Remote Assistance for Repair Tasks Using Augmented Reality

Lu Sun

A thesis submitted in partial fulfillment of the requirements for the
Doctorate in Philosophy degree in Computer Science

uOttawa

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

In the past three decades, using Augmented Reality (AR) in repair tasks has received a growing amount of attention from researchers, because AR provides the users with a more immersive experience than traditional methods, e.g., instructional booklets, and audio, and video content. However, traditional methods are mostly used today, because there are several key challenges to using AR in repair tasks. These challenges include device limitations, object pose tracking, human-computer interaction, and authoring. Fortunately, the research community is investigating these challenges actively.

The vision of this thesis is to move the AR technology towards being widely used in this field. Under this vision, I propose an AR platform for repair tasks and address the challenges of device limitations and authoring. The platform contains a new authoring approach that tracks the real components on the expert's side to monitor her or his operations. The proposed approach gives experts a novel authoring tool to specify 6DoF movements of a component and apply the geometrical and physical constraints in real-time. To address the challenge of device limitations, I present a hybrid remote rendering framework for applications on mobile devices. In my remote rendering approach, I adopt a client-server model, where the server is responsible for rendering high-fidelity models, encoding the rendering results and sending them to the client, while the client renders low-fidelity models and overlays the high-fidelity frames received from the server on its rendering results. With this configuration, we are able to minimize the bandwidth requirements and interaction latency, since only key models are rendered in high-fidelity mode.

I perform a quantitive analysis on the effectiveness of my proposed remote rendering method. Moreover, I conduct a user study on the subjective and objective effects of the remote rendering method on the user experience. The results show that key model fidelity has a significant influence on the objective task difficulty, while interaction latency plays an important role in the subjective task difficulty. The results of the user study show how my method can benefit the users while minimizing resource requirements. By conducting a user study for the AR remote assistance platform, I show that the proposed AR platform outperforms traditional instructional videos and sketching. Through questionnaires provided at the end of the experiment, I found that the proposed AR platform receives higher recommendation than sketching, and, compared to traditional instructional videos, it stands out in terms of instruction clarity, preference, recommendation and confidence of task completion. Moreover, as to the overall user experience, the proposed method has an advantage over the video method.

Acknowledgements

I would like to thank my supervisors Jochen Lang and Hussein Al Osman, for their guidance, insight, and encouragement during the past years. They were always patient and provided invaluable advice that guided me through the mist.

I would like to thank my colleagues in the VIVA lab, for their insight and kindness during the together time.

I would like to thank the members of the committee for their time, effort and advice.

This work was partially supported by Lockheed Martin Canada.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AR	Augmented Reality
BCI	Brain-Computer Interface
CAD	Computer-Aided Design
CMR	Cloud Mobile Rendering
DoF	Degree of Freedom
DoT	Difficulty of Task
GM	Grand Mean
GPS	Global Positioning System
GUI	Graphical User Interface
HCI	Human-Computer Interaction
HHD	Hand-Held Displays
HMD	Head-Mounted Display
LDI	Layered Depth Images
NCR	Number of Corrected Errors
NUR	Number of Uncorrected Errors
ODoT	Objective Difficulty of Task
PC	Personal Computer
QoE	Quality of Experience
SD	Standard Deviation
SDK	Software Development Kit
SDoT	Subjective Difficulty of Task
SURF	Speeded-Up Robust Features
VR	Virtual Reality
XML	Extensible Markup Language

Chapter 1

Introduction

In our increasingly digital world, remote assistance in repair tasks is a novel application. Repair tasks take place when a user needs to repair a broken product, which may involve disassembling the product, replacing a failing component, and assembling it back together again. The capacity to repair products, such as electronic devices and appliances, is important to consumers and the environment. In the US, approximately 20 states are said to have the right to repair legislation in progress. Under the European Commission's new standards, manufacturers will have to make spare parts available to professional repair businesses. Campaigners argue that individual consumers should also be allowed to buy spares and mend their own machines [37]. During the COVID-19 crisis, many repair shops are shutdown. Remote assistance becomes essential for product repair in some regions. Remote assistance can also reduce physical contact during this crisis. However, repairing can be challenging for customers. An example of a repair task is to change a component in a broken mobile phone. A user who has no experience in phone repair must obtain some instructions before repairing it. Otherwise, they may further damage the device.

Currently, traditional methods, e.g., instructional booklets, audio, and video clips, are still mostly used in this scenario [29]. However, there are some difficulties in the traditional methods. First, an instructional booklet or a video tutorial covers only a limited number of assembly and disassembly paths. If a user encounters a situation that the instructional booklet or the video tutorial does not cover, it is difficult to successfully complete the repair task. Especially for electronic products, e.g., mobile phones and computers, due to their high complexity. Second, an instructional booklet or a video tutorial does not provide users with feedback. It is on the users to recognize the errors should they occur. Failure to notice an error or unsafe operation may cause unexpected results. Third, if a user calls an available customer service hotline, the communication between the user and the support

staff is usually performed via voice, and the user can only describe the problem and receive instructions by speech without any visualization. Therefore, the communication can be less than effective.

The emergence of Augmented Reality (AR) provides a promising way to perform remote assistance for repair tasks. AR is an interactive experience that allows the user to see the real world, with virtual objects superimposed upon or composited with the real world [3]. It is different from the Virtual Reality (VR) technology that immerses the user inside a completely virtual environment. With a VR application, the user cannot see the real world around her or him. In contrast, AR technologies bridge the gap between the virtual and the real by allowing the user to see them at the same time.

