Arttu Niemelä

MOBILE AUGMENTED REALITY CLIENT FOR CITIZEN PARTICIPATION

Master’s Thesis
Degree Programme in Computer Science and Engineering
January 2018
ABSTRACT

The objective of this master’s thesis was to develop a mobile augmented reality (MAR) client application that enables citizen participation in design of services, products and city plans, for example. The application was developed to be a part of an existing citizen participation platform. The interest was on finding out what functions such an application should provide and what kind of user interface (UI) it should have.

The application was developed iteratively, by building prototypes in parallel with researching best practices from existing literature. Especially finding suitable tracking solutions for MAR applications were of concern, so that desired functionalities could be provided. The final prototype enabled users to explore urban environment through augmented reality (AR) and find location-based surveys. The surveys could present city plans in AR and ask the users’ opinions on the plans.

For evaluation, the application was tested with real users (n = 9) in authentic environment. Quantitative data was gathered with questionnaires to evaluate the usability of the application. In addition, semi-structured interviews were used to gather qualitative data.

The results give some indications that simple graphical user interface together with the AR provides an approachable way to control such an application. However, more interactive functionalities could be required to make such an application interesting for the users. In addition, much effort is needed to address tracking and content management, before large-scale MAR citizen participation can be fully realized.

Keywords: mobile augmented reality, citizen participation, living labs, user involvement

TIIVISTELMÄ

Tämän diplomityön tavoitteena oli kehittää mobiilin lisätyn todellisuuden (MAR, mobile augmented reality) sovellus, jolla kansalaiset voivat osallistua esimerkiksi tuotteiden, palveluiden ja kaupungin suunnitteluun. Sovellus kehitettiin osaksi jo olemassa olevaa käyttäjien osallistamisalustaa. Työllä pyrittiin selvittämään, mitä toiminnallisuuksia sovelluksen tulisi tarjota ja millainen käyttöliittymä sillä tulisi olla.


Sovellusta evaluotiin käyttäjätesteissä (n = 9) autenttisessa ympäristössä. Sovelluksen käyttävyyttä arvioitiin kvantitatiivisilla kyselyillä. Lisäksi kerättiin kvalitatiivista tietoa haastatteluilla.

Tulokset viittaavat suuntaan antavasti, että yksinkertainen graafinen käyttöliittymä yhdistettiynä lisättyyn todellisuuteen tarjoaa helposti lähestyttävän käyttöliittymän. Jotta sovellus kiinnostaisi käyttäjiä, tulisi sen samalla kuitenkin tarjota interaktiivisempia toiminnallisuuksia. Lisäksi, laajamittainen seuranta ja sisällön hallinta vaativat vielä paljon työtä, ennen kuin mobiilila lisättyä todellisuutta voidaan mielekkäästi hyödyntää käyttäjiänsä osallistamisessa.

Avainsanat: mobiili lisätty todellisuus, kansalaisten osallistaminen, living labs, käyttäjien osallistaminen
# TABLE OF CONTENTS

**ABSTRACT** .................................................................................................................. 8

1. INTRODUCTION ........................................................................................................... 8
   1.1. Research objectives and method ........................................................................... 8
   1.2. Overview of the thesis ......................................................................................... 8

2. MOBILE AUGMENTED REALITY ................................................................................. 9
   2.1. Definition and related concepts ........................................................................... 9
   2.2. Tracking ................................................................................................................ 10
      2.2.1. Tracking characteristics and requirements .................................................... 10
      2.2.2. Sensor-based techniques ............................................................................. 11
      2.2.3. Optical techniques ..................................................................................... 12
   2.3. Displaying ............................................................................................................ 12
      2.3.1. Visual display characteristics and requirements ......................................... 13
      2.3.2. Visual display types ................................................................................... 13
      2.3.3. Other sensory displays .............................................................................. 15
   2.4. Interaction ............................................................................................................ 15
      2.4.1. Interaction characteristics and requirements ............................................ 16
      2.4.2. Input modalities ......................................................................................... 16
   2.5. Networking and data management ...................................................................... 17

3. RELATED WORKS ....................................................................................................... 19
   3.1. Involving users in urban planning ...................................................................... 19
   3.2. CityScope ........................................................................................................... 19
   3.3. SmartSantander ................................................................................................. 20
   3.4. KioskAR ............................................................................................................ 21

4. DESIGN ....................................................................................................................... 22
   4.1. Context of design and development ................................................................ 22
   4.2. Purpose and requirements ............................................................................... 25
   4.3. User interface design ....................................................................................... 25
   4.4. System design ................................................................................................... 28

5. IMPLEMENTATION ................................................................................................... 30
   5.1. Implementation process ..................................................................................... 30
   5.2. Hardware documentation .................................................................................. 30
   5.3. Software frameworks ....................................................................................... 31
   5.4. Preceding iterations ......................................................................................... 31
      5.4.1. The first iteration ....................................................................................... 32
      5.4.2. The second iteration .................................................................................. 33
      5.4.3. The third iteration ..................................................................................... 33
   5.5. Final user interface and use cases .................................................................... 35
   5.6. Final software architecture .............................................................................. 40
      5.6.1. Application management .......................................................................... 41
      5.6.2. Geological position management .............................................................. 42
      5.6.3. Map management .................................................................................... 42
      5.6.4. User object ............................................................................................... 42
<table>
<thead>
<tr>
<th>Section</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.6.5.</td>
<td>Survey Bubble objects</td>
</tr>
<tr>
<td>5.6.6.</td>
<td>Target management</td>
</tr>
<tr>
<td>5.6.7.</td>
<td>Cameras and views</td>
</tr>
<tr>
<td>5.6.8.</td>
<td>Sensor based tracking</td>
</tr>
<tr>
<td>5.6.9.</td>
<td>Point cloud tracking</td>
</tr>
<tr>
<td>5.6.10.</td>
<td>Touch input</td>
</tr>
<tr>
<td>6.</td>
<td>EVALUATION</td>
</tr>
<tr>
<td>6.1.</td>
<td>Test users</td>
</tr>
<tr>
<td>6.2.</td>
<td>Test case</td>
</tr>
<tr>
<td>6.3.</td>
<td>Test setup and devices</td>
</tr>
<tr>
<td>6.4.</td>
<td>Test flow</td>
</tr>
<tr>
<td>6.5.</td>
<td>Data collection</td>
</tr>
<tr>
<td>7.</td>
<td>RESULTS</td>
</tr>
<tr>
<td>7.1.</td>
<td>Quantitative results: questionnaire score comparison</td>
</tr>
<tr>
<td>7.2.</td>
<td>Quantitative results: deviations in the scores</td>
</tr>
<tr>
<td>7.3.</td>
<td>Qualitative interviews</td>
</tr>
<tr>
<td>8.</td>
<td>DISCUSSION</td>
</tr>
<tr>
<td>8.1.</td>
<td>Limitations</td>
</tr>
<tr>
<td>8.2.</td>
<td>Findings</td>
</tr>
<tr>
<td>8.3.</td>
<td>Future visions</td>
</tr>
<tr>
<td>9.</td>
<td>SUMMARY</td>
</tr>
<tr>
<td>10.</td>
<td>REFERENCES</td>
</tr>
</tbody>
</table>
FOREWORD

Studying can sometimes feel like juggling with bowling ball-weight and-sized 28x28x28 Rubik’s cubes. Fortunately, memories grow sweeter over time, so I already feel happy having worked with an inspiring project, from which I personally learned a lot. I hope at least some of that managed to leak onto these pages too.

I wish to thank Dr. Timo Koskela (who was also the main supervisor for this thesis) and Prof. Timo Ojala for helping me to get to this point. I also wish to thank Dr. Matti Pouke and Paula Alavesa for acting as the other supervisors. I’m also thankful for all the help and support I got from people at the Center for Ubiquitous Computing-research unit, especially from OULLabs team and PATIO development team, which include Satu Väinämö, Lotta Haukipuro, Minna Pakanen, Sari Komulainen, Leena Arhippainen, Marta Cortez Orduna, Weiping Huang and Ciprian Florea.

I’m grateful for my parents, partly for encouraging education, but especially for the food and money.

Also, special thanks to the student counsellors for being there to sort out the spaghetti study schemes. And finally, special-special thanks to all my friends along the years, if any of you is actually reading this, tell me, and I’ll buy you a beer.

Oulu, 28.1.2018

Arttu
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D</td>
<td>Two-dimensional</td>
</tr>
<tr>
<td>3D</td>
<td>Three-dimensional</td>
</tr>
<tr>
<td>6DOF</td>
<td>Six-degrees-of-freedom</td>
</tr>
<tr>
<td>A-GPS</td>
<td>Assisted global positioning service</td>
</tr>
<tr>
<td>AR</td>
<td>Augmented reality</td>
</tr>
<tr>
<td>AV</td>
<td>Augmented virtuality</td>
</tr>
<tr>
<td>BLE</td>
<td>Bluetooth Low Energy</td>
</tr>
<tr>
<td>FOV</td>
<td>Field of view</td>
</tr>
<tr>
<td>GNSS</td>
<td>Global navigation satellite system</td>
</tr>
<tr>
<td>GPS</td>
<td>Global positioning service</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical user interface</td>
</tr>
<tr>
<td>HCI</td>
<td>Human-computer interaction</td>
</tr>
<tr>
<td>HMD</td>
<td>Head-mounted display</td>
</tr>
<tr>
<td>HRTF</td>
<td>Head-related transfer function</td>
</tr>
<tr>
<td>IMU</td>
<td>Inertial measurement unit</td>
</tr>
<tr>
<td>MAR</td>
<td>Mobile augmented reality</td>
</tr>
<tr>
<td>MR</td>
<td>Mixed reality</td>
</tr>
<tr>
<td>NFT</td>
<td>Natural feature tracking</td>
</tr>
<tr>
<td>ODG</td>
<td>Osterhout Design Group</td>
</tr>
<tr>
<td>OS</td>
<td>Operating system</td>
</tr>
<tr>
<td>POI</td>
<td>Point-of-interest</td>
</tr>
<tr>
<td>RR</td>
<td>Real reality</td>
</tr>
<tr>
<td>SDK</td>
<td>Software development kit</td>
</tr>
<tr>
<td>SLAM</td>
<td>Simultaneous localization and mapping</td>
</tr>
<tr>
<td>TUI</td>
<td>Tangible user interface</td>
</tr>
<tr>
<td>UI</td>
<td>User interface</td>
</tr>
<tr>
<td>UX</td>
<td>User experience</td>
</tr>
<tr>
<td>VC</td>
<td>Virtuality continuum</td>
</tr>
<tr>
<td>VR</td>
<td>Virtual reality</td>
</tr>
<tr>
<td>WIMP</td>
<td>Windows, icons, menus and pointers</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless local-area network</td>
</tr>
</tbody>
</table>
1. INTRODUCTION

Citizen participation is becoming crucial in efficient design and development of services, products and city plans that satisfy the end-users, the citizens. This makes it necessary to provide the right digital tools to facilitate such citizen involvement [1]. Augmented reality (AR) on the other hand is showing potential to become the championing interface for computing, directly linking digital information with physical world [2]. Thus, it is in question, if combining citizen participation with AR would produce useful and beautiful results.

This Master’s thesis is a story about developing a mobile augmented reality (MAR) client application, which helps to involve citizens in research and design of e.g. services, products and city plans. The application was developed to be a part of PATIO, which is an existing online community and platform for citizen participation [1] [3]. The motivation was that MAR could enhance the way citizens can participate, especially in city planning, as MAR can provide an intuitive link between digital plans and the physical city.

1.1. Research objectives and method

The work treaded with a constructive study approach. The high-level objective was to empirically find out the best practices to design and implement a MAR application that could involve citizens. Moreover, answers were sought for what functionalities such an application should provide. Specifically, its user interface (UI) and perceived user experience (UX) were of interest.

