UNIVERSITY OF TARTU
Institute of Computer Science
Software Engineering Curriculum

Ott Saar

RematoAR – An Augmented Reality Assignment Aid

Master's Thesis (30 ECTS)

Supervisor(s): Ulrich Norbisrath, PhD
Ranno Käomägi, CTO of Remato

Tartu 2022

RematoAR – An Augmented Reality Assignment Aid

Abstract:

This thesis describes the design, development and testing of RematoAR, an augmented reality application with a focus to improve the communication in the construction industry by giving the users a new form of communication. The goal of the application is to reduce the needed communication between the project manager and the worker by providing the assignment related information through virtual markers. During the thesis the main features of the application are set and the application flow with the consideration to the nature of the augmented reality applications is designed. Similar available applications are found and analysed, and the descriptions of the differences with RematoAR are given. Users are provided with tools to place basic 3D elements such as texts, polylines and 3D objects, and request the task related information afterwards using the QR codes. For remote assistance a screen sharing feature is implemented. A usability testing session with four different construction companies was made. The tests focused on the evaluation of the idea, the usability of the made solution and the ease of use. Additionally, it was found if and how the construction companies would benefit from this kind of solution in their everyday work.

Keywords:

Augmented Reality, AR, Computer Graphics, Mobile AR, Construction, Construction AR

CERCS: P170 Computer science, numerical analysis, systems, control

RematoAR – Ülesande seletamise liitreaalsusrakendus

Lühikokkuvõte:

Antud magistritöös kirjeldatakse RematoAR rakenduse disaini, arendust ja testimist. Rakenduse eesmärgiks on parandada ehitusettevõtete kommunikatsiooni, andes selleks uus ülesande seletamise viis. Eesmärgiks on anda ülesande teostamiseks vajalik informatsioon edasi läbi virtuaalsete objektide ja nii vähendada projektijuhi ja töölise omavahelist suhtlust. Magistritöö jooksul pandi paika tähtsamad rakenduse funktsionaalsused ja disainiti rakenduse kasutatavus võttes arvesse liitreaalsusrakenduste eripära ja standardeid. Leiti ja analüüsiti sarnaste ideedega turul olevaid rakendusi ning toodi välja nende erinevused valmistatava rakendusega. Rakendus pakub kasutajatele võimaluse asetada lihtsamaid 3D elemente nagu tekstid, jooned ja 3D objektid, samuti võimaldab rakendus sama infot taas näha kasutades selleks QR koodi. Pakkumaks kaugabistamist on rakendusse implementeeritud ekraani jagamise funktsionaalsus.

Veendumaks rakenduse kasutatavuses viidi läbi testid nelja erineva ettevõttega. Testide eesmärgiks oli saada kinnitust idee väärtuses, aru saada loodud rakenduse kasutatavusest ja hinnata kasutuslihtsust. Lisaks selgitati välja, kas ja kuidas saaksid ehitusettevõtted kasu sellisest rakendusest ja kuidas parandaks see nende igapäeva tegemisi.

Võtmesõnad:

Liitreaalsus, arvutigraafika, ehitus, mobiilirakendus, liitreaalsusrakendus

CERCS: P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Table of Contents

1 Introduction

In the construction business, like in any business where there are a lot of people, communication is key in order to get the fastest and the most accurate results. Having the best results requires a lot of preparation like planning and marking of the construction site from the project manager[1]. There is also the need to communicate the information to the workers and this usually requires multiple visits to the construction site or phone calls with the workers to specify certain aspects. In a software like Remato, task description is usually done through taking numerous pictures of the construction site, writing some description about the task and attaching this information to a task. This info can then be accessed by the worker from the task's module, where they can see all the task related information with the taken photos. However, often capturing all this information requires the creation of multiple subtasks and navigating between these tasks to have a full picture which can be quite tedious.

To make the communication of tasks more effective, I propose here creating an augmented reality application. By giving the task virtual information, the workers could simply look around the room to find the extra information needed to complete the task. With today's availability of performant mobile devices and ever-evolving augmented reality capabilities, an application that is able to display the necessary information related to the current task is more accessible than ever before[2]. The main purpose of the application is to give the project managers and construction workers a possibility to store virtual information that would describe or draw attention to the things necessary in completing the assigned tasks. This way the need to contact the project manager with simple questions would be removed as the application would already display the needed information visually. Additionally, a video call system is introduced to eliminate any confusions with the markers remotely. The manager can be contacted via video call to add the missing information into the augmentation and clear any potential confusions.

The first chapter of the thesis describes the problem, Remato's current available solutions and introduces augmented reality to the reader. In addition, a brief overview of the proposed application and the capabilities of this application are outlined and described. The second chapter focuses on the analysis of the existing similar applications, brings out the similarities and differences between them and compares them to RematoAR. The third chapter gives an

[1] https://resources.workable.com/construction-project-manager-job-description
[2] https://www.techdim.com/the-history-and-evolution-of-smartphones/

overview of the technologies used in the creation of the application and describes the principles and ideas that were used in the application design process. The fourth chapter goes more in depth about the implementation part of the application, describing the implemented features and explaining the problems that arose and found solutions to these problems. The fifth chapter describes the test flow and focuses on the results received from the testing of the application. Additionally, the chapter focuses on the suggested improvements and discusses the future for the created application.

2 Problem statement

While working in a start-up company called Remato, the importance of communication in construction companies can be seen with each new client Remato acquires. Every client has some new suggestions on how to display the information of a task better, being it through photos or through additional comments about the tasks.

2.1 Remato

Remato Solutions OÜ (Remato) is an all-in-one construction software that allows the construction companies to manage assets, plan projects, assign tasks to workers and track time to accurately see the time spent on a specific task[3]. The goal of Remato is to be the all-inclusive software in making the everyday tasks of a construction company more efficient. Remato provides a construction company with digital solutions to manage its workers and assets through one platform.

Different features of Remato are separated into modules. To track the work time, users can access the tracker module, in which they can choose the current project and task they are working on. Additionally, the used materials can be marked in this module. Usually, the worker starts their time using their mobile devices, but in cases where the user does not have access to their device, a tracker can also be started via one of the Suprema devices. Suprema has a variety of devices for access control and assessment tracking, like Biometric Scanners, key cards, and FaceStations, which can be connected to Remato to start task tracking through them[4].

The crew module contains all of the information about the people of the company, like their contacts, a live list of who is currently working, and reports of the work done. Through the crew module, new people can be invited to the company, and the worker information can be managed from one place.

In the project module, the user has overview of the current projects, create proposals, and budget their different projects. Additionally, all of the project-related documents and photos can be accessed from the project module.

The asset module is for managing the tools and available materials. All of the companies' tools are visible from this module, containing the information about the condition of the tool, who is currently using it and at what was the last location of the tool.

[3] https://remato.com/

[4] https://www.supremainc.com/en/

Assigning and explaining the tasks to the workers is done through the task's module, which allows users to create tasks and describe them through comments and pictures. For additional details, project managers can upload descriptive documents about the tasks and create custom fields.

Remato is cross-platform, meaning it is usable on the desktop and on different mobile devices (Figure 1).

Figure 1. Remato desktop and mobile application[5]

Because of the focus on creating something intuitive and beneficial to the construction companies, I proposed to make an application that uses augmented reality to make the communication between the project manager and a worker more meaningful. Instead of writing every task separately into the tasks list, the project managers could use only one of the tasks and attach a QR code to it. The task specific QR code can be used to start the augmented reality session. The AR session would contain additional information about this task using virtual markers. Instead of needing multiple photos and additional drawings on those photos to explain a task, the worker could visually see the information at the construction site through their device.

[5] https://remato.com/

2.2 Augmented Reality

Augmented reality is a way of combining the virtual and physical world into one through a complementary device that can do this combining [1]. The augmentation does not need to be only through visual media. It can also be audio or anything that can improve or alter our perception of the physical world [1]. In the case of RematoAR, we are focusing on the visual part of augmented reality using a smartphone, but it can also be done through some head-mounted display like HoloLens[6] or Google Glass[7]. The word augmentation comes from the idea that with augmented reality, we try to improve our perception of the physical world by conveying some virtual information on top of it [1]. When considering visual augmented reality, there are four common techniques to do it: marker-based AR, markerless AR, project-based AR, and superimposition-based AR[8]. RematoAR will be using the combination of marker-based AR and markerless AR. Marker-based AR techniques will be used to determine the QR code in the world and position the virtual objects relating to it into the world. Markerless AR will be used to navigate in the world freely and to keep the placed objects in the correct positions after pointing the camera away from the marker.

Virtual reality is at the other end of the spectrum considering the forms of reality, where the complete environment is created virtually, and the user interacts with virtual objects [1]. A virtual reality that is primarily virtual but has some aspects of the physical world augmented into it is called augmented virtuality [1]. Any form of reality augmented with virtual or real content is called mixed reality (Figure 2) [1].

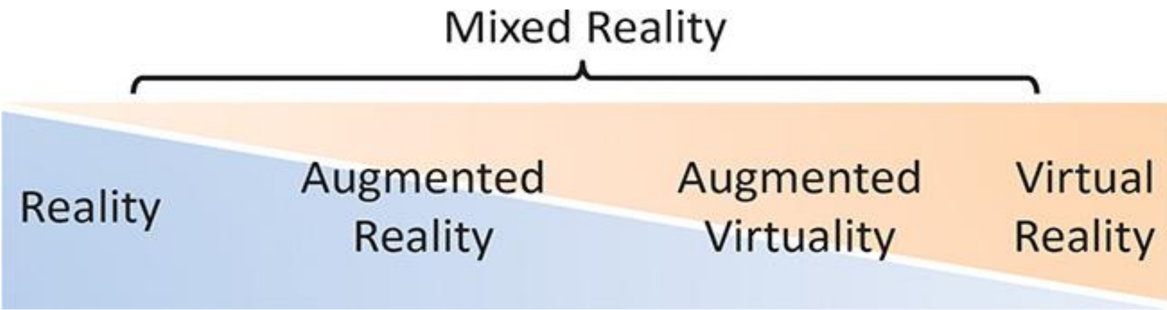

Figure 2.The scale where one form of reality transitions over to another [1]

[6] https://www.microsoft.com/en-us/hololens
[7] https://www.google.com/glass/start/
[8] https://clevertap.com/blog/what-is-augmented-reality/

2.3 Overview of the idea

The basic functionalities of the application would provide the project managers the tools to mark down helpful information into the world. The ability to place text and drawings should be enough to fill this requirement as most of the needed information could be drawn, and with the text, measurements or descriptions could be written anywhere. This virtual content would be shown to the worker who could use the provided information to understand their task through the smartphone. Any confusion with the marked information could be solved via a video call. Through the video call, the project manager can add additional information remotely to the virtual scene via screen share, and the added information can be accessed by the others immediately. Since the construction sites change overtime, the virtual content should be erasable.

