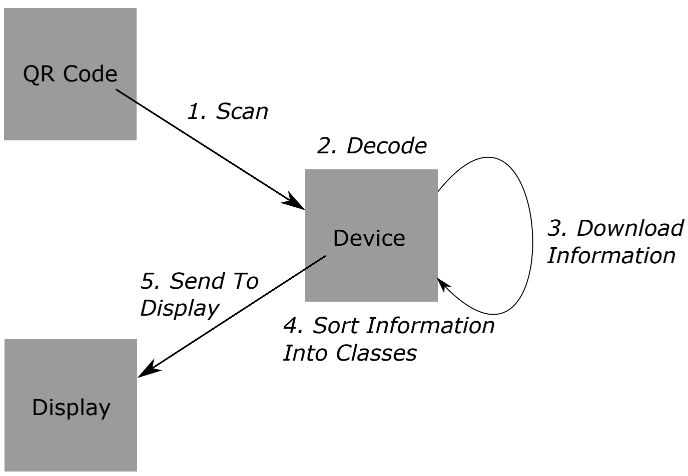


Figure 3.1: Application functionality.

As seen in Figure 3.2, the application is designed as a slide view, with only one slide being displayed at a time. Instructions are to be divided into several steps, each presented on a separate slide which users may scroll through at their own pace. On Google Glass, users may scroll through the application using voice commands in order to make the application truly hands-free. By applying the slide view design users may focus on one instruction at a time.

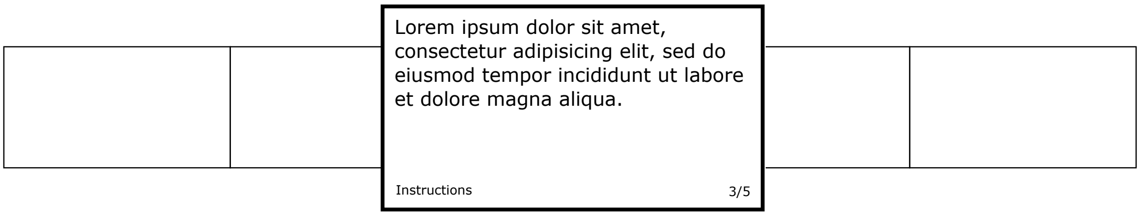


Figure 3.2: A simple sketch of the application's GUI design.

The instructions are the major focus of each slide, with each instruction using most of the slide. At the bottom of each slide additional information may be found, such as a label which, for instance, specifies that the information being presented on the current slide is an instruction (other information may also be presented, such as components required to complete the instructions). The design has been heavily inspired by Google's own guidelines [49] as to how applications for Google Glass should be designed.

3.1 Glassware Flow Designer

Google provides developers with a design tool to help them visualise applications prior to implementation. The design tool, called "Glassware Flow Designer" [47], allows developers to discover recommended design patterns and to draw out the flow of their application prior to implementation.

In Figure 3.3 the Glass Flow Design for the Google Glass application described in this report is shown. When the application launches the user should scan a QR code. After successfully doing so, the product name will be displayed, in order to help the user confirm that the correct instructions have been loaded in. This is followed by the slides which show the required components. After the list of components follows the instructions. The slides can be controlled either by swiping on the Google Glass touchpad or via voice commands.

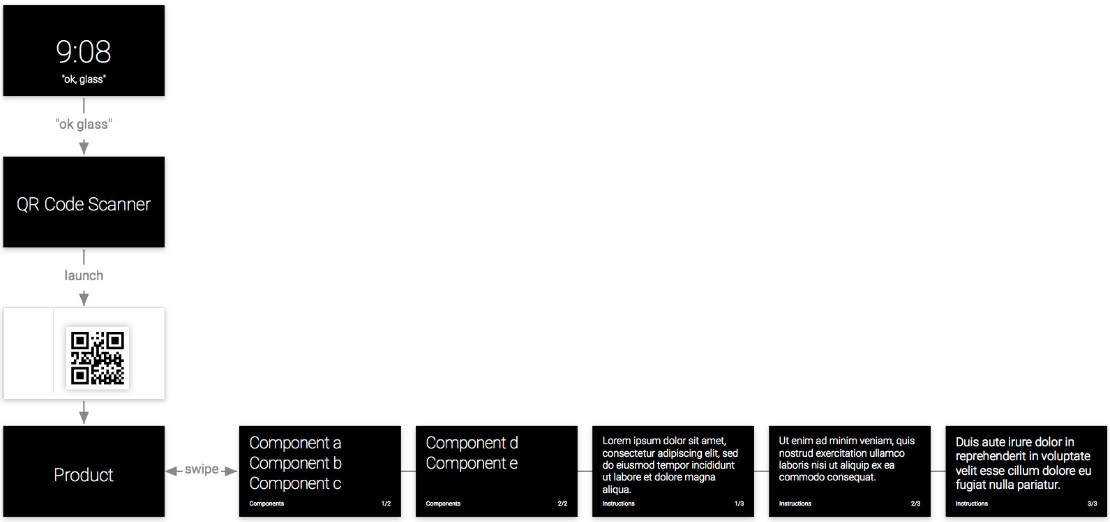


Figure 3.3: Glass Flow Design of the Google Glass application.

3.2 Presenting Information on Google Glass

As part of the design guidelines for Google Glass, Google provides developers with a card layout template, seen in Figure 3.4. The different coloured regions are intended for different types of information. The red area is the main area intended for presenting information in text form with the green squares representing the preferred margins. The thick blue stripe almost at the bottom marks the footer. The footer should hold supplementary information, such as a user name or a timestamp. The blue, slightly transparent, area to the left is mainly intended for images with associated text being presented to the right. The grey area, seemingly appearing behind all the other coloured areas, represent the entire card, with a size of 640 pixels wide and 360 pixels high [46].

Google goes even further in providing developers with guidelines for the design of cards.

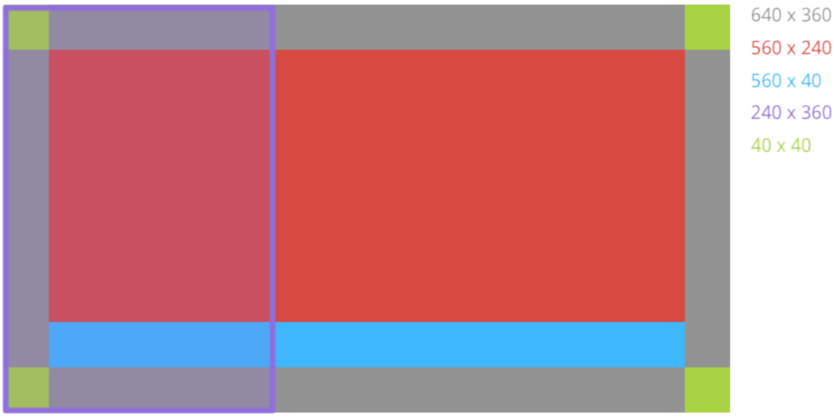
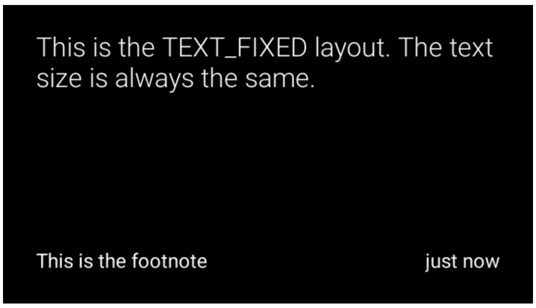


Figure 3.4: Google's design guidelines include a card layout template [46].

Google provides developers with a set of fixed card layouts. Specific card layouts set up the necessary margins and leave the developers to input the information to be displayed. Four examples of fixed card layouts can be seen in Figure 3.5.

(a) Text layout.



(b) Fixed text layout.

(c) Columns layout.

(d) Title layout.

Figure 3.5: Four different standard card layouts [45].

In terms of the information displayed, Google's default typeface family is called "Roboto". Google states that Roboto's geometrical forms and open curves makes for a natural reading rhythm [50]. Roboto is the typeface family used on all of Google's standard card layouts, some of which are seen in Figure 3.5. Google uses different typfaces from the Roboto typeface family for different texts [46]. Roboto Light is most common, with Roboto Regular being used for footnote text and Roboto Thin being used for larger texts, such as titles on the title card layout seen in Figure 3.5 (d).

One of the advantages of using Google's default layout is the fact that the text is dynamically resized to fit the card. Dynamically resized text means that the text is only as small as the text needs to be. However, there is a minimum size text may be downsized to. At 32 pixels the text is as small as the text may be and any text that does not fit on the card at that point is truncated.

Due to the limitation on the amount of text that may be presented on screen at the same time Google have provided developers with guidelines on how to present written information on Google Glass [46]. The guidelines for writing are five in total and read as follows:

- **Keep it brief.** Be concise, simple and precise. Look for alternatives to long text such as reading the content aloud, showing images or video, or removing features.
- **Keep it simple.** Pretend you're speaking to someone who's smart and competent, but doesn't know technical jargon and may not speak English very well. Use short words, active verbs, and common nouns.
- **Be friendly.** Use contractions. Talk directly to the reader using second person ("you"). If your text doesn't read the way you'd say it in casual conversation, it's probably not the way you should write it.
- **Put the most important thing first.** The first two words (around 11 characters, including spaces) should include at least a taste of the most important information

in the string. If they don't, start over. Describe only what's necessary, and no more. Don't try to explain subtle differences. They will be lost on most users.

- **Avoid repetition.** If a significant term gets repeated within a screen or block of text, find a way to use it just once.

Another part of Google Glass applications where Google have provided guidelines is voice commands [54]. However, voice commands might be the most restrictive area of all in terms of what Google recommend developers to do. Any voice command not officially approved by Google is not allowed in an application if the application is to be released on MyGlass. MyGlass is the official store from which users may buy their Google Glass applications. However, MyGlass is not installed on Google Glass but rather on, for instance, a smartphone [30].

Only by using the approved main voice commands [56] will applications be approved and allowed on MyGlass. These approved main voice commands does not, however, include for instance "next slide". Unapproved voice commands may be used during development and for separate releases. Developers may also use the built-in speech recognition. However, using speech recognition would mean not being able to launch the voice recognition simply saying "ok glass", which is the case with approved and unapproved voice commands.

In terms of getting a voice command approved, Google has set up a checklist for developers to cross off as they design their voice commands [53]. The checklist includes not designing a voice command which is similar to an already existing voice command. The voice command should also be long enough to ensure high recognition quality, yet short enough to fit on a single line on the Google Glass display.

3.3 Presenting Information on Smartphones

Despite being two different devices, the smartphone and Google Glass, Google's design recommendations for smartphones share similarities. For instance, similar to Google's design guidelines for Google Glass Google recommend developers to keep information brief. Google recommend developers to use short phrases with simple words [42].

However, in contrast to Google's design guidelines for Google Glass, Google does give developers freedom to make their own decisions. "Deviate with purpose", as Google states. Google recommend developers to develop applications which are easy and fun to use.

One design guideline for applications on smartphones provided by Google is to highlight what is most important in the application. For instance, in a camera application the shutter button is the most important button. As such the shutter button should be the most prominent component in the application and easy to find, similar to Figure 3.6, making the application easy to use in terms of the core feature.

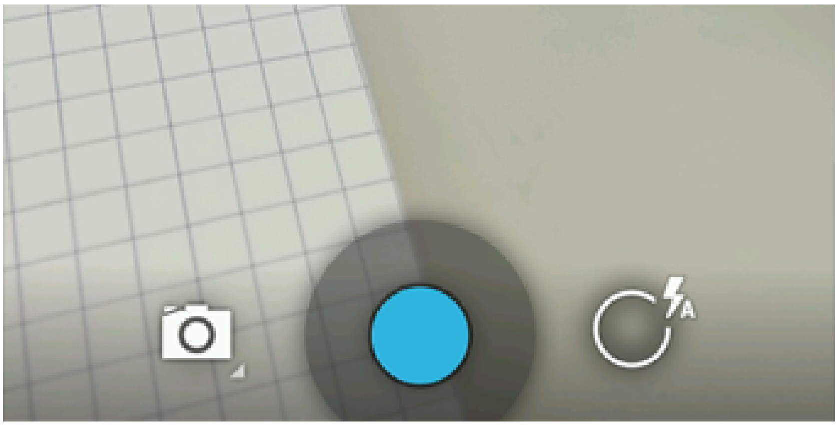


Figure 3.6: The most important component should be the most prominent [42].

In terms of clear cut differences between designing applications for smartphones compared to designing applications for Google Glass the most important one is perhaps where the user touches on the device. The Google Glass touchpad is mounted on the right hand side of the user, while the display sits in front of the user. On a smartphone the display and the touch-area is the same area. As such, buttons and other similar, intuitive, touch-objects are well suited for applications on smartphones. On Google Glass the user does not have any pointer on the screen, meaning that, for instance, menus can not be dependent on the user selecting an option by touching the option on the screen.

The use of voice command on Google Glass does help menus on Google Glass seem more similar to those on a smartphone since the users may select an option with their voice. However, the use of buttons and icons is less practical. On the other hand, since Google Glass may be controlled by voice command, developers might also hide functionality and decide not to show buttons and icons on the screen since the user does not need a target to touch in order to interact with the application.

An example of where Google Glass might hide functionality is in a camera application. In Figure 3.6 part of a camera application for smartphones is shown, with three buttons on screen. An application for Google Glass would not need to display any buttons since the users need only to say what they want to do. If, however, the Google Glass user did not use the voice command functionality, a simple touch of the Google Glass touchpad might for instance bring up a menu, displaying possible options, similar to how saying "ok glass" would bring up the voice menu.