To find answers, a prototype client application was developed. The development was done with an iterative process. Many different prototypes were implemented, and background literature was explored in parallel, to achieve the best possible outcome for the final result. To evaluate the UI and the UX of the final prototype, user tests were conducted, where the participants tested the prototype. Data was gathered with two questionnaires: The Handheld Augmented Reality Usability Scale (HARUS) and the MEC-Spatial Presence Questionnaire (MEC-SPQ). In addition, semi-structured interviews were conducted to gather qualitative data.

Finding answers for even elementary questions is important, as clear guidelines how MAR applications should be designed do not exist yet. Even less so, there are no design rules on how a MAR client for citizen participation should function. Thus, this thesis may work as one reference, when developing such application in the future.

1.2. Overview of the thesis

This thesis begins by exploring the existing knowledge about MAR. Further, previous implementations that were deemed relevant to the topic, are studied. After establishing a theoretical basis, the design and implementation process of the work is described. Finally, evaluation and results are presented, and the outcomes of the project are discussed.
2. MOBILE AUGMENTED REALITY

Augmented Reality (AR) aims to present virtual information merged with physical reality, as if it was natural part of it. MAR aims to make this experience mobile and ubiquitously accessible.

There are three major technological challenges in developing AR: (1) tracking; (2) displaying; and (3) interaction. Tracking seeks solutions to how computers can understand physical reality and place the virtual information accordingly. Displaying is concerned with how to show the combination of virtual and real to users. Interaction is also tricky; in order to benefit from AR, novel input methods are required. In addition, networking and data management is a special issue regarding MAR.

2.1. Definition and related concepts

AR as a concept is perceived to be the merging of virtual information into the real world, or at least creating an illusion of such merge [4]. Virtual information is often perceived as computer generated graphics, which provides a visual experience. However, AR systems may provide other sensory experiences, like audio and haptic, too. Real world is thought as the physical environment the user experiences without the use of any devices and is sometimes referred to as the real reality (RR) [5]. It is also debatable, if there is a philosophical difference between real and virtual realities, but that is way out of the scope of this thesis.

As generally accepted in literature, AR is defined as technology that fulfills the following requirements [6]:

1. It combines real and virtual;
2. It is interactive in real time;
3. It is registered in 3D.

These three requirements also reflect the three key technical requirements of AR: displays, interaction and tracking. Displays are needed to present the combination of real and virtual to the user. Interaction requires novel input techniques to manipulate the virtual information. Tracking techniques are needed to measure the real world and align the virtual information with it [6].

The same definition and technical requirements apply for MAR. However, the requirement of mobility adds more demands in terms of scalability and performance [7]. In practice, MAR system should work on a large scale, globally at best, and execute in real time on mobile devices.

The above definition specifies the required technology. Still a broader context is useful to understand the potential of AR and MAR. In this regard, the virtuality continuum (VC) is often brought up. The continuum depicts a spectrum from completely real world to fully virtual reality (VR) (Figure 1). On this continuum, AR is closer to the real world, adding some virtual information, while keeping connected to it. VR on the other hand is fully virtual without any relation to the RR. Augmented virtuality (AV) is a less used term, but it refers to systems where part of the VR experience is still tied to some part of real world [8]. The whole spectrum is called mixed reality (MR). Sometimes term MR is used instead of AR, as it captures a broader view of mixing real and virtual in varying concentrations [2].
The big idea behind AR can be traced to the concept of “ultimate display”. Ultimate display is described as a “room within a computer can control the existence of matter” [9]. Thus, the ultimate objective of MAR could be thought as the ability to control matter through a computer ubiquitously, meaning outside the room too. Though this approaches the concept of dual reality, where sensors and actuators are used to measure and influence the real world [10].

2.2. Tracking

To align virtual content with the real world, AR systems must determine their spatial relation to the physical environment they operate in. This process is commonly referred to as tracking [4]. AR systems should at least track their orientation and position relative to the real world. More believable systems need to perceive the 3D structure of the environment too. In any case, tracking must work in real-time. In MAR, the already challenging nature of tracking is only pronounced.

Tracking approaches can be roughly categorized into sensor-based and optical approaches. Sensor-based approaches measure inertial, magnetic and radio waves to define the orientation or the location of the device. Optical approaches employ machine vision algorithms recognize features in the environment and thus define the pose of the device. Some vision-based methods can also model the environment at runtime. For note though, hybrids of different tracking techniques are usually required for accurate and scalable MAR systems [2].

2.2.1. Tracking characteristics and requirements

AR systems need to understand their surrounding environment to be able to “anchor” the virtual information to the real world. Overall, tracking is a fundamental technical challenge in AR. Still, even as a widely researched topic, it remains largely an unsolved problem [11]. Main challenge in MAR tracking is being scalable efficient in unknown, large-scale environments. Choice of tracking solution is critical, as in MAR it has a strong impact on the design of any application [12].

A user holding an AR device can move unpredictably in six-degrees-of-freedom (6DOF). Thus, AR system needs to precisely define the position (x, y and z-coordinates) and the rotation (yaw, pitch and roll) of the device in its reference frame. Together the translation and rotation are often referred to as the pose of the device.
When the pose is calculated continuously in real-time, e.g. 3D graphics can be moved accordingly on the screen, appearing to be part of the environment. For improved immersion, AR system should understand the 3D structure of the world around it. Otherwise, the virtual objects would not interact with the real world, for example collide with real objects, or be occluded by them. Additionally, by simply tracking the environment, AR system can only place virtual content in arbitrary places around the user. Also, the system would not remember the location after a reboot. To make AR contextual, the system must recognize its location and relocate itself in the environment.

In MAR, the tracking should work ubiquitously on a large scale, in real time. Large-scale environments are often uncontrolled, and obviously large. So, they are difficult to condition for tracking, for example by installing suitable infrastructure. MAR systems usually cannot have prior knowledge about such environments. Thus, they must rely on built-in tracking solutions. Also, outdoor environments can be very dynamic, objects in the environment move around and lighting and weather can range dramatically [14].

### 2.2.2. Sensor-based techniques

Sensor-based tracking techniques estimate the movement of the device using sensors like accelerometers, gyroscopes and magnetometers. These technologies are cheap and exist on most mobile devices. However, error correction is difficult, even impossible, resulting them to drift over time. Additionally, infrastructure like Global Navigation Satellite System (GNSS) and Wireless Local-Area Networks (WLAN) can be used to trilaterate the location of the device, even on global scale. Unfortunately, accuracy of such trilateration is usually very coarse [2].

A common way to calculate the rotation of an AR device is to measure its inertia [7]. Combination of sensors for this are called inertial measurement units (IMU). An IMU usually includes an accelerometer and a gyroscope. Accelerometers measure the acceleration of a device, while gyroscopes measure with rotation velocity. IMUs can be sampled very fast, resulting in smooth tracking of rotation. These sensors are also low-cost and commonly available on mobile devices. They also do not have range-limitations, as they do not depend on any external signal source. However, a major disadvantage of IMUs is that slight biases and noise accumulate as drift from the correct rotation, which can only be corrected with other tracking techniques [15].

Magnetometers are another common sensor on mobile devices. They are used to determine yaw of AR systems with Earth’s magnetic field as reference frame. The problem with magnetometers is that they are very susceptible to ambient electromagnetic fields, which cause large errors [7]. Magnetometers can also be used to define location. This is done by mapping the magnetic fingerprint of an area, but it generally works only in indoor environments [16].

Trilateration of the position with GNSS or base station radio waves, e.g. GPS (Global Positioning System), WLAN and Bluetooth, has been done since earliest MAR works. For example, in [17] GPS position accuracy to one meter was achieved with special equipment. However, on most mobile devices GPS accuracy is 10 to 15 meters. GNSS trilateration is still useful in MAR, as it can define the coarse position on global scale. With WLAN, the position can be accurate to 2.5 meters, but trilateration works mostly in indoor areas. Using Bluetooth beacons, accuracy can be even 0.1 meters, but
these beacons have very short range, making them costly for large-scale implementation [5]. Bluetooth signal is also easily obstructed, even by human bodies.

2.2.3. Optical techniques

Vision-based tracking means using optical sensors, mainly digital cameras, along with machine vision algorithms to track the pose of the camera. Earliest optical techniques track purposefully designed fiducial markers. Nowadays, state-of-the-art techniques can track natural features of the environment, thus they can work in unknown and unconditioned environments. Most advanced methods can also map these features modeling the environment in 3D. Vision-based methods can accurately determine pose in 6DOF, but they are still difficult to scale for large environments.

One of the oldest solutions for tracking is to track predefined fiducial markers that are placed into the environment. Fiducial markers are often 2D images representing simple shapes or binary code to distinguish them. These markers are also usually black-and-white, as it gives the best contrast for cameras to handle [2]. Fiducial marker tracking is very accurate and there are many ready-made solutions available. For MAR however, it is impractical to implement and maintain these markers in large environments. Additionally, fiducial markers suffer greatly from occlusion [7].

More advanced optical techniques track visually distinct features that are already present in the images from the environment. This approach is referred to as natural feature tracking (NFT) [2]. By keeping track of the features between successive images, the movement of the camera and therefore its pose can be determined. Important distinction between NFT implementations is that they can be either model-based or model-free. Model-based means predefining the trackable features in the environment. With predefined features, specific location can be recognized, and virtual content can be placed precisely where desired. However, defining these features for large areas is very laborious. Model-free means that tracking can work in arbitrary locations, without predefined features. A major problem however is that without predefined features, tracking system cannot recognize a certain location. This means that placement of virtual content can neither be predefined exactly. Also, if tracking is paused, it probably does not continue from where it was left. A solution is simultaneous localization and mapping (SLAM), where both methods are used. SLAM tries to both optically track the pose and map the environment at the same time. By mapping the environment, the system can recognize if it is a place where it has been before. Here, previously created maps can be used to recognize predefined environments. Major challenge is that for large areas the amount of mapping data grows very large. Both acquiring such mapping and maintaining it is labor-intensive. Still, SLAM is thought to be the go-to solution for accurate mobile tracking in the foreseeable future [2].

2.3. Displaying

AR systems require a way to display the virtual information mixed into reality. AR displaying is influenced by challenges between display technology and human perception, and it is important to acknowledge these issues. With visual display types, there are many to choose from, like hand-held and head-mounted displays (HMD), and the choice can significantly affect the application design and the UX. AR has focused
on visual perception, but for example audio or tactile feedback may greatly enhance the experience. There are even some works done on augmenting gustatory and olfactory senses.

2.3.1. Visual display characteristics and requirements

Creating a convincing illusion of AR has many challenges. They mostly spring from biological basis of human visual perception and the complexity of real world environments [18].

The displays stimulate users’ visual perception to create the illusion of AR. This results in many issues regarding the interplay between displays and human visual perception [19]. Fidelity of the image is one issue. Maximum field of view (FOV) of human vision is approximately 270 degrees and the closer the display is to the eyes, the higher the pixel density must be to produce a sharp image [19]. Another major issue is accommodation-vergence conflict. This occurs when eyes converge on seemingly distant virtual object on the screen, but the lenses accommodate at the typically constant depth (the distance to the screen) [18]. Individual biological differences are also a prominent issue, as they can require laborious calibrations to the display device [18]. Additionally, true 3D perception requires stereo displaying.

Environment brings another set of challenges [18], especially regarding MAR. The real-world environment is unknown and dynamic. Outdoor lighting can range from 100 to 130,000 lux. It is a challenge to devise screens that can work both in dim and very bright environments. Also, real-world environments have aspects like reflections and shadows, which are difficult to display virtually. It is also beneficial to light the virtual imagery according to the real-world lighting. Another issue is that real-world environments can have complex 3D structures. To occlude virtual objects by real world objects, the structures must be understood, which also poses a problem for tracking. All in all, very sophisticated display technology is needed to make the AR experience believable and comfortable.