As the AR can be quite a different experience compared to the traditional use of a mobile device, it is necessary that the application would provide as much explanation as possible. Therefore, a simple tutorial view, before the start of the application, would be needed to explain the tools to the new users.

2.3.1 An example of the application flow

An example of the application flow is as follows:

1. Project manager places a QR code into the room;
2. Project manager enters the construction site and scans the QR code;
3. Project manager draws the position of the power outlet in the app and writes the distance it should have from the wall;
4. Worker enters the construction site and scans the QR code;
5. Worker can see the position of the marked outlet;
6. Worker calls the project manager to clarify the other outlet positions through a video call;
7. Through the video call project manager discusses the positions with the worker and adds the necessary virtual markers;
8. The other workers entering the construction site can see the initial outlet position with the additional outlets that were added through the video call;

Through the use of the application, the project manager and workers should have the necessary tools to describe the tasks and reduce the amount of needed communication. Some companies have dedicated site managers for these occasions, someone who oversees the completed work, knows the details of the construction site, and has received plans or a task list from the project manager. However, considering the nature of the construction sites and possible plan changes, communication with the project manager is still needed. Oftentimes a simple call is not enough to solve these issues, and numerous photos and messages are exchanged in the process. However, with the built application, if there are any issues or confusion with the markers, they can be solved through a video call feature and can be marked down.

2.3.2 Requirements

To summarize the discussed aspects of the application, a more concrete list of requirements was formulated.

Requirement id	**Description of the requirement**
Req. 1	The application must provide the users the necessary tools to mark any virtual information into the world.
Req. 2	The virtual information must be accessible after the marking session to the workers to visually see the tasks.
Req. 3	The application must be able to teach the basics of the application, so that the user could simply download the application and start using it without prior training.
Req. 4	The application should contain the possibility to assist the user remotely by the project manager.
Req. 5	During the video call the project manager should be able to mark additional information to the workers AR session.

These five requirements are the main features the application must contain and will be considered when creating the application. Other features might be added, but these are the core aspects that make the RematoAR.

3 Related work

Before developing something new, it is necessary to find existing solutions and analyse them. Finding their strengths and weaknesses can help to outline the basic aspects that need to be filled within the created application to match the current market standards and user expectations. For this purpose, five augmented reality applications were analysed. Some of the found applications are not targeted at the construction industry, but they are included as they share some similar ideas with the created application.

3.1 Plop

Plop is an augmented reality application developed in Estonia. Its main purpose is to offer a new and fun way to leave messages to others[9]. The virtual objects are placed in a specific location in the world and can be viewed by others through their mobile device (Figure 3).

Figure 3. Placable ghost emoji in Plop[9]

The main difference between the application built during the thesis and Plop is that the virtual objects in Plop are tied to a specific location and shown to the user when they are in the radius of the object. The goal of RematoAR is to use QR codes as the reference points for positioning the virtual objects so the positioned objects would always be in the correct positions. Unfortunately, GPS positioning alone is not accurate enough to place virtual objects in the corresponding real-world positions as the GPS position is only accurate to about 5m[10]. That is why the Plop is placing the objects around the user in relation to the user's rotation without considering the distance the object has from the user.

3.2 TeamViewer Assist AR

TeamViewer is a remote access and control software that allows users to provide computer maintenance and assist others remotely[11]. TeamViewer Assist AR was introduced in the year 2019, and it provides a way to assist users or workers on the field through a video call

[9] https://plop.global/
[10] https://www.gps.gov/systems/gps/performance/accuracy/
[11] https://www.teamviewer.com

with the help of augmented reality[12]. Augmented reality is used to guide the user using virtual drawings or pointers to visually explain and solve the problems (Figure 4).

Figure 4. TeamViewer Assist AR[12]

The main difference between RematoAR and TeamViewer Assist AR is that the drawn virtual objects are not accessible or usable after the assist session in TeamViewer Assist AR. With RematoAR, the plan is to be able to access the notes and explanations even after the call has ended. The benefit of this is that the need to make repeated calls to the site manager is reduced because the information is there for the other workers. TeamViewer Assist AR focuses more on the aspect of remote help, whereas RematoAR focuses on the marking and information communication aspect.

3.3 VisualLive

VisualLive is a CAD/BIM viewing software that can overlay the virtual information onto the physical world[13]. It uses specifically placed markers or cloud markers as a reference to start the augmentation process and can overlay the digital information over the physical world using these markers. It's support for Android, iOS, and HoloLens makes it quite a versatile software for any construction company using BIMs (Figure 5).

[12] https://www.teamviewer.com/en/augmented-reality/
[13] https://visuallive.com/

Figure 5. VisualLive showing the BIM of the inner pipes[14]

Although the augmentation process of using markers as a reference to place the virtual objects is similar in both RematoAR and VisualLive, their core purposes are different. The main purpose of VisualLive is to aid the user in viewing BIM schemas and comparing them to the real results. Compared to RematoAR's approach, VisualLive requires creating a BIM schema before the information can be used on the site. Considering this, RematoAR's approach should make it more dynamic and easier to use as the needed information can be marked while being on the construction site.

[14] https://visuallive.com/

3.4 InspectAR

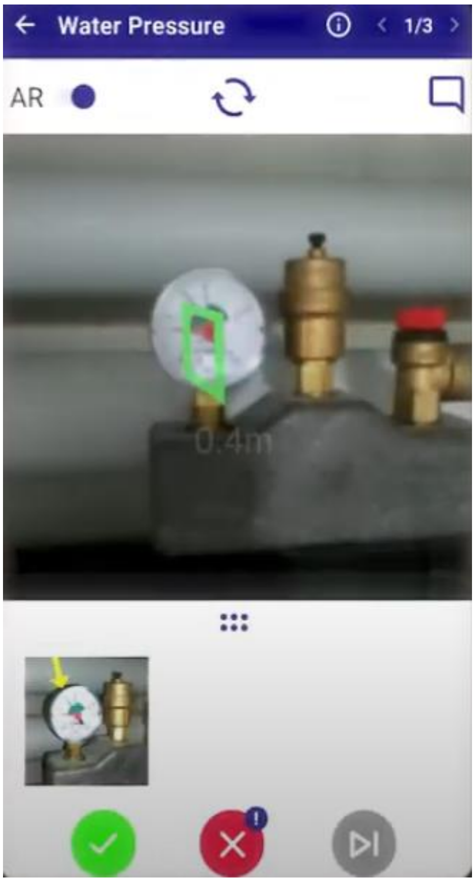

Figure 6. InspectAR showing what should be inspected in the view[15]

InspectAR is an augmented reality application developed by Augment IT, and it is mainly targeting maintenance, inspection, and training-related tasks[15]. It tries to make the process as simple as possible by having an overview of each task with complementing photos and videos. If the device supports AR, it can show some visual cues on the inspectable equipment. InspectAR is one of the applications similar to RematoAR, using the idea of having visual cues to help the user to navigate and complete their tasks at hand. In the case of InspectAR, they are using virtually placed reference points to place the visual cues correctly into the world (Figure 6). Virtually placed reference points are placed manually by the user during the AR session. Compared to the physical references like QR codes, the virtual ones are more convenient as they do not need much setup to be used. If the inspected equipment is the same and the virtual references are placed correctly, information will be the same everywhere. QR codes can do the same thing. Just the setup would take more time as the QR code must be positioned on the same spot on every instance of the equipment. The benefit of QR codes is that it removes the human error that might come when placing virtual references manually. Additionally, the user does not have to choose their equipment from the list. The equipment could be identified with the QR code. That is why RematoAR uses QR codes, as it simplifies the process for the worker. By just scanning the code, the needed virtual information is retrieved from the database and visualized for the worker. In Remato, it has been heard numerous times that it is more difficult to onboard older construction workers to work use the digital platform than it is to onboard younger workers. According to Atlanta Agent Magazine the median worker age is 41, meaning the less technical the process of viewing the visual information is, the better[16].

[15] https://augment-it.com/en/inspectar/
[16] https://atlantaagentmagazine.com/2021/06/30/whats-the-median-age-for-construction-workers/

3.5 IndustrialAR

IndustrialAR is an application that introduces a new way of assigning tasks to the workers. By 3D scanning an area of the working space and viewing it through the smartphone, the application is able to place virtual markers with messages and show indoor navigation to the task[17]. It additionally has the capability of AR remote assistance similar to the TeamViewer Assist AR.

Figure 7. IndustrialAR with the visible text marker[17]

This application is something that RematoAR sets out to do. It can display messages and helpful markers in the room and is capable of remote AR assistance. The difference between the two applications is that RematoAR is more "traditional" AR, whereas the IndustrialAR relies on the 3D scans of the rooms to place the virtual objects. Regarding the accuracy, the latter is better as potential movement errors are being removed by recognizing the position of the placed objects straight from the room. Its downside is that it requires more setup time and special equipment for scanning. In the promo video of IndustrialAR, they are using special 3D scanners to get the scanned room as accurately as possible and only after they can create their AR experiences. Small to medium companies, which is Remato's current client base, are unwilling to pay for a full 3D scan of a construction site that is bound to change quickly during the building process.

All of the found applications utilize a similar markerless AR approach, except for VisualLive, which combines the marker-based AR and markerless AR as the marker is used as a basis for starting the augmentation. Considering the provided tools in the analysed applications, the most common tools seem to be the text, drawing, and general 3D objects. With these, the users could mark down the necessary information and thus these will be used in the implementation of RematoAR. However, the most accurate marker placement and overall tracking seemed to be provided by IndustrialAR, as it has the 3D scan of the room and can use the scanned data to track the devices movement more accurately. Although a full 3D scan is not possible, a similar approach to IndustrialAR of feature detection can be used in RematoAR to improve the accuracy of the created application.

[17] https://industrial.viewar.com/

4 Design

Before starting the development, it was necessary to analyse the information obtained from the related work and decide on the technologies that could be used to create such an application. Additionally, user interfaces and the general flow of the application were designed.

4.1 Technologies used

There can be many different approaches when creating an AR application. For example, Android Studio for Android[18] or XCode for iOS[19] can be used to build it separately for each platform or use something that could easily export to both of the platforms.

As a part of the distributed systems research seminar, numerous AR frameworks were found and analysed. The research found the most used AR frameworks and compared them to find the most suitable one for RematoAR. One of the most common ways to develop AR applications is using Unity Game Engine with its cross-platform development and the support for many different augmented reality SDKs, so the Unity compatible frameworks were tested[20]. It was concluded in the research that the chosen framework depends a lot on the purpose of the application. By the assigned criteria, the most capable of the chosen frameworks was 8th Wall (Figure 8) [2].