All the above guidelines and restrictions have been taken in to consideration when designing the application for smartphone as well as Google Glass. While the application does have the same functionality, the most noticable difference between the two applications is that the smartphone application have buttons, enabling the user to jump forward, past slides to reach, for instance, the instructions, where as in the Google Glass application such features have been limited to voice commands.

3.4 Test Cases

The following aspects of the application are to be tested in order to help determine wether Google Glass is a viable option to use as replacement for an instruction manual when assembling components. Each test will be performed 30 times, for statistical significance [61].

3.4.1 Text Length

Since the Google Glass display is small and limited in space the amount of text that may by displayed on screen is as a result also limited. As such one interesting test case is to see where the text limit lies. The test consists of trying to find the the text length limit, on both the Google Glass display as well as a smartphone display. The text used should consist of characters distributed in similar fashion to English text [1].

When the smartphone display has been filled with information, how many slides does the same amount of information require on Google Glass? If the number of slides that the Google Glass application must use in order to display all the information is significantly larger than that of the smartphone application, the use of a Google Glass application might not be preferred as the user must use the slide more often than when using a smartphone equivalent application.

3.4.2 Distance to the QR Code

Generally, the recommendation regarding at what distance the user should be positioned in relation to the QR code is the size of the QR code times ten [4]. However, different devices will have different delay time before registering the QR code in frame. As such, one test regards the time from start of the camera until Google Glass has registered the QR code. The time is to be compared with that of smartphones.

As seen in Table 5.5 three different distances will be tested. The size of the QR code will be optimised according to the formula stated above, with the scanning distance set at two decimeters. The reason for testing for other scanning distances, yet with the same QR code size is because most users might not be aware of the optimal scanning distance, as well as to determine wether Google Glass has any advantages when scanning either closer or further away from the QR code.

The size of the QR code is calculated as

$$\frac{2}{10} = 0.2 \; decimeters$$

.

Table 3.1: Average time of registering a QR code with varying distances.

Distance (dm)	**Google Glass (ms)**	**Samsung Galaxy SII (ms)**	**Samsung Galaxy SIII (ms)**
1.0			
2.0			
3.0			

3.4.3 Complexity of the QR Code

Depending on the number of characters encoded by the QR code, the density of the QR code changes. The density of the QR code increases as the number of characters encoded by the QR code increases, where the number of black and white squares increase. As such one interesting test case is where the variable is the density of the QR code, or more specific; the number of characters encoded in the QR code.

The number of characters will vary between 1, 50 and 100. The values were chosen in order to give the results big enough room so that potential difference can be determined while still keeping the number of characters to a realistic minimum. Table 5.6 shows how the results will be presented, where the results will be the average of 30 test runs.

Table 3.2: Average time of registering a QR code with varying density.

Encoded Characters	**Google Glass (ms)**	**Samsung Galaxy SII (ms)**	**Samsung Galaxy SIII (ms)**
1			
50			
100			

3.4.4 Display Time

The speed at which Google Glass registers the QR code is important to wether the device is to prefer over regular smartphones. However, another interesting aspect is how fast downloaded information may be displayed on screen, from the point that the information has been downloaded. As seen in Table 5.7 the test will evaluate three different information sizes, meant to represent three different ways of presenting information. 100 kB represent text, 1 MB represent an image and 10 MB represent video.

Table 3.3: Average display time for Google Glass with varying information size.

Information Size (Byte)	**Google Glass (ms)**	**Samsung Galaxy SII (ms)**	**Samsung Galaxy SIII (ms)**
100 k			
1 M			
10 M			

3.5 Test Units

- Google Glass
- Samsung Galaxy SII
- Samsung Galaxy SIII

The reason for using these three units was the fact that the testing required physical devices. Since the testing included scanning a QR code from a various distances physical devices were necessary. What led to the two specific smartphone models (Samsung Galaxy SII and Samsung Galaxy SIII) were partially due to availability during testing. However, another reason was the fact that both models were among the most widely distributed during the time of Google Glass' release. Samsung Galaxy SII was released in 2011 and

Samsung Galaxy SIII in 2012, with Google Glass being released in 2013. Samsung Galaxy SII and Samsung Galaxy SIII have in total been sold in 100 million units to date [21, 27].

Another reason for not using more modern smartphones is due to the Google Glass version used. The Google Glass unit used was the so called Explorer Edition 1". Google Glass Explorer Edition 1 was released in February, 2013 [36]. An updated version, "Explorer Edition 2", was released in the summer of 2014 [31].

A few updates were done to Google Glass with the new version, most noticeably a doubling of available RAM. Google Glass Explorer Edition 1 has only 1 GB RAM [20] where Google Glass Explorer Edition 2 has 2 GB RAM [31]. Samsung Galaxy SII and Samsung Galaxy SIII both have 1 GB RAM as well [76, 77]. As such using more modern smartphones in testing could be seen as unfair for Google Glass since Google Glass also exists in a more modern version.

Additional technical specifications regarding the test units can be found in Table 3.4.

Table 3.4: Technical specifications of the three test units.

Component	**Google Glass [70]**	**Samsung Galaxy SII [76]**	**Samsung Galaxy SIII [77]**
RAM	1 GB [20]	1 GB	1 GB
CPU	OMAP 4430 dual core [81]	1.2 GHz dual core ARM Cortex A9	1.4 GHz quad core Cortex A9
Screen Size	Equivalent of a 25 inch screen from 2.5 meters [52]	4.3 inches	4.8 inches
Screen Resolution	640 * 360 pixles	480*800 pixles	720*1280 pixles

3.6 Summary

The application designed and implemented in parallel with this report is a proof-of-concept of an application used for assembling components. The functionality of the application may be divided in to several steps, as seen in Figure 3.1. First the application scans a QR code. The QR code is then decoded and the decoded information from the QR code is used in order to download instructions on the specific product belonging to the QR code which was scanned. The downloaded instructions are sorted in to classes and displayed on screen.

The application was designed to be run on Google Glass, however a proof-of-concept of a similar application for smartphones has been designed and implemented as well. The smartphone application will provide a point of reference as well as help evaluate the pros and cons of using Google Glass. The application is designed as a slide view, with each instruction appearing on a separate slide. The slide view design was used so that users may focus on one instruction at a time.

The design of the application also follows the design guidelines provided by Google. For instance Google recommend developers to keep their application and the content within the application simple. Google recommend developers of Google Glass applications especially to mind the small screen and as such to keep the information brief, and the most important information first.

Google provides developers designing applications for Google Glass with even more specific recommendation, as Google recommend using the predefined card layouts, some of which can be seen in Figure 3.5.

The design guidelines for smartphone applications are not as specific, although Google does recommend keeping information simple. Google also recommend developers to highlight the most important feature of an application. One example of how the guidelines have been taken in to consideration when design the application is how both the Google Glass application and the smartphone application starts in camera view, where the user may scan a QR code. There is no start menu, instead the most important feature is the first screen the user sees when launching the application.

The application will also be tested, both on Google Glass as well as smartphone. One test case is text length, as one limitation of Google Glass is the small display. The test will consist of maximising the number of characters that may fit on one slide in the smartphone application, and comparing the result with how many slides the same number of characters would require in the Google Glass application.

Another test comprise the distance from the device to the QR code. Are there any differences between the smartphone application and the Google Glass application when the distance to the QR code is altered? The complexity of the QR code is another test which will be performed. Does the complexity of the QR code, which increases with the number of characters encoded, have any impact specific to either the smartphone application or the Google Glass application. The fourth and final test to be performed measures the display time. Does the display time differ between the smartphone application and the Google Glass application for different sizes of information?

All of these tests will be performed 30 times for statistical significance, and all of these tests will be performed on Google Glass as well as the smartphones Samsung Galaxy SII and Samsung Galaxy SIII. The reason for using the two specific smartphone models was for one the need of physical devices used for testing, but also the fact that these two smartphones were very prominent on the market when Google Glass was released in 2013. Samsung Galaxy SII and Samsung Galaxy SIII has been sold in a total of 100 million units and are as such good representations of smartphones comparable to Google Glass.

4 Implementation

As the application launches, the first screen the user sees, in both the Google Glass version and the smartphone version, is the camera screen. The user must, in order to proceed further within the application, scan a QR code. Scanning a QR code is done by positioning the device's camera such that the QR code can be seen on screen, as seen in Figure 4.1. The user does not need to press any shutter button as the application automatically recognises the QR code pattern if seen on screen.

(a) The Google Glass application

(b) The smartphone application.

Figure 4.1: The application is scanning a QR code.

The reason for not providing a menu on the start screen or even requiring the user to press a shutter button is because the application should be simple, easy to use and focus on what is important. Since the the first step when using the application is to scan a QR code in order to receive the necessary information on the specific product, the scanning is also the main focus on the first screen of the application.

When the QR code has been scanned the application decodes the QR code. The decoding process is handled by the Zebra Crossing (ZXing) library [79]. ZXing is an open source barcode image processing library which uses the default QR code reader installed on the device.

The smartphone application was based directly on the ZXing library, where as the Google Glass application was based on a port of the ZXing library to Google Glass, called "BardcodeEye" [25]. The main difference between ZXing and BarcodeEye is the fact that BarcodeEye is an example application ready to be run on Google Glass, in contrast to the ZXing library which is only a library and as such needs to be attached to a runnable application.

The BarcodeEye application for Google Glass is however a bare bone application, used as an example and introduction as to how ZXing may be implemented in an application for Google Glass. BarcodeEye displays the decoded information from the QR code and also gives the user the option to search the internet using the information previously decoded from the QR code.

As the QR code is meant to encode only a product ID, and the application will then use the ID to download the product information (rather than having all of the instructions encoded directly in the QR code), the BarcodeEye application had to be modified. The first modification was on the graphical layout of the BarcodeEye application. The change of layout was mostly done due to the fact that the BarcodeEye application only displayed plain text, not taking in to account for instance a mix of image and text.

In order to display information BarcodeEye used the deprecated class `Card`, as seen in Listing 4.1. The application now instead uses the `CardBuilder` class, as seen in Listing 4.2, as recommended by Google [40]. The `CardBuilder` class allows users to input a desired layout style as an argument to the constructor of the `CardBuilder` class. With the `Card` class developers could in a separate method call set where on the card the image would appear. For instance, in order to create a columns card, as seen in Figure 4.6, the code would look as the code in Listing 4.1. Using the recommended `CardBuilder` class instead would look as the code in Listing 4.2. With `CardBuilder` developers can now use default layouts, enabling more consistent application design across Google Glass applications. If a developers wishes to use a custom layout instead, the developer can simply use the custom layout as input to the `CardBuilder` constructor instead.

Listing 4.1: Instancing of the deprecated class Card

```
Card card = new Card(context);
card.setImageLayout(Card.ImageLayout.LEFT);
```

Listing 4.2: Instancing of the recommended class CardBuilder

```
CardBuilder cardBuilder = new CardBuilder(context, CardBuilder.Layout.COLUMNS);
```

Since the smartphone application also used the ZXing library, but without any preexisting application, no changes similar to those done to the Google Glass application had to be done for the smartphone application. Instead the smartphone application was built from the ground up to make use of the ZXing library's functionality, similar to how the ZXing library was integrated in the base Google Glass application.

Having scanned and decoded a QR code the decoded information is then used to download information on the specific product. The downloaded information contains the product name, as well as necessary components and instructions for assembling the product. The components and instructions may be represented by text, images or both.

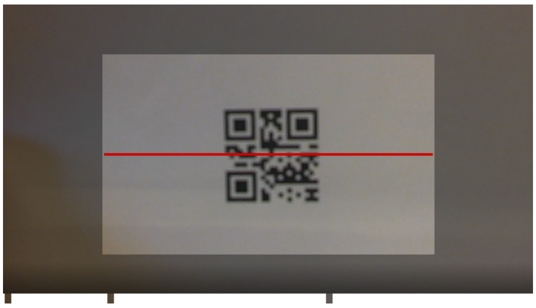
The way the download process uses the decoded product ID is by concatenating the product ID with a URL adress, which connects to a database containing all necessary information regarding the specific product. The download process was done using a `AsyncTask`. An AsyncTask is a class which performs operations in the background on a different thread than the user interface, such that the user interface is not frozen when the download process is taking place [44].

In the Google Glass application a loading bar pops up at the bottom of the screen when information is being downloaded, as seen in Figure 4.2 (a), indicating that the application is loading. Google calls the bar an “ndeterminate Slider”, as the the class used in implementation is the same as for a regular Slider [51]. A similar animation was added to the smartphone application. However, in contrast to the Google Glass application's loading bar the smartphone application instead displays a spinning wheel at the center of the screen, seen in Figure 4.2 (a), indicating that the application is loading. The spinning wheel is implemented by the `ProgressDialog` class [?].