The first AR prototype was invented in 1962 by Sutherland [96]. It leveraged a Head-Mounted Display (HMD) that used half-inch monochrome Cathode-Ray Tubes (CRT) and half-silvered mirrors and incorporated head-position-sensing to stabilize the projected graphics on a real-world scene. Due to hardware limitations in the 1960s, the system only drew transparent wireframes. Unlike the gyroscope, accelerometer, or other devices used today for sensing head positions, they utilised an ultrasonic system with four receivers mounted in a square array on the ceiling. Thomas P. Caudell, a former Boeing researcher, is believed to have coined the term "Augmented Reality" in 1990 [58].

After decades of research, AR technologies have gained great progress in terms of both hardware and software. In 2012, Google unveiled its AR glasses project [27]. The Google Glass has a horizontal frame and a strip of a computer that rest on a wearer's nose. It is equipped with a small display in front of a wearer's right eye, which superimposes a virtual world upon the wearer's view. A wearer can get map navigation, take photos, and conduct video chats. On the other hand, Microsoft announced its AR HMD "HoloLens" in 2015 [67]. Unlike Google Glass, which is lightweight, HoloLens has powerful hardware [67]. The newest HoloLens 2 is equipped with a Qualcomm Snapdragon 850 CPU, 4GB RAM, and 64GB storage. The sensors include an 8MP camera, a 1MP time-of-flight depth sensor, two eye-tracking cameras, etc. There are many other AR glasses available besides Google Glass and Microsoft HoloLens [98]. However, while AR glasses are still in development, the most popular AR devices among customers are still Hand-Held Displays (HHDs), e.g., mobile phones and tablets.

Various AR applications have been developed. They are used in gaming, tourism, navigation, training, etc. [17]. Shi et al. [88] proposed an AR application for individual location recommendation. It displays information that relates to users based on the users' images. It has shown that using both a Global Positioning System (GPS) and image

recognition can improve the location inference. Chang et al. [15] proposed an AR mobile guidance system that improves the learning effectiveness of visitors in art museums. They used tablets to superimpose related information on real paintings. They also conducted a user study to evaluate the effectiveness of their proposed system. The results showed that the AR guide effectively enhanced visitors' learning effectiveness, promoted their flow experience, and extended the amount of time the visitors spent focusing on the paintings. In 2016, an AR mobile game, Pokémon GO, first reached the top of the download charts for mobile applications [78]. It enables users to use smartphones and tablets to view and interact with virtual creatures superimposed on the real world.

The use of AR in repair tasks has also been explored for decades [107, 73]. Researchers have exploited various hardware, such as HMD, tablets, desktop Personal Computers (PC), and projectors. The fields of application include the aviation industry, mechanical maintenance, consumer technology, etc. As early as in 1997, in a survey of AR, Azuma [3] stated that instructions might be easier to understand if they were available as 3D representations superimposed upon the real scene, not as 2D texts and images. Westerfield et al. [109] considered AR as an ideal tool for situations that require the manipulation of objects. Researchers have conducted a lot of user studies to compare AR platforms to traditional ones. Henderson and Feiner [39] have evaluated their proposed AR platform against a traditional instructional software on the PC and concluded that the AR platform could effectively accelerate task localization. Another work by Henderson and Feiner [40] showed that their proposed AR platform led to fewer alignment errors than a traditional instructional booklet. A qualitative study by Zhu et al. [114] showed that, compared to paper-based manuals, the AR platform provided a more intuitive and satisfying user experience.

Although previous work for remote assistance has targeted various task types, such as repair, maintenance, assembly and disassembly, assembly training, etc., those task types share many common characteristics [73]. For example, the AR applications for repair and maintenance tasks always involve assembly and disassembly procedures. Similar authoring methods, interaction designs, and instruction designs are used in many types of tasks. In fact, some authors have discussed the applicability of their methods for other types of tasks [73]. Therefore, when introducing the previous work, I include the ones that have been designed for repair tasks, maintenance tasks, assembly and disassembly, and assembly training. In the next section, I will discuss the use of AR for remote assistance.

1.1 Augmented Reality in Remote Assistance for Repair Tasks

Augmented Reality (AR) is very suitable for assembly tasks since it can overlay instruction as a 2D demonstration or 3D virtual objects on a real product. The devices vary from mobile phones, tablets, to HMD.

Wang et al. [107] and Palmarini et al. [73] are good sources for a comprehensive understanding of the use of AR in assembly tasks. In this section, I only list some prominent work in the past decades.

Boeing [12] used AR technology in its wire bundle assembly training. They leveraged an HMD combined with head position sensing to superimpose computer-produced diagrams on the real environment. In the system, only simple wireframes, 2D notations, and texts are displayed and animated due to the limited computing power of the standard and inexpensive microprocessors at that time. They built four applications using the proposed platform: 1) wire bundle assembly on a wiring formboard, 2) connector assembly, 3) composite cloth layup, and 4) component assembly or removal. The motivation for them to build the AR platform is to close the gap between engineering designs for parts and processes stored in Computer-Aided Design (CAD) systems and assembly guides, templates, and other instruction materials. In this way, the expense and delay in aircraft manufacturing coming from the requirement to mirror changes in the engineering design in the instructional materials are minimized. Also, they expected that such a system could improve the performance of technicians. A list of the existing work that uses AR technology for remote assistance can be found in Section 2.1.

By reviewing existing approaches, I noticed that the types of AR for remote assistance can be divided into two categories: offline approaches and online approaches. In the offline approaches, an expert hardcodes or uses a script to integrate the guidance into the system. When a user operates with the help of the assistance platform, there is no real-time and direct feedback available from the expert. In the online approaches, an expert guides users from a remote place in an online manner. Therefore, the user receives feedback from the expert in real-time.