Framework	***Tracking - Ranking***	***Objects Rendered - Ranking***	***Android version support - Ranking***	***Final ranking (points)***
8th Wall	3.5/5 - 5	1600+ - 1	7 >= - 1	1-2(7)
Wikitude	4.5/5 - 1	360 - 5	7 >= - 1	1-2(7)
ARCore	3.75/5 - 4	970 - 2	10 >= - 2	3(8)
Vuforia	4/5 - 3	180 - 6	10 >= - 2	4(11)
MAXST	3.25/5 - 5	450 - 3	7 >= - 1	5(10)
Easy AR	2/5 - 6	400 - 4	10 >= - 2	612)

Figure 8. Comparison table of the tested frameworks [2]

However, as Unity has quite an unfavorable license for start-ups, it was decided with the Remato developer team to use React Native with Viro Media SDK as the AR library instead.

4.1.1 Viro Media

Viro is a platform that provides developers with the tools to create VR or AR applications [3]. For AR, it provides a set of virtual objects (called ViroObjects) to be used and places

[18] https://developer.android.com/studio
[19] https://developer.apple.com/xcode/
[20] https://invisible.toys/best-augmented-reality-sdk/

them in an augmentation session using ARCore for Android devices and ARKit for iOS devices. Viro is working as a middleman between the ARCore and ARKit so that the functionnality could be written for both platforms fairly easily. There are some platform-specific functionalities available. However, to have similar flows for both platforms, the provided functions will not be used.

4.1.2 React Native

React Native is a development framework that allows developers to develop for many different platforms through React components[21]. React components are wrappers around each platform's native components, making the development of individual platforms easier because learning each platform's individual components is not needed. React Native was chosen as the native part of the current Remato application is already using it, making it easier to merge the existing application and RematoAR. Additionally, using a framework that Remato already uses makes it easier for other developers to adapt to it.

4.2 Design approach

As AR applications for smartphones are highly visual, it is necessary to design them correctly without losing much of the immersion. When designing the application flow, the suggestions and guidelines made by the ARCore and ARKit teams were followed. According to Apple’s Human Interface Guidelines, it is necessary to display as much of the physical world and virtual objects as possible when designing an AR application[22]. In the made designs, the available tools and settings are hidden in the corners, and only when clicked they would be expanded. By keeping the controls in the corners, most of the screen space would be used for the camera feed. I hoped to work with the designers of Remato on the designs of the AR application. However, as Remato was entering a new market during the development of the RematoAR, I had to make the designs and figure out the best user experience myself.

The important controls will be placed to the bottom part of the screen. According to an article written by Steve Hoober, most users use their thumbs to navigate around the applications [4]. Considering the limited reach of the thumb, most of the navigation should be placed on the bottom as that is the most comfortable position to be reached [4].

[21] https://reactnative.dev/
[22] https://developer.apple.com/design/human-interface-guidelines/ios/system-capabilities/augmented-reality/

Although the article is more than ten years old, the ideas it puts forward are still relevant, and with the smartphone displays getting larger in recent years, the problem is more apparent than before (Figure 9).

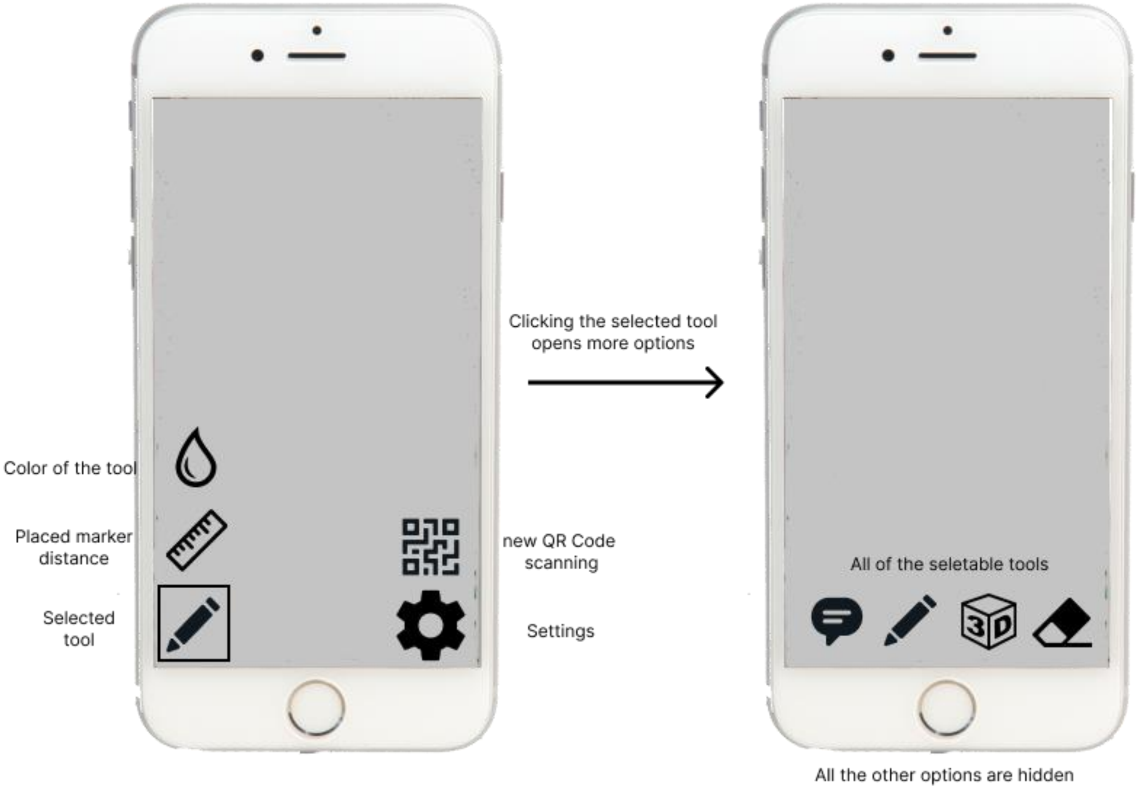

Figure 9. The designed user interface

Most of the used icons will be taken from Remato's current icon collection. Missing icons will be taken from UXWing[23], a website providing a variety of free-to-use icons. In addition, some special icons that cannot be found from the available free libraries will be made.

4.3 Application flow

To have the application work as intended, it is good to have the flow of the application designed beforehand. Having a specific flow assures us that the different steps and situations within the application are handled and thought of. For example, in Figure 10 it can be seen that at the start of the application, the ground plane needs to be determined before doing anything considering the AR session. After finding the ground plane, the user is allowed to scan the QR code of the application. If the scanned QR code does not contain any virtual information, the user is asked to create a new data object (Req.1). If the user declines, the

[23] https://uxwing.com/

application will continue searching for the next QR code from the camera feed until a new one is found. However, if the user creates a new data object, they will be sent to the next state of the application. If this is the user's first time running the application, they will be shown a simple onboarding message (Req.3). This is done to explain the meaning behind different icons in the view before letting them use them.

During the AR session, the users can receive and initialize video calls with the project manager (Req.4). If the project manager declines the call, both users will continue using the application as normal. If the callee accepts the call, the callee's AR session will be ended, and the caller's screen will be shared with them. The participants of the call can talk to each other, and both of them can place virtual objects into the caller's AR session (Req.5). After the call ends, the caller's AR session will continue as normal, but the callee's AR session must be started again as all of their local AR sessions tracking information is lost during the video call.

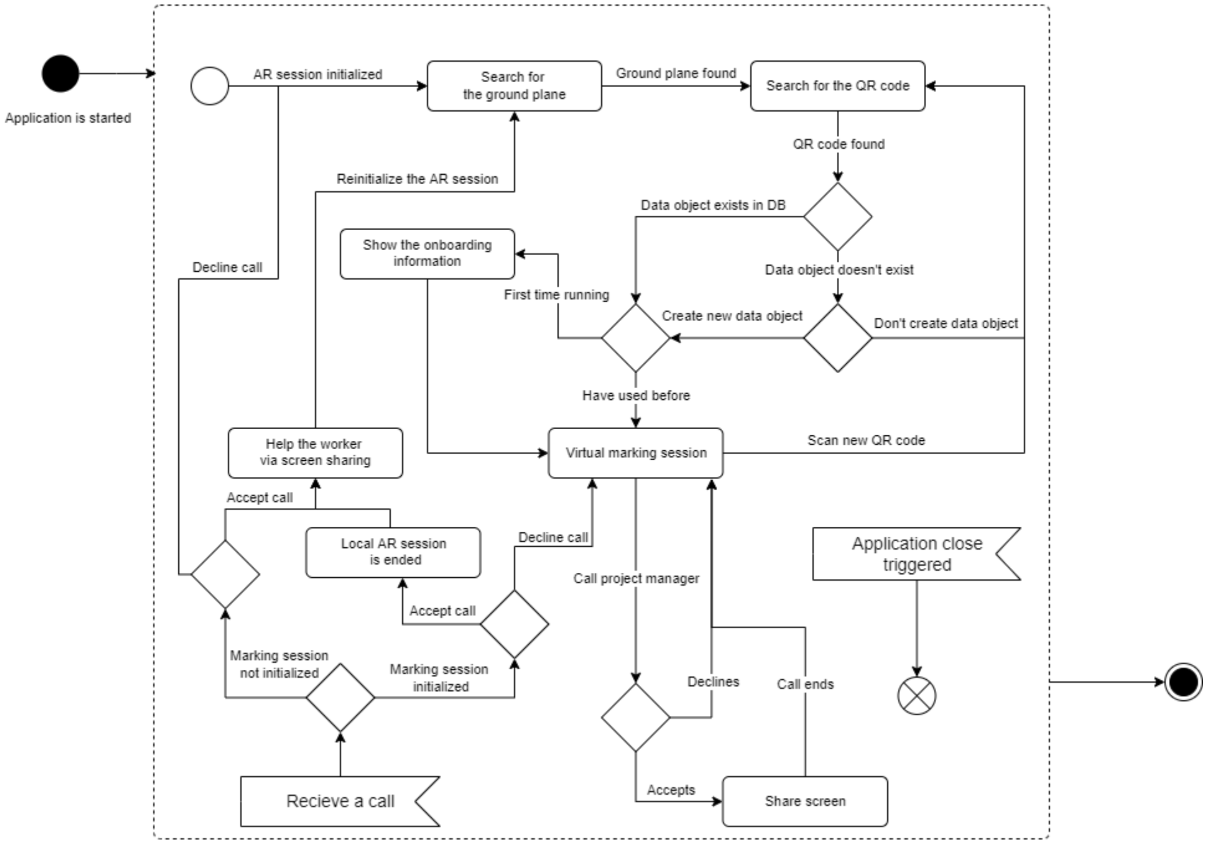

Figure 10. Flowchart of the application

5 Implementation

To make RematoAR work, along with the application additionally a backend service needed to be created. The backend server will contain information about the added virtual markers so that anyone could have access to the marked information, while the AR application mainly handles the displaying of this information.

5.1 Backend service

For storing the virtual information, a simple NodeJS server with a PostgreSQL database is used. The database consists of 4 types of data objects: rooms, texts, polylines, and objects. All of the data objects, along with their general type-specific information, hold information about their position and rotation. The room object does not have any information about its position but is tied to a specific QR code and is a parent object of the other objects (Figure 11). All of the information about the virtual objects is received through the room objects.