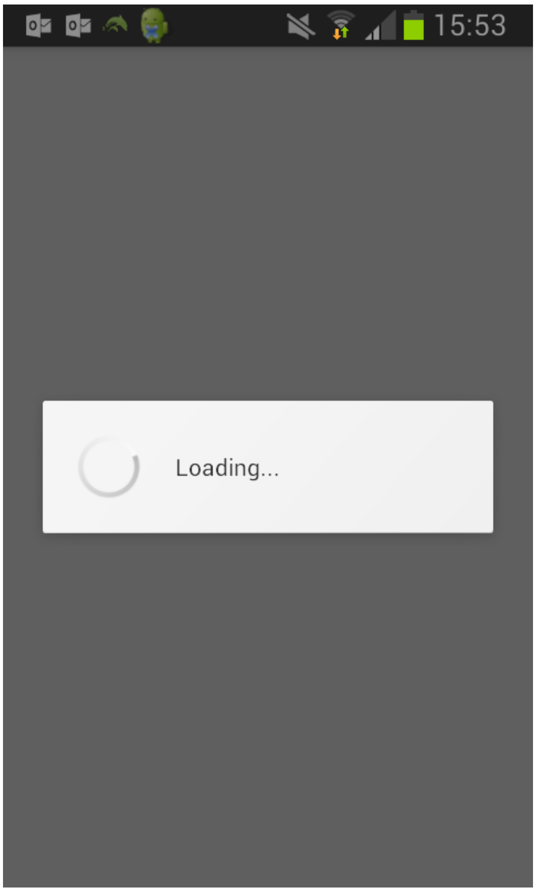
in both cases of loading animation the animation is started in the `AsyncTask`, just before the download begins. The loading animation stops when the `AsyncTask` has finished downloading the product information.

The download process also includes creating and initialise an instance of the `Products` class. The instance contains the name of the product and an image of the product as the product will look when the user is done assembling all the components (the existence of an image is dependent of whether there was an image of the product stored in the database or not).

The Products class instance will also contain a list of components as well as a list of instructions. Both components and instructions are classes themselves. Similar to the `Products` class, instances of both the `Components` class and the `Instructions` class will contain a string and potentially an image, depending on wether there is an image stored in the database or not. In the case of components the string attribute will contain the name of the component, in contrast to instances of the `Instructions` class where the string instead will contain the instruction itself.



(a) The Google Glass application



(b) The smartphone application.

Figure 4.2: The loading screens.

When the downloaded information is being displayed, the first screen the user sees contains the product name as well as an image of the product (if an image existed in the database). In the example case, lego parts are to be assembled in order to construct the so called “Space Pirate”, seen in Figure 4.3.

The first information the user sees displayed on screen after the QR code has been scanned and the information has been downloaded is the title page for the Space Pirate product, seen in Figure 4.4.

The next slide in line after the title slide is the first slide containing information on the components necessary for constructing the specific product. Each component has their

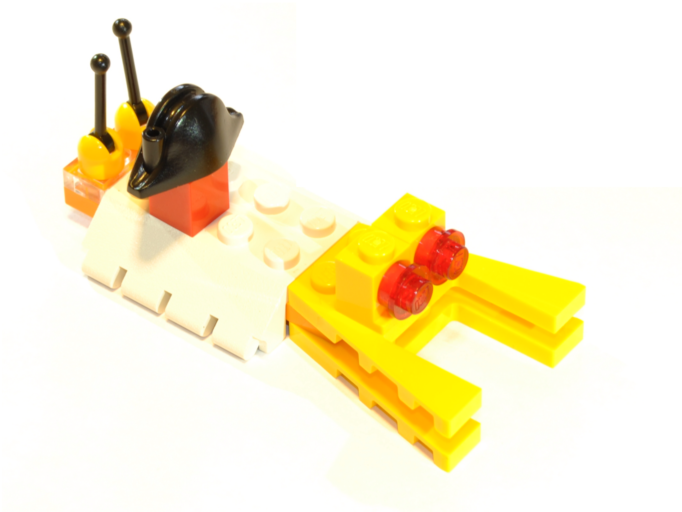


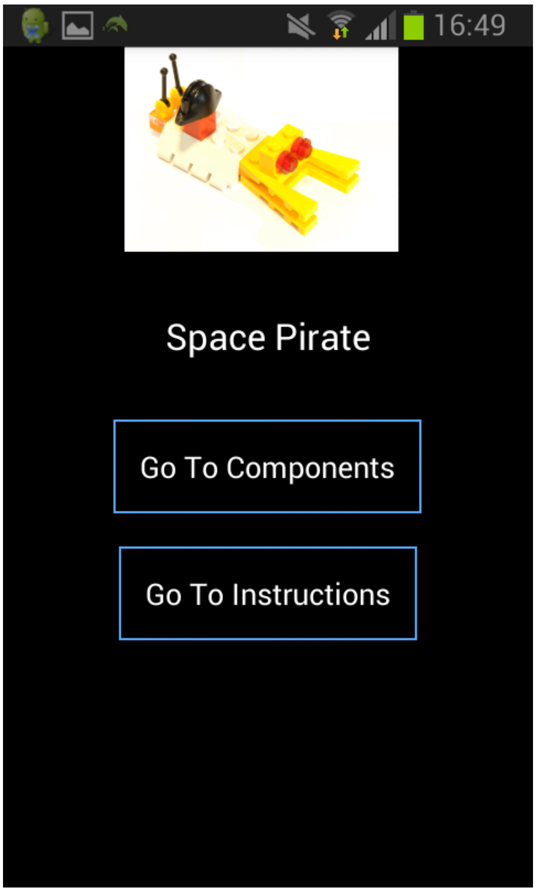
Figure 4.3: The product.

own slide as the component may contain an image in complement to the name of the component. Examples of a component described in only text can been seen in Figure 4.5 and examples of when the component has been described with both text and image can be seen in Figure 4.6.

As seen in Figure 4.5 the text in the Google Glass application is larger than the text in the smartphone application. The difference in size is due to the automatic scaling in the Google Glass application. If the text in the Google Glass application where to cover the entire screen the text size would automatically be downscaled. In the smartphone application, however, the text is always of the same size. The size of the text in the smartphone application was set to `MEDIUM` which is one of the three different default sizes of text used in android applications. The footer text, which in Figure 4.5 says “Components”, as well as “1/4”, is of the default size `SMALL`. In Figure 4.4 the size of the product name text is of the default size `LARGE`. The reason behind the choice of the different text sizes is to keep the smartphone application similar to the Google Glass application, making the comparison of the two version easier.

The card design used for the component slide containing only text, seen in Figure 4.5,

(a) The Google Glass application



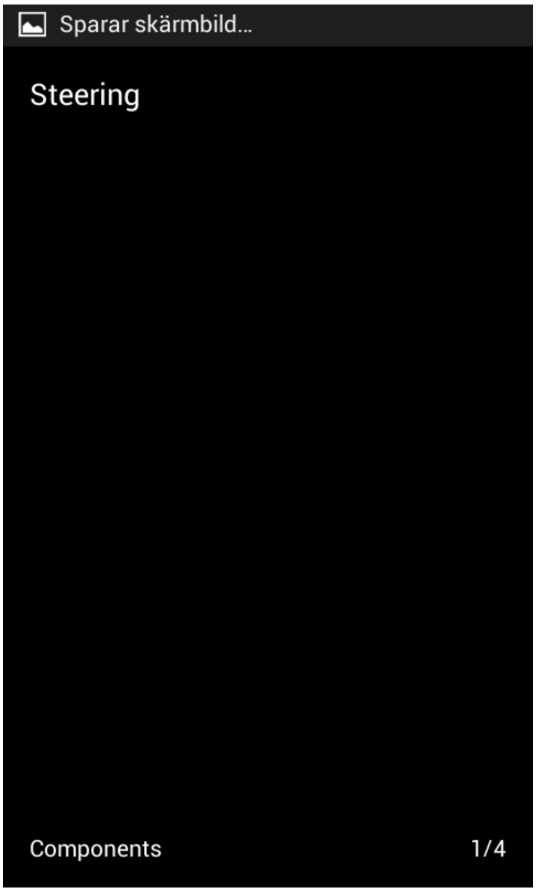
(b) The smartphone application.

Figure 4.4: The title card of the demo application.

is the predefined `TEXT` layout, used in similar fashion to how the title card design was specified in Listing 4.2. A component slide may also contain an image as complement to text. The card design used for the component slide containing both text and image, seen in Figure 4.6 is the predefined `COLUMNS` layout, used in similar fashion to how the title card design was specified in Listing 4.2

After all the component slides comes the instruction slides. These slides may also contain either only text or both text and an image, and these layouts are specified the same way as the component slides. However, an instruction slide may also consist of only an image, and no text at all. The reason for giving an image exclusive layout as a possible

(a) The Google Glass application

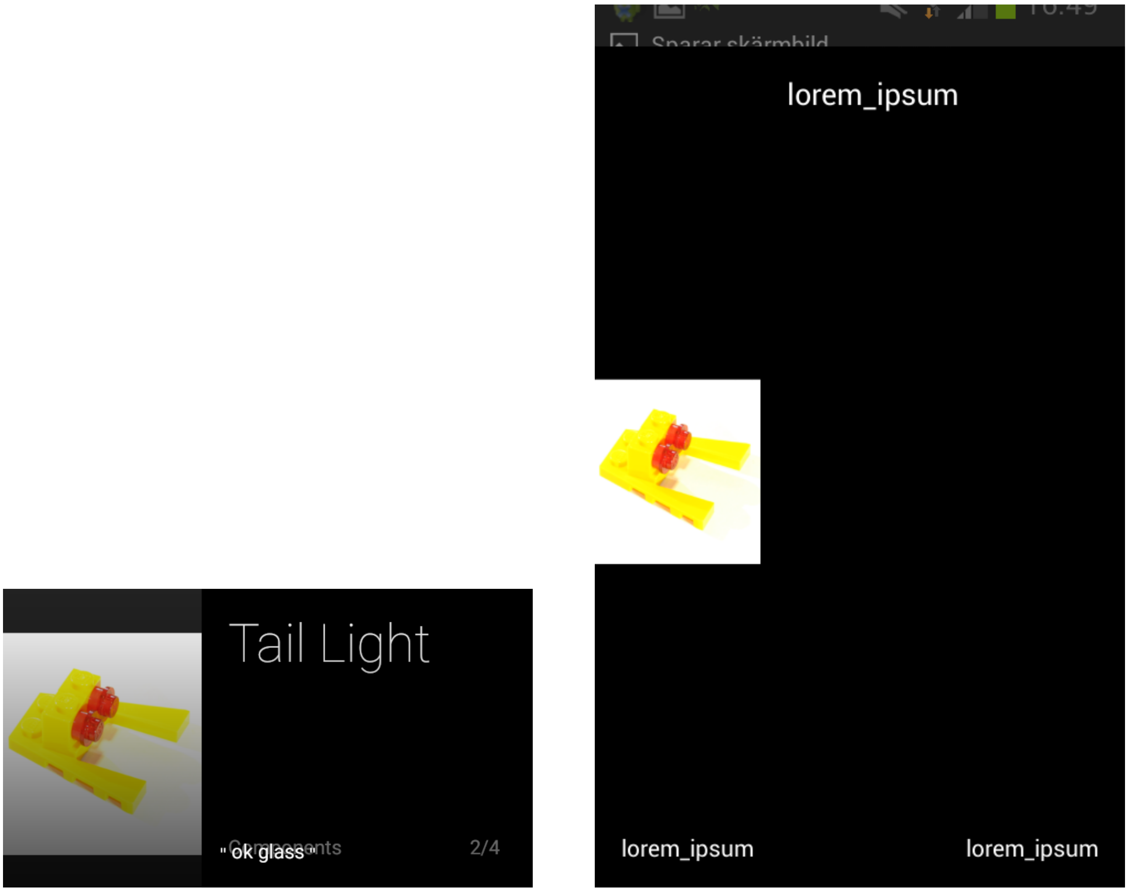


(b) The smartphone application.

Figure 4.5: A component slide from the demo application.

card layout for an instruction slide is because an instruction may be best described with an image. Describing the same instruction in text may take up more slides than the image, as the image will only take up one slide. A slide containing only an image and no text will also mean that the image will be scaled larger than when using both text and an image. As such more detail can be shown and users may get a better understanding for how the components should be assembled.

As seen in Figure 4.7, one instruction is described with only an image and no text. Since the placement of the components are important, and potentially hard to specify in text, an image may describe the placement better. Note also that the smartphone application still



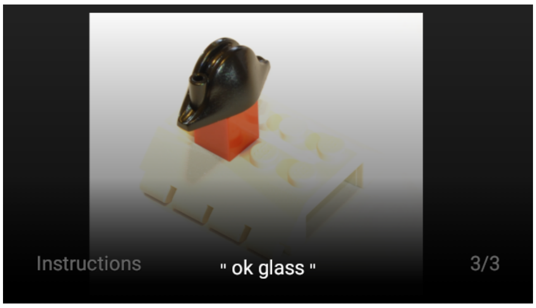
(a) The Google Glass application

(b) The smartphone application.

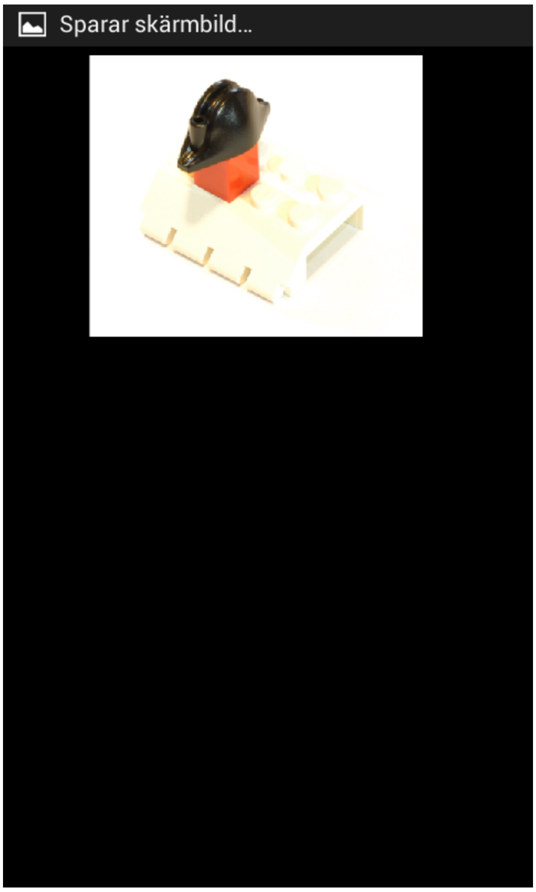
Figure 4.6: A component slide from the demo application.

has room for text. Potentially the image in the smartphone application could be expanded, for instance by giving the user the ability to zoom by pinch. However, such a feature does not exist.