Moreover, the authors of previous work used different terms for the roles involved in a task. For instance, in the category of offline approaches, Caudell and Mizell [12] used the term "user", Webel et al. [108] used the term "technician", and Gavish et al. [30] used the term "trainee". In the category of online approaches, Gurevich et al. [35] used the terms "worker" and "helper" to denote a person who has no experience on a task and a

person who is an expert of a task, respectively, while Ranatunga et al. [77] used the terms "worker" and "expert". In this thesis, I use the term "user" to denote a person who has no experience on a repair task and the term "expert" to denote a person who has considerable expertise in a repair task.

Currently, the use of AR platforms in repair tasks is seeing rapid progress. The traditional methods, e.g., instructional booklets and videos, are still the most frequently used. Most of the AR platforms I listed above were prototypes and have only been explored in laboratories. Researchers are facing many challenges that prevent the AR platforms from wide deployment.

Although the use of AR technology has been around for almost 30 years, there are still limited examples of its concrete implementation in industry [73]. And in my experience, the assembly manuals of the products we have bought in our daily life are in the forms of printed manuals and video tutorials. The next section aims to show the existing challenges in this field.

1.2 Existing Challenges

A. Device Limitations

AR devices have experienced rapid growth recently. However, there are still limitations that prevent remote assistance platforms from becoming widely used [98]. An AR device usually consists of sensors and one or two displays. The most accessible AR devices are mobile phones and tablets that are equipped with cameras. However, compared to wearable devices, mobile phones and tablets provide a less immersive user experience. Wearable devices such as smart glasses use two displays, one for each eye, to provide stereo vision, while mobile phones and tablets have only one display. Also, users wearing AR glasses have a wider field of view. Moreover, the sensors on board of the mobile phones and tablets are not specifically designed for AR, which limits their capacity for data sensing for AR applications.

Although wearable devices such as AR glasses seem more suitable for AR applications, they are still in fast progress. Researchers are seeking a balance between comfort and performance. The devices with a higher performance are usually heavier, more expensive, and less comfortable, but the devices that are more portable are less computationally powerful. For example, the Microsoft HoloLens 2 [67] is equipped with very powerful hardware but is 566 grams. The less heavy Google Glass [34] is 46 grams but has weaker

computing power. Note that, according to a survey [98], Google Glass is the lightest device on the market but is much heavier than conventional glasses that are usually around 15 grams. Reducing the weight of AR devices is necessary for users to wear them for a prolonged time. Besides weight, battery life is another challenge. Increasing battery capacity also increases the weight. Therefore, a balance between performance, comfort, and portability is the key to the widespread use of AR devices.

The simplest and most effective method that reduces the weight and increases battery life is to degrade the hardware, e.g., using power-saving processors, batteries with shorter battery life, fewer sensors. However, this degradation requires specific algorithms and applications that rely on less powerful hardware but produce a decent user experience.

The advent of new communication technologies, such as 5G, has enabled low latency communication and high bandwidth and made cloud computing more attractive. Leveraging the internet to relieve the rendering burden of end devices has been studied by researchers [46]. The basic idea is to offload computationally intensive tasks to a remote server. For example, remote rendering methods offload rendering tasks to a remote server that has a higher rendering capacity [86]. In this way, end devices no longer need powerful graphic chips, and therefore their size and weight can be further reduced. Besides offloading rendering tasks to a remote server, we can also offload some other tasks, e.g., object tracking and object recognition [84, 76].

B. Object Pose Tracking

Although researchers have spent a substantial effort in the investigation of object pose tracking, it remains challenging to incorporate such technique in AR remote assistance systems. Most of the existing augmented reality applications for AR assistance rely on fiducial markers as marker-based object pose tracking methods are considered robust and accurate [73]. However, there are often small components involved in AR remote assistance. For instance, mobile phone repair tasks may involve the handling of pegs and tiny chips. It is difficult to attach fiducial markers to these small components. Moreover, it is challenging to track the markers on the non-rigid components, e.g., wires, since the markers may not remain flat during tracking. Nonetheless, even for methods that do not use fiducial markers, it is still a challenging problem, since these methods are typically less robust than those that use markers [73].

In AR remote assistance, object pose tracking methods suffer from long-lasting and severe occlusions. Contrary to typical scenarios of object pose tracking, an assembly task component is often a part of the entire product. A vast part of a component may be hidden

in another component during the assembly process.

Since the nature of the components in AR assistance is somewhat different from typical everyday objects (e.g., they are small and similar in appearance, and may be severely occluded during tracking), dedicated datasets are required. Existing datasets, e.g., YCB-Video [111], LineMOD [41], and KITTI [31], do not focus on the challenges of assembly task components. For example, the YCB-Video dataset contains videos of twenty-one objects in our daily life, e.g., cans, boxes, and cups. There are three main differences between those objects and the components used in assembly tasks. First, most of the objects in the YCB-Video dataset have rich textures. Second, these objects are bigger than many of the components used in assembly tasks. Third, these objects are very different in terms of appearance. The fifteen objects of the LineMOD dataset are also different from the components used in assembly tasks. A limited number of researchers have worked on such datasets. Hodaň et al. [42] proposed a dataset for 6 DoF pose estimation using thirty industry-relevant components. These components are texture-less and share similar appearances. They tested a state-of-the-art object pose tracking method [43] on this dataset. The results suggested that the tested method has ample room for improvement. More specifically, the tested method achieved a 95.4% mean recall on the LineMOD dataset while it achieved a 67.2% mean recall on their proposed dataset.

C. Human-Computer Interaction

Human-Computer Interaction (HCI) design is another challenge for AR platforms for repair tasks. Instructional booklets and videos often provide step-based instructions for the users. Some researchers have explored this direction, such as Webel et a. [108] and Speicher et al. [92]. Some projects focused on the interactive nature of AR systems [40]. With the progress of object tracking methods, a few researchers investigated the use of AR in providing the users with feedback automatically, such as Gavish et al. [30] and Westerfield et al. [109]. Researchers also have explored online remote assistance during the past decades [94].