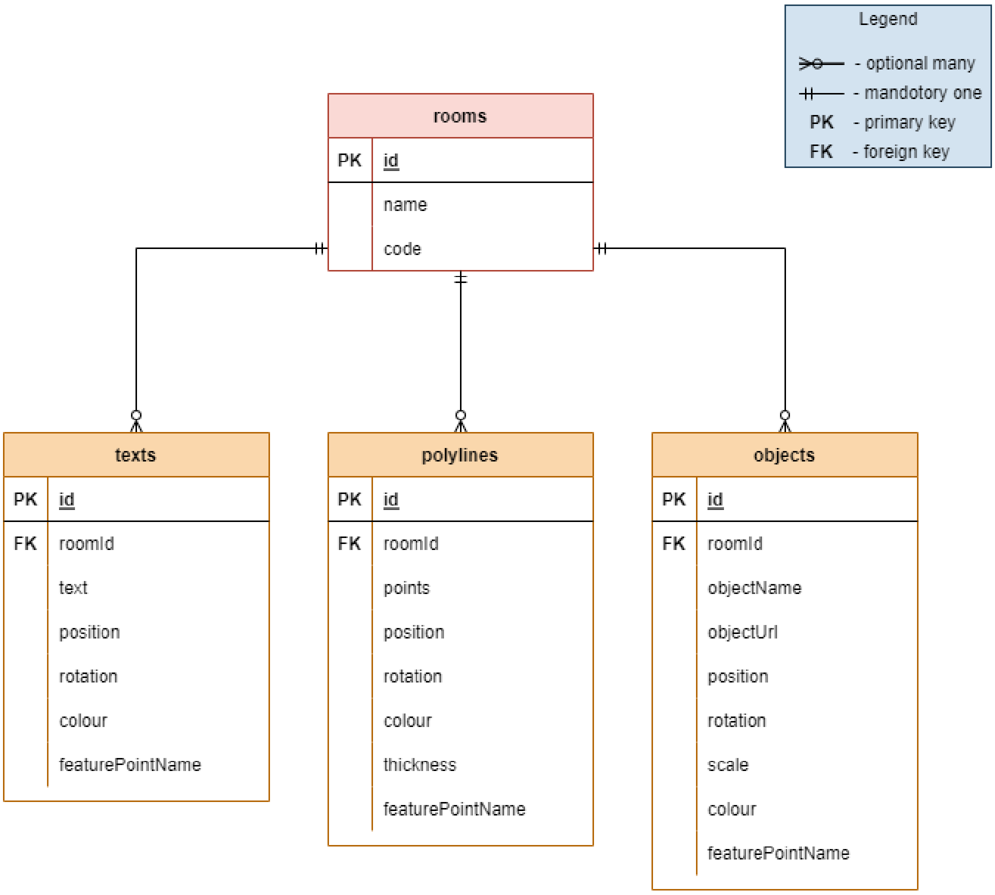

Figure 11. Relations of the data types in the backend

Position, scale and rotation columns of all of the data objects hold the corresponding number arrays in the jsonb format. FeaturePointName is the reference to the feature snapshots that are used by feature detection to try and position the objects more accurately. The colour is a simple string that corresponds to the specific-coloured materials that are assigned to the virtual objects. Points field stores the points that create the polyline in jsonb format and thickness is a number that corresponds to a Viro thickness unit. 3D objects have two ways to be defined, by passing in an object URL, which will take the 3D object from the web, or by using the existing sample objects, which are defined by using the objectName value.

Besides holding the inserted virtual information, the backend also handles the calculations for the feature detection through OpenCV. OpenCV is an open-source library providing the developers with the necessary functions to perform real-time computer vision[24]. For feature detection, the ORB method (Oriented FAST and Rotated BRIEF), a combination of FAST keypoint detector and BRIEF descriptor for rotations is used [5].

5.2 Augmentation Process

As Viro SDK is used to create the AR experience, some core functionalities and features are already coming with it, like the ability to place texts, 3D objects, polylines, portals, and other virtual objects [3]. With the existing features, developers can create a basic AR application with ease. However, for RematoAR purposes a bit stricter interaction and placing of the virtual objects needed to be implemented.

5.2.1 Implemented virtual objects

From the variety of available virtual objects offered by Viro, only three of them are used to create the application. The chosen virtual objects are text, polyline, and 3D objects. These are the three most basic elements that can be easily combined to create more advanced markings (Req.1) (Figure 12).

In the current iteration of the application, these objects have been implemented in their simplest forms, meaning that the tools that the users can use are currently not combined with any of the other

Figure 12. Three basic elements: text, polyline and 3D object

[24] https://opencv.org/

objects and are just plain Viro objects. This is done to keep the first iteration as simple as possible, allowing the users to suggest themselves which combinations should be made, and which tools work the best together.

5.2.2 Placement of the virtual objects

For retrieving and placing data correctly, QR codes are used. Each QR code is assigned to a data set containing all the previously placed virtual information (Req.2). On scanning the QR code, the data is requested from the backend. This data is then parsed into the correct types and presented to the user as virtual objects. Additionally, the QR code position and rotation are used to place the retrieved virtual objects accordingly into the room.

While trying to implement the placement of the objects, a few problematic features of the Viro SDK started to complicate things. Each time the AR application is started, Viro starts an AR session and uses the current rotation and position as the world origin [3]. An easy solution for this would be to reset the AR session at the scan moment and take the QR code as the world origin. Then it would not be needed to calculate the correct rotations and positions for the objects. Unfortunately, while Viro offers session resetting, it is only available for iOS devices. As the focus is to support Android devices, the placement and correct calculations need to be made.

This issue is solved by marking down the QR code's position and rotation on the scan moment, and while saving the data to the database, the virtual objects relative rotations and positions to the QR code are calculated. Once the data is requested from the database, the objects can be placed around the QR code again using QR code’s position and rotation. With this approach, it will not matter how far from the QR code the application is started. The objects would still be placed correctly into the room.

To get the correct rotation and position of the QR code, the existing ViroImageTracker element was used. This basic marker-based augmentation searches for the specified image from the camera feed and places a 3D object according to the matched result [3]. Usually, these images are predefined, but since QR codes are used, a reference image can be generated for the image tracker during run-time. After finding a match from the camera feed, ViroImageTracker places an object into the room. Instead of placing a 3D object, the ViroImageTracker is set to place an empty object so as to not draw any un-necessary objects into the user's view. Once its position and rotation are retrieved, the empty object is deleted along with the ViroImageTracker.

5.2.3 Correct rotation of the objects

When trying to calculate the correct rotations and positions, another Viro issue became apparent. Viro is providing the developers with the possibility to request the camera's position and rotation at any moment, but the camera information is returned as Euler angles. Euler angles are three rotation angles that describe the object's rotation with respect to the defined coordinate system [6]. The Euler angles can have 12 combinations of sequences, but a common XYZ sequence represents the rotation first about the X-axis, followed by a rotation about the Y-axis, and finally about the Z-axis [6]. With these three angles, we can define almost all the possible object's rotations in 3D space. Although Euler angles are easy to read and understand, they have a problem with locking up on certain angles. This phenomenon is called a gimbal lock. Gimbal lock occurs if the alignment of one of the axes matches that of the other, meaning instead of having all three of the axes change the rotation, only two of the axes will affect the rotation [7]. Because of the sequence used by Viro to describe the camera rotation, gimbal locks will start to occur on certain Y-axis rotations. The locking is handled by the Viro automatically, so visually, nothing changes. However, the resulting angles are not predictable anymore and cannot be used to calculate rotations for other objects. To overcome this issue, quaternions and quaternion operations were used to correctly position and rotate the objects.

5.2.4 Object placement onto detected planes

Viro provides the developers with the functionality of detecting planes in the world. According to the ARCore, it does this by looking for clusters of points, and if they have similar properties in relation to the camera, they are determined as being on the plane[25]. This plane is then returned to the user to be interacted with. It works great out of the box. However, there are some edge cases where the detected plane is a bit shorter than reality. If trying to draw something on that plane, some of the information will be placed on the plane, and other points will not, which results in visually broken polylines (Figure 13).

[25] https://developers.google.com/ar/develop/fundamentals

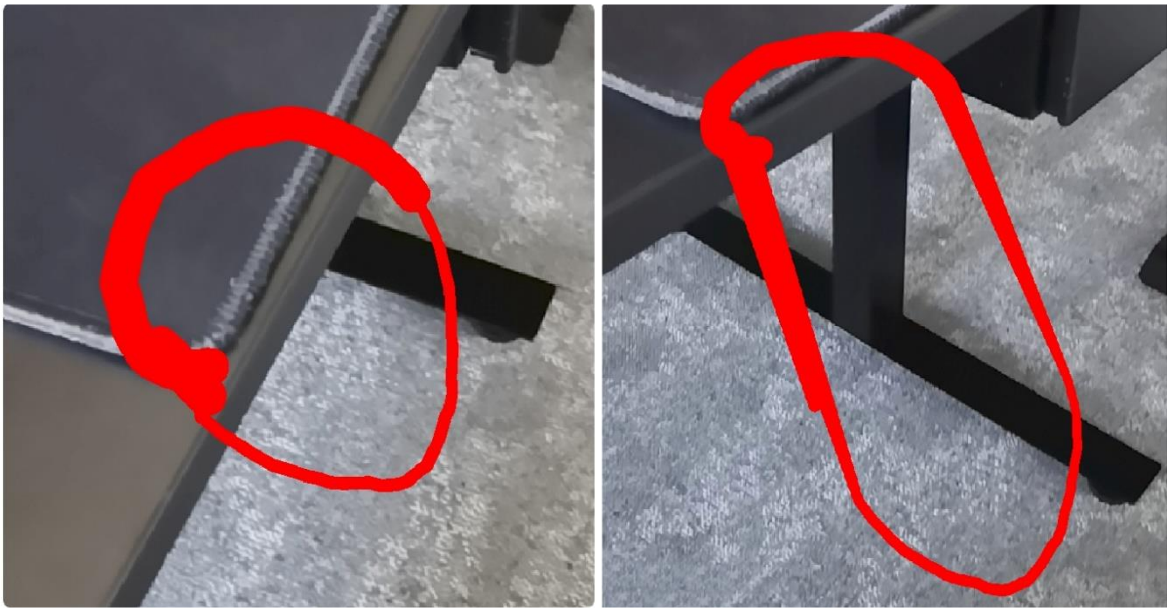

Figure 13. User input (left) and the resulting polyline (right)

To overcome this problem, if the drawing is started on a detected plane, the position and rotation of the plane are taken and formed into a plane formula. Then it is possible calculate the point where the line cast from the camera intersects with the computationally infinite plane. The plane formula is defined like this: $n \cdot (p - a)$, where *n* is a plane normal, *a* is our known point on the plane, and *p* is an unknown point on the plane [7].