2.3.2. Visual display types

Displays differ on approach to merging virtual graphics with real world. Most used approaches in MAR are optical see-through and video see-through [4]. Displays also come in different form factors, like hand-held or head-mounted displays. Display forms have their advantages and disadvantages when considering their mobility, UX and availability.

Most optical see-through displays use optical combiners to let real-world light through and guide virtual graphics from video display to user’s eyes. There is also some research on displays that are transparent themselves [4]. With optical see-through, users have direct view to real world. However, the FOV is limited to 50-degree on state-of-the-art displays [20]. Miniaturizing the optical combiner technology, while maintaining wide FOV and high resolution, is a great technical challenge [20].

Video see-through displays combine virtual graphics and video stream from real world, rendering the composition on common flat screens. Required technology (cameras and flat screens) are easily available, making this approach popular. With video see-through the view is completely digital, thus controlling and manipulating the
view is easier (e.g. lighting, occlusion or diminished reality). However, with indirect view to real world, usability is limited and even users’ safety is a concern [4].

With display form factors, there is a whole spectrum of different shapes and sizes. This spectrum illustrates how AR displays can range from direct retinal contact lenses to static spatial projectors [21]. For MAR however, the currently most applicable choices are hand-held and head-mounted displays, both having their advantages and disadvantages.

Hand-held displays practically mean mobile devices like smartphones and tablets. These devices are currently the best choice for mass market MAR applications because of their wide adoption and familiarity to the public [13]. Hand-held display devices commonly merge the real and virtual using video see-through (Figure 2). With hand-held screen, there is no need for stereoscopy, but the image is also restricted to the flat screen. Also, hand-held devices hinder interaction, as they occupy at least one hand of the user. This can even be a safety issue, when using hand-held AR in outdoor areas [22]. All the strengths of AR are not reached with hand-held displays, but they are currently the best solution for creating scalable consumer AR today, while waiting for more immersive displays.

HMDs are display devices that users wear on their heads. They often come in the form of large goggles [4] (Figure 3) and are also referred to as smart glasses. They provide a natural and hands-free view to AR. However, they are not yet widely available for consumers. HMDs also suffer from not being socially accepted, mainly because of having always-on cameras [23]. Still, they are considerable platform, when developing industrial MAR applications. Most current HMDs use optical see-through [24], although video see-through HMDs also exist. With optical see-through HMDs, the FOV is currently limited; the displayable area can be roughly emulated by holding an A4 paper at arm’s length. Another problem for HMDs seems to be that they hinder the cognitive capabilities [25], instead of enhancing them, as is the purpose of AR. In addition, the accommodation-vergence problem applies especially to HMDs.
2.3.3. Other sensory displays

Most research and development has focused on visual displays [11]. However, audio and haptic sensory systems are also considered important for AR [4]. There is also some work done on virtually stimulating olfactory and gustatory senses.

Augmenting the sounds of the real world should not be neglected. Using audio along with visual AR enhances the experience [4]. 3D directed sound can be achieved even with stereo speakers imitating cues that audial perception takes from direction of sound, e.g. the delay of input audio signal between ears, and modeling a person-specific Head-Related Transfer Function (HRTF) [26]. Vice versa, it could also be useful to diminish real-world sounds with noise-cancellation.

For stimulating haptic sense, meaning touch, there is technology like robotic mechanical gloves and vibro-tactile feedback [4]. Making visual augmentations tactile could be a very important aspect regarding interaction with them. There is also some development on bodysuits, which can stimulate full-body haptic feedback and even have climate control, for example the Teslasuit [27].

Augmenting the olfactory and gustatory senses is a little researched subject. Still, some implementations exist. One example is the Meta Cookie, which augmented a plain cookie with virtual toppings (visual), but also with different scents using a special olfactory headset [28]. Another work is the Digital Sour Lollipop, which used electrical and thermal stimulation of tongue to produce sensations of taste. Augmented tastes though were limited to sour, bitter and salty [29].

2.4. Interaction

Interaction is where the potential of AR truly lies: in bridging the gaps between digital information, the user, and the physical world. While tracking and displaying are characterized mainly by technological challenges, interaction brings the focus closer to human-computer interaction (HCI) issues. Messy combination of factors should be considered to develop AR interaction methods that bring value over the classical methods, like desktop interfaces. Also, to interact with a world, where digital and physical are blended together, new input modalities are required for a fluid experience.
2.4.1. Interaction characteristics and requirements

Interaction in AR has the potential to directly link the digital and physical worlds. It can provide a natural user interface (UI) to digital information, as it is presented as part of the real world [2]. In MAR this would be especially powerful, as computing is becoming ubiquitous and the whole world may work as an interface. Though here it is arguable, if the digitalized physical world is the interface, and AR is merely interaction medium [30].

To be useful, AR must bring more value than the conventional interaction metaphor WIMP (windows, menus, icons, pointing). AR should support WIMP methods, but it must also enable 6DOF interaction, like direct spatial manipulation [13]. This requires new interface metaphors and techniques specific for AR. Currently everything is borrowed from other fields, like VR, although research in VR interaction also support AR [4].

It is not yet known, what such metaphors should be, but many important aspects to consider have been identified. AR interaction design must address three major components: (1) the physical world; (2) the virtual information; and (3) the metaphors that links them together [4]. These components bring many possibilities and raise many questions. For example, the physical and virtual worlds may both be interfaces to a common application and interact with each other. On the other hand, the digital information could also be stand-alone, without a link to the physical environment. Also, for example in WIMP a collection of data is represented as a folder. What should be the corresponding metaphor in AR? Another thing to consider is the nature of the augmentations. The digital information can either abide the laws of physical reality, which can make the interaction very intuitive, or they can have their own laws, which can possibly provide a better UX [19]. There are also many options for placement of the digital information. They can be simply placed in free space, floating around the user. However, they also may be attached to the physical environment, or the user itself [2]. Placing augmentations on the user, for example on hands or the torso, can be very powerful, as humans have very fine control and sense over their body parts, known as proprioception [31]. Yet another consideration is that AR can be either collaborative experience, or presented only to individuals [4]. Finally, it should be carefully considered, when it is appropriate to use AR and when it is an unnecessary complication. In other words: “Use AR where it makes sense” (S. Julier, personal communication, 13.6.2017).

All in all, there is much left to explore before MAR can realize its potential in interaction. In addition, current MAR applications do not fulfill the user expectations, which are reportedly “empowerment, surprise, awareness, liveliness, playfulness, tangibility, collectivity, inspiration and creativity”, instead the applications provide much narrower experiences [30]. Another issue is that interaction design is currently very dependent on the accuracy of tracking and the level of immersion from displaying [30].

2.4.2. Input modalities

Many input modalities have been explored for AR ranging from conventional graphical UIs (GUI) to multimodal tracking of body parts and speech recognition.
A common and basic interaction method is to use conventional GUIs to interact with AR information. This approach is familiar to users, but there is no direct manipulation of the digital information, so the potential of AR is not realized [4].

A very notable approach is the tangible UI (TUI) [2]. TUI means instrumenting physical objects into interaction devices. For example, graspable objects may be used for manipulating digital objects, or digital interfaces can be displayed on physical surfaces. However, implementing TUI's requires tracking of such physical objects and surfaces [32]. If successful, TUI realizes the potential of AR as it is a natural and convenient way to influence the AR experience [2]. Also, closely related way is to use specific input devices that have built-in tracking, such as the Nintendo Wiimote.

Using body parts is also a popular input method. The whole human body can work as an input device, when tracked appropriately. Most popular however is to track the hands. Using hands enables fine-grained and natural manipulation, but precise hand-tracking is very challenging, even with specialized sensors [33]. Related to hand-tracking, gestures can be used to control AR applications. Gesture recognition has been demonstrated on mobile devices [34], but gesture-based interfaces are harder to learn [2]. Gesture-control is also less direct than full hand-tracking. In addition, gaze tracking is another researched way for interaction. It however requires additional sensors to track the eyes and filtering of eye-movements that are not meant as interaction [13].

Speech recognition is used in AR too. Robust speech recognition systems have been demonstrated to run on mobile devices [35]. Speech input shines when used as quantitative input, while body-tracking is better for qualitative manipulation [4]. General audio input can also be used in more subtle manner, for example interacting by whistling [36].

### 2.5. Networking and data management

MAR systems must handle large amount of data that is not plausible to be stored on mobile devices. And, truly large-scale MAR applications may have to outsource the demanding calculations, for which advanced network infrastructure is required.

Large-scale MAR applications require large quantities of data. This consists of the digital content for augmentations (e.g. 3D models) and the data for model-based tracking. Applications may also depend on other location-based information. This creates a three-fold problem: acquiring, organizing and storing of all that data [7]. Firstly, the data needs to exist. There are services, e.g. Google Poly, from where community-made 3D models can be acquired. But if applications desire custom content, creating it is laborious. Also, large-scale tracking requires modeling the complete operation environment and this is currently done manually. Secondly, the data needs to be continuously organized. The 3D models must be connected to the tracking data, so that they can be placed in desired positions. And, large-scale tracking data must be organized, because searching through a complete model database becomes implausible on a large scale. Another problem is to maintain the model data, as the physical world changes. Thirdly, all this naturally needs storage space. For city-wide AR systems, the tracking model databases alone would be very large [2].

Offloading the calculations is a proposed solution, especially for the large-scale tracking problems. This means creating a client-server model, where a mobile client acquires image data, which is sent to a server that determines the pose, which is sent back to the client. In [37], the maximum latency requirement for such interactive
applications was identified to be only few milliseconds. In the same paper, a proposed solution was to develop “cloudlet”-infrastructure, where many small servers are distributed on a large-area, so that physical distance is always short. In addition, for real-time systems, the required bandwidth is roughly estimated to be around 9 to 12 Gb/s [5]. Regarding the problem of searching through a large-scale tracking database, one proposed solution is to organize the model database by location and compass-direction, so the search is narrowed to close-by models [38]. To update the database for changes in the environment, a possible solution could be to develop a client-server architecture, where the server continuously updates a SLAM map from data obtained by the client [2]. This idea might also be expanded to solve the problem of acquiring large-scale tracking models by crowdsourcing it to clients.
3. RELATED WORKS

In general, large-scale MAR applications are not a new or uncommon idea. Pioneering work was done in 1999 [39], for example. However, these types of work explore the technical requirements, which were considered in Chapter 2, but the main interest in this Chapter is in works that explore implementing MAR in the context of user involvement in research or decision making. Alas, these types of applications proved hard to find.

In any case, four earlier works were picked for discussion. These were a paper on a prototype application titled as “Smart-phone Augmented Reality for Public Participation in Urban Planning”, CityScope, SmartSantander and KioskAR.

3.1. Involving users in urban planning

One comparable work is presented in [40]. The work resulted in a smartphone application, which displays architectural designs over existing architecture in AR. The application was developed to involve users in urban planning. It has a GUI with 7-step smiley face scale to rate the displays designs, and the ratings were to be made available for stakeholders in urban planning. The authors stated that using MAR in presenting architectural designs for experts is common, but not much research is done with regards to public participation. Authors’ hypotheses were:

1. Use of MAR in public urban planning increases public willingness to participate in the planning process;
2. Users of the system are satisfied with the level of perceived participation in the planning process.

The developed MAR application was evaluated with user tests, gathering data mainly with a quantitative questionnaire. The questionnaire was developed by the researchers, as no suitable questionnaire for measuring people’s willingness to participate and level of involvement was found. Qualitative interviews were used to complement the questionnaires. Results indicated that the application increased younger people’s willingness to participate in urban planning, while for older people the willingness stayed at the same levels. Also, using the application was easier for younger people. The application was generally received as useful for displaying architectural designs, however, the users questioned if the ratings would truly be considered by the stakeholders. Users also expressed wishes for extra information about the presented designs. The study was reportedly unable to evaluate the level of perceived participation. For future improvements, the authors painted ideas of displaying extra information, allowing the users to create their own designs, or modify the existing ones, and adding web functionality for a sense of community. The sense of community was hypothesized to increase the perceived participation.