AR technology can provide rich information to the users in various media formats, e.g., images, 3D animations, and notations [73]. Some researchers used 2D graphics, e.g., notations, tags, sketches, and images, such as Caudell and Mizell [12] and Ladwig et al. [56]. However, others chose to use 3D graphics, e.g., 3D virtual models, animations, or mixed 2D and 3D graphics. For example, Hoppe et al. [45] used 3D virtual objects in their AR platform, while Sukan et al. [93] mixed 3D virtual objects and 2D texts in their method.

Furthermore, AR has enabled more intuitive input methods, e.g., gaze tracking, ges-

tures, voice control, and Brain-Computer Interface (BCI). Although, to the best of my knowledge, they have not been used in repair tasks, they may improve the performance in assembly and maintenance tasks since these input methods provide users with a hands-free experience. The research community has spent huge efforts on this topic [112, 74], but further exploration and testing are needed before they can be widely accepted.

With rich media formats, large information volume, content awareness, and collaboration, designing instructions using AR is more flexible than designing traditional instructions. However, an effort must be made to understand which is the best way of visualizing information, and to present more complex and multi-step tasks [107, 73]. However, there are few analyses on how different interaction and visualization patterns may influence the effectiveness of a platform on a specific task. The existing evaluations mainly focused on the comparison between a specific AR platform and a traditional method (e.g., [30, 45, 55]). Although some researchers have paid attention to evaluating several design decisions for their proposed platforms [72, 109], they used very different measures and based on very different scenarios. I believe, to guide the design of HCI for AR remote assistance platforms, more general analyses are needed.

D. Authoring

Authoring plays an important role in the widespread use of AR in real scenarios. For instance, if a desktop computer vendor wishes to provide customers with an AR application to help them with repair tasks, it must consider the cost of development. Developing an AR application can be costly and beyond the means of small vendors. So adapting an existing AR platform to various tasks is important towards the democratization of AR technology for remote assistance.

Of the challenges of AR authoring is that, contrary to text and images, AR content has richer media formats, e.g., 3D virtual objects, audio, animations, etc. Integrating different formats of AR content with the spatial and temporal conditions requires specialized tools. Most of the previous AR platforms used manual authoring to create AR content, which can be expensive. The research community has proposed a variety of tools. Some researchers have used existing software [36], while others developed new authoring tools [68, 92]. These tools require the author to manually specify the spatial correspondence, i.e., the relative positions and orientations between the real world and the virtual objects. Bhattacharya and Winner developed a tool can generate such spatial correspondence in a more convenient way by tracking the components [8]. Other researchers explored the use of programs to generate AR content from traditional materials for remote assistance, e.g., instructional

booklets and videos [70, 69].

For online AR platforms, authoring is more challenging since it requires that the authoring tools produce AR content in an online manner. A vast body of research has focused on authoring 2D AR content, usually by allowing a remote expert to do freehand sketching or put notations in the user's view [32, 35]. Some researchers have developed the tools that enable remote experts to present three-dimensional contents, e.g., 3D virtual objects, 3D arrows, and virtual hands [72, 94]. The key difficulties are how to facilitate the use of those tools and how to apply physical and geometrical constraints more naturally.

In this thesis, I will propose an AR platform that aims to address two of the above challenges: device limitations and authoring. In the next two sections, I will introduce the two solutions that address the challenges.

1.3 Problem Statement

1.3.1 Authoring

In Section 1.1, I divided AR platforms into two categories: offline platforms and online platforms. By comparing the methods from the offline category and the ones from the online category, I noticed that, in addition to the presence of a remote expert, there is an obvious difference between the methods from those two categories. The methods from the offline category typically present the content as virtual 3D models, while most of the online approaches present the content as 2D drawings. Only presenting 2D drawings can affect the performance of users [2], due to the limited capacity of conveying complex spatial translations and rotations in 3D space. The experiment described in Chapter 5 shows that the participants who receive instructions in 2D sketch tended to make more corrected errors than those who receive instructions in virtual 3D models.

The cause of this difference is that, in an offline approach, an expert has enough time and more efficient tools to prepare the 3D representations for the instructions. In contrast, in an online approach, an expert must present the instructions to the user immediately, and there are no efficient tools to create 3D representations in a real-time manner.

The authoring tools for the offline methods can help the experts create complex 3D representations of the instructions, such as the work of Westerfield et al. [109]. Many of them have a Graphical User Interface (GUI) that enables content creation without any programming knowledge. However, for online composing, the experts use online methods,

e.g., drawing and pointing, to guide users, such as in the work of Gurevich et al. [35], which limits the experts in creating complex 3D contents.

Many researchers have explored the authoring of complex contents other than 2D sketches in online AR remote assistance. Ranatunga et al. [77] used an object tracking method to enable the expert to more accurately and more conveniently specify the locations and rotations of the components. However, the drawback of this method is that it only supports operations on a 2D plane. Oda et al. [72] proposed a tool for the experts to interact with and present 3D models to users. It has the potential to enable composing more complex instructions in the field of online approaches. But there are still some drawbacks. First, a software to create instructions introduces some difficulties for the experts since they need to specify 3D positions and orientations on a 2D display. Second, it is difficult to apply real-world physical and geometrical constraints.

Therefore, I propose a novel AR remote assistance platform that contains an authoring tool for the experts to compose complex instructions and present them to new users using 3D virtual objects. This authoring tool tracks the operations of the experts using the Vuforia Engine [75] so that it only requires the experts to demonstrate the operation by simply performing it rather than using a non-intuitive tool. The advantages of the proposed authoring tool are two-fold: 1) it is efficient to present operations with 6 DoF (Degree of Freedom); 2) because the experts operate on the real products, the physical and geometrical constraints are applied natually. The details of the proposed authoring tool are in Section 3.2.