By knowing the rotation of the plane, a plane's normal vector can be easily calculated by finding the quaternion and rotating the original world direction vector by the found quaternion. To transform Euler angles into quaternions, the following equation is used (1), where ψ is the rotation angle on the x-axis, ϕ is the angle on the y-axis, and θ is the angle on the z-axis [6].

$$Q = \begin{pmatrix} \cos\left(\frac{\psi}{2}\right)\cos\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right) + sin\left(\frac{\psi}{2}\right)sin\left(\frac{\phi}{2}\right)sin\left(\frac{\theta}{2}\right) \\ sin\left(\frac{\psi}{2}\right)\cos\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right) - \cos\left(\frac{\psi}{2}\right)sin\left(\frac{\phi}{2}\right)sin\left(\frac{\theta}{2}\right) \\ \cos\left(\frac{\psi}{2}\right)sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right) + sin\left(\frac{\psi}{2}\right)\cos\left(\frac{\phi}{2}\right)sin\left(\frac{\theta}{2}\right) \\ \cos\left(\frac{\psi}{2}\right)\cos\left(\frac{\phi}{2}\right)sin\left(\frac{\theta}{2}\right) - sin\left(\frac{\psi}{2}\right)sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right) \end{pmatrix} \tag{1}$$

To find the casted line, the projected point and the camera position are used to calculate the projected points normal vector. Combining the projected point and the projected points normal vector results in the line formula: $p_l + t \cdot n_l = 0$, where p_l is the known point on the line, n_l is the normal vector of the line and *t* is a multiplier to find the possible points on the line.

Replacing the unknown point of the plane formula with the line formula will give the following equation: $n \cdot ((p_l + t \cdot n_l) - a) = 0$. By solving this equation, a value t is found that, if applied to the original line formula, will give the point where the line intersects with the plane. The result in Figure 14 is obtained using the described calculations.

Figure 14. User input (left) and the resulting polyline (right)

5.2.5 Correction via feature detection

To detect motion and track the device's position, AR frameworks usually use inertial measurement unit (IMU) sensors and simultaneous localization and mapping (SLAM) algorithms[26]. However, the more the user moves from the originally detected position, the larger the error becomes in calculating the correct position of the device. This is caused by the IMU sensor's inaccuracies and the nature of SLAM algorithms. According to a study made in 2017, where they compared different SLAM algorithms, it was proven that even on basic human movements, the positioning errors could be quite large (several meters) [8]. To compensate for these errors on a smaller scale, a simple correction system through feature detection was implemented.

As the smartphone calculates and preserves AR session-related processes, which are computationally demanding, the feature detection is handled by the backend. A feature detection request is made to the API if the marking is visible in the camera feed and is near the user. These conditions were added as otherwise, the device would make a lot of

[26] https://developers.google.com/ar/develop/fundamentals

unnecessary requests to the API and most probably would receive many false-positive results.

The feature that the API is supposed to be searching for is determined when placing the object into the virtual world. At that moment, a snapshot of the scene is taken, and an area around the touched point is cropped out. The result is saved, and a relation between the feature and the snapshot is made.

When the requesting conditions have been met, the object's id with the current camera feed and the position in screen space are sent to the API. Using these parameters, the API will try to find the matching features from the camera feed. The matching features are used to find the perspective in the current feed and transformed to receive an outline of the result (Figure 15). The outline of the warped result is received, and the centre point of the quadrilateral is found through the diagonals. If the difference between the current screen position and the found position is large enough, the point is returned and a new position for the object is calculated.

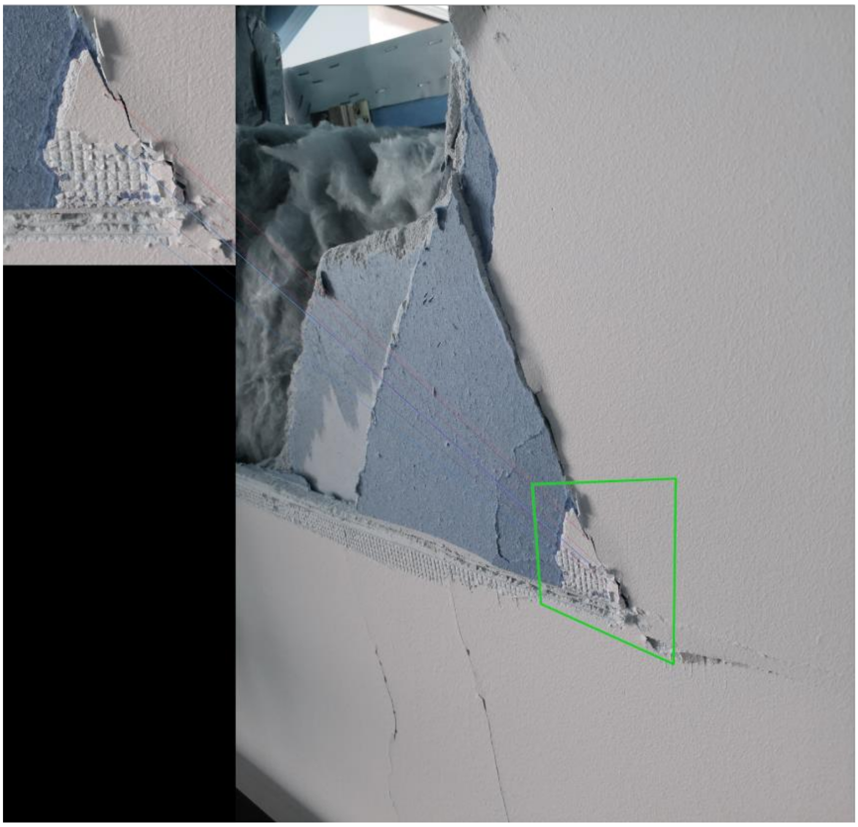

Figure 15. Feature matching result, outline of the match is in color green

If the application required only two coordinates to place the objects, the current approach would be sufficient enough. However, as we work in the 3D coordinate system, it is unknown which coordinates the received values should be manipulating. This is solved by taking the object's position and rotation and calculating a plane formula out of it. Then using the similar approach used in the plane placing chapter, a new position on this plane is found and assigned to the object.

5.3 Manipulation through video

The video call feature is done using the WebRTC API. The WebRTC API enables the web application to initialize real-time communication and transfer data between the peers [9]. The initial peering is done through a signalling protocol called STUN (Session Traversal Utilities for NAT) [9]. This protocol is then used to connect the public IP addresses of the peers to create connections between them [9]. For it to work with React Native, react-native-webrtc is used, providing the user with most of the features WebRTC has. However, as it does not have all of the features provided by WebRTC, it caused some complications when trying to build the screen sharing of the video feed.

The initial idea was to share the whole application screen with the other person, as WebRTC has this functionality. However, after implementing the basic calling feature and screen sharing using the existing screen sharing functionality, the application's performance suffered quite a bit. Moreover, although the captured screen was sent successfully to the other device, its framerate was quite bad. The video feed came in at about one frame per second, making it difficult to understand and interact with. Because of the poor performance of the whole screen sharing, it was decided to try and send the Viro rendered image - containing the camera feed and virtual information - through a data transfer channel instead.

For sending the rendered image, a simple data transferring channel is initialized within the communication session. Viro provides the developers with a simple feature to take screenshots that capture the current camera feed with virtual objects [3]. The function results were compressed and sent over to the other device. The compression is needed to lower the initial file size to send it as one message, making the sending of the images a bit faster between the peers. The received information is fitted and displayed on the receiving device (Figure 16).

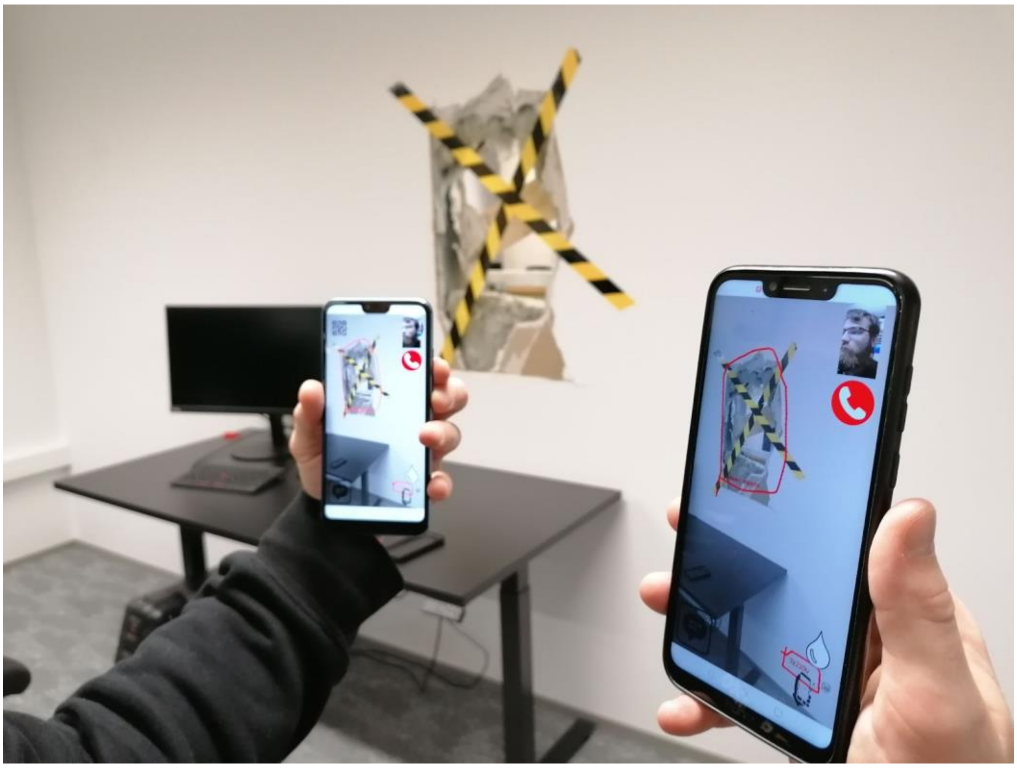

Figure 16. Left device is screen sharing, right device is the receiver of the screen share

While decoding and loading the next image to the native component, it tended to flicker a bit because of the constant loading and unloading of the displayed images. To remove the flickering, two image elements are used, and while the new image is loading, the previous frame is shown. Although this approach is not as effective as a properly working video feed or screen sharing on the desktop, it is much better than the screen sharing provided with the react-native-webrtc package. Using this method, the average frame per second is about 4-5, which is not great but can give enough information to understand what is going on. The slowness of this approach comes from the Viro screenshot-taking method that is used. Testing it showed that it takes approximately 200ms to get the image. A way to improve it would be to create a video feed listener, but for this, the native code of Viro needs to be manipulated, as currently this functionality is not provided. Additionally, the creation of a custom WebRTC MediaStream object is needed. This is because simple data transferring is not as optimal as MediaStream, which uses proper encoders and buffering to receive the smoothest video feed possible [9]. Currently, the react-native-webrtc package does not provide the developers with such features, so this is something that must be implemented as well.