3.2. CityScope

Another interesting work is presented in [41]. The paper discusses CityScope, a TUI platform adaptable to different use cases, in handling the 2015 refugee crisis in Hamburg, Germany. CityScope’s basic idea is to display geographic information on a
table, which both experts and the public can use to tackle problems. The system is described to have tangible objects for interaction, representing for example buildings and massing elements. AR is described to be an element in displaying, but other methods are used too, e.g. conventional displays and VR. In the context of the refugee crisis, the system was used to enable both experts and the public to find suitable locations to accommodate the refugees.

Authors assessed the strengths, weaknesses, opportunities and threats regarding the work. Strengths were that authorities received high quality information based on citizens’ local knowledge, while the citizens felt they were actively participating in the process, which also built up acceptance towards refugee accommodation. Weaknesses were that due time constrains setting up the system was challenging and pre-processed urban data easily approachable for non-experts was not available. The system was thought to provide plethora of opportunities due to how successfully it was adapted to a problem it was not designed for. Discussed threats were that selection bias of participants, lack of transparency, manipulated data and political interests may render the system misleading and invalid. All in all, CityScope may not qualify exactly as a MAR application, but it adds a lot to the discussion about designing such systems for user involvement. In addition, CityScope is used in Andorra Living Lab project to present geographical information in AR [42]. Unfortunately, no published papers about this were found.

3.3. SmartSantander

Paper [43] describes a MAR application that was developed as part of a project called SmartSantader. The aim of the project was to develop a smart city platform in Santander, Spain, which would ultimately engage citizens in city development, thus improving their lives. A large part of the project focused on a sensing network that provided various data for the platform, so the MAR application was not the only focus of the work. The authors also emphasized the need for infrastructure in creating such a platform.

The MAR component was aimed for visualizing diverse data provided by the platform. The data included points-of-interest (POI) and real-time information on e.g. traffic, cameras and forecasts. The data was presented location- and context-sensitively, but users were also enabled to choose their preferences on what they wanted to explore. The MAR application was complemented by tags placed in around the city, which the system could recognize and thus provide contextual information.

The sensing network consisted of various sensors around the city. In addition, users’ smart phones were harnessed to gather data, e.g. on geographic location, temperature and humidity. The users themselves were also able to provide data to the network, for example, by taking picture of a hole they had found in the pavement. A local newspaper was also linked to the system to deliver news information.

The authors emphasized the need for networking infrastructure. One main server was built to combine all other services in the platform and to work as an application programming interface for the other entities to carry out their developments. MAR component added two servers, one for providing the AR content and another one to register all the AR nodes in the network. Yet another server was set up to register the nodes in the sensing network. One more server provided information on social-related information, such as nearby events. Additional interfaces were also developed to facilitate management and development of new services by third-parties.
In conclusion, the authors saw the platform as promising, involving authorities, researchers and individuals and providing useful data in various ways. However, they raised concerns about the privacy of the users and the threat that technologically less experienced citizens could be neglected due to such development.

3.4. KioskAR

KioskAR is a prototype for an AR game that is meant for art students to compete and collaborate with each other [44]. Users of the game establish AR kiosks in physical locations, where they can present artworks, e.g. 3D models, videos or photos. Other users’ kiosks can be visited by moving to the according physical location. Kiosks can be interacted with by leaving comments on them. The users compete for points that can be used to enhance a kiosk. The points are received for adding artworks to one’s own kiosk, but also for visiting and interacting with other users’ kiosks. Thus, the game rewards socializing.

UI of KioskAR includes a map view, an AR view. The map view displays the environment and the kiosks on a 2D map to give a big picture of the game area. Tapping a kiosk on the map view gives a description of it. The AR view presents the kiosks in 3D and it is the main interface for interacting with them. In the AR view, the tracking is achieved combining GNSS positioning and IMU-based orientation. In addition, a spot view is a complimentary mode to the map view, presenting available spots for kiosks on the map. There is also a kiosk view, which is used to browse available kiosk templates, when establishing a new kiosk. Finally, a configuration view allows viewing and editing users’ profile information.

The authors evaluated the usability and sense of presence regarding the application. They emphasized the importance of finding suitable evaluation methods, stating that UX in AR has not been studied rigorously. Chosen method was to use HARUS [45] to evaluate the usability and MEC-SPQ [46] to measure sense of presence. Questionnaires were answered on 5-point Likert scales. Results from 48 participants were used in analysis.

The usability was rated high on average. However, inputting information and controlling the application in general was rated to be lower in comparison to the other scores. The authors argue that this is inherent in all handheld AR applications, as the device needs to be held in the other hand. Proposed solutions for this were to design the AR view in such way that it is not required to tap on the screen for interaction or just use HMDs. Regarding the presence, it was also rated high in general. One exception was that the application only stimulated the visual sense, thus the score for this was comparatively very low. In addition, the authors stated that KioskAR did not adequately support social interaction, if the users did not know each other beforehand. Typing text in the AR view was also observed as difficult.

Although KioskAR was not a large-scale MAR application, it provided a solid baseline to compare the work in this thesis to.
4. DESIGN

AR PATIO is part of a much larger context of PATIO, which is an already publicly available platform for involving users in the design of new products and services. The design is thus affected by the already existing functionalities. This context also includes other new major parts of PATIO, which were developed in tandem with this thesis project. Also, design for AR version of PATIO was discussed by the PATIO team before this thesis project began. Thus, there were prior designs for AR PATIO’s UI and use cases, which influenced the outcome.

4.1. Context of design and development

To understand the design of AR PATIO and the decisions made while developing it, it is worthwhile to consider the broad concept of PATIO [3] and its software components as a context.

As an overview, PATIO is an internet community, which both people and organizations can join [1] [47]. People join as test users for new products and services. Organizations and companies join to display a new product or service and to recruit test users to evaluate their products or services. As such, PATIO’s mission is to provide an innovation platform. Through PATIO, test users can participate in development of new services or products and express their opinions about them. Companies, research centers and organizations get to evaluate their new designs. For each evaluable product or service, an “activity” is created. Activity is an entity, which encloses information related to the product or service at issue. An activity also provides participation and evaluation tools, like surveys and forums.

PATIO software environment comprises of a web implementation, mobile application and more experimental VR and AR implementations. There is also an Admin Tool component, which connects the four implementations and provides the administrators an interface to handle the content of PATIO. Out of these components, only the web implementation was publicly available at the beginning of the development. Other parts were developed in parallel with AR PATIO.

The web implementation, referred to as Classic PATIO, is the oldest form of PATIO. Figure 4 shows the front page of Classic PATIO website. Through the website, the test users can login and access all the services currently provided. Users first see the currently available activities. These activities are created for the customer organizations and their contents are customized to whatever the evaluable case is. Opening an activity gives a more detailed description of the activity. After joining an activity, users can view and participate in all the content of it. Common contents include discussion forums and surveys. Through the activities, there can often be some organized event, where the subject of an activity is accessed. For example, if the subject is some product, the users might get a chance to test the product.
Figure 4. The activity page of the PATIO website, showing a list of activities.

Mobile PATIO is a smartphone application with the purpose of providing most or all the Classic PATIO’s functionalities in a convenient mobile application. As such, it displays the same content, activities, forums, surveys etc. Figure 5 shows two screenshots from the mobile application. Mobile PATIO was not published at the time, when the development of VR and AR PATIOs began, but it was further in development.

Figure 5. Two screenshots from Mobile PATIO. On the left: overview of an activity. On the right: a discussion on the activity forum.

VR PATIO is an experimental project to enhance PATIO services with VR. It provides a virtual model of the city of Oulu, where users can engage in activities. The activities are related to specific real-world locations. With virtual content, use case can be, for instance, to present urban development plans. Figure 6 shows a screenshot from the application. VR PATIO can be thought as a sibling of AR PATIO. They may have common content, which the users can explore either in the virtual city or in the real world through AR. VR PATIO’s development began approximately at same time as AR PATIO’s.
Figure 6. Screenshot of VR PATIO use case on assessing opinions on moving a statue in the city of Oulu.

The Admin Tool is intended for the PATIO administrators. It provides an interface to create content, e.g. the surveys. The surveys are placed at desired locations using the same virtual city model as in VR PATIO. Figure 7 shows a screenshot of the Admin Tool. Under the hood, the Admin Tool’s purpose is also to tie together the different implementations, providing a common interface to access content and user data. The admin tool was developed in tandem with VR PATIO, but it did not readily provide a common interface that AR PATIO could use.

Figure 7. Screenshot of the Admin Tool, which is intended for the administrators to create content, e.g. surveys. Admin Tool was under development when finishing this thesis.
4.2. Purpose and requirements

At the beginning of the project, the overall purpose of creating an AR version of PATIO was discussed with the development team of PATIO. Also, certain requirements for UX and implementation were included in the initial designs made by the team before this thesis project began.

The forth bringing idea for AR PATIO was that it could make users more willing to participate in location-based surveys and thus provide value for research. The AR PATIO was not to replace the Classic PATIO by including all the current functionalities of it. Instead, it was to complement the PATIO service with AR content.

From the UX point of view, users should be able to explore the city through AR and find surveys at specific locations. Surveys may simply present location-based questions, but some of them may also present related 3D content in AR. When a user is close to a nearby survey, but is not using the application, a push notification should be shown to make the user aware of the survey.

On technical side, the implementation should work reliably on a large scale, meaning that surveys can be placed anywhere in the city of Oulu. However, focus is on outdoor areas, so surveys need not to be placed indoors. The application should also be usable by the public. Therefore, it should be deployable on currently available mobile devices, meaning smart phones. Even though the application was designed to be practically deployable on the mass market, the project was decided to be research oriented. Thus, final goal was not to develop an application that satisfies commercial requirements.

4.3. User interface design

Designs for the UI of AR PATIO were done by a designer before this thesis work began. The UI and the application flow are presented in Figures 8 to 12. First, the user receives a notification about a nearby survey (Figure 8). Second, the user has opened the application and is guided with arrows to find the survey (Figure 9). Then, as the user has found the survey, the application asks, if the user wants to initiate the survey (Figure 10). In this case, the survey is about relocating the famous Toripolliisi statue in the city of Oulu. User starts the survey by swiping ‘Yes’. Then the user is guided to the relocation site by following a trail presented in AR (Figure 11). Finally, at the new location, the statue is shown in AR and user is asked for opinion on the proposed location (Figure 12).
Figure 8. User receives a notification about a nearby survey. Image © Minna Pakanen.

Figure 9. User opens the application and is guided to find the survey. Image © Minna Pakanen.
Figure 10. User opens the survey. The survey is about moving a famous statue to a proposed new location in the city of Oulu. Image © Minna Pakanen.

Figure 11. User is guided to the proposed location for the statue. Image © Minna Pakanen.
In the beginning, there was no overall design of the system architecture. After some time into the development, the planned system design formed to the one shown in Figure 13. The different implementations, AR, VR, Mobile and Classic PATIOs would be individual components that communicate and are administrated through the common admin tool interface. They would mainly communicate the user and survey data. External survey tool component is also included, as it would be used to form the survey through the Admin Tool.

Looking more deeply at the AR PATIO’s system design, it was to use GPS and sensor-based tracking to overlay survey and 3D models on the city in the correct coordinates. The system would also need a server where the 3D models can be downloaded at runtime. Figure 13 also shows a connection to a map service. The addition of map was done later in the implementation phase and the system designed in the figure was formed at that time. It was also considered, if AR PATIO implementation should be embedded into the Mobile PATIO. This was thought to be beneficial from the UX point of view. Decision was they would be developed as separate applications, as goal of AR PATIO application was to be research oriented and this would allow more freedom for experimentation.
Figure 13. Architecture design of PATIO. Image © Weiping Huang.
5. IMPLEMENTATION

Implementation of AR PATIO was an iterative process of trial and error, but ultimately a suitable solution was found. The process explored many different tracking solutions ranging from a simple GPS and sensor-based tracking to sophisticated optical tracking systems.