The Google Glass application may be controlled by swiping across the Google Glass touchpad, but the Google Glass application may also be controlled using voice commands. The voice commands can be used as simple replacement for the touchpad, where users may swipe cards both backwards and forwards. However, using voice commands users may also “jump” in the application. By for instance saying “ok glass, show components” the application jumps to the first component slide, regardless of which slide the user is



(a) The Google Glass application



(b) The smartphone application.

Figure 4.7: An instruction slide from the demo application.

currently viewing. If the user is currently already viewing the first component slide nothing will happen. The voice command menu of the Google Glass application can be seen in Figure 4.8. Although the smartphone application does not have any voice command features the title slide contains buttons, as seen in Figure 4.4 (b) giving the user the ability to skip straight to the instructions is so desired.

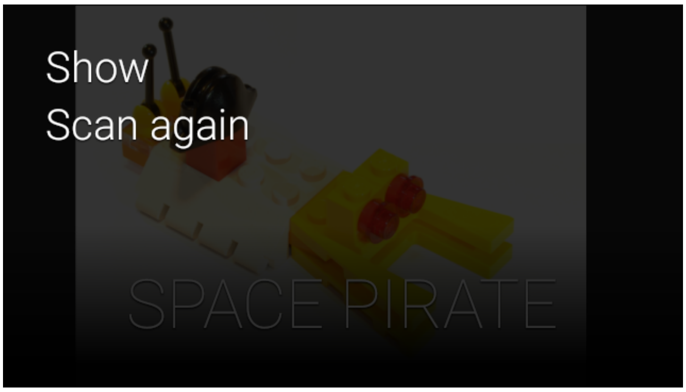


Figure 4.8: The voice command menu in the demo application.

4.1 Android Studio

Both the smartphone application as well as the Google Glass application were developed in Android Studio [43]. Android Studio is a development environment developed by Google. Both applications were initially being developed in Eclipse [35], however development soon shifted to Android Studio as Android Studio is now the official integrated development environment (IDE) for Android [41]. The shift was done without complications as Android Studio contains an import feature enabling developers to import projects previously not developed in Android Studio [41].

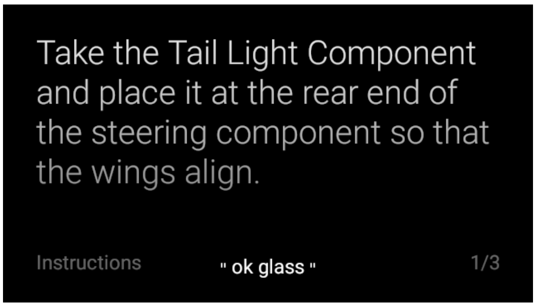
4.2 Card Layout

Google provides developers with a set of predefined layouts for different types of cards, which were used in the Google Glass application and used as basis for the design of the different layouts for the slides in the smartphone application. The following predefined layouts were used in the implementation: “Title”, “Columns” and “Text”. The Title layout was used for the first card of the slide view, which shows the product name as well as an image of the product as it is supposed to look when finished.

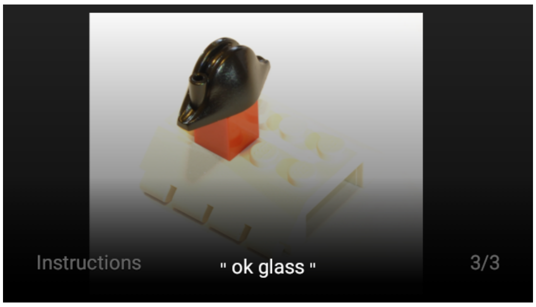
The Columns card layout, seen in Figure 4.9 (b) was used for when an instruction or component was to be presented with both text and an image. Since the Columns layout

(a) The title card layout.

(b) The column card layout.



(c) The text card layout.



(d) The image card layout.

Figure 4.9: The different layouts used within the Googla Glass application.

split the card, with an image to the left and text to the right, the Columns layout was the most reasonable choice when presenting both text and an image. An alternative would have been to display the text on top of the image, the image could potentially have been hidden behind a larger amount of text.

Such a layout design was instead used for the title card as the amount of text being displayed is only the name of the product, and the image is only to give an idea of what the finished product will look like. The layout design where the text overlapped the image was called Title and can be seen in Figure 4.9 (a).

If the information, either a component or an instruction, instead were to be presented only as text the Text layout, seen in Figure 4.9 (c) was used. The Text layout displayed dynamically sized text. In other words, if there was a lot of text being displayed the text would be resized to fit the screen.

When the card were to only contain an image and no text the Caption layout was used, seen in Figure 4.9 (d). The Caption layout is similar to the Title layout, but in contrast to the Title layout the Caption layout also have both a footer and a timestamp at the bottom of the card on the left and right side respectively. The Caption layout enables the use of an actual caption, however a caption is not necessary and neither is a caption used in the application as the Caption layout was used when no text was meant to be displayed (not counting the footer and the timestamp which appears on all cards except for the title card).

Using the predefined layouts in the implementation was easily achieved as the process consisted mostly of plug-and-play. The `CardBuilder` class constructor took the layout as an argument, as seen in Listing 4.3. When and instance of the `CardBuilder` class was created what remained was to simply input the necessary information, such as the instruction text. Setting an image was done slightly differently than written information as images were loaded in using a separate thread. As soon as the `CardBuilder` method `getView` was called the card was built with the information that had been inputed.

Listing 4.3: Initialisation of the CardBuilder class

```
CardBuilder cardBuilder = new CardBuilder(context, CardBuilder.Layout.COLUMNS)
   .setText(getText())
   .setFootNote(mFootNote)
   .setTimestamp(mTimeStamp);

cardBuilder = (new LoadImage(isTitleCard(),
    getByteArray()).doInBackground(cardBuilder));

return cardBuilder.getView();
```

4.3 Voice Commands

The Google Glass application gives users the option to use voice commands in order to navigate the slides. The user opens the voice command menu by saying "ok glass" at any point in the application when "ok glass" is shown at the bottom of the screen. The voice command feature is available at all times except when the camera is active. In other words the voice commands are unavailable when the application is waiting to scan a QR code.

The voice command menu contains the following options.

- **Show next slide**

  The application scrolls to the next slide. If the current slide is the last slide, and in other words no other slides are following, the application does nothing.

- **Show previous slide**

  The application scrolls to the previous slide. If the current slide is the first slide, and in other words no other slides comes before the current slide, the application does nothing.

- **Show components**

  The application scrolls to the first slide showing information on a component. If the user is currently on the first slide showing information on a component the application does nothing.

- **Show instructions**

  The application scrolls to the first slide showing an instruction. If the user is currently on the first slide showing an instruction the application does nothing.

- **Scan again**

  The application launches the camera and expects the user to scan another QR code.

Implementing voice command in the Google Glass application is done by following Google's step-by-step guide on how to implement voice commands in Google Glass applications [55]. Listing 4.4 shows the XML written according to Google's step-by-step guide, which gives the voice commands used in the application. Using contextual voice commands according to Google's step-by-step guide means that "ok glass" is displayed at the bottom of the screen at all times when voice commands are available. "ok glass" also comes with a dark overlay, seen for instance in Figure fig:cardLayout, which is transparent yet darkens the slides a bit, especially near the bottom of the slides where "ok glass" is shown. Although the dark overlay could potentially distort the slides a bit, especially when the slide contains only an image and no text, alter the "ok glass" overlay in any way is not possible [32, 33]. The dark overlay does however ensure that "ok glass" is always visible, no matter what the background image look like.

Listing 4.4: The voice command menu XML file

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
  <item
    android:id="@+id/next_menu_item"
    android:title="Show next slide" >
  </item>
  <item
    android:id="@+id/previous_menu_item"
    android:title="Show previous slide" >
  </item>
  <item
    android:id="@+id/components_menu_item"
    android:title="Show components" >
  </item>
  <item
    android:id="@+id/instructions_menu_item"
```

```
        android:title="Show instructions" >
    </item>
    <item
        android:id="@+id/scan_menu_item"
        android:title="Scan again" >
    </item>
</menu>
```

Although none of the voice commands have been sent in for official approval by Google most of the voice commands follows the design guidelines provided by Google [53]. As seen in Table 4.1 11 of the 15 voice command guidelines provided by Google has been followed. However, some of the guidelines has been applied to some or most of the voice command used within the application, and not all. For instance “Show components” does not follow the first guideline of the voice command checklist. However, the “Show components” voice command remains a part of the application as “Show components” is a key feature of the application.

The reason for not following guidelines 11–14 is because they would make the voice commands longer. As the voice commands may potentially be said often while using the application, as the user may proceed through the slides quite fast, shorter voice commands makes for more comfortable use. As the voice commands still follows guideline 6 the voice commands were deemed to be long enough to still ensure high recognition quality.

4.4 Test Cases

The following section describes how the tests were set up and carried out.

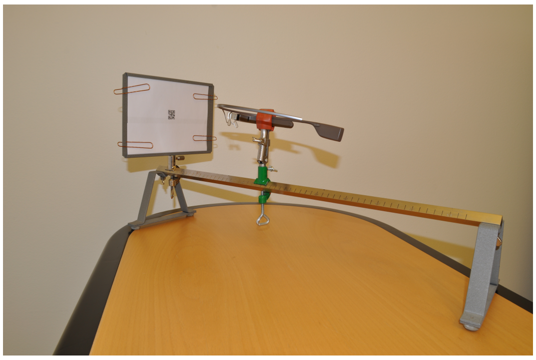
Table 4.1: Voice Command Checklist [53].

	Guideline	**Acheived**
1	Is general enough to apply to multiple Glassware, but still has a clear purpose	Yes
2	Is colloquial and can explain Glass features in a conversation	Yes
3	Is comfortable to say in public	Yes
4	Brings the user from intent to action as quickly as possible	Yes
5	Avoids brand words	Yes
6	Is long enough to ensure high recognition quality (at least three syllables)	Yes
7	Fits on a single line	Yes
8	Does not sound similar to existing commands	Yes
9	Does not require immediate interactivity in Mirror API Glassware.	Yes
10	Has an imperative verb with an object	Yes
11	Uses articles when possible	No
12	Uses definite articles only when the object is definite	No
13	Uses "this" when there is only one relevant instance of the object	No
14	Uses me and my when appropriate	No
15	Refers to Glass as the subject carrying out the action	Yes

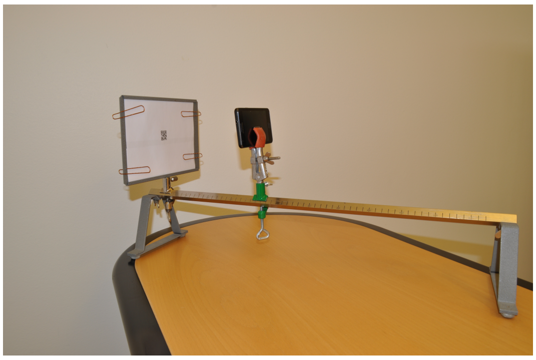
4.4.1 Experimental Setup

The tests were carried out using an optical bench to guarantee scientific accuracy. The experimental setup contained an optical bench, with a screen holder at the zero point, where the QR code was positioned. The device being tested, Google Glass or smartphone, was then positioned at the specified mark on the optical bench using a clamp and pointed towards the QR code. See Figure 4.10 for a better understanding of the experimental setup.

As seen in Figure 4.10 Google Glass were mounted in such a way that the camera sat a bit closer to the QR code than where the clamp marked on the optical bench. In order to compensate for the slight misalignment the clamp was positioned a few centimeters back from the specified mark and as such not used to determine the distance to the QR code. Instead the camera on Google Glass was used to pin point the exact distance to the QR code, and the clamp was positioned in such a way that the camera on Google Glass was at the distance specified in each experiment. In other words, even though the clamp was not at the same distance to the QR code for the smartphone tests as for the Google Glass tests, the camera of each device was.



(a) The experimental setup for Google Glass.



(b) The experimental setup for Samsung Galaxy SII.

Figure 4.10: The experimental setup.

Although not shown i Figure 4.10 each device was connected to a computer via a USB cable. The result time of each test was obtained from the log within Android Studio after each run, since when running an android application via Android Studio, log information may be obtained through the log within Android Studio.

In order to measure the time needed for the results of each test a specific class was built, called `Timer` (seen in Listing 4.5). The `Timer` class was built using the singleton design pattern. A singleton class is a class that can only be instanced once during the entire execution of an application. However the instance of a singleton class lives throughout the entire execution and may be accessed from anywhere in the application.

Using the singleton pattern meant that the timer could be started in one class, and stop in another without having to pass the instance around, which could potentially affect the performance of each device.

Listing 4.5: The Timer class