For sending the input from the remote device the same data transferring channel is used (Req.5). While the device sharing the screen sends information about the video, the other device could send back any executed input. The remote device is given the same tools and user interface as the screen sharing device to have a similar marking flow to the local AR session. To determine where the information provided by the remote device needs to be placed, the touched screen point is taken and compared to the camera feed. The difference between the shared screen and the current screen size is found, and the corresponding screen coordinates are calculated out of it. These coordinates, along with the input type and other related information, are then sent to the device with the AR session. After the AR session receives the remote input data, it executes the received input the same way the local input would execute it: 1) get the world point from the screen coordinates and 2) handle the Viro object placement using the world point.

5.4 Implemented UX

When creating the user experience, the nature of the AR applications were considered and most of the buttons were kept in the corners of the screen to have as much free space for the camera feed as possible. The designs were implemented into the application by mostly following the previously made designs. However, after testing the application, some changes to the layout were made (Figure 17).

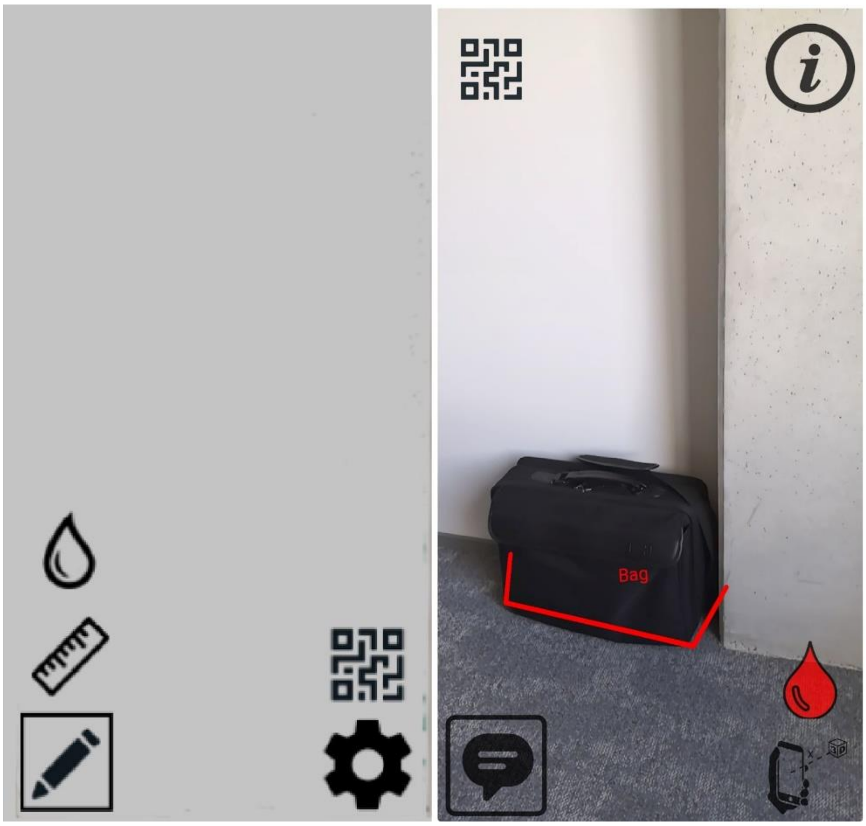

Figure 17. The comparison of the designed and implemented UI

In the design chapter, it was said that it is not good to place buttons at the top of the screen as it is not as comfortable to reach. However, in the implemented designs, some of infrequently used buttons were placed at the top. This change was made as the original placement of the icons started to take up too much height and blocked the visual information. Most of the activities and object placement happens at the centre of the screen rather than on the top, so it made sense to move infrequently used buttons to the top.

5.5 Timeline

This chapter gives an overview of the timeline about the work and research made during the thesis. The timeline table shows months with the specific tasks and most notable achievements made during that month. The timeline table does not include the hours spent writing the thesis, as this is not part of the implementation. For each of the tasks, an estimation of the spent hours is given.

In September of 2021, the initial meeting with the product manager was made to discuss the idea of using augmented reality to visualize tasks for the workers. From this, an initial document was made with the requirements of the application. Research about the available AR libraries was made and presented to the developers' team, after which the technologies were agreed on.

The full development of the application started in January of 2022, with the first feature developed being the placing of the basic objects. After this, the reproduction of the visual information through the QR code was integrated into the system. A plane detection and correct placement onto the detected planes were made after that. To improve the overall accuracy of the application, feature detection using OpenCV was implemented. The video call feature was one of the last things to be implemented, as the other parts of the application needed to be ready before it.

The month of April focused on the tests and implementing some of the suggestions made by the testers.

Month	**Description of the tasks done**
September 2021	The initial idea is formulated with the project manager and documented (3h) Initial set of similar applications are researched (10h)
October 2021	Research of the available frameworks (30h)

November 2021	Continued research of the available frameworks (28h) Presenting the results and choosing a framework (2h)
December 2021	Setup of the development environments and the libraries used (20h)
January 2021	The basic object placement is implemented (50h) Placement of the objects according to the QR code (45h)
February 2021	Correct rotation of the objects according to the QR code (50h) Placing objects correctly on the detected plane (55h)
March 2021	Implementation of the feature detection (80h) Started work on the video call feature (40h)
April 2021	Finished the video call feature (60h) Creating a testing flow and testing the application on companies (18h) Implementing suggestions (24h)

6 Evaluation

To evaluate how well the created application accomplishes the set goals and how easy the application is to use, usability testing was conducted [10]. The application was tested on four construction companies. Evaluating the ease of use was done by having the testers follow a basic todo-list with various tasks. To see how and what a construction worker would mark down using the application, an unrestricted marking session was conducted.

6.1 User testing flow

When conducting the tests, a simple flow for each of the visited companies was followed. The core idea of the application and its purpose were given to the participants, which was followed by two forms of testing sessions: restricted and unrestricted testing.

6.1.1 Restricted testing

To test the ease of use, the testers were given a simple task list in which they would be marking three things in the room. No explanation on how to accomplish any of these steps was given at the beginning of the testing sessions. All of the information the tester needs should have been captured or handled by the descriptions in the application (Req.3). Before the testing, a basic setup of the room was made where an empty QR code was placed on the wall and a computer bag underneath it. The bag was added to have the user mark something specific with reference to the world. The basic flow of the testing was planned as the following:

- The user opens the application and is greeted with a request to point the phone towards the ground plane. As a result of this, the AR session should be successfully initialized;
- The user is asked to scan a QR code on the wall, after which he is informed that the data for this QR code does not exist yet;
- The user fills in the simple form, creating a data set of the current task;
- A descriptive overlay is shown to the user that prevents the user from interacting with the world before understanding the basic meaning of each UI element;
- The user must mark down a red coloured circle around the bag with a simple text that instructs to remove the bag;
- The user places a virtual 3D object to the room to plan the object placements out;

- The user tries to draw a doorway onto the wall of the room, showing where the door should be installed in the room;
- The user closes the application and reopens it, confirming that the information they inserted in the previous session was indeed captured and placed correctly in the new AR session;

Although the markings the users were doing in the test flow were not real cases, it still demonstrated the application's capabilities. The application provides the user with different marking tools, and how these tools are used depends a lot on the specifics of the company itself.

6.1.2 Unrestricted testing

After the initial testing, the users were given time to try things out on their own to find the nuances that would work in their company's workflow. This phase was for finding the missing functionalities of the application and to really focus on things that would benefit the construction industry. As the construction industry is big, it is impossible to satisfy everyone involved. However, trying to find similar ideas will help to improve the application in the future.

From these sessions, the missing features and needed improvements of the application would come out. Without these clear specifications and requests from the end-user, we will eventually be drifting away from the core functionality and will start focusing on the things that are not so beneficial. There are different ways to interpret things, and without testing and discussing the ideas at the right time, the actually important features cannot be found.

6.2 User test results

Following the defined testing flow, the tests were conducted with the project managers of each company. After the testing session, the participants filled in the survey and gave feedback about the application.

6.2.1 User background

To have an overview of the participants' current flow of describing and assigning tasks, and to capture the main field they work at, the first part of the survey was about their background. By chance, each surveyed company focused on slightly different aspects of the construction industry. This was beneficial for getting different opinions and views about the different aspects of the construction field.

Out of the participants, only one of the four testers had previously seen or heard about augmented reality, and only one of them had used a marking software of any sort in their work. The current flow is usually through paper and pen. One of the companies, which focuses on apartment repairs and other interior-related work, marks their work down on an apartment plan together with the client, and from this, the task list is formulated.

The common task assigning and describing approach among all of the questioned companies was that the site manager is usually the person the project manager talks to, and the site manager then shows and describes to the workers what is needed to be done. Site manager is a person who oversees the work done on the field and is usually there with workers, whereas a project manager is there only at the planning time or at the end of the project to inspect the completed work. Only one of the companies, whose focus was flooring, did not have a list of tasks as the work is quite repetitive in all of the sites they work at. The steps to creating a concrete floor are similar in every construction site. Thus, the tasks do not need much description if the team has done it once already.

6.2.2 Marking tasks

To evaluate the current state of the application and to find the shortcomings of the current solution, the participants were asked about the task marking process: the accuracy of the placed virtual information, the usability of the presented marking tools, and the difficulty of placing the markers.

6.2.2.1 Accuracy of the placed information

The testing showed that the placing of the information after the initial sessions was rather accurate. In some cases where the phone lost the anchor positions, the results tended to drift, but around the area of the QR code, the information was where it was placed. When asked to rate the accuracy of the placed information from 1 (“really inaccurate”) to 6 (“really accurate”), the answers were more on the positive side (Figure 18).