1.3.2 Remote Rendering

As I discussed in Section 1.2, the limitations of AR devices is one of the challenges before such devices can be widely used in repair tasks. The rapid growth of communication technologies inspired researchers to tackle this problem with cloud computing. The target is to reduce the weight and increase battery life of the AR devices while maintaining a good user experience. One way to address this is through Cloud Mobile Rendering (CMR). CMR offloads rendering to cloud servers: the server initializes a rendering engine and an encoder that serve every connected client. The models are rendered on the server-side and encoded in video frames streamed from the server to the client [90, 66].

Although a considerable effort has been put into this research direction, CMR systems can still suffer from bandwidth limitation and interaction latency [18]. Various attempts have been made to improve such systems. Hemmati et al. [38] proposed a method for

cloud gaming, which selectively renders important objects and reduces video bitrate by not rendering unimportant objects. Boukerche and Pazzi [10] used environment maps that are rendered on a server and sent to the client to reduce bandwidth. The client can respond to user interactions resulting in panning and tilting without latency based on the environment map. Shi et al. [87] proposed a method that leverages depth maps to reduce interaction latency. They take advantage of 3D image warping to synthesize the mobile display from the depth images generated on the server. However, theses two image-based rendering techniques assume static scenes and only support limited user interactions. For example, the use of environment maps by Boukerche and Pazzi [10] accelerates panning interactions. Still, a new environment map needs to be generated for a novel viewpoint, which increases interaction latency and bandwidth requirements. Similarly, during scene changes, the environment maps or depth maps also need to be regenerated.

Therefore, I propose a hybrid remote rendering approach that minimizes the network bandwidth requirements and interaction latency associated with remote rendering. This approach enables a trade-off between rendering quality of less important objects in the scene and network requirements. The details of the proposed remote rendering approach are in Section 3.3.

1.4 Thesis Statement

The vision of the thesis is towards making AR technology for repair tasks become widely used in the real world. This work aims to address two challenges, device limitations and authoring, among the four existing challenges described in Section 1.2. To achieve this goal, I propose an authoring tool and a hybrid remote rendering method. The authoring tool aims at addressing the challenge of authoring by enabling a remote expert to compose complex instructions using 3D virtual models. The hybrid remote rendering method aims at addressing the challenge of device limitations by reducing the rendering burden of the AR devices. I integrate the authoring tool and the hybrid remote rendering method into an AR platform.

1.5 Overview and Contributions

In this thesis, I propose an AR online assistance platform for assembly tasks in Chapter 3. This platform exploits mobile phones as AR devices. A remote expert uses a mobile

phone to capture her or his operations on a real product. The platform converts the operations into animations of virtual 3D models and displays them on the user's mobile phone. Moreover, the users' operations are also captured by a mobile phone and sent to the remote expert as virtual 3D models. Therefore, the remote expert can inspect the user's operations and give feedback to her or him. In this way, the expert does not need to create the instructions before the repair task and can create the instructions online. Figure 1.1 shows the platform setup. Note that the setup is the same for users and remote experts because both sides use a mobile phone as the AR device and repair a real product.

Figure 1.1: Platform setup for a user or a remote expert.

This platform addresses the authoring challenge by capturing the operations of a remote expert automatically. In this way, the presentation of instructions in an online remote assistance system for complex tasks is not limited to 2D sketches and notations. Moreover,

by allowing the experts to use a real product to demonstrate the operations, the geometrical and physical constraints are applied naturally. It also addresses the challenge of device limitations by using a hybrid remote rendering approach that can help AR applications get beyond the device limits by offloading the rendering burden to a remote server. I propose the following contributions:

1. A novel AR platform that assists a remote expert on guiding users through repair tasks in an online manner.

2. A novel authoring tool for repair tasks that enables a remote expert to compose complex instructions and present them to users using 3D virtual objects (see Sections 3.2).

3. A novel distributed rendering method that enables a trade-off between rendering quality of less important objects in the scene and network requirements (see Section 3.3). It has the ability to simultaneously reduce bandwidth requirements and interaction latency compared to existing CMR approaches. A performance analysis of my method is presented (see Section 4.1).

4. A comprehensive user study that evaluates the impact of my proposed hybrid remote rendering method on the user experience (see Section 4.2).

5. A user study that compares the proposed AR platform to a 2D sketching and an instructional video methods in terms of time, number of corrected errors, and number of uncorrected errors. I also use questionnaires to evaluate the participants' subjective opinions of these methods (see Chapter 5).

Chapter 2

Background and Related Work

I begin by reviewing existing work that uses AR technology for remote assistance. I first list the significant work in order of time, followed by discussing various aspects of AR assembly platforms: devices, tracking modules, visualization, and guidance, see Section 2.1. However, since this thesis focuses on authoring and using remote rendering to address hardware limitations, we use separated sections to review the current research on these two topics. Section 2.2 reviews work related to the challenge of authoring, while Section 2.3 reviews work related to using remote rendering to address hardware limitations.

2.1 Background

Kuzuoka [54] proposed one of the first online AR systems that supports a remote expert to monitor a user's workspace through a monitor and instruct the user by pointing or gesturing to a live video. Using a table-based arrangement task, Kuzuoka showed that using gestures can improve task completion times over using audio instructions only. Another early work was done by Reiners et al. [80] in 1999. The work described an AR demonstrator for the task of the doorlock assembly into a car door. This is an HMD-based training application that guides users through the linear assembly process in a step-by-step fashion. The authors demonstrated the system to a large non-expert audience for one of the first times. In 1999, Baird and Barfield [4] conducted an experiment that evaluates the performance of the AR platform in a motherboard assembly task. The involved platforms include paper manual, computer-aided, opaque AR display, and see-through AR display. The results suggested that AR techniques were the most effective industrial aids for the assembly task given that the participants using the AR techniques achieved the fastest assembly time and made

fewer errors.