```

public class Timer {
  private static Timer ourInstance = new Timer();
  public static Timer getInstance() { return ourInstance; }
  private Timer() { }

  private boolean timerRunning = false;
  private Long startTime;
  private Long stopTime;

  public void startTimer() {
    if(timerRunning) { Log.d("TIMER", "Timer already running"); }
    else  { startIme = System.nanoTime(); }
  }

  public void stopTimer() {
    if(!timerRunning) { Log.d("TIMER", "No timer running"); }
    else { stopTime = System.nanoTime()); }
  }

  private long getElapsedTime(int timerID) { return stopTime - startTime; }

  public void logElapsedTime(String information) {
    Log.d("TIMER", information + ": " + String.valueOf(getElapsedTime() + "
       nano seconds");
  }
}
```

4.4.2 Text Length

When evaluating the text length the text string used was not a predefined one, but rather a randomised one. The text was randomly generated using the distribution of characters in regular English text. One might argue that technical texts have a slightly different distribution of characters, but Google recommend developers to be personal when writing text meant to be displayed to the user [46]. The text was also short enough to fit a smartphone screen and as such the difference using slightly different distribution of characters would not have any major effect on the results.

Listing 4.6 shows how each character was randomly selected. `randchar` was called from within a for loop where the character was added to a string. The number of loops determined how long the text was going to be. The `doubleList` list contained the distribution of each character according the their distribution in the english language [1] and the `alph` list contained all the individual characters, including whitespace. As a decimal number between zero and one was randomly selected a corresponding character was picked out based on the distribution of that character.

A recursive method located the corresponding character and returned the character back up to the calling method `randchar` which in turns returned the character to the for loop in which a string of random characters were collected.

Listing 4.6: The randomizer class

```
public RandomizeEnglishText() {
  doubleList = new ArrayList<>();
  doubleList.add(0.0651738);          // A
  doubleList.add(doubleList.get(0) + 0.0124248);  // B
  doubleList.add(doubleList.get(1) + 0.0217339);  // C
    [...]
  doubleList.add(doubleList.get(23) + 0.0145984); // Y
  doubleList.add(doubleList.get(24) + 0.0007836); // Z
```

```
    doubleList.add(doubleList.get(25) + 0.1918182); // _

    alph = Arrays.asList("a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k",
        "k", "m", "n", "o", "p", "q", "r", "s", "t", "u", "v", "w", "x", "y",
        "z", " ");
}

private double randfrom(double min, double max) {
    Random rand = new Random();
    double range = (max - min);
    return min + range * rand.nextDouble();
}

private String getChar(int pos, double rand) {
    return (rand <= doubleList.get(pos) || pos+1 <= alph.size()) ?
        alph.get(pos) :
        getChar(pos+1, rand);
}