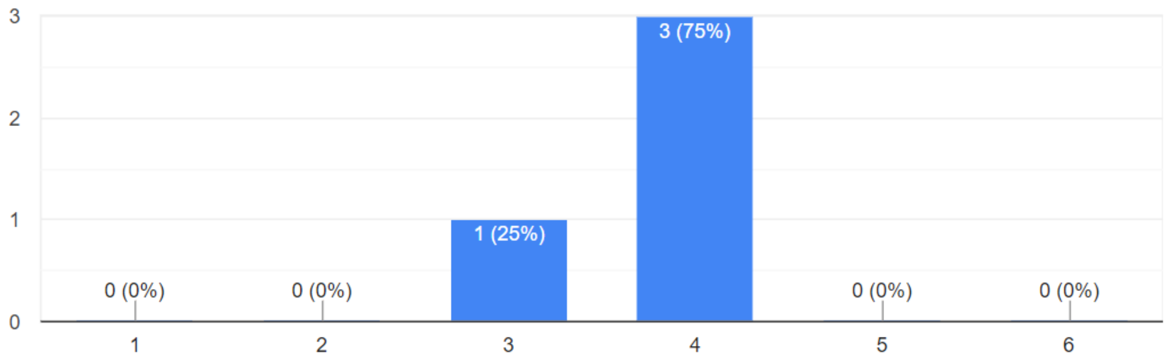

Figure 18. The tracking accuracy

One of the participants rated the accuracy with a rating of 3. During his testing session, the QR code's rotation was determined wrongly, causing all of the placed information to be rotated into the wrong positions. After rescanning the QR code, the application was able to position the objects correctly, but since the initial scanning showed wrong results, it caused a bit of concern in the overall accuracy of the application. The project manager knows the correct positions of the marking and knows when the QR scan was not successful, but a random worker with no prior knowledge of the site might use the wrongly placed information as a reference, causing them to work or do something that does not need doing. To prevent this from happening, some improvements to the rotation determination are needed, and some virtual control elements should be placed that the user can see near the QR code after the scanning. If the control elements are not in the places the application is hinting them to be, then the user knows that the position determined through the QR code is wrong and can re-scan the code.

6.2.2.2 Marking tools

From the tests, it was concluded that the presented tools were easy enough to use to mark the information. From 1 (“really difficult”) to 6 (“really easy”) most of the users answered with a rating of 5 - the tool did what they expected it to do (Figure 19).

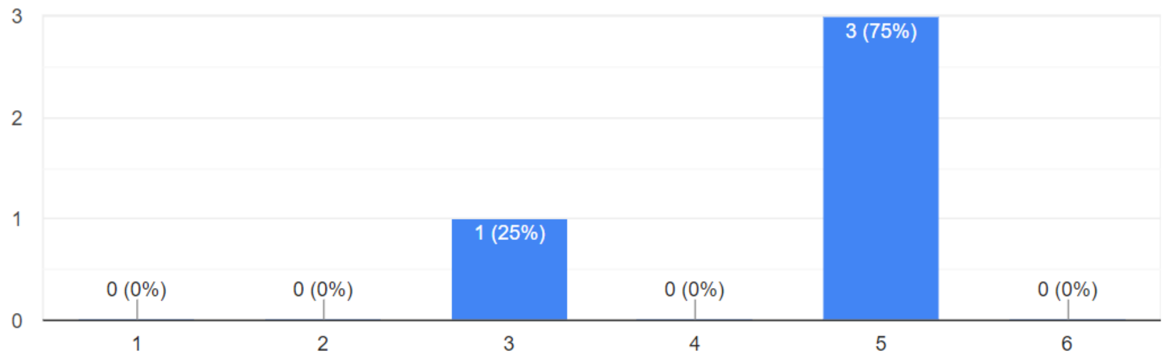

Figure 19. The ease of marking the information with the provided tools

One tester rated the ease of marking with a rating of 3. In his case, the lower rating was not caused by the difficulty of marking the information but rather by the used tools. He felt that the pencil tool, although providing with the freedom of drawing anything, will cause too much human error when trying to draw, for example, straight lines. He suggested that there should be more predefined shapes and some additional tools in the application. These same suggestions were made by the other testers as well.

For a question, “Were the provided tools enough to convey the needed information?” except for one of the participants, all of the others thought that there should be more tools to mark the information down. The most suggested tool was predefined shapes. With these shapes, the rectangles and circles would be easier to make, and instead of drawing a rectangle by hand, the users could have simply dragged a rectangle into the scene. Additionally, one tester suggested creating a simple hyperlink tool that could be placed as a marker. Through this hyperlink, some references or more accurate schematics about the task could be shown. Previously mentioned straight-line drawing was one of the suggestions as well.

When considering the provided tools there was not a tool that could be considered the “best” one. Participants felt that the combination of drawing and adding text was the way to go as the text described the task, and the drawn lines showed visually what region or object was meant. None of the participants thought the 3D object placement added as much benefit to the markings as the text and pencil together did. It could be a good tool for visualizing things in some furniture installment or an interior design company, but for a construction company it was not that beneficial.

6.2.3 User interface

As augmented reality applications are not the most traditional form of application, it was good to test the designed flow and to find any shortcomings of it. When asked about the ease of choosing different tools, most of the users concluded that the switching between the tools was relatively simple and straightforward (Figure 20).

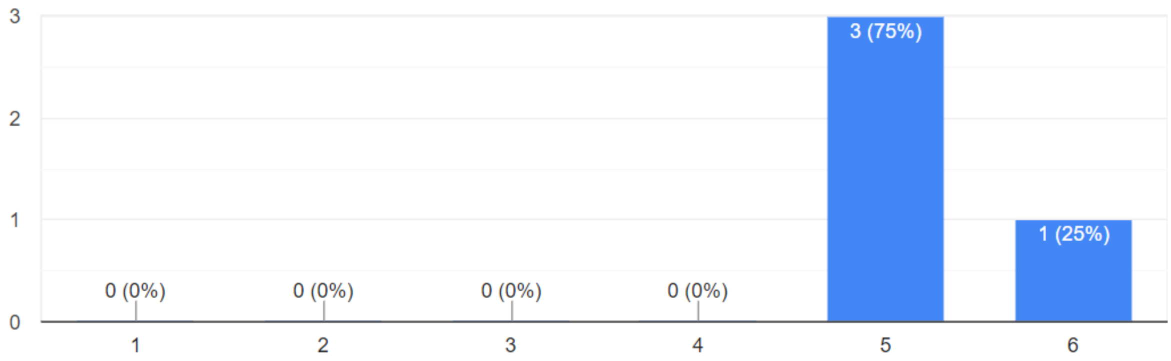

Figure 20. The comfort of switching between the tools.
1 (“really difficult”) to 6 (“really easy”)

As seen by the answers, the tool switching and changing the options of the current tools was quite easy to do. However, the addition of new tools and options might make the current flow a bit overwhelming. Thus, the flow might change in the next iteration of development. The users felt that although the application contained a simple onboarding view, which explained the meaning of the icons and the overall layout to the user, the general explanation on how to mark the items was missing. A simple text instructing the user to tap on the screen to add markings could have potentially solved this issue. One of the users suggested including some simple demo videos in the application. This way, the user could see the application in action instead of figuring things out themselves. All of the users confirmed this by saying that once they got the hang of the application, it was quite easy to use.

6.2.4 General questions

Some general questions about the application were asked to receive more feedback and to determine the potential benefits of using the created application in their workflow.

6.2.4.1 Use of the current solution

When asked how the companies would use the current solution in their work, different use cases were given. Most of the surveyed companies saw the presented solution as a way of validating the completed work. The project manager could mark the tasks in the application at the start of the project, and before calling the client, he could confirm and recheck if and how each of the tasks were completed. Combined with the current Remato task system, the project manager could even see who was responsible for the completed work and could contact them.

One potential use case was that the project manager could mark the information together with the client, giving the client a visual representation of what would be done and creating a visual aid for the workers in the process. For a work that does not require the use of plans or does not have any plans, it would be perfect. However, for anything requiring the use of plans, the current solution would be potential extra work for the project manager.

Most companies saw the current solution as a better communicator between the project manager and object manager, then a way to communicate the work straight to the worker. The main argument was that the average worker is not interested and might not even have the technical knowledge to use smartphones in this capacity. Many of the companies using Remato have said that even the use of Remato's application is too much for some of the workers.

This shows that in order to make this application usable for a wider user base, a redesign of the user experience is needed to make things simpler. Many functionalities and extras are good for an experienced user but can make things rather overwhelming at first. Additional user testing must be conducted to find which parts of the application make things complicated and what would make it easier. Another improvement regarding this is to add a way to tune down the application and limit its functionality so only a project manager could mark the objects and the worker has only the visual cues. This makes the application marginally simpler and allows the worker to use it as a simple visual aid.

6.2.4.2 Video call feature

All of the participants thought that the feature to add markings and explain the tasks remotely over a video call is beneficial, but it is not that necessary. In these cases, a simple phone call is often enough to give the missing information, and if needed, the worker could mark the information down as well.

As the video call feature is not that fluid in the current solution, it was rather unpleasant for the testers. The stuttering image made marking the additional information difficult for the testers.

6.2.5 Additional suggestions

It was suggested by the testers to combine the existing Remato task system with the AR application. This way, the task information could be transferred to the AR application, and the task-related descriptions and photos could be accessed without needing to switch between the two applications. Along with this, an on-site task tracking start and stop functionality could be built into the application, meaning that the worker would not need to open the Remato application at all to start their work tracking. It would work similarly to the mentioned Suprema devices in chapter 2, just without the need for any special devices for the tracking starting.

One of the testers suggested that the marked elements should be shown in the corners of the screen relative to where they are positioned from the user. Even if the user is not facing any of the elements, the visual aid in the corners of the screen will hint to the user that there is something to be found from the room. For example, a marking made about ceiling mouldings would be missed in the current solution as the user is usually not looking upwards when using the application. Additionally, it would save time it would take workers to find the markings as they would not need to look around to find them, instead the available markings would be hinted to the worker through visual hints. A similar solution was suggested by another company. They found that a list of the marked elements would be useful as then the worker would have a quick overview of the tasks and could then look for them if needed. The filtering and categorization of the elements could be useful as well as then the worker could hide the completed work or inspect only the information related to their current work.

An interesting idea from one of the testers was that an additional 2D planner application or a feature could be made to accompany the AR application. In this planner, the project manager could use the existing construction site plans or use the measurements made on the site to create a more accurate representation of the data and the tasks. Similar marking tools like texts, lines, and 3D objects could be used in this 2D planner. On the 2D plan, the QR code spot would be marked with the reference distances and height from the ground. After the planning, the object manager would place the QR code onto the construction site according to the specified measurements, and then the marked information could be used similar to the current solution. The only difference being that the task information was marked in the planner instead of marking the information on-site. Another benefit of this approach would be that the marked information could be improved later through the accompanying planner. The project manager could do quick markings on the construction site and later improve upon the made markings to have the workers use more refined information.

One of the participants felt that this solution could be suitable for main contractors. While Remato's target users are subcontractors who have workers with a need to track their work time and assets, the main contractors do not have this need. If this AR application could be used to create tasks into Remato, which the sub-contractors could later use then, it would potentially be something that the main contractors would like. The AR application could be used by the main contractors to validate the subcontractors' work and mark any additional work needed to be done. Sub-contractors usually are not interested in looking at pictures and protocols themselves as well, if they could receive this information through the application, it would potentially make their work faster. To add this feature and to have the interest of the main contractors, Remato core software would need a lot more work as the current modules are too subcontractor focused.