The resulting application fulfilled most of the design requirements, being able to present surveys to the user on a large scale and show related 3D content in AR. The UI did evolve from the initial design. For example, a major addition was a map view to give the user a better understanding of what content is nearby.

The final implementation was done with a hybrid solution for tracking, using GPS and sensors to place the surveys and optical tracking to place the 3D content precisely in predefined places.

5.1. Implementation process

Implementation was an iterative process. The process consisted of four major iterations. These iterations produced prototypes, where some technical obstacle rendered the design unsatisfactory. The greatest difficulty was finding a suitable tracking solution for the use case. After each prototype a new plan was made culminating in the fourth and the final implementation.

All in all, the implementation from the beginning to the first user test took about 8 months. Implementing the final prototype took the last 3 months, so most of the time went to exploring different tracking solutions and redesigning the application.

5.2. Hardware documentation

The application was developed on a Samsung Galaxy S6 smartphone. Table 1 presents the technical specifications of this device.

Usefulness of running the application on smart glasses was also discussed during the development. The later prototypes were built for ODG R-7 smart glasses as they also run Android operating system (OS). However, to make the application usable, significant changes to UI would have been required and idea was not pursued further.

<table>
<thead>
<tr>
<th>Display</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>71.5 cm²</td>
</tr>
<tr>
<td>Resolution</td>
<td>1440 x 2570 pixels, 16:9 ratio, (~577 PPI density)</td>
</tr>
<tr>
<td>Type</td>
<td>Super AMOLED capacitive touchscreen, 16M colors</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Platform</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>OS</td>
<td>Upgraded to Android 8.0 (Oreo)</td>
</tr>
<tr>
<td>Chipset</td>
<td>Exynos 7420 Octa</td>
</tr>
<tr>
<td>CPU</td>
<td>Octa-core (4x2.1 GHz Cortex-A57 &amp; 4x1.5 GHz Cortex-A53)</td>
</tr>
<tr>
<td>GPU</td>
<td>Mali-T760MP8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Memory</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal SSD</td>
<td>64 GB</td>
</tr>
<tr>
<td>RAM</td>
<td>3 GB</td>
</tr>
<tr>
<td>Card slot</td>
<td>No external SD card slot</td>
</tr>
<tr>
<td>----------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td><strong>Camera</strong></td>
<td></td>
</tr>
<tr>
<td>Primary</td>
<td>16 MP (f/1.9, 28mm, 1.12 um)</td>
</tr>
<tr>
<td>Video</td>
<td>2160@30fps, 1080@60fps, 720@120fps</td>
</tr>
<tr>
<td><strong>Communication</strong></td>
<td></td>
</tr>
<tr>
<td>WLAN</td>
<td>Wi-Fi 802.11 a/b/g/n/ac, dual-band, Wi-Fi direct, hotspot</td>
</tr>
<tr>
<td>GPS</td>
<td>A-GPS, GLONASS, BDS</td>
</tr>
<tr>
<td>Bluetooth</td>
<td>4.1, A2DP, BLE, aptX</td>
</tr>
<tr>
<td><strong>Battery</strong></td>
<td></td>
</tr>
<tr>
<td>Capacity</td>
<td>Li-Ion 2550 mAh</td>
</tr>
<tr>
<td>Talk time</td>
<td>Up to 17 h (3G)</td>
</tr>
<tr>
<td>Music play</td>
<td>Up to 49 h</td>
</tr>
<tr>
<td><strong>Performance</strong></td>
<td></td>
</tr>
<tr>
<td>Basemark OS II</td>
<td>1663</td>
</tr>
<tr>
<td>Basemark OS II 2.0</td>
<td>1641</td>
</tr>
<tr>
<td>Basemark X</td>
<td>22752</td>
</tr>
</tbody>
</table>

### 5.3. Software frameworks

Application was developed for Android operating system (OS). Many different software development kits (SDKs) and libraries were used throughout the implementation. These included game engines, tracking solutions, map data services and a touch input library.

Implementations were built on many different versions of the Android OS, but all the components were found to work on versions 6 or higher. Android Studio itself, or Android libraries were not used directly, as the used game engines call the building processes and have function calls for the Android libraries.

Game engines were found very useful in creating AR applications. They provide a robust platform to work with 3D computer graphics and an intuitive GUI to handle the content. Many AR related SDKs also support popular game engines. Unreal Engine [49] and Unity [50] were both used during the implementation. However, Unity was deemed as the better platform for this project, as it has more mature support for mobile development. It also has more AR SDKs supporting it.

Most major AR SDKs available at the time were tried during the implementation. These were Vuforia [51], Wikitude [52], Kudan [53] and ALVAR [54]. The final application was implemented using the ALVAR’s point cloud tracking.

Additionally, map services were used to acquire map data. Google Maps data was used at first, but in the end, MapBox was found to provide better tools to use maps in game engines.

### 5.4. Preceding iterations

There were three major iterations, or prototypes, before the application architecture was finalized. With the iterations before the final implementation, many different solutions were explored. Especially, many different tracking solutions were tested. All of them were deemed unsatisfactory in the end. However, they gave useful insight into how different tracking solutions and design for UX affect each other.
5.4.1. The first iteration

The initial idea for implementation was to overlay a 3D model of Oulu in the correct coordinates as user looked through the screen. This way, the application would also be able to position surveys in desired coordinates. Tracking solution was a combination of GNSS, gyroscope and a compass. GNSS was used to determine the position of the user. Gyroscope was used to orientate the camera. Compass was used to initialize the camera heading to align it with the real world and correct the orientation at intervals.

Application was at first developed on Unreal Engine, as many previous VR works in the research team used it. Unreal Engine was also seen beneficial as it provides open source code for the engine. Later, however, Unity replaced it, because mobile development on Unity is faster. Possibly because Unity has focused more on supporting mobile development than Unreal Engine. Unity also offers better documentation for its mobile library. For example, Unity provides an easy solution to use GPS location service, while Unreal Engine completely lacks documentation for its respective implementation. Unity also uses C# versus Unreal Engine’s C++, which compiles faster into an Android application.

The application was able to overlay the 3D city model on top of the real city (Figure 14). GPS accuracy varied approximately between 1 to 10 meters, which was deemed good enough for presenting the bubbles. Gyroscope was accurate enough for user to freely look around. However, initializing the model with correct heading proved difficult, as the acquired heading was not precise. The heading accuracy was approximately 15° at best. Also, depending on the orientation of the device, the compass had rapid changes in the heading up to 180°.

Also, while the GNSS and gyroscope were precise enough for placing surveys, they were not satisfactory for aligning 3D content, e.g. overlaying models on top of real buildings. Together with the compass problem, this pushed the development into the next iteration, where better tracking solutions were sought.

Figure 14. A 3D model of the city overlaid on top the real one.
5.4.2. The second iteration

Second iteration was characterized by exploration of different tracking SDKs available. These were Vuforia, Wikitude and Kudan. The SDKs were ultimately insufficient to implement the initial design. However, exploration of them provided insight into strengths and weaknesses of different tracking methods and how they enable different use cases.

Vuforia provides many tracking methods. The simplest one is “device tracking”, which in practice means sensor-based tracking, where gyroscope is used to track the rotation of the device and accelerometer is used to correct the rotation with the Earth’s gravity. Vuforia is best known for its marker-based tracking. It can robustly track any image with enough features. Image tracking however is not very suitable for large-scale mobile AR, as trackable images would be needed everywhere the application should work. Another downside was that Vuforia does not allow manual rotation of the camera. This prevented aligning the camera heading with compass heading. However, a fix for this was found with the third prototype.

Wikitude provides model-free optical tracking. It can track a plane near the user quite precisely. The downside was that the plane must be chosen by the user. This means that the user looks through the camera and chooses an area to track and to place AR content on it. Therefore, the application was not able to place AR content on predefined locations, so Wikitude was regarded insufficient for AR PATIO use cases.

Kudan provides model-free optical tracking similar to Wikitude. Likewise, it enables the user to choose a plane to track and place content on it. For the same reason, it was found insufficient. Another problem was that Kudan tracking applies the transformation to the world, not the camera. Thus, combining the GNSS positioning with Kudan would have been complicated as the world transformation would have to be converted to the camera transformation, before combining them.

All in all, the second iteration, along with the first, indicated that designing the application to only give a direct AR view to the content might not form a good UX. The current state of tracking technology did not provide a precise enough solution on a large scale. This ignited a redesign and the third phase of the implementation.

5.4.3. The third iteration

Third prototype was much further developed than the previous iterations. It began to take the shape of the final application providing most of the functionalities and the UI. A significant change in design was addition of map view for the user. This was to give a better sense of nearby surveys and thus, overcome the limitations of imprecise tracking. When desirable, Vuforia’s image tracking was to be used to recognize building facades and align 3D content precisely with them.

In the third prototype, user was able to look around for nearby surveys in the AR view (Figure 15). This was implemented by combining Vuforia’s device tracking with GPS coordinates, as surveys did not need to align with the real world perfectly. The problem of manually correcting the Vuforia’s heading with compass was solved by making the camera a child of another game object and manipulating the parent object’s heading. Even though Vuforia does not allow manipulating the camera’s pose, parent object’s pose can be manipulated.

Tapping a survey (AR bubble in the view) opened an interface containing a question and a slider to give numerical answers. Survey data could be downloaded from a server
and it used the same format as the surveys in VR PATIO. The answers were stored in the application, but they were not sent back to the server.

Figure 15. AR view with an opened survey.

A significant change in the design was adding a map view (Figure 16). The map view shows the user a 3D map of the nearby area, along with nearby surveys. User was represented as an avatar on the map. This avatar was moved according to the values received from the AR view camera. This gave the user a better sense of direction and location. As the user saw the avatar from bird’s-eye view, the tracking did not have to be perfect. Initially, the plan was to only provide the map view without the AR view to surveys. When the user would open a survey in the map view, a direct AR view would open. In the AR view 3D content would be aligned with the real world using Vuforia’s image tracking. However, PATIO team argued the surveys should be found and opened in the AR view, as this would provide a more engaging UX. Thus, tapping a survey on the map opened only a preview of what the survey was about. The map tile data was acquired from Google Maps.
To show 3D content precisely in predefined locations, the plan was to use Vuforia’s image tracking to track facades of buildings and other recognizable features around the city. This was to be used only when a survey required to show some 3D content. The idea was tested early on with locations nearby where the project was developed. However, after the other functionalities were implemented, the image tracking was tested in a real use case in the city center, and it did not work well. Vuforia’s image tracking is meant to track planar images, and urban environment rarely has perfectly planar surfaces. Thus, the system was able to recognize the facades but very unreliably, sometimes not at all. Also, it was very difficult to even get a picture of the facades to track. Taking a picture from the ground makes the plane distorted and a translation is required, which made the tracked images less precise representations of the real facade. Taking a picture from a higher perspective, like inside the opposite building was also difficult, as the streets are generally too narrow to get a wide enough picture.

Leaving thorough testing of the image tracking so late was possibly the biggest neglect during the implementation. A lot of time went into implementing the core functionalities, while a critical component was not working. However, a solution for precise tracking was found and the implementation of the final software began. Also, the core functionalities, like the basic AR view and the map view, were found useful and kept in the final implementation.