public String randchar() {
    double rand = randfrom(0, 1);
    return getChar(0, rand);
}
```

4.4.3 Distance to the QR Code

Using the `Timer` class, seen in Listing 4.5, the elapsed time from the start of the application until the QR code was decoded was measured for each device. The test was done 30 times for each device, with 3 different distances between the QR code and the device. Figure 3.1

described the application functionality and this test measures steps 1 and 2 in Figure 3.1. The results of the test can be seen in detail in Appendix B but will also be presented in a smaller version with the average time in a table similar to Table 5.5. Although the `Timer` class will give the elapsed time in nano seconds the result will be presented in seconds in order to clearly show the significance of each time as small, nano seconds, differences between devices will not matter as much as seconds.

Table 4.2: Average time of registering a QR code with varying distance.

Distance (dm)	**Google Glass (sec)**	**Samsung Galaxy SII (sec)**	**Samsung Galaxy SIII (sec)**
1.0			
2.0			
3.0			

4.4.4 Complexity of the QR Code

Using the `Timer` class, seen in Listing 4.5, the elapsed time from the start of the application until the QR code was decoded was measured for each device. The test was done 30 times for each device, with 3 different complexities of the QR code. Figure 3.1 described the application functionality and this test measures steps 1 and 2 in Figure 3.1. The results of the test can be seen in detail in Appendix B but will also be presented in a smaller version with the average time in a table similar to Table 5.6. Although the `Timer` class will give the elapsed time in nano seconds the result will be presented in seconds in order to clearly show the significance of each time as small, nano seconds, differences between devices will not matter as much as seconds.

Table 4.3: Average time of registering a QR code with varying density.

Encoded Characters	**Google Glass (sec)**	**Samsung Galaxy SII (sec)**	**Samsung Galaxy SIII (sec)**
1			
50			
100			

4.4.5 Display Time

Using the `Timer` class, seen in Listing 4.5, the elapsed time from the point that the product information had ben downloaded until the information was sent to the display was measured for each device. The test was done 30 times for each device, with 3 different sizes of product information. Figure 3.1 described the application functionality and this test measures steps 4 and 5 in Figure 3.1. The results of the test can be seen in detail in Appendix B but will also be presented in a smaller version with the average time in a table similar to Table 5.7. Although the `Timer` class will give the elapsed time in nano seconds the result will be presented in seconds in order to clearly show the significance of each time as small, nano seconds, differences between devices will not matter as much as seconds.

Table 4.4: Average display time for Google Glass with varying information size.

Information Size (Bytes)	**Google Glass (sec)**	**Samsung Galaxy SII (sec)**	**Samsung Galaxy SIII (sec)**
1 000			
100 000			
1 000 000			

4.5 Summary

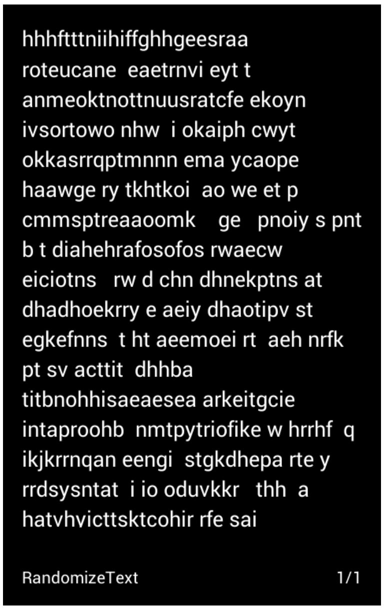
5 Results

The following are the results of the tests performed on the application, both on the Google Glass version as well as the smartphone version.

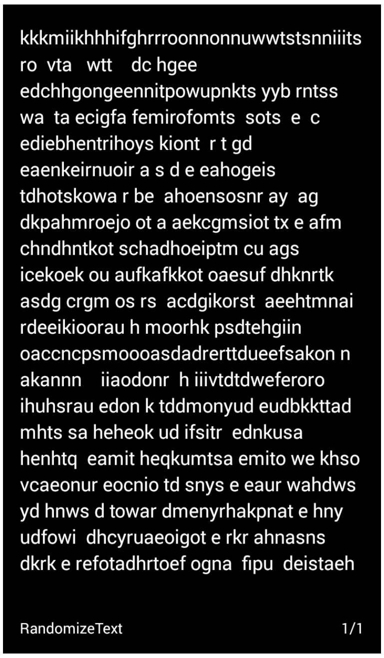
5.1 Text Length

The following are the results from evaluating the number of characters that may fit on a card in the Google Glass application, both when using only text and when using both text and an image. The test was performed only once for each device as despite randomising the characters used the text length would not differ significally from one case to another. As seen in the following results none of the tests resulted in an additional card with only one character. Had that been the case, different combinations of characters may have changed the number of cards necessary to display all of the information. As that is not the case only one test was deemed sufficient enough to determine the number of card required to display the same information as displayed on a smartphone device.

However, some trial and error was part of the process of finding how many characters would fill the smartphone screens. 550 was deemed a suitable number for Samsung Galaxy SII och 750 was deemed a suitable number for Samsung Galaxy SIII, as that many characters filled up the screen and yet still left some room for longer characters or longer words, as seen in Figure 5.1. For Samsung Galaxy SII, 600 characters did not fit on the screen, and for Samsung Galaxy SIII, 800 characters did not fit on the screen. As such 550 and 750 respectively was set as the maximum number of characters, although depending on the characters being displayed the text may use up more or less space as an 'i' does not take up as much space as a 'w'. The filled smartphone screens can be seen in Figure 5.1.



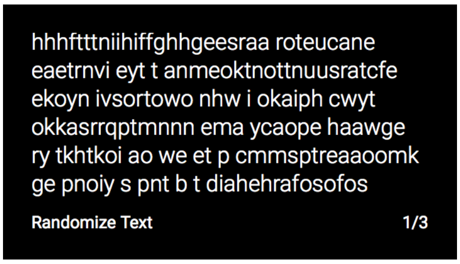
(a) The Samsung Galaxy SII screen.



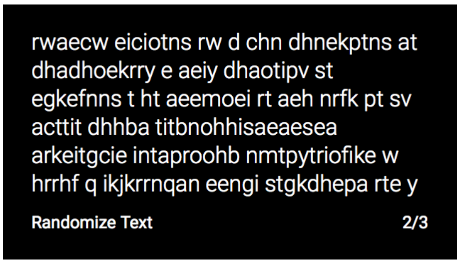
(b) The Samsung Galaxy SIII screen.

Figure 5.1: The maximum text length on the smartphone application.

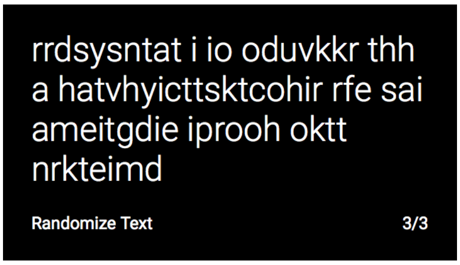
When translating the results from the smartphones to Google Glass the same text was used and simply copied over to the Google Glass application. The result of the text that filled the Samsug Galaxy SIII screen, seen in Figure 5.1 (a), can be seen in Figure 5.2. As seen the text that filled the Samsung Galaxy SII screen took up roughly two and a half card in the Google Glass application. Although the screen is nearly filled in Figure 5.2 (c) the text is dynamically sized depending on the amount of text being displayed. As such the card can be deemed about half full. Table 5.1 shows the exact number of characters and words on each card, showing that the last card contains about half the number och characters as the other two cards.



(a) The first slide.



(b) The second slide.



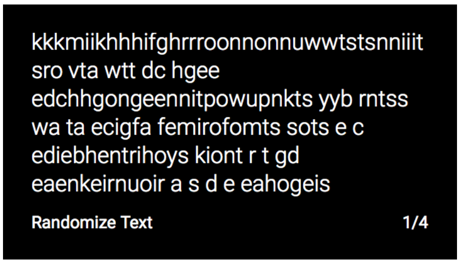
(c) The third slide.

Figure 5.2: Samsung Galaxy SII text.

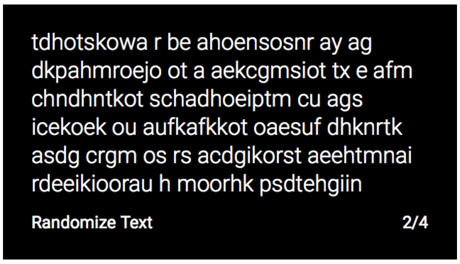
Table 5.1: Details on text length on Google Glass from Samsung Galaxy SII.

Figure	**Number of words**	**Number of characters**
Figure 5.2 (a)	30	229
Figure 5.2 (b)	35	229
Figure 5.2 (c)	13	92
Figure 5.2 (Total)	78	550

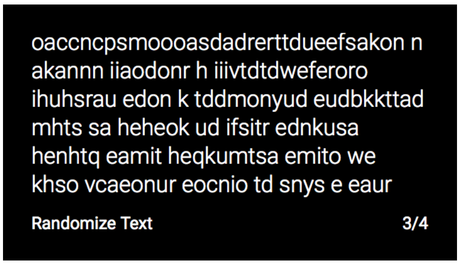
The result of using the text that filled the Samsung Galaxy SIII screen, seen in Figure 5.1 (b), in the Google Glass application can be seen in Figure 5.3. As seen the Samsung Galaxy SIII screen fit more characters than the Samsung Galaxy SII screen and as such three and a half cards in the Google Glass application was needed to display all the text that filled the Samsung Galaxy SIII screen. The exact number och characters and words on each card can be seen in Table 5.2.



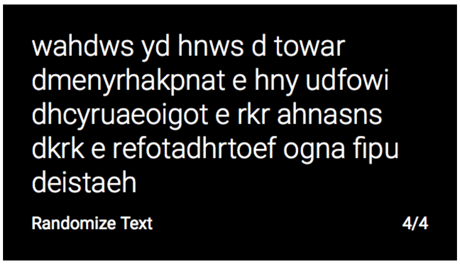
(a) The first slide.



(b) The second slide.



(c) The third slide.



(d) The fourth slide.

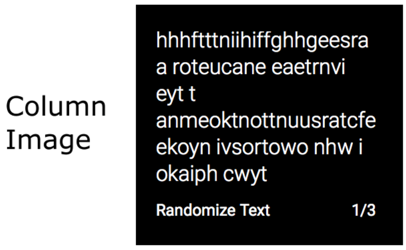
Figure 5.3: Samsung Galaxy SIII text.

Table 5.2: Details on text length on Google Glass from Samsung Galaxy SIII.

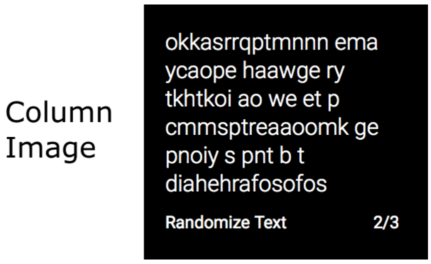
Figure	**Number of words**	**Number of characters**
Figure 5.3 (a)	26	199
Figure 5.3 (b)	32	213
Figure 5.3 (c)	29	217
Figure 5.3 (d)	19	121
Figure 5.3 (Total)	106	750

Instructions described in text may also be displayed in combination with an image. When text combined with an image is used as the design layout for a slide the maximum number of characters that can be displayed changes. The image, according to Google's predefined card layout design [46], make up 3/8 of the card, and as such the text make up the other 5/8. As such the maximum number of characters can be calculated using the maximum number of characters used when presenting only text.

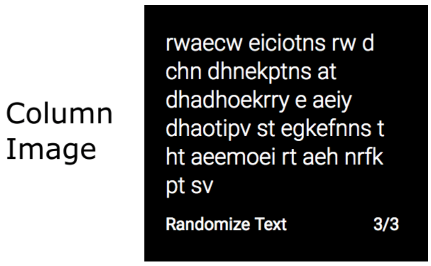
For Samsung Galaxy SII the maximum number of characters when using both text and an image was calculated as follows: $(5/8) * 550 = 343.75$. As three quarters of a character is of no use the maximum number of characters on the Samsung Galaxy SII screen is 343 characters. The result of using the same text as when displaying only text on Samsung Galaxy SII, only shorten down to 343 characters can be seen in Figure 5.4, with specific numbers in Table 5.3.



(a) The first slide.



(b) The second slide.



(c) The third slide.

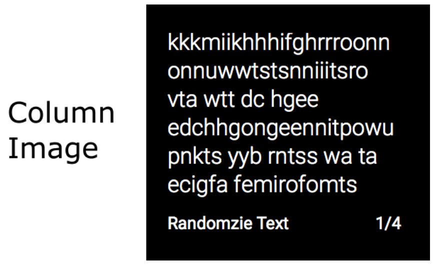
Figure 5.4: Samsung Galaxy SII text and image.

Table 5.3: Details on text length on Google Glass from Samsung Galaxy SII.

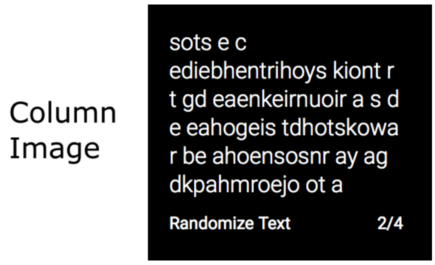
Figure	**Number of words**	**Number of characters**
Figure 5.4 (a)	12	118
Figure 5.4 (b)	18	111
Figure 5.4 (c)	21	114
Figure 5.4 (Total)	51	343

The maximum number of characters when using both text and and image on Samsung Galaxy SIII was calculated the same way as for Samsung Galaxy SII, as follows:

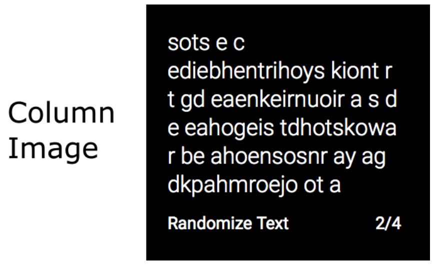
(5/8 = 468.75). As three quarters of a character is of no use the maximum number of characters when displaying both text and an image is as such 468 characters. The result of using the same text as in Figure 5.1, only shorten down to 468 characters, in the Google Glass application can be seen in Figure 5.5. The application requires about three and a half card to be able to display all of the text. The exact number of characters and words on each card can be fount in Table 5.4.



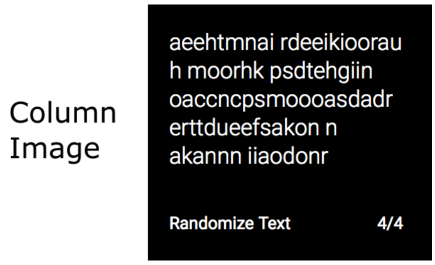
(a) The first slide.



(b) The second slide.



(c) The third slide.



(d) The fourth slide.

Figure 5.5: Samsung Galaxy SIII text and image.

Table 5.4: Details on text length on Google Glass from Samsung Galaxy SIII.

Figure	**Number of words**	**Number of characters**
Figure 5.5 (a)	12	127
Figure 5.5 (b)	23	124
Figure 5.5 (c)	18	115
Figure 5.5 (d)	9	102
Figure 5.5 (Total)	62	468

Overall a slide in the Google Glass application, which contains only text, can fit around

200 characters, which corresponds to somewhere between 25 to 30 words. When using both text and and image the number of characters that may fit on a slide in the Google Glass application is about 115, which corresponds to about 15 words. In total the Google Glass application requires 3 to 4 cards for every slide in the smartphone application, depending on the size of the smartphone screen, in order to display the same text as a full slide in the smartphone application.

5.2 Distance to the QR Code

In Table 5.5 the average time for registering and decoding a QR code while varying the distance to the QR code can be seen. The size of the QR code was optimised for a distance of two decimeters between the QR code and the device, which also shows in the results as all devices registered the QR code fastest at a distance of two decimeters. The QR code used encoded only one character, an 'a'. The results from every individual run can be seen in Appendix B

Table 5.5: Average time of registering a QR code with varying distance.

Distance (dm)	**Google Glass (sec)**	**Samsung Galaxy SII (sec)**	**Samsung Galaxy SIII (sec)**
1.0	1.919976807	1.965959850	1.839656134
2.0	1.831889852	1.649790392	1.487449920
3.0	2.227158610	1.921767210	1.568837591

5.3 Complexity of the QR Code

Table 5.6 shows the average time for each device to both register and decode a QR code of varying complexity. The distance to the QR codes where two decimeters, as the size of the QR code was optimised for a distance of two decimeters between the QR code and the device. As seen in Table 5.6 some of the results are infinite, meaning that the device was never able to register the QR code due to the high complexity of the QR code. Samsung Galaxy SIII was able to both register and decode all three of the QR codes, Samsung Galaxy SII was not able to decode the most complex QR code and Google Glass was only able to register the simplest QR code.

One explanation as to why Google Glass was not able to register any QR code other than the simplest one would be that the Google Glass camera did not auto focus as well as both Samsung Galaxy SII and Samsung Galaxy SIII. As such the complexity of the QR code made it difficult to register the QR code. The difference can be seen in Figure 4.1. Every single test run for each device can be found in Appendix B.

Table 5.6: Average time of registering a QR code with varying density.

Encoded Characters	**Google Glass (sec)**	**Samsung Galaxy SII (sec)**	**Samsung Galaxy SIII (sec)**
1	1.831889852	1.649790392	1.487449920
50	*Infinite*	2.096434673	1.913361687
100	*Infinite*	*Infinite*	1.949743816

5.4 Display Time

The average display time of each device, with varying information sizes, can be seen in Table 5.7. As the size of the information increased, so did the average display time. The increase in time came as a result of all information being downloaded at the same time, and is then handled by the device. Although only the currently visible information is sent to the display, there are still more information handled by the device and as such the device must search through more information in order to determine what information is to be displayed.

The display time is also affected by other background processes running at the same time. The average display time also only shows the time from when the information was sorted into classes until the information was sent to the display, and as such does not take in to account the potential delay the display might add in order to actually display the information to the user. Every single test run for each device can be found in Appendix B.

Table 5.7: Average display time for Google Glass with varying information size.

Information Size (Bytes)	**Google Glass (sec)**	**Samsung Galaxy SII (sec)**	**Samsung Galaxy SIII (sec)**
1 000	0.310802205	0.008374156	0.025498950
100 000	0.442440796	0.020444892	0.034109654
1 000 000	0.582170613	0.050938751	0.078170083

6 Conclusion

As seen in the test results the Google Glass application was almost always the slowest in all of the tests. However, as the test results only shows the times from when the QR code is being scanned until when the information is being displayed on screen the difference in time might not be of significance.

After the user has scanned a QR code and the information is being displayed, that is when the actually usage of the application starts. A user might be able to look past the fact that the Google Glass application is slower at scanning and presenting information as the advantages of Google Glass lies not in the speed of the device but rather in the user interface.