6.3 Future improvements

All of the suggestions made by the testers will be taken into consideration when continuing with the development. Features like the control elements around the QR code for confirming the correct initialization and showing the direction of existing markings in the corners of the screen are features that seem obvious now after the testing sessions and are something that this sort of application needs. Additionally, the predefined shapes and the improvements to the onboarding of the application will be done. Onboarding will be focused on as any confusion caused within the application will probably make the company not use this application. It is also impossible to introduce everyone to this sort of application in person.
Whether the extra features like a 2D planner will be developed must be decided by Remato's product team. To convince them of the need for this 2D planner, additional surveys must be done to figure out if this is something that would benefit other companies as well. As the Remato already contains a basic blueprint editor and marking, it should not be that difficult to include.

The video call feature will also be improved upon in the future. As described in the video call chapter, the current screen sharing provided by the react-native-webrtc library is even worse than the implemented solution. There are two ways this could be solved. 1) by manipulating the ViroMedia source files, which is doable as it is open-source, and inserting a listener to the scene renderer, or 2) wait for the react-native-webrtc screen sharing improvements. Working with ViroMedia source files would be preferred as this way, something could be given back to the Viro Community. Additionally, the views received from the ViroRenderer would be cleaner as they would only contain the camera feed and the rendered virtual objects.

From the suggested improvements, the basic line creation and the video tutorials have already been implemented (Figure 21).

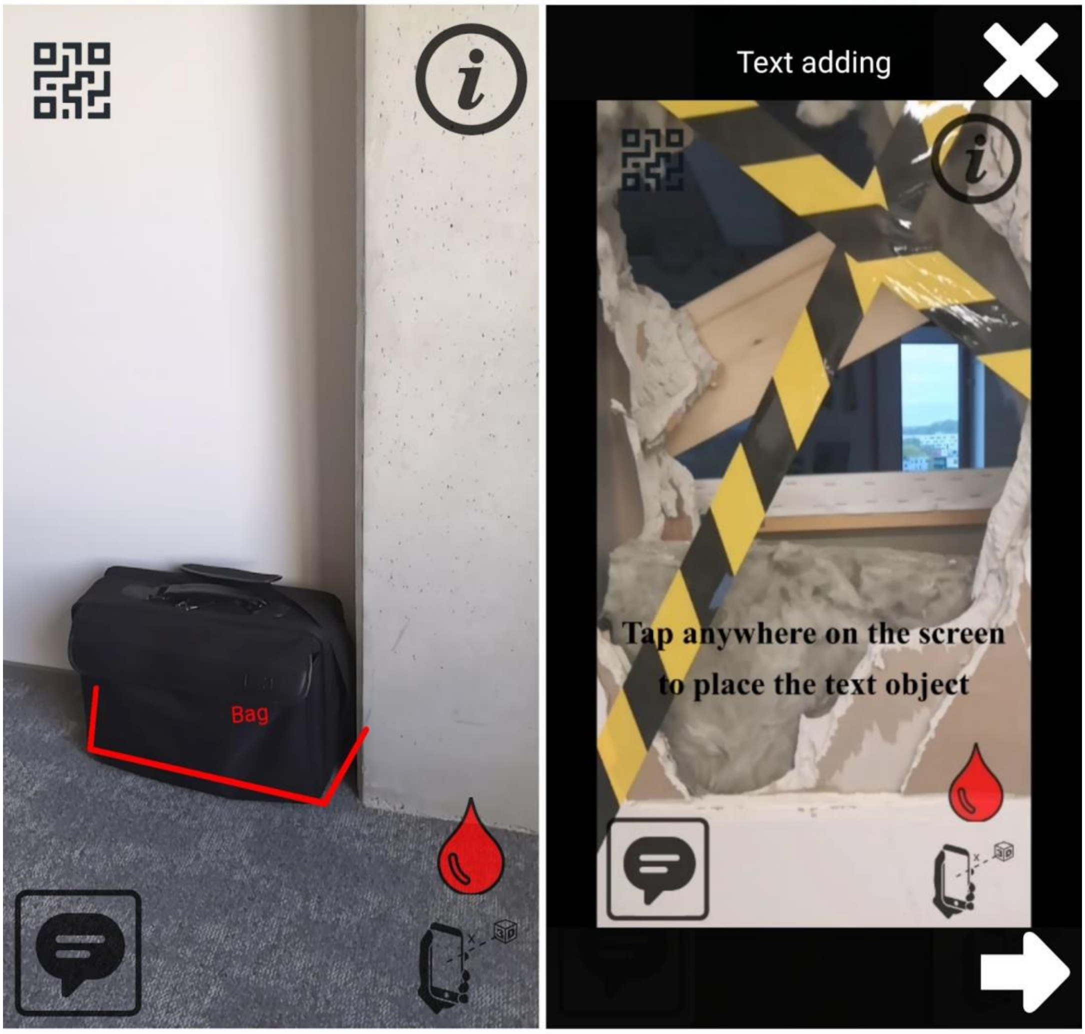

Figure 21. Implemented line adding (left) and the video tutorials (right)

These were implemented as with the basic line creation users can start creating different shapes easily before the actual implementation of predefined shapes. Video tutorials were implemented to onboard the next testers and users. The other improvements required a lot more work and were not done in the scope of the current thesis. Overall, the testing was a success as a lot of great feedback, and ideas were received, and a list of needed improvements was formulated.

7 Conclusion

During the thesis, a RematoAR application was made to improve the communication between the project manager and worker by providing the workers with virtual tools to leave virtual markers at the construction site. The added virtual information is accessible to the worker by scanning the QR code at the construction site. The QR code holds the virtual information and is used as a reference to place the markers correctly into the room.

For the application to work, a backend service with the application itself was made. The implemented features of the application are the placement of the basic 3D elements such as texts, polylines, and 3D objects, the ability to receive the previously made room information through the backend API, and the ability to initialize a video call. The video call feature was implemented using the WebRTC library. Using the video call, the worker can share their current view of the AR session with the project manager and solve any confusion or inconsistencies regarding the virtual information. A remote input handler was implemented, which takes the screen input from the callee and allows them to remotely insert information into the caller's AR session. This information will be saved, and the added information can be used by the others.

To determine the usability of the RematoAR and find the improvements the application needs, usability tests were conducted. There was a total of four testers from various construction companies. The testers were asked to rate the main features of the application, comprehension of the user interface, and the ease of use. Many new features and ideas for the application were suggested, and two of the suggestions were implemented. In the end, it was concluded that the created application needs additional features, better tracking of the markings, and improvements to the user experience. The user experience needs improvements as it was said that the average construction worker is not so tech-savvy. Most of testers suggested to implement an easier version of the application specifically for the workers who are quite unexperienced with using their smartphones. Overall, the testers liked the idea of the application and were interested in trying to implement the created application in their workflow. If not as a tool to communicate the tasks to the worker, then as a tool to communicate information between the project manager and the object manager. Additionally, many of the testers saw the potential in the application to be used to evaluate the work that was made.

All the proposed suggestions received from the testers will be considered in the next development iteration, but if and in what capacity the application will be implemented depends on the Remato product team. Potentially another testing session with some implemented improvements is needed before the full commitment to the RematoAR development is made. However, as the feedback received from the testers was quite positive and supportive of the idea, with some improvements to the flow of the application and accuracy, I am quite certain that RematoAR will be implemented in some form. If not as a part of the Remato main application then at least as a supporting application, providing the interested users with a unique way of describing tasks.

8 References

[1] D. Schmalstieg and T. Höllerer, Augmented reality: principles and practice. Boston: Addison-Wesley, 2016.

[2] O. Saar, 'Survey of Existing Augmented Reality Frameworks'. Accessed: Mar. 26, 2022. [Online]. Available: https://courses.cs.ut.ee/MTAT.08.024/2021_fall/uploads/Main/OttSaar-FinalReport.pdf

[3] 'Viro Community API Documentation', Viro Community. https://viro-community.readme.io/docs (accessed May 08, 2022).

[4] S. Hoober, 'How Do Users Really Hold Mobile Devices? :: UXmatters'. https://www.uxmatters.com/mt/archives/2013/02/how-do-users-really-hold-mobile-devices.php (accessed Mar. 21, 2022).

[5] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, 'ORB: An efficient alternative to SIFT or SURF', in 2011 International Conference on Computer Vision, Barcelona, Spain, Nov. 2011, pp. 2564–2571. doi: 10.1109/ICCV.2011.6126544.

[6] R. Mukundan, Advanced methods in computer graphics: with examples in OpenGL. London Heidelberg: Springer, 2012.

[7] S. Marschner and P. Shirley, Fundamentals of computer graphics, Fourth edition. Boca Raton: CRC Press, Taylor & Francis Group, 2016.

[8] D. J. J. LaViola, B. M. Williamson, D. R. Sottilare, and P. Garrity, 'Analyzing SLAM Algorithm Performance for Tracking in Augmented Reality Systems', 2017, pp. 8–9.

[9] L. Salvatore, Real-Time Communication with WebRTC, First Edition. Beijing: O'Reilly, 2014.

[10] J. Nielsen, 'Usability 101: Introduction to Usability'. https://www.nngroup.com/articles/usability-101-introduction-to-usability/ (accessed Apr. 21, 2022).

Appendix

I. Glossary

Software Development Kit (SDK)	Software development kit is a collection of software development tools used in the creation of an application[27].
Quaternions	Quaternions are hyper-complex numbers of rank 4, that along with other mathematical applications can be used to calculate three-dimensional rotations. It is quite a common way to calculate rotations as it works well with other methods of rotation [6].
NodeJS	Node.js is an open-source back-end JavaScript runtime environment. It allows the developers to write server-side scripts using JavaScript[28].
Building Information Modeling (BIM)	Building information modeling (BIM) is a process that is supported by various tools, technologies and contracts. It involves the generation and management of digital representations of physical places. The main field of use is the construction industry[29].
Usability testing	Usability testing is testing with a goal to find how easy the application is to use. In the usability testing 5 components are viewed: learnability, memorability, efficiency, errors, and satisfaction [10].

[27] https://en.wikipedia.org/wiki/Software_development_kit
[28] https://en.wikipedia.org/wiki/Node.js
[29] https://en.wikipedia.org/wiki/Building_information_modeling

II. Accompanying files

The archive files consist of the following items:

- demo.mp4 – the demo video of the use of the application
- rematoar_survey_original.csv – the question and answers by the testers in Estonian
- rematoar_survey_translated.csv – the question and answers by the testers translated to English

.apk of the build will be available from the repository of the application.

III. Source code

Source code is accessible from the following link: https://github.com/ottsaar/RematoAR.

This is currently a private repo, if access to the code is needed, please write an email to: ott.saar1@gmail.com.

IV. Photos of RematoAR

V. License

Non-exclusive licence to reproduce thesis and make thesis public

I, **Ott Saar**,

(*author's name*)

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to:

   1.1. reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and

   1.2. make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright,

of my thesis

RematoAR – An Augmented Reality Assignment Aid,

(title of thesis)

supervised by Ulrich Norbisrath, Ranno Käomägi

(supervisor's name)

2. I am aware of the fact that the author retains these rights.

3. I certify that granting the non-exclusive licence does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu, **17.05.2022**