### 5.5. Final user interface and use cases

The final UI consists of two AR views and a map view. The first AR view is referred to as *IMU-based AR view*, as it was implemented with IMU tracking. The IMU-based AR view is used for looking around to find the surveys, represented as speech bubble-shaped objects, or “Survey Bubbles” (Figure 17). Arrows in the bottom of the screen guide the user to move the camera around until a Survey Bubble is visible.

Along with the UI flow, the following images illustrate the two fleshed-out use cases that were materialized. First one presented 3D models of canopies planned for the city center of Oulu. The second use case was developed for filming demo-material of
PATIO applications. In this use case, the Toripolliisi statue was presented as moved to another location, following the initial design plans.

User can also get an overview of the location from a map view (Figure 18). The map view can be opened at any time by tapping the map icon in the top right corner. The map view shows an aerial view of the surrounding environment. The Survey Bubbles are overlaid on the map in corresponding coordinates. Survey Bubbles can be tapped in the Map view too, but this opens a preview window describing the content of the surveys. This preview window is closed by tapping on it. In addition, the map view shows an avatar of the user (Toripolliisi statue in this case), which moves according to the pose of the device.
When a Survey Bubble is tapped in the AR view, a survey opens. If the survey simply presents questions without any AR content, the survey is opened in the IMU-based AR view (Figure 19). This presents a speech bubble with a question, a slider to give a numerical answer, and a button to send the results. If the survey is accompanied with 3D AR content, a prompt to open it appears. The purpose of this prompt is to ask, if the user wants to activate the survey with AR content and to give instructions, where the user should look for the content.
When agreed to start a survey with AR content, the view is changed to the ALVAR AR view (Figure 20), implemented with ALVAR. From the user’s point of view, ALVAR AR view does not differ much from the IMU-based AR. The switch is more of a way of hiding technical issues of using two tracking methods. Guiding arrows are at the bottom of the screen to prompt the user to look for the target of the survey. In the final implementation, the view is also changed from portrait mode to landscape, as this worked better with the ALVAR SDK. This was a quick fix, which had to be done hastily before the first user tests. Previously the tracking had worked in a portrait mode. However, the target environment changed over time (e.g. Christmas lights were installed) and a new point cloud was created to have as accurate tracking as possible.
When the application recognizes a target location, it can place the canopies accordingly. A survey regarding the content appears at the instance the target is found. Figures 21 to 23 illustrate this, as well as the three options for different canopies. The options can be cycled back-and-forth with the Back and Next buttons. For each option, the according question is also cycled. In the case the option is the last one, or there is only one option, the Next button changes to Send button. Pressing the Send button takes the user back to the IMU-based AR view.

Figure 21. Open survey with AR content (the first canopy option in this case). UI design by Minna Pakanen.

Figure 22. The second canopy option. UI design by Minna Pakanen.

Figure 23. The third and the final canopy option. UI design by Minna Pakanen.
5.6. **Final software architecture**

The implementation of AR PATIO was built on basis of Unity application running on Android OS. Three plugin SDKs for Unity were used in the implementation. These were ALVAR to enable precise tracking of predefined locations, MapBox for managing map data and Lean Touch for diverse touch input functionalities. Figure 24 shows the high-level software architecture of the application.

![Software stack for AR PATIO](image)

**Figure 24. Software stack for AR PATIO**

The application was implemented in heavily object-oriented architecture. This system architecture is shown in Figure 25. User Interface is an object that handles most of the application management, along controlling UI elements. Navigator is an interface to retrieve geolocation data, meaning GPS coordinates and compass heading. Map object handles the map shown in the application and the placement of content in correct positions. User object controls the movement of IMU Camera and Map Camera. These cameras are used for the IMU-based AR and map views. The ALVAR camera is used by the ALVAR view. Survey Bubbles are the survey content visible in basic AR and map views. They also hold the survey data. Targets are objects that the ALVAR uses for tracking. They are managed by the Target Manager, which downloads point clouds and respective 3D content as needed. This object-oriented architecture was found to fit well with the Unity’s framework, which is formed of objects added to the application and their components. To tie everything together, singletons, state machines and listeners were found very useful.
5.6.1. Application management

The main business logic of AR PATIO is handled by the User Interface object. This object is a singleton that manages the UI and overall system flow. For managing the system flow, it uses most of the other components in the architecture, combining information from them.

UI has three main states. These are the IMU-based AR view, ALVAR AR view and the map view. State of the UI controlled with a state machine. In the basic AR view, user can see the survey bubbles and open the surveys by tapping the bubbles. Tapping a bubble either opens a plain survey or switches UI state into the ALVAR AR view. The ALVAR AR view looks like the IMU-based AR view on most parts but uses point cloud tracking. Thus, it can align 3D content precisely in the predefined locations. The surveys open like they do in the IMU-based AR view, but present the related 3D content. By tapping the map button, the UI state switches to the map view. In the map view, user’s own location is indicated with an avatar and nearby surveys are also shown. When a survey is opened in the map view, a preview information about the survey is displayed, prompting to find the survey in the basic AR view.
To handle the UI and the system flow, the User Interface object needs information from the other objects. Survey data is loaded into respective survey objects, so the User Interface object communicates the survey data between the survey objects, as needed. It also communicates with the Tracking Manager to get the state of the tracking, as in the ALVAR AR view the surveys must open after the target has been found.

5.6.2. Geological position management

Geological position is acquired and managed by the Navigator object. A few components in the system require a geological position and they retrieve it by calling the Navigator object. These objects are the Map, IMU camera, User and the survey objects.

The Navigator object uses Unity’s library to initialize location services on Android to read the GNSS position of the device. Initialization lasts for an undefined time as it needs to find enough satellites, so it is implemented as a script that runs independent of the Unity’s main update-loop. When the initialization is ready, other objects can call location coordinates from the Navigator object. Additionally, Android location service also handles the compass, so compass heading too is called from the Navigator object. For testing purposes, the Navigator object can be set to provide a static fake location.

5.6.3. Map management

Map object manages the map of the application. This includes creating the map and updating it as the user moves. The map was implemented with MapBox SDK, which provides a database to load map data and a comprehensive library for using maps.

The Map object initializes by loading map tiles according to the user’s location. The Map object will wait until the Navigator object has initialized, after which the map can be initialized with map tiles according to the GNSS coordinates. After the map is initialized, other objects can use the Map object to place themselves in the correct coordinates. This is done by converting the geographic GNSS coordinates into the Cartesian coordinate system used by Unity. User and survey objects use the map to place themselves.

5.6.4. User object

User is a simple object, with functions only to align its pose with GPS coordinates and inertial tracking. However, it works as a centerpiece of the application to support other objects and their functions. It has the IMU Camera and Map Camera as components.

For tracking the user pose, User object uses inertial tracking for rotation and GPS coordinates for position. To position itself in the game world, the User object uses Map object to calculate its relative position on the map. Additionally, the User object has an avatar that is visible on the map and moves accordingly with tracking, giving a sense of location and heading. By moving on the map, the User object forces the Map object to load new map tiles, as the user is reaching the current edges of the map. Also, by this movement, the user’s relative position to surveys is maintained.
5.6.5. **Survey Bubble objects**

Survey data is stored by the Survey Bubble objects. These objects store their respective questions and answers. This data can then be used by the User Interface object. Survey objects also store their designated GNSS coordinates. They use the Map object to place themselves in the correct coordinates in the game world. Each survey object also has a model that represents them as 3D speech bubbles in the IMU-based AR and the map view. The surveys can be opened by tapping on the bubbles.

Survey objects can be loaded and created during runtime. In the earlier prototypes, there was a manager to load the surveys from a server. However, in the final application the surveys are offline, built into the application. This is because the format of the surveys changed a few times during the development and resources were not used to create a comprehensive backend service for retrieving the survey data. Thus, it was left as open case for future development. The survey objects still introduce a JavaScript Object Notation (JSON) format that contains the required survey data:

```json
{
  "surveydata":
  {
    "questions": [<an array of strings> ],
    "previewtext": <string>,
    "usersalvar": <Boolean value>,
    "startalvartext": <string>
  }
}
```

The JSON format contains the questions as an array of strings. Preview text shown in the map view is provided as a string. As the User Interface object must know if a survey uses the ALVAR AR view or not, the format contains a Boolean value to determine this. And, as the ALVAR AR view is preceded by a prompt to open it with a survey-specific introduction text, this is provided as a string as well. The format is not complete though. For example, the images displayed in the prompt were hard-coded. Also, the downloaded survey data might have to contain references to the corresponding 3D models and the point clouds too. However, this is handled by the Target Manager object.

5.6.6. **Target management**

Targets are objects, which ALVAR SDK uses for tracking. Each target engulfs a point cloud to track and the 3D content that is aligned with the point cloud. The final implementation uses individual targets for each trackable area. Thus, there is a Target Manager object that manages the Targets.

A Target is an object from the ALVAR SDK that holds the respective point cloud data. Target objects in the application are added into the ALVAR tracking system as active targets to track. The Target objects also hold camera calibration data that can be created for each device model to optimize the tracking. The Target object also has the 3D content related to its point cloud as children. It can contain multiple 3D objects and display them separately. This is used when a survey contains multiple questions each related to a different 3D model.
Target Manager object engulfs all the targets in the application as children. It can load point clouds and 3D contents as needed on runtime. It first loads a package which is a Target object with 3D models pre-placed in correct positions and initialized the object. Then, it loads the respective point cloud into the Target object. Finally, it notifies the User Interface object what is the currently active target.

5.6.7. Cameras and views

There are three UI view states in the application, hence there are three cameras to facilitate the views. One is for the IMU-based AR, one for ALVAR AR and one for the map view. Cameras are separate as this was the easiest way to switch between UI view states and tracking techniques, while maintaining the camera pose in the other states. At any state, the currently unused cameras are not rendering anything, so the effect on performance is negligible.

The IMU-based AR view is implemented by the IMU Camera. It is called IMU as it uses IMU sensor-based tracking. The IMU-based AR view is used for showing the survey bubbles around the user in correct coordinates. To be precise, the IMU Camera is a component of the User object, and the tracking is applied to the User object. This way, the User object pose has the same pose visible on the map view as the IMU Camera has in IMU-based AR view.

The ALVAR AR view is implemented by the ALVAR Camera. It is an individual object provided in ALVAR SDK. Its purpose is to render the AR content more precisely using the ALVAR point cloud tracking. The ALVAR Camera also provides the camera image of the real world that is rendered in the background. The IMU-based AR view uses the same rendered background.

The Map view is implemented by the Map Camera. It is a component of the User object and located above the user. It renders the map with the user and the surveys on it. The map view can also be zoomed in and out for a better view.

5.6.8. Sensor based tracking

A custom sensor-based tracking solution was required for the final implementation, even though Vuforia’s device tracking was quite accurate and could be corrected with the compass. This was because different AR SDKs were difficult to combine at the time, as they collide using same resources, e.g. the camera. Sensor based tracking was implemented using the gyroscope, accelerometer and magnetometer readings from the device through Unity’s libraries. Gyroscope is used for smooth orientation of the User object, while accelerometer corrects gyroscope’s drift with the defined orientation of the device relative to Earth’s gravity, along with compass heading defined with magnetometer.