The fact that users does not have be use their hands is the major advantage of using Google Glass over a smartphone. Although voice commands are possible to implement in the smartphone application as well the user must still hold the smartphone or place the smartphone on a surface in order to see the information being displayed. Google Glass puts the display in front of the user and does not require the user to hod the display in any way.

However, the fact that Google Glass was not able to scan more complex QR codes is definitely a negative aspect. Although the product ID:s used in this test were not long enough as to were the complexity would become an issue, the complexity of the QR code will be an issue when the database contains many, many more products and the product ID:s are such that the complexity of the QR code is too high for Google Glass to handle.

Hopefully Google will replace the camera on Google Glass with a better one.

6.1 Personal User Experience

6.2 Future Work

6.2.1 Official approval of Voice Commands

The voice commands should be officially approved by Google.

6.2.2 Customised Voice Command

Construct own or use `https://github.com/RIVeR-Lab/google_glass_driver/blob/master/android/RobotManager/src/com/riverlab/robotmanager/voice_recognition/VoiceRecognitionT java`

6.2.3 TextResultProcessor

In the Google Glass application the class TextResultProcessor is not needed any more and should as such be removed. At this point the class in only used as middleware between an instance of the Products class and a list of CardPresenter. Instead the CardPresenter class should only be instanced once for each product, and keep the instance of the Products class.

The reason the TextResultProcessor class exists in the first place is due to how the Google Glass application was built originally, were all information presented was encoded directly in the QR code. At that point the TextResultProcessor was used when the encoded information was a text string. At this point the only information encoded in the QR codes are product ID:s.

The smartphone application already functions in this way, where the information stored in the instance of the Products class is used directly when a slide is created, instead of first being sorted through a middleware class.

[TODO possibly uml diagram of how the application works now and how it should work]

6.2.4 A General Fragment

The smartphone application should only have one general fragment instead of a bunch of different ones for different purposes. This should be done in order to be even more similar to the Google Glass application which uses the CardBuilder class, that is a general case that takes the layout as input.

6.3 Concluding Remarks

References

[1] Electronic Frontier Foundation (2004). *Statistical Distributions of English Text.* http://www.data-compression.com/english.html [2015-03-23].

[2] Ascii Table (2010). *Ascii Table.* http://www.asciitable.com.

[3] United States Census Bureau (2010). *Language Spoken At Home.* http://factfinder.census.gov/faces/tableservices/jsf/pages/productview.xhtml?pid=ACS_10_1YR_S1601&prodType=table [2015-03-06].

[4] QR Struff (2011). *What Size Should A Printed QR Code Be.* http://www.qrstuff.com/blog/2011/01/18/what-size-should-a-qr-code-be [2015-03-24].

[5] Wikipedia (2011). *QR Code Mask Patterns.* http://commons.wikimedia.org/wiki/File:QR_Code_Mask_Patterns.svg [2015-03-30].

[6] Google (2012). *Project Glass: Live Demo At Google I/O* [Online video]. http://youtu.be/D7TB8b2t3QE [2015-02-24].

[7] Google (2012). *Project Glass: One day...* [Online video]. http://youtu.be/9c6W4CCU9M4 [2015-02-16].

[8] Olsson, Isabelle (2012). *Google Glass Frames.* https://plus.google.com/110625673290805573805/posts/Nmc8LuwFw5M [2015-02-16].

[9] Parviz, Babak. Lee, Steve. Thrun, Sebastian. (2012). *Google Glass.* https://plus.google.com/u/0/wm/4/+GoogleGlass/posts/aKymsANgWBD [2015-02-16].

[10] Penny (2012). *Penny - Company.* http://www.penny.se/eng/company/company.html [2015-03-04].

[11] Bill Casselman (2013). *How to Read QR Symbols Without Your Mobile Telephone.* http://www.ams.org/samplings/feature-column/fc-2013-02 [2015-03-30].

[12] Christina Bonnington (2013). Smartphone screen sizes keep on growing—but not for much longer. http://www.wired.com/2013/04/why-big-smartphone-screens/ [2015-03-02].

[13] Dan Farber (2013). *GlassUp takes on Google Glass and Google legal.* http://www.cnet.com/news/glassup-takes-on-google-glass-and-google-legal/ [2015-03-02].

[14] Demo Wave (2013). *Information capacity and version of the QR Code.* http://www.qrcode.com/en/about/version.html [2015-03-02].

[15] Denso Wave (2013). *History of QR Code.* http://www.qrcode.com/en/history/ [2015-02-24].

[16] Denso Wave (2013). *Types of QR Code.* http://www.qrcode.com/en/codes/ [2015-02-25].

[17] Denso Wave (2013). *What is a QR Code?* http://www.qrcode.com/en/about/ [2015-02-25].

[18] Elise Ackerman (2013). *Could Google Glass Hurt Your Eyes? A Harvard Vision Scientist And Project Glass Advisor Responds.* http://www.forbes.com/sites/eliseackerman/2013/03/04/could-google-glass-hurt-your-eyes-a-harvard-vision-scientist-and-project-glass-advisor-responds/ [2015-02-23].

[19] Google (2013). *Google Glass How-to: Getting Started* [Online video]. http://youtu.be/4EvNxWhskf8 [2015-02-07].

[20] Jay Lee (2013). *I realize the with innovative products like Glass, the experience is more...* https://plus.google.com/+JayLee/posts/GwvagwVN6Hz [2015-03-25].

[21] Kevin Kwang (2013). *Samsung Galaxy S smartphones slaes hit 100M.* http://www.zdnet.com/article/samsung-galaxy-s-smartphones-sales-hit-100m/ [2015-03-25].

[22] Matt Welsh (2013). *Running a software team at Google.* http://matt-welsh.blogspot.com/2013/04/running-software-team-at-google.html [2015-03-03].

[23] Sergey Brin (2013). *Why Google Glass?* Long Beach, United States of America. http://www.ted.com/talks/sergey_brin_why_google_glass [2015-02-02].

[24] Andrew Fuller (2014). *Decoding small QR codes by hand.* http://blog.qartis.com/decoding-small-qr-codes-by-hand/ [2015-03-30].

[25] BarcodeEye (2014). *BarcodeEye.* https://github.com/BarcodeEye [2015-01-27].

[26] Ben Taylor (2014). *Why smartphone screens are getting bigger: Specs reveal a surprising story.* http://www.pcworld.com/article/2455169/why-smartphone-screens-are-getting-bigger-specs-reveal-a-surprising-story.html [2015-02-26].

[27] Fionna Agomuoh (2014). *Samsung Galaxy S5 Hits 10 Million Sales Mark: Smartphone Beats Galaxy S4 Initial Sales Record By Two Days.* http://www.ibtimes.com/samsung-galaxy-s5-hits-10-million-sales-mark-smartphone-beats-galaxy-s4-initial-sales-1582476 [2015-03-25].

[28] Google (2014). *Find a Preferred Provider.* http://www.google.com/glass/help/frames/providers/ [2015-02-02].

[29] Google (2014). *Frames.* http://www.google.com/glass/help/frames/ [2015-02-02].

[30] Google (2014). *MyGlass.* https://www.google.com/glass/help/myglass/ [2015-03-17].

[31] Google (2014). *Thanks to you, Glass just got even better.* https://plus.google.com/+GoogleGlass/posts/1tSYsPCCsGf [2015-03-25].

[32] Stack Overflow (2014). *content near "ok glass" gets grayed out.* http://stackoverflow.com/questions/25533765/content-near-ok-glass-gets-grayed-out [2015-04-29].

[33] Stack Overflow (2014). *Hide "ok glass" menu when enable contextual voice commands.* http://stackoverflow.com/questions/24733681/hide-ok-glass-menu-when-enable-contextual-voice-commands [2015-04-29].

[34] Augmented Reality (2015). *Encyclopædia Britannica Online.* academic.eb.com/EBchecked/topic/1196641/augmented-reality [2015-02-19].

[35] Eclipse (2015). Eclipse. https://eclipse.org [2015-04-17].

[36] Glass Almanac (2015). *The History Of Google Glass.* http://glassalmanac.com/history-google-glass/ [2015-03-25].

[37] GlassUp (2015). *GlassUp.* http://www.glassup.net [2015-02-23].

[38] GlassUp (2015). *GlassUp: Augmented Reality glasses that display messages from your smartphone.* https://www.indiegogo.com/projects/glassup-augmented-reality-glasses-that-display-messages-from-your-smartphone [2015-02-23].

[39] GlassUp (2015). *GlassUp Features.* http://www.glassup.net/features.html [2015-02-26].

[40] Google (2015). Card. https://developers.google.com/glass/develop/gdk/reference/com/google/android/glass/app/Card.html [2015-04-13].

[41] Google (2015). Migrating to android studio. http://developer.android.com/sdk/installing/migrate.html [2015-04-17].

[42] Google (2015). *Android Design Principles.* https://developer.android.com/design/get-started/principles.html [2015-02-16].

[43] Google (2015). *Android Studio Overview*. http://developer.android.com/tools/studio/index.html [2015-03-24].

[44] Google (2015). *AsyncTask*. http://developer.android.com/reference/android/os/AsyncTask.html [2015-05-02].

[45] Google (2015). *Card Design - Layout*. https://developers.google.com/glass/develop/gdk/card-design#layouts [2015-03-13].

[46] Google (2015). *Card Regions*. https://developers.google.com/glass/design/style [2015-02-16].

[47] Google (2015). *Glassware Flow Designer*. https://developers.google.com/glass/tools-downloads/glassware-flow-designer [2015-03-12].

[48] Google (2015). *Head Wake Up*. https://support.google.com/glass/answer/3079855 [2015-03-02].

[49] Google (2015). *Principles*. https://developers.google.com/glass/design/principles [2015-02-02].

[50] Google (2015). *Roboto*. http://www.google.com/fonts/specimen/Roboto [2015-03-16].

[51] Google (2015). *Slider*. https://developers.google.com/glass/develop/gdk/slider [2015-05-05].

[52] Google (2015). *Tech spechs*. https://support.google.com/glass/answer/3064128?hl=en [2015-02-24].

[53] Google (2015). *Voice Command Checklist*. https://developers.google.com/glass/distribute/voice-checklist [2015-03-16].

[54] Google (2015). *Voice input*. https://developers.google.com/glass/develop/gdk/voice [2015-03-02].

[55] Google (2015). *Voice Input*. https://developers.google.com/glass/develop/gdk/voice [2015–04-29].

[56] Google (2015). *VoiceTriggers.Command*. https://developers.google.com/glass/develop/gdk/reference/com/google/android/glass/app/VoiceTriggers.Command [2015-03-17].

[57] IGN (2015). *Microsoft's HoloLens Live Demonstration* [Online video]. http://youtu.be/b6sL_5Wgvrg [2015-03-04].

[58] Microsoft (2015). *Microsoft HoloLens.* http://www.microsoft.com/microsoft-hololens/en-us [2015-02-23].

[59] Microsoft (2015). *Microsoft Hololens Concept Video* [Online video]. http://compass.surface.com/assets/b2/15/b215c2a2-1c5b-4707-9a92-fffca6fb82fe.mp4?n=Microsoft_HoloLens_TransformYourWorld_816p23.mp4 [2015-02-23].

[60] Oculus (2015). *Low Latency 360° Head Tracking.* https://www.oculus.com/rift/ [2015-02-09].

[61] Pennsylvania State University (2015). *Confidence Intervals and the Central Limit Theorem.* https://onlinecourses.science.psu.edu/stat506/node/8 [2015-03-25].

[62] Penny (2015). *PDS Projecting Display System.* http://www.penny.se/eng/products/pds.html [2015-03-02].

[63] Penny (2015). *Penny.* http://www.penny.se/eng/index.html [2015-02-23].

[64] Penny (2015). *Penny Products.* http://www.penny.se/eng/products/c-moreig.html [2015-02-26].

[65] Recon Instruments (2015). *Recon Jet.* http://www.reconinstruments.com/products/jet/ [2015-02-26].

[66] Recon Instruments (2015). *Technical Specification.* http://www.reconinstruments.com/products/jet/tech-specs/ [2015-02-26].

[67] Virtual Reality (VR) (2015). *Encyclopædia Britannica Online.* http://academic.eb.com/EBchecked/topic/630181/virtual-reality-VR/ [2015-02-19].

[68] Wikipedia (2015). *Bone conduction.* http://en.wikipedia.org/wiki/Bone_conduction [2015-02-24].

[69] Wikipedia (2015). *Cuneiform.* http://en.wikipedia.org/wiki/Cuneiform [2015-02-23].

[70] Wikipedia (2015). *Google Glass.* http://en.wikipedia.org/wiki/Google_Glass [2015-04-07].

[71] Wikipedia (2015). *Head-Mounted Display.* http://en.wikipedia.org/wiki/Head-mounted_display [2015-02-23].

[72] Wikipedia (2015). *Head-Up Display.* http://en.wikipedia.org/wiki/Head-up_display [2015-02-23].

[73] Wikipedia (2015). *iPhone*. http://en.wikipedia.org/wiki/IPhone [2015-03-04].

[74] Wikipedia (2015). *Optical Head-Mounted Display*. http://en.wikipedia.org/wiki/Optical_head-mounted_display [2015-02-23].

[75] Wikipedia (2015). *QR Code*. http://en.wikipedia.org/wiki/QR_code [2015-02-24].

[76] Wikipedia (2015). *Samsung Galaxy SII*. http://en.wikipedia.org/wiki/Samsung_Galaxy_S_II [2015-03-25].

[77] Wikipedia (2015). *Samsung Galaxy SIII*. http://en.wikipedia.org/wiki/Samsung_Galaxy_S_III [2015-03-25].

[78] Wikipedia (2015). *Virtual Image*. http://en.wikipedia.org/wiki/Virtual_image [2015-02-27].

[79] Zxing Project (2015). *Zxing project*. https://github.com/zxing [2015-01-27].

[80] robomatics (2013). *How to Decode a QR Code by Hand* [Online video]. https://youtu.be/KA8hDldvfv0 [2015-03-30].

[81] Star Simspon (2013) Scott Torborg. *Google Glass Teardown*. http://www.catwig.com/google-glass-teardown/ [2015-04-07].

[82] Thad Starner. Project glass: An extension of the self. *Pervasive Computing, IEEE*, 12(2):14–16, 2013.

[83] Einar Selander Sture Allén. *[Information on Information]*. Studentlitteratur, Lund, 1985 (In Swedish).

[84] Laramee, Robert S. Ware, Colin. Rivalry and interference with a head-mounted display. *ACM Trans. Comput.-Hum. Interact.*, 9(3):238–251, sep 2002.

A Abbreviations

2D Two-Dimensional

3D Three-Dimensional

BCT Bone Conduction Transducer

GUI Graphical User Interface

HMD Head-Mounted Display

HUD Heads-Up Display

IDE Integrated Development Environment

OHMD Optical Head-Mounted Display

QR Code Quick Response Code

ZXing Zebra Crossing

B Results

Table B.1: Results of scanning QR code at different distances with Google Glass

Test Number	**1 decimeter**	**2 decimeters**	**3 decimeters**
1	1884216308	1798065186	2296325683
2	1682800293	1705902100	2415893555
3	2043151856	1937561035	2408782959
4	2487091065	1779327392	2346679688
5	1316070557	1948822022	2336975098
6	1631652832	1777801514	2341278076
7	1576843262	1941070556	2184875488
8	1772125244	2041961670	2347503662
9	2140167236	2060516358	2296203613
10	1911987304	1757171630	2170745849
11	1884277345	1767211914	2357238769
12	1929656983	2313415528	2337341308
13	1709838868	1731719971	2460479736
14	1819946288	1789154053	2208862304
15	1881225586	1959869384	2198150634
16	1790405272	1852996826	2317962646
17	1574829101	1649383545	1774505615
18	1825531006	1702728271	2231719971
19	1731201172	1674377441	2281311035
20	2475097657	1735839844	1698455811
21	2370330810	1776123047	2375762940
22	1872375489	1714141846	2186004639
23	2327819824	2099090575	2258178712
24	1753479003	1826843263	2193664551
25	1721496582	1849182128	2280761718
26	2481842039	1802490234	1814788818
27	1602935791	1912353515	1650085449