Boulanger [11] described a collaborative industrial tele-training system that allows a remote trainer to interact with trainees by audio, 2D notations, and 3D virtual objects. It supports multiple trainees and multiple trainers. Since only one of the trainees has access to the real product, the other trainees and the trainers share the same view as that trainee using video streams. Trainees use the 2D markers to move the 3D virtual objects to show trainers their movements. Therefore, the other trainees can also participate in the discussion. Trainers use audio and remote pointers to demonstrate the instructions. With remote pointers, trainers can overlay a 2D arrow on the real-world product. The system uses 2D markers for object and camera tracking. With the system, the cost of training can be greatly reduced since multiple users can share one expensive product during the training process. Moreover, by using overlapped 2D notations and 3D virtual objects, trainees and trainers can describe operations and movements effectively and precisely.

Gurevich et al. [35] leveraged projectors to display instructions as 2D hand-drawn notations. When a worker works on a product, a camera records the workspace and transmits it to a remote expert. Since the camera is mounted on top of a robotic arm, the remote helper can control the viewpoint by moving the camera remotely. The projector is mounted together with the camera, so that the instructions can be projected correctly in the workspace. The system is equipped with a set of annotation tools to allow the helper to perform freehand sketches. What the helper draws is projected on top of the real objects in the worker's environment. Both, the worker and the helper can carry a voice conversation. It provides the helper with a more powerful tool to establish a common understanding than voice-only communications. They also conducted a user study for a LEGO assembly task to show the effectiveness of their method.

Webel et al. [108] proposed an AR platform for assembly task training. They used tablets to capture the real-world environments and display the AR content in various media formats, e.g., images, text, and 3D virtual object animation. The platform uses a step-based interaction design to present the instructions. They showed 3D virtual object animations and other contextual information for each step. They also integrated haptic hints using vibrotactile bracelets to provide additional translational and rotational movement cues or present error feedback. Their object tracking is based on 2D markers. They evaluated the proposed platform against an instructional video on an electro-mechanical actuator assembly task. Twenty participants were assigned into two groups, where one group used the proposed AR platform, and the other group used an instructional video. The results showed that the participants that trained with the AR platform outperformed

those that trained with an instructional video with a smaller number of errors in the testing after training. This indicates that the AR platform can result in a better skill transfer.

The work of Ranatunga et al. [77] used projectors to display instructions. As opposed to the work by Gurevich et al. [35], they used an object tracking method to project rectangles on real objects based on their sizes, rather than freehand drawings. The remote expert can see the workspace of the user using a camera set up in the worker's environment. She or he can move or rotate any virtual rectangles on the screen, and the projected virtual rectangles move accordingly on the local worker's view. This facilitates instruction authoring since it allows the expert to directly specify the positions and orientations of the objects. The system also provides a clearer demonstration than pure freehand sketching methods. Adcock et al. [2] conducted a user study for this work. The results showed that compared to a pure freehand sketching method, the proposed method improves the accuracy of movements and rotations for the workers, with a camera perpendicular to the work surface. However, one disadvantage of this system is that the expert can only demonstrate planer translations and rotations.

Bhattacharya and Winer [8] proposed an automatic authoring tool for AR systems. The tool captures assembly parts with a depth+RGB camera and generates instructions for the captured movements. Compared to previous authoring tools, it minimizes the effort needed to adapt an AR application for various tasks. Therefore, no manual editing is needed since the tool generates the AR content of 3D virtual objects automatically

Gavish et al. [30] evaluated the performance between an AR platform, a VR platform, and two traditional methods, i.e., an instructional booklet and an instructional video. In the AR group, the participants were trained using the AR platform once, while in the Control-AR group, the participants were trained using the real actuator and a filmed demonstration once. In the VR group, the participants were trained using the VR platform twice, while in the Control-VR group, the participants were trained using a filmed demonstration twice. The results showed that the VR and AR groups required longer training time, and there were fewer unsolved errors in the AR group compared to the Control-AR group. However, there were no significant differences in the performance between the VR and Control-VR groups. Note that the objective study did not directly compare the AR and the VR platforms since they were different in several pedagogical features. However, a subjective study showed that the AR platform achieved significantly higher scores in eight of the thirteen questions.

In most of the methods in which remote experts are involved, the remote experts use 2D notations or sketches to present the instructions. In contrast, Oda et al. [72] presented

a method for remote experts to author instructions with 3D virtual objects. The expert uses a tracked mouse and lazy susan turntable to move the 3D virtual objects in a virtual environment, where the local user wears an HMD and operates on the real product with the 3D virtual objects superimposed on the view. The authors asked the participants to fill a questionnaire about their experience. The results of the questionnaire showed that using 3D virtual objects was significantly more favorable compared to 2D sketches.

Sun et al. [94] developed an AR platform named OptoBridge that enhances remote presence by displaying a virtual hand on the local environment. In their AR platform, a remote teacher can guide a local student through several steps of an adjustment task of an interferometer with hand telepresence. When the local student is operating on a real interferometer, a camera is recording her or his workspace and sending a video stream to the remote teacher. On the remote side, a hand tracking device tracks the hand of the teacher and presents it to the local student in augmented reality. As opposed to most of the other AR platforms, this system does not present the instructions as text, image, 2D sketches, or 3D virtual objects. The authors conducted a user study to investigate the influence of two different viewpoints on task performance and task completion time. The results indicated that the task performance is better with the third-person viewpoint than the first-person viewpoint. However, there was no significant difference between the two viewpoints concerning the completion time.