To determine the orientation, accelerometer and magnetometer are used first. Acceleration is retrieved as three-axis radian values. These values are then low-pass filtered to reduce shakiness. Unity’s library for compass heading returns the heading as one-axis degree value that is already calculated from the values of three-axis magnetometer. The heading is converted into radians, separated into sine and cosine and low-pass filtered, as the compass heading is especially shaky. After acquiring filtered values, yaw, pitch and roll of the device is calculated. Yaw is calculated from
sine and cosine of heading and converted back to degrees. Pitch and roll in degrees are calculated with following formulas commonly used in industry [55]:

\[
Pitch = \text{Atan2} \left( z, \sqrt{x^2 + y^2} \right) \times (-57.3)
\]

\[
Roll = \text{Atan2} \left( x, (\neg y) \right) \times (-57.3)
\]

, where \( x \), \( y \) and \( z \) are the three-axis values of the accelerometer. After this, the values are applied to the User object. In this, the unreliability of the compass must be considered. It was noticed that when the device is held in portrait orientation, the compass heading is reasonably accurate only between certain pitch angles. At the extremes, outside \( 0^\circ \) to \( 90^\circ \), the compass heading is completely unusable. Thus, the yaw is applied to the user orientation only, when the pitch is between \( 15^\circ \) and \( 75^\circ \) (Figure 26). Otherwise, only pitch and roll are applied. One problem with this approach is that the device is often held around the zero-degree pitch, so compass heading for correct yaw is not applied often. For note, the reported compass heading worked better, when the device was held in the landscape orientation. This may be due to way it is implemented in Unity’s library, which the documentation does not disclose however. A future fix would be to calculate the heading manually from three-axis magnetometer readings, but at the time resources were used for other components.

![Figure 26. The pitch rotation of the device where compass yaw is reasonably reliable.](image)

Gyroscope is also used to define the orientation. Gyroscope’s readings are acquired as already processed and unbiased values [56]. Thus, they can be directly applied to User object’s orientation, which alone would yield quite smooth tracking. However, the gyroscope has noticeable drift. It also cannot recover correct alignment, when the application is paused and resumed, for example. Therefore, the accelerometer and compass heading are important as they can correct the orientation relative to the real world.
Point cloud tracking is provided by ALVAR for Unity SDK. To track a point cloud and align 3D content with it, the point cloud must be created and added into a Target. Any desired 3D content is placed as a child of the Target. Then, the ALVARTracker can keep a list of all the Target objects in the application and interface with ALVAR’s dynamically linked libraries to recognize the point clouds and define the device’s pose. Finally, the received pose is used to orient the ALVARCamera.

Point clouds are created outside the application. This is done by taking pictures of the target environment. A recommended amount is 30 to 100 images. In outdoor environments, it is also recommended that multiple sets of images are taken in varying lighting and weather conditions [54]. Although ALVAR was possibly the best solution available at the time for tracking predefined locations outdoors, the creation of point clouds is a major strain on the scalability of the application. After taking the pictures, point clouds can be created on the server provided by ALVAR. Though, they can also be created with various 3D reconstruction software. For example, VisualSFM [57] was used in this project.

The created point cloud data is imported into Unity. Point cloud data is set into the Target object and the tracking is basically ready to use. Desired 3D content should of course be added as the child of the Target. The 3D content can be placed in in the Unity Editor, where the point cloud is visualized. One major problem with the point clouds was that their data type could not be imported at runtime into Unity, making downloading them impossible. However, this was solved by packaging the point cloud data into an AssetBundle, which is a Unity data structure that can package any content in the application, and which can be loaded at runtime.

After the Targets are set and activated, they are used by ALVARTracker. It iterates through every active Target in the application. If ALVAR recognizes any of the targeted environments in the camera feed, it calculates the pose of the device. This pose is used to align ALVARCamera with the device. Whenever a Target is active and recognized, its child content is rendered in ALVARCamera. ALVARCamera also creates a shader that renders the device camera image in the background.

Most of the input is done by tapping buttons or dragging the sliders, which is simple to implement in Unity. In addition, Lean Touch asset was used as Unity does not provide more sophisticated touch input functions. Lean Touch, for example, enabled to easily make the survey bubbles interactable by tapping them. It was also used for pinch-zooming in the map view.
6. EVALUATION

The aim in this study was to evaluate, how usable and valuable the AR PATIO is perceived by users. Evaluation was also planned so that there would be good object of comparison. The chosen benchmark was the KioskAR study (introduced in the Chapter 3.4), as it presented a very applicable method to evaluate usability in MAR applications.

A user evaluation was conducted in authentic setting with 9 test users. Users were presented a typical case, where AR PATIO would be used. Test case location was in the city center of Oulu. Test users tried the application by going through the case. Lastly, qualitative interviews and quantitative surveys were used to gather data on the usability and the UX provided by the application.

6.1. Test users

The Classic PATIO itself was used to promote the test and recruit participants. Activity was open for all interested PATIO users. A total of 9 test users were recruited. Out of these, 8 test users were recruited from PATIO with one additional user recruited from another source.

Background questionnaire was used to gather demographic data and test users’ familiarity with mobile devices, AR or 3D technologies in general. Their age group and gender distribution are shown in Figure 27. Age groups yielded a representation of young and old, which could be used analyzing the results. Most of the users were men. For education, majority had a bachelor’s degree as the highest level of education. Major subjects and the current occupations varied, although some had education background in information processing science or software engineering. Only two of the test users had used AR with smartphones or smart glasses. Users had more experience with 3D virtual environments. Almost all had watched 3D movies.

![Age group and gender distributions of the test users.](image)

6.2. Test case

The test case was the survey on the planned canopies presented in Chapter 5.5. The topic was to view the planned canopies for the city center and rate each one.

The test case began with the users standing near the location of where the content of the application would be. They were thought to already know there is content nearby
and they should open the application. As users opened the application, they were to find the surveys. There were two surveys. One survey presented the virtual canopies in AR and asked to rate those. Another one was a simple one question survey that asked how the user’s felt about the application. Its purpose was to represent a survey without AR content, such as the virtual canopies.

The case was chosen, because the same scenario was implemented and studied in VR in the research unit. Thus, a study comparing the AR and the VR implementations could be conducted in the future. Also, due to the previous work, the 3D models for the canopies were easily available.

6.3. Test setup and devices

The environment of testing was obviously the location of the test case. This was a pedestrian crossroad at the very center of Oulu. Tests were conducted between 11-14 hours, as these are the lightest hours of the day in November-December on high latitudes. The tests were spread across six days, but the weather was quite consistent. Temperature was between -5 to 2 degrees. Thin gloves were given for the test users, so that the cold would not affect the experience too much. The gloves’ fingertips were designed to use with a touch screen, which worked well. However, the gloves made the handling of the device more slippery. Five days had cloudy forecast and one day had very light rain. On the rainy day, an umbrella was held over the test user, so that raindrops would not affect the application usage.

The tests were done on the same smartphone used for development. On a note, people commonly have protecting covers on their phones, which also make them less slippery. The test phone did not have such cover, which made handling the phone difficult for some participants. Additionally, a voice recorder was used to record interviews.

6.4. Test flow

Individual tests proceeded as follows. At first the broad idea of the application was explained to users that is, they would have to find surveys in the area and proceed as the application guides. Users were also told to test the application for as long as they liked. Users were then handed the device and asked to open the application. After opening the application users would either start to look for surveys through the AR view, or after a while they were guided to do this. There were two surveys nearby. After either one was found users would either tap the survey open, or they were guided to do this. The users would then answer the opened survey. If the survey was the one with the AR decks, they would view the decks how they pleased. After closing the first survey, the users would either try to find more surveys or they were told about the second survey. After the users had answered both surveys they were done with the test. When the actual testing was done, the data was gathered with the questionnaire and the interview.
6.5. Data collection

For data collection, both qualitative and quantitative methods were used. Qualitative data was gathered with interviews and quantitative with post-test surveys. It should be noted, the questions in the actual tests were given in Finnish.

Semi-structured interviews were used as qualitative method. Interview questions touched the overall experience, UI and if users would use such application in the future. The questions were also the same as in the VR study which implemented the same use case. The questions were following:

1. Describe your experience.
2. Was interaction easy or difficult, and why? Or what influenced that?
3. Was switching between the options easy or difficult and why?
4. Was rating the options easy or difficult, and why?
5. What do you think of this way to rate things related to urban planning?
6. Were the ideas easier to understand, than from pictures?
7. If this method was available with all architecture and urban planning related evaluations, would you use it rather than pictures, and why?

A post-test survey was given to the users to gather quantitative data. The survey questions used were adopted from Handheld Augmented Reality Usability Scale (HARUS) [45] and partly from MEC Spatial Presence Questionnaire (MEC-SPQ) [46]. The relevant parts of these surveys were used. From HARUS, the comprehensibility and manipulability scales were used. From MEC-SPQ, only the involvement scale was deemed relevant for this case.

The survey consisted of 21 questions in total. Answers were given on a 5-point Likert scale from strongly disagree to strongly agree. The questions were following:

For comprehensibility:

1. Interacting with this application requires a lot of mental effort.
2. The amount of information displayed on the screen was appropriate.
3. The amount of information on screen was difficult to read.
4. The information display was responding fast enough.
5. The information displayed on the screen was confusing.
6. The words and symbols on screen were easy to read.
7. I felt that the display was flickering too much.
8. The information displayed on screen was consistent.

For manipulability:

1. Interacting with AR PATIO requires a lot of body muscle effort.
2. Using the application was comfortable for my arms and hands.
3. The device was difficult to hold while operating the application.
4. It is easy to input information through the application.
5. My arm and hand became tired after using the application.
6. The application is easy to control.
7. I was losing my grip and dropping the device at some point.
8. The operation of this application is simple and uncomplicated.
And for involvement:

1. I thought mostly about things having to do with the application.
2. I imagined precisely where the decks were.
3. I thought the application could have personal meaning to me.
4. I considered things that were presented in the AR view.
5. The experience activated my thinking.
7. RESULTS

Quantitative scores of AR PATIO were compared to the ones of KioskAR. Unfortunately, KioskAR paper did not provide exact numerical results, nor were these able to be acquired. To further analyze quantitative results for AR PATIO alone, the deviations for each question are also presented. Questions with large deviations were analyzed for differences between different age-groups and levels of experience with AR technologies. Statistical significance in the differences was calculated with Student’s t-Test. Lastly, the quantitative data from the interviews was transcribed and categories were sought in it.

7.1. Quantitative results: questionnaire score comparison

Even though numerical results for KioskAR were not available, there were some comparable differences between the two applications. AR PATIO scored partly lower on comprehensibility. For manipulability the two applications had their own strengths and weaknesses. With involvement there was no noticeable differences.

For comprehensibility (Figure 28), AR PATIO scored low on Q1 ("Interacting with this application requires a lot of mental effort") and Q7 ("I felt that the display was flickering too much"), while KioskAR scored quite well on all fronts. Q7 tells that AR PATIO suffered more from flickering. This was possibly because the AR content in the ALVAR view flickered, as the tracking was lost during quick movements. Users also mentioned this in the interviews. In comparison, the tracking in KioskAR was implemented with GNSS, a compass and an accelerometer [44], which seems to result in high ratings, even though it provides less accurate tracking. Q1 indicates that AR PATIO required much more mental effort to use. It is not clear why this is so, as most other scores are relatively high, and the application was regarded as simple and easy to use.

![Comprehensibility](image)

Figure 28. Average comprehensibility scores for AR PATIO and KioskAR.

For manipulability (Figure 29), AR PATIO scored higher on Q4 ("It is easy to input information through the application") and Q6 ("The application is easy to control"). Vice versa, KioskAR scored higher on Q3 ("The device was difficult to hold while..."
operating the application”) and Q7 (“I was losing my grip and dropping the device at some point”). Scores on Q4 and Q6 indicate that inputting information and generally controlling the application was easier in AR PATIO. This is possibly due to AR PATIO providing simpler functionality. Q3 and Q7 were about holding the device. AR PATIO’s low scores are most likely explained by the very slippery device, which many of the users complained about.

For involvement, both applications scored reasonably well (Figures 30) KioskAR’s results for the Q5 were missing for some reason. Also, comparing the results between the applications is not meaningful without numerical results. Relatively lower scores for Q1 (“I thought mostly about things having to do with the application”) and Q3 (“I thought the application could have personal meaning to me.”) may indicate that the application was not too engaging. Still, the perceived presence of the 3D canopies was rated relatively high. And in contrast to Q1 and Q3, the application was perceived to activate the users’ thinking on some level (Q5).
7.2. Quantitative results: deviations in the scores

There was deviation in some of the scores in all categories. Scores with the largest deviations were further analyzed by comparing the absolute scores between two major groups in the participants. One type of comparison was done between age groups 25-44 and 45-65, as they were almost evenly sized groups. Another type of comparison was done between participants who had some experience with AR (3 participants) and those who had none (6 participants).