28	2181121826	1636383056	2349945068
29	1797210694	1716491699	2250000000
30	2422576904	1698699951	2444274902

Table B.2: Results of scanning QR code at different distances with Samsung Galaxy SII

Test Number	**1 decimeter**	**2 decimeters**	**3 decimeters**
1	2139868085	1486211750	1643240375
2	1467511210	1433458333	1645772458
3	1531975085	1544185626	2269734332
4	1999989252	1474127458	1652179501
5	2117759919	1437559418	1784042751
6	1590932000	1346899542	1959105167
7	2309159125	1542022917	2183700002
8	1722160542	1421151250	1430372917
9	1442118042	1537669377	2517021085
10	2225043835	1922686667	1584863210
11	2559335792	1634340084	1544045376
12	2087568125	1493966751	2455554542
13	2399355752	1654488126	2096796210
14	1835165960	1681548458	1444782750
15	2013470417	1577887875	2794100627
16	1947420919	1971720543	2164583833
17	1914936335	1497237917	1976264377
18	1964812292	1473154916	2001674627
19	1836791835	1332783794	2090852043
20	1291630877	1760558543	1331759376
21	1715746333	1808396667	1840536334
22	2281021168	1940200418	2039864458
23	2106378627	1551906084	2214860250
24	2048848293	2127196127	1541294919
25	1953401960	1389226667	1873071125
26	1901472292	1534745500	2040191418
27	2009275501	1439262916	1916855625
28	1843962793	2145346709	1627921959
29	2639261585	2315900375	1939550709
30	2082421543	2017870959	2048423918

Table B.3: Results of scanning QR code at different distances with Samsung Galaxy SIII

Test Number	**1 decimeter**	**2 decimeters**	**3 decimeters**
1	1749676834	1779497917	1513588627
2	1980956916	1389859375	1552651834
3	1784749169	1264649875	1513143459
4	1842951960	1513206585	1352977542
5	1754621419	1473651001	1307146084
6	1644033335	1392401627	1453716167
7	1730130585	1282980458	1582548250
8	1672027044	1688112459	1311798583
9	1737328751	1432261543	1462818377
10	2373175125	1317017667	1316425376
11	1848027542	1463129292	2047718001
12	1576518375	1676956292	1675308210
13	1642278708	1709122334	1916630376
14	1750943335	1423191250	2168636127
15	1792001168	1402313542	1774322918
16	1733112083	1544061083	1798746335
17	2075140127	1554050960	2053470334
18	1940637085	1406057500	1583177875
19	1879888500	1666031252	1327241125
20	1690253250	1350420626	1350043708
21	1694755291	1506406918	1429665709
22	1966512835	1445166958	1401702668
23	1741643376	1664807709	1242497667
24	2036178543	1692449501	1511735834
25	1694977000	1474073667	1383570126
26	1851916252	1655224458	1549562168
27	1653187293	1203773709	1560395085
28	1966861500	1297300542	1317753292
29	1885895208	1287375710	1785062417
30	2499305419	1667945793	1821073459

Table B.4: Results of scanning QR code of different densities with Google Glass

	1 Encoded Character	**50 Encoded Characters**	**100 Encoded Characters**
1	1798065186		
2	1705902100		
3	1937561035		
4	1779327392		
5	1948822022		
6	1777801514		
7	1941070556		
8	2041961670		
9	2060516358		
10	1757171630		
11	1767211914		
12	2313415528		
13	1731719971		
14	1789154053		
15	1959869384		
16	1852996826		
17	1649383545		
18	1702728271		
19	1674377441		
20	1735839844		
21	1776123047		
22	1714141846		
23	2099090575		
24	1826843263		
25	1849182128		
26	1802490234		
27	1912353515		
28	1636383056		
29	1716491699		
30	1698699951		

Table B.5: Results of scanning QR code of different densities with Samsung Galaxy SII

	1 Encoded Character	**50 Encoded Characters**	**100 Encoded Characters**
1	1486211750	2024130959	
2	1433458333	1969146751	
3	1544185626	2236472876	
4	1474127458	2076327083	
5	1437559418	2219363335	
6	1346899542	2231630084	
7	1542022917	2143448876	
8	1421151250	1874537335	
9	1537669377	1969802000	
10	1922686667	1959907251	
11	1634340084	1999703669	
12	1493966751	1916870585	
13	1654488126	2090464252	
14	1681548458	2093513208	
15	1577887875	2254044209	
16	1971720543	2120755668	
17	1497237917	2253129752	
18	1473154916	2184249042	
19	1332783794	2064304543	
20	1760558543	2360741002	
21	1808396667	2101858835	
22	1940200418	2167193710	
23	1551906084	2046369502	
24	2127196127	2310106833	
25	1389226667	2130918084	
26	1534745500	2046775085	
27	1439262916	1942044501	
28	2145346709	2240394250	
29	2315900375	1967590667	
30	2017870959	1897246252	

Table B.6: Results of scanning QR code of different densities with Samsung Galaxy SIII

	1 Encoded Character	**50 Encoded Characters**	**100 Encoded Characters**
1	1779497917	2032751544	1887171167
2	1389859375	1998325293	1819777001
3	1264649875	1955494667	1988476627
4	1513206585	1656547125	1916022668
5	1473651001	2189044751	1880171334
6	1392401627	1972979041	1727605668
7	1282980458	2132110417	2092458209
8	1688112459	1718851126	2213131667
9	1432261543	1866888252	1979007460
10	1317017667	1697766835	1803592752
11	1463129292	1723378125	2000521042
12	1676956292	2041532793	2135961500
13	1709122334	1988374500	2045468750
14	1423191250	1913362543	1959357169
15	1402313542	1738970334	1991219667
16	1544061083	1995461668	2075882709
17	1554050960	1822996709	1895777792
18	1406057500	2099073044	2025029959
19	1666031252	2030953417	2111278042
20	1350420626	2145817792	1914733292
21	1506406918	1953587583	1947725500
22	1445166958	1875210209	2060683710
23	1664807709	1959188459	1814745458
24	1692449501	2109525960	1782884626
25	1474073667	1846324500	1886035542
26	1655224458	1911180292	1872026709
27	1203773709	1646990960	2026801625
28	1297300542	1744290210	1683916542
29	1287375710	1667269792	1893676792
30	1667945793	1966602667	2061173500

Table B.7: Results of displaying information of different sizes on Google Glass

Test Number	**1 kB**	**100 kB**	**1 MB**
1	253570557	442565918	605865479
2	302947998	368041992	542236329
3	309509278	388549804	591369630
4	330413818	382720947	637756348
5	325225830	408355713	436096192
6	308532715	345550537	547424317
7	339141846	468597412	575408936
8	163146973	405181885	383300781
9	350677490	363525391	563415528
10	317047119	258819580	511657714
11	289825439	387298584	581054688
12	334625243	368591309	683319091
13	357788086	358184814	531799316
14	286682129	256286621	553283693
15	342498779	376190186	551635742
16	309417724	403717041	580993652
17	298553467	462371827	549407959
18	283508300	394836426	582153320
19	348236084	396270752	706787110
20	319580079	516387940	476898193
21	313873291	540405274	576416016
22	320098878	521606446	563262940
23	288299561	438934327	603668214
24	308593750	727264404	553710937
25	338897705	550872803	564880371
26	321868897	534332277	582336426
27	332733154	549041748	693267823
28	297607421	557159424	748901368
29	317657470	543640137	506378174
30	313507080	557922364	880432129

Table B.8: Results of displaying information of different sizes on Samsung Galaxy SII

Test Number	**1 kB**	**100 kB**	**1 MB**
1	6797250	14496709	64364083
2	6326709	43183958	100282458
3	6746292	24071374	18386459
4	7058875	31881374	80973374
5	11829542	28462958	84470625
6	10667375	13489125	109992750
7	8890167	12720376	18701084
8	21200959	28104084	11930708
9	5742833	14041624	37545749
10	7117500	34152708	37839583
11	6092042	13555542	32261416
12	7110333	14802667	26862500
13	11012625	14330666	32655250
14	9552500	29462791	38692001
15	8780000	40578625	73218541
16	10533500	11147917	11013791
17	5902250	13767874	10906625
18	10135791	12711084	137211250
19	8074709	13150458	29192460
20	8935250	13934084	102380958
21	6071583	14262083	29384416
22	6577125	15152625	31167750
23	6078875	23965041	32326793
24	5932333	29859749	38545333
25	6505000	28264250	35596375
26	8270792	19016750	79581583
27	7437583	13678375	88178667
28	10711625	13444250	69627500
29	5428708	19024708	26571250
30	9704542	14632958	38301208

Table B.9: Results of displaying information of different sizes on Samsung Galaxy SIII

Test Number	**1 kB**	**100 kB**	**1 MB**
1	45458667	39401666	87967500
2	19902667	31100959	84084292
3	41686916	26440126	74972542
4	20914917	29435958	90547042
5	23337708	47066459	73427167
6	22120000	52401791	37535458
7	22092541	32144417	78092125
8	26526917	35520667	23304625
9	21858542	47763958	69872875
10	21574041	29292875	70221041
11	26992500	36875791	118110042
12	22648708	37272000	113794584
13	20634042	26331666	69038042
14	23051125	31668000	102973167
15	24605791	27136583	72928792
16	34179333	29959792	66249875
17	24432041	29541250	81965833
18	22108541	26804375	67999167
19	22469167	31402334	79773458
20	20666375	28152458	112654583
21	21058209	32599208	69238500
22	31040333	25747000	66882792
23	22904250	33012250	67235334
24	25762126	37820708	94330208
25	29134500	33178042	81698416
26	23086958	39990250	74808583
27	26097626	36171749	73372666
28	29608500	44156125	86602001
29	25589501	38544333	72212833
30	23425959	26356833	83208958

C Code

D Project Specification (In Swedish)

Uppdragsbeskrivning

Google Glass

Version 1.0

Mats Persson

Distributionslista

Befattning	Bolag/enhet	Namn	Åtgärd	Info.
Student	KaU	Richard Hoorn		
Student	KaU	Johan Häger		
Konsult/handledare	Sogeti	Mats Persson		
Konsultchef	Sogeti	Åsa Maspers		
Säljare	Sogeti	Bengt Löwenhamn		

Innehållsförteckning

Ändringsförteckning

Version	Datum	Ändring
1.0	2014-10-15	Dokumentet skapats

1. Allmän beskrivning av uppdraget

1.1 Bakgrund

Sogeti Sverige AB (Sogeti) är ett IT-konsultbolag med bred verksamhet, stort fokus på kompetens och modern teknik.

Sogeti jobbar mycket med applikationer inom mobilitet och en av de nyaste teknikerna här är Google Glass.

Google Glass introducerar ett helt nytt sätt att tänka på angående hur man interagerar med applikationer och helt nya användningsområden för mobila applikationer.

2. Google Glass

Syftet med detta uppdrag är att tillverka en Proof of Concept på en applikation till Google Glass, applikationen ska kunna hämta/spara data från en web service. Applikationen ska hjälpa användare i en produktion genom att bidra med vilka delar som behövs för att kunna montera något och även en enklare instruktion över hur det ska se ut när det är klart.

Ett annat viktigt syfte med detta uppdrag är att ta reda på hur Google Glass är ur ett användarperspektiv, är det behagligt att använda en hel dag? Osv.
Även prestandamässigt är det intressant för att veta om Google Glass är nog stabila för att kunna rekommenderas till kunder på företag inom tillverkningsindustrin.
Och slutligen så är bra/dåliga sidor med denna nya teknik intressant. Finns det begräsningar? T.ex. hur länge räcker batteri m.m.

I uppdraget ingår en back-end och en front-end och nedan ser vi lite exempel på saker man kan fokusera på inom de olika delarna.

I back-end finns följande att fokusera på:

- Prestanda, hur kan man optimera en back-end som pratar med liknande appar för att skicka så lite data som möjligt så snabbt som möjligt?
- Eftersom vi jobbar med Microsofts plattform så är det intressant att veta om man ser tydliga plus/minus med att hosta tjänsten i Azure iställer för att man t.ex. hostar den själv på sitt egna nätverk.

Vi ser helst att back-end utvecklas med WebAPI eller WCF då Sogeti har stort fokus på Microsofts plattform.

I front-end som ska köras på Google glass ser vi följande man kan fokusera på:

- Användarvänlighet. Som när touchenheter kom så är detta också ett nytt unikt sätt att interagera på, hur görs detta på bästa sätt? Vad skiljer sig från applikationer till telefoner?
- Multitasking, hur mycket kan göra samtidigt och hur hanteras detta?
- Prestanda

2.1 Case

Följande case ska uppdragstagarna utgå ifrån.

Ett företag har en produktionslinje med ett antal stationer där man monterar ihop saker vid varje station. Delarna man använder för monteringen har en kod på sig som man kan scanna av. Problemet företaget har är att det är många variationer av saker som monteras på varje station och arbetarna byter ofta platser så de har svårt att hålla reda på vad som behövs för att montera och hur det ska monteras.

2.2 . Optioner

Följande är förslag på vidareutveckling av detta som uppdragstagarna själv får plocka från om tid finns.

2.2.1 Option 1 – Jämföra Google Glass med andra liknande produkter

Jämföra Google Glass med en annan liknande produkt genom att välja en mindre del av applikationen som man implementerar. Ett förslag på annan produkt är företaget Pennys motsvarighet till Google Glass.

2.2.2 Option 2 – Bildigenkänning

För att slippa att ha koder på varje produkt så skulle det vara bra om applikationen kan känna igen produkterna genom att man bara tittar på produkterna, detta dels för att se vilka möjligheter det finns för bildigenkänning på Google Glass och för att ta reda på vad som redan finns färdigt inom detta område och vad som krävs av företaget för att få bildigenkänning att fungera.

2.2.3 Option 3 – Känna av position

För att underlätta för arbetarna så skulle det vara bra om applikationen kunde känna igen vart i produktionslinjen man sitter och jobbar och då filtrera listan av produkter man har att välja mellan så det blir lättare att hitta produkter.

2.2.4 Option 4 - Röststyrning

Kunna styra applikationen med rösten och ta reda på vad det finns för begränsningar för detta. T.ex. hur tyst måste det vara runt användaren för att det ska fungera?

3. Genomförande/arbetssätt

3.1 Rutiner

Sogeti tillhandahåller arbetsplatser, datorer, hårdvara samt erforderliga utvecklingsverktyg.
Uppdragstagarna kommer att ha access till Sogetis nätverk och förväntas nyttja vår TFS-server för versionshantering.

3.2 Genomförande

Uppdragstagarna planerar själv genomförandet och Sogeti tillhandahåller stöttning både projektstyrningsmässigt och rent implementationstekniskt. Sogeti tillhandahåller all programvara och hårdvara som behövs.

Förslagsvis används SCRUM med en sprintlängd på 2-3 veckor som sätts upp där uppdragstagarna specificerar vad de tror att de hinner med i början av varje sprint och har en demo för en eller flera på Sogeti i slutet på varje sprint.

4. Stöd/kvalitetssäkring

4.1 Granskningar

Vid behov genomförs granskning som kan initieras av både handledare och uppdragstagare.

Lämpligen definieras några granskningspunkter vid planeringen av projektet.

4.2 Testarbete

Funktions-, system- och integrationstest görs av uppdragstagarna.

5. Leveranser

5.1 Dokumentation

Systemdokumentation görs av uppdragstagarna. Dokumenten lagras i projektarkiv hos Sogeti.

Lämpligen levereras en enkel användarinstruktion.

6. Konfigurationsstyrning

All programkod och tillhörande specifikationer och andra utvecklingsdokument ska versionshanteras med hjälp av Microsoft TFS.

7. Miljö

Utvecklingsverktyg väljs av uppdragstagarna tillsammans med handledare. Hårdvara som behövs för projektet tillhandahålls av Sogeti.

8. Uppföljning och Rapportering

8.1 Rapportering internt/externt

8.1.1 Statusrapportering

Rapportering av status och framskridande i utvecklingen beslutas i samråd vid projektuppstart.

8.1.2 Möten

Möten hålls vid behov. Vid uppstart läggs lämpligt antal avstämningsmöten in i projektplanen.

8.1.3 Slutrapportering

Arbetet presenteras för Sogeti i samband med lämpligt månadsmöte alternativt lunchmöte.