The devices mentioned and utilized in previous articles include HMDs, HHDs, desktops, projectors, and haptic devices, among which HMDs are the most used devices [73]. An HMD, usually worn on the head or as part of a helmet, has a small display in front of each eye. Kolkmeier et al. [53] Utzig et al. [103] are two examples that use an HMD. Compared to HHDs, HMDs have the advantages of portability. Users of HMDs, such as glasses, can navigate or change their viewing direction while performing operations using both hands. HHDs are usually more accessible than HMDs since mobile phones and tablets can serve as AR devices. Kim and Moon [51] developed a car maintenance training application for smartphones and tablets. Gavish et al. [30] used a tablet to capture the real world and overlay virtual objects and tags on it. Serván et al. [85] used a tablet in a demonstrator for electrical harness routing in the frame 36 of the AIRBUS A400M. Desktops were widely used on the expert's side. For example, Re and Bordegoni [79] enabled an expert to monitor the user's work from a remote location. The users used mobile devices to store information for each maintenance step on a remote database. Experts could check the maintenance activity from a remote PC at any time. Haringer and Regenbrecht [36] proposed an authoring approach with a GUI on a PC. Researchers have used projectors in their AR platforms. Using projectors, the field of view is not limited by devices. Sakata et

al. [81] used a laser pointer and a camera mounted on the user's shoulder to allow a remote expert to instruct the user by pointing to physical objects using the laser pointer. Adcock and Gunn [1] allowed a remote expert to draw and project instructions onto the user's environment using a laser projector. A limited number of articles have explored the use of haptic devices. Webel et al. [108] used a tactile bracelet to present additional movement hints, such as rotational or translational movements cues, and to alert users about errors during the training.

The tracking module in AR platforms calculates the relative transformations of the camera and the objects in real-time. Therefore, it is "the heart" of an AR platform [89]. Most of the existing AR platforms for assembly tasks have used marker-based or feature-based techniques. Zauner et al. [113] used markers in their AR platform and tested it in a furniture assembly task. Liverani et al. [62] stamped fiducial markers on components that have uniform colors. Salonen and Sääski [82] did not attach markers on each assembly component, but attached them on the base of the product instead. The relative transformation between each component and the base is treated as a priori knowledge and has to be measured beforehand. Marker-based methods are considered robust and accurate [73], but rely on the maintenance of markers which may not always be available. For example, the aerospace industry considers the need to put markers on the aircraft is unacceptable [23]. Koch et al. [52] leveraged so-called "natural markers" that are fiducial markers existing in the environment, such as position marks of fire extinguishers and device ID tags. Crescenzio et al. [23] developed a tracking method that relies on SURF (Speeded-Up Robust Features) algorithm [7]. However, feature-based methods are sensitive to illumination changes and partial occlusions. Many researchers have proposed model-based approaches. Khuong et al. [50] developed a non-marker-based tracking approach in their AR platform and experimented with it in a LEGO assembly task. They leveraged a Kinect with a depth camera to reconstruct and track each LEGO block.

The AR platforms for assembly tasks utilized various visualization methods, e.g., virtual 3D models, 2D images, texts, and sketches. Limited by the computing power, AR platforms of the early stage only display 3D wireframes, 2D images, and texts, such as the ones proposed by Sutherland [96] and Caudell and Wizell [12]. For recent work, the hardware is no longer a bottleneck for visualization, and therefore, the main consideration is to make instructional visualizations suitable to users. Engelke et al. [25] presented an interface that allows a user to personalize the visualization method. Many existing platforms used more than one visualization method. Wang et al. [105] used virtual 3D models and texts to display instructions. Ranatunga et al. [77] used a projector to project rectangles and arrows on a plane. Some researchers considered telepresence as a visualization method.

Sun et al. [94] developed a system to capture the hands of an expert and reconstruct them on the user's view. However, in the online AR platforms, displaying virtual 3D models is difficult. This involves the development of a complex interface for the remote expert to allow her or him to specify the positions and orientations of models in real-time and with the geometrical and physical constraints that correspond to the manipulated objects. In this case, displaying 3D animations is more difficult. Oda et al. [72] proposed an interface that allows an expert to edit the virtual scene using a tracked mouse and a lazy susan turntable. However, geometrical and physical constraints need to be specified beforehand.

An AR assembly guidance enables users to perform a task in a higher quality since the quality of work does not completely rely on the experience and skills of a user. It even allows a user to learn by performing the task [107]. Many existing AR platforms for repair tasks provide step-by-step guidance to users. Speicher et al. [92] proposed an authoring tool that facilitates a non-programmer to make guidance with fixed assembly steps. Syberfeldt et al. [99] implemented a step-based AR assembly platform and experimented with it in a puzzle task. Some researchers used AR technology to provide additional information to users. Schall et al. [83] presented an AR system for viewing underground infrastructure, e.g., pipelines under a road. Sukan et al. [93] developed an AR platform called ParaFrustum that supports users to find an appropriate view angle of an object to avoid occlusion. Some projects focused on the interactive nature of AR systems. Henderson and Feiner [40] presented an AR system for a psychomotor phase of procedural tasks. This system provides dynamic and prescriptive instructions according to the user's current activity. Zhu et al. [116] proposed a wearable AR mentoring system that can recognize the user's current status and provide position-aware feedback. With the presence of a remote expert, AR platforms can facilitate the communication between the expert and user [11, 77].