Figure 31 shows that for comprehensibility, most deviated scores were given for Q1 (“Interacting with this application requires a lot of mental effort”). However, there was no significant difference between the compared groups.

![Figure 31. Stacked percent deviations for comprehensibility scores.](image)

For manipulability (Figure 32), Q2, Q3, Q6 and Q7 show the greatest deviations. For Q2 (“Using the application was comfortable for my arms and hands”) and Q6 (“The application is easy to control”), there was no significant difference between the groups. However, there were differences (Figures 33 and 34) regarding Q3 (“The device was difficult to hold while operating the application”) and Q7 (“I was losing my grip and dropping the device at some point”). Both the younger age group and the participants who had experience with AR, had less trouble holding the device.
Figure 3.2. Stacked percent deviations for manipulability scores.

Figure 3.3. Difficulty of holding the device between major age groups ($p = 0.146$), and between experienced and non-experienced AR users ($p = 0.143$).
Difficulty of keeping grip on the device between major age groups ($p = 0.005$), and between experienced and non-experienced AR users ($p = 0.047$).

For involvement (Figure 35), there was deviation in the Q1 (“I thought mostly about things having to do with the application”). This was not reflected in the comparison of the groups though. Still, it could imply that the application was engaging for some and not at all for others. On the other hand, it may also reflect that the application required more mental effort for others, because of the phrasing of the question.

Figure 35. Stacked percent deviations for involvement scores.
7.3. Qualitative interviews

The recordings from qualitative interviews were transcribed and the data was analyzed to find the underlying themes and categories. Interview data was very rich and complex and no clearly defined categories were identified. Still, three main themes rose from the interviews. Firstly, the UI was thought easy to use, but interaction with AR felt harder. Secondly, the AR viewing of urban plans were thought to give better understanding how they would look like in reality, but that AR would not replace the conventional methods. Thirdly, users commented on wishes for more interaction with the 3D content.

The comments on usability of the application fell roughly into two categories, interacting with the familiar survey UI and the less familiar AR. The UI was thought very easy to use. UI was simple and elements, like the slider, were familiar. The simplicity of the UI was thought as a good thing: “Other AR things have been more confusing, demanding to browse many menus. Here everything needed was on the screen already”. On the other hand, IMU-based AR view caused some confusion, at first at least. Many of the users did not realize the survey bubbles were supposed to be tapped: “I’d probably still be confused if you hadn’t told me what to do”. However, after learning the idea, it was not thought as complicated anymore: “On the second time, it was pretty easy”. Also, there were many comments about the flickering in the ALVAR AR view and the canopies being hard to keep visible. From observing the tests, the flickering was due to tracking being lost. It also seemed difficult to understand where the camera should be pointed. The small screen size seemed to also be a problem, as the canopies were never shown completely on the screen, and neither was the target on the camera image.

Many of the users thought that AR can give a generally better understanding how things would look like in the future. However, many commented that AR would not replace images in newspapers etc., but rather complement them.

Some users voiced hopes for more interaction. A common wish was to be able to inspect the 3D content closer and from wider angles. 3D content could also be interacted with: “…could switch to see it in night lighting, or on a sunny day …”. As the UI was simple, voice recording method was brought up: “…writing text would be cumbersome, but you could leave a comment by speaking to the microphone”. Also, it was thought that more interactive content would be beneficial: “If I had nothing to do, I could open the application, if there were some small games, information or history about the area”.

A few comments rose about the value for user participation, such as: “If it’s this simple, I might tap it [the application] open occasionally”, “It [the application] would require more users, if only few are using it, it has no relevance” and “You can better express opinions, when you see something with your own eyes”.
8. DISCUSSION

For starters, the outcome of the thesis work did not come without limitations. Still, the thesis can provide interesting findings and educational mistakes. The evaluation results may be taken as hints, on which direction to further develop UIs and the UX in MAR applications within citizen participation paradigm. Moreover, the design and implementation process, especially due to the problems faced, gives insight on how to evolve the idea, and what are the requirements, to achieve better results in future works.

8.1. Limitations

There were many limitations regarding the whole study, but especially the evaluation. Firstly, with nine participants, the sample size was small. Another limitation could be that the users were biased to being interested in participation, as they were recruited from PATIO.

The comparison between the two applications was superficial, as numerical results were not available for the benchmark application. The questionnaires utilized can mostly assess the usability, so definite conclusions about anything higher-level cannot be drawn. Earlier, the plan was to compare the results to another study done in the research unit, which implemented the same canopy case in VR, but the results were not available in time.

The test case also explored a narrow set of functionalities. Main part of the use case was rating architectural plans, which is not a new idea. The users also moved in a small area, where the surveys were located. Due to that, for example the map view was not used at all for navigation. In that, opportunities, or threats, of large-scale MAR were not truly assessed.

In addition, the implementation had some shortcomings. Some planned features were not implemented, e.g. the notification for nearby surveys. Also, parts of the software architecture were not fully fleshed out in their functionality. For example, many features in the application would in production depend on back-end services, which were limited to one simple file-server for the point clouds. In addition, many obstacles were faced, mostly with implementing a satisfactory basis for tracking. This consumed time which could have been used to further develop other features of the application.

8.2. Findings

The objectives of the work were to design and implement a MAR client application that enables efficient citizen participation in urban environment. The main questions were what functions such an application should provide and how should the UX and the UI be designed.

The UI of the application was thought to be simple and easy to use. This may be due to using familiar 2D GUI elements. Despite being reported as simple and easy to use, many users had trouble at some point, especially the AR views were thought to be confusing at first. Also, use of the application was reported to require mental effort. The 2D guiding arrows might not be enough, as many users had trouble to understand what to do in the AR views, e.g. where to turn the camera. Some users even turned
their head, instead of the device, when instructed to look around in the AR views. This could be an inherent limitation with hand-held AR. With HMDs the purpose of AR view could be clearer to new users. However, in any case the reliability and functionality of the used tracking method affects the usability. For example, the flickering was reported as annoying and participants often had trouble finding the targets. Generally, it was deemed easy to control and input information in the client, reportedly easier than in some other AR applications. This may indicate that it is good practice in MAR to keep 2D GUI elements minimal, as the AR views demand a lot of attention. Otherwise, the connection between the abstract UI on the screen, and the supposed augmented reality behind it may not be understandable.

The perceived UX was a mixed bag. The simplicity of the application could also be a shortcoming. Many users wished for more interactivity, or additional information. Merely presenting surveys and related 3D content may not be enough to provide value for the users for them to engage in participation processes. Still, it was thought to be valuable for getting a better understanding how designs would look like in reality. However, such application may not be able to replace conventional methods, e.g. pictures completely. It could be justified to design the application to be a complementary component for other methods. It was also unexpectedly difficult to hold the device. For example, the physical buttons of the smartphone were in the way all the time. It may be important to address this in future designs. The model-based tracking method introduced problems, as it was challenging to find the targets and keep them in focus. Though the point clouds were not created perfectly: only images in one lighting condition were used and the point cloud also did not cover all the angles the 3D content could be viewed. For example, the highest canopy was mostly outside the trackable boundaries. The benchmark application only used sensor-based tracking, but it is not clear if this would give any better UX in the long run. In MAR, the sensor-based tracking may not give sufficient understanding of how the virtual content relates to the real world. In addition, benefits of the added map view could not be assessed in this study. It could be very important in giving an overview, especially in large-scale MAR applications.

In the implementation, fast prototyping was a very useful method for finding best solutions. Used technologies, particularly for the tracking, were probably the most applicable ones for the use case. This was at least at the time: the MAR field is advancing fast, as during later stages, new impressing model-free tracking solutions were published, such as the Google ARCore. What is not clear, is that were the chosen solutions applied in the best possible way. The tracking approach is strongly entwined with the design of UX. With the state-of-the-art in MAR, it must be considered carefully, how different tracking techniques support different kinds of UXs within citizen participation. In addition, different physical environments can have unique characteristics. Therefore, what works in one environment, may not work well in another. The environments can also change over time, which may render a previously working solution unstable.

8.3. Future visions

The provided functionalities in terms of citizen participation should probably be thought over, considering the possibilities, but also the technological challenges of MAR. There is also much to improve with the UI and the UX of the finished prototype. Likewise, there is much to do to support large-scale MAR applications in general.
The client application could be improved for its functionality. It should be studied, what sort of interaction would be interesting to the citizens, while also providing value for the researchers and stakeholders in citizen participation processes. One example could be to allow the users to place 3D content and manipulate it, instead of just looking at it. Users could be instructed to build the canopies themselves or place the statue in a location they feel best. Also, other input methods, such as speech recognition could be tried. However, these additions would in turn increase strain on administrators in terms of how to analyze the data and gather results. It should also be studied, how to design the high-level UX flow. For example, it would be interesting to see how a map view compares to an AR view in MAR navigation. It could also be reviewed, if a MAR client should be a comprehensive stand-alone application, or more of a complementary visualization service for other applications. In addition, it could be explored if adding sense of community would be beneficial, and how should it be done. For example, the users of VR PATIO could be connected to the ones of AR PATIO.

Earlier, the plan was to compare the results to another study done in the research unit, which implemented the same canopy case in VR, but the results were not available in time. The results may be used for comparative studies in the future.

On a higher-level, supporting infrastructure for large-scale MAR clients need to be developed to enable tracking and content management on large-scale. Without such, the functionalities of citizen participation MAR clients will probably remain severely lacking. Particularly for the tracking, back-end service for fetching point clouds according to GNSS location is required to enable large-scale MAR. Also, tools to manage this are needed, as for example the content needs to be aligned with the point clouds by the administrators. Moreover, the tracking and content data must also be created. This is very labor-intensive, so supporting tools are required. Acquiring and managing a tracking database could also be outsourced. For example, the data from users’ devices could be used to map new locations and update existing ones. Another possibility could be to create an application that enables easier creation of the tracking data, which could be used by administrators and stakeholders.
9. SUMMARY

The main objective of this thesis work was to develop a MAR client application that facilitates citizen participation in design of e.g. services, products and city plans. The interest was in what functionalities should such an application provide and what kind of UI should it have.

The work was done empirically, searching for best practices in the literature, while iteratively further developing the prototype application. The resulting final prototype enables users to explore the city in AR and find location-based surveys. These surveys can present 3D content in AR, such as displaying planned architecture overlaid with the physical world. Thus, users can view the plans in the authentic location and express their opinions by giving numerical ratings for the plans. In terms of technology, many tracking solutions were tried, and in the end, sensor-based tracking was used for finding the surveys and model-based point cloud tracking was used for presenting the 3D content.

The prototype was evaluated with real users in authentic location. HARUS questionnaire were used to evaluate the usability of the UI. Also, MEC-SPQ was used to assess the perceived UX. The prototype was also compared to an earlier similar work [44], which had used the same quantitative questionnaires in its own evaluation. This comparison was lacking however, as there were no exact numerical results for the benchmark application. In addition, interviews were used to gather additional qualitative data.

The results suggest that simple 2D GUI combined with AR view may be a good way to design an approachable UI for hand-held AR applications. However, more interactive and interesting functionalities may be required, than just presenting surveys with static 3D content. Otherwise, users may not be interested enough to use such application to participate. It is also important to consider that a MAR application for citizen participation may not be able to replace conventional methods, but rather complement them. In addition, back-end services to facilitate large-scale tracking and 3D content management are required, before large-scale MAR citizen participation can be meaningfully realized. All in all, this thesis work provides useful starting points, when developing such application in the future.
10. REFERENCES