2.2 Authoring

As discussed in Section 1.1, the use of AR in remote assistance can be categorized into two categories: offline approaches and online approaches. In offline approaches, an expert hardcodes or uses a script to integrate the guidance into the system. In the online approaches, an expert guides the user online from a remote place in an online manner. The bodies of work in the two categories are independent from each other to a high degree. The proposed platform in this thesis belongs to the online category.

Many approaches have been proposed in the industry [12, 23]. However, these approaches are still far from being widely used in real production environments. The reasons

are multifold, and include the accessibility of AR devices, relatively immature user experience, and a lack of efficient authoring tools [73]. Here we only review authoring tools proposed by previous work for repair tasks.

2.2.1 Offline AR Systems

Many of the existing methods are task-specific, i.e., lacking the capacity to be adapted for a wide range of different tasks. Many existing platforms do not take authoring into consideration or do not focus on non-programmer accessibility of the authoring tools [23]. Authoring has gained increasing interest during the past years [107]. The authoring tools are used by non-programmers to construct augmented reality content. In practical aspects, the expert must access the authoring tool to adapt an AR system to a specific task.

One direction is to create scripts using software. Haringer and Regenbrecht [36] proposed an approach to augmented reality authoring with the help of Microsoft PowerPoint. The expert can create an XML-base (Extensible Markup Language) extensible description of a PowerPoint file, followed by importing the PowerPoint file into 3D scenes and using it in an AR-viewer. Zauner et al. [113] developed an XML-based tool for non-programmer expert to structure the assembly steps with mouse clicks. Mitrovic et al. [68] developed an authoring tool named ASPIRE that includes a GUI. With ASPIRE, experts are able to create AR solutions, domain models and instructions without programming knowledge. Speicher et al. [92] demonstrated a web-based tool for creating AR content for industry 4.0. Users are able to design and share AR assembly instructions worldwide. However, using software to create scripts still has a learning curve. Furthermore, those methods are not suitable for online remote assistance platforms, since the scripts cannot be generated in real-time.

Another direction is to generate AR content automatically by various techniques. Currently, limited work has focused on this. Mura et al. [70] reported a proof-of-concept system to create an interactive AR manual by segmenting a video of a task into segments. Mohr et al. [69] developed an approach to transfer a printed assembly manual into a 3D AR training guidance. The input of this system is a printed manual, 3D models and the definition of various 2D annotations. The output is an interactive AR application. Bhattacharya and Winer [8] described an authoring tool to capture assembly parts with a depth+RGB camera and with markers to generate the instructions for the captured movements. They used this authoring tool to generate AR content and instructions for an offline platform. This approach can be more efficient than using software to create scripts, and, most importantly, has the potential to adapt to real-time use cases. We use the same idea presented

in the work of Bhattacharya and Winer [8], and adapt it to an online remote assistance platform.

2.2.2 Online AR Systems

Online AR remote assistance platforms for repair tasks have another roadmap in terms of authoring. The most difficult challenge is creating AR content and instructions in real-time.

Many existing methods use 2D hand-drawn notations as AR guidance. REFLEKT ONE [32] enables real-time assistance from a remote expert. The expert shares the same view with a user, and what the expert draws appears on the user's display in real-time. Boulanger [11] presented a system that allows a remote expert to interact with a user by audio and pointing. The expert shares the same view with the user via a video stream. For a larger field of view, some researchers exploited projectors to project the 2D hand-drawn notations on the workspace. Gurevich et al. [35] is an example. The expert views the assembly process remotely via a monitor and draws on the screen. Meanwhile, the projector in the user's workspace projects the instructions onto the real environment.

However, the 2D hand-drawn notations have typically not made use of any semantic information about the physical properties of the environment. The work of Ranatunga et al. [77] allows the expert to directly specify the object movements required of a local user. In other words, the expert is able to more easily specify accurate positions and orientations. The drawback of this method is that the instructions are still on a 2D plane so that it is difficult to present 3D translations and rotations. Telepresence is a widely studied direction. Tecchia et al. [101] enables an expert who wears an HMD to monitor the workspace of the user and help her or him by presenting virtual hands on the view of the user. A pilot user study demonstrated the use of this platform on a LEGO toy assembly task. Observations indicated that the expert is able to guide the user through the task with hand gestures and verbal communication. Sun et al. [94] developed a system that tracks the hand gestures of an expert and displays virtual hands to the users. This method is able to show the spatial position and orientation changes. But showing the hand movements without 3D virtual objects is still not enough to precisely demonstrate the instructions.

For a more effective spatial referencing and action demonstration, Bottecchia et al. [9] proposed an AR platform that facilitates the expert to present instructions to the user using 3D animations. However this platform lacks flexibility since the 3D animations are precomputed. It does not handle unexpected situations encountered in a task. Chastine et

al. [16] enables an expert to place an arrow in the AR environment using a paddle. However, aligning the arrow is difficult and time-consuming. Oda et al. [72] used 3D virtual objects to present the instructions. They enabled the expert to compose the guidance with the software and display 3D virtual objects as overlays on a user's view. This approach also tracks the real components on the user's side so that it is possible to assess whether a user is following the guidelines precisely. There are still some drawbacks to this method. First, composing with a software introduces some difficulties to the expert since they need to specify 3D positions and orientations on a 2D display. Like using a 3D creation software, modifying the 3D scenarios on a 2D display involves a significant degree of training which can be costly and time-consuming. Second, it is difficult to apply real-world physical and geometrical constraints.

My proposed method takes a further step towards composing guidance online by leveraging the idea of tracking real components on the expert's side to specify the 3D movements [8]. In this way, it is easier for the expert to accurately place the virtual components, while the real-world physical and geometrical constraints are applied naturally.

2.3 Remote Rendering

Remote rendering leverages the computational capacity of a remote server to address the limitations of mobile devices in terms of processing power, limited storage and rendering hardware. The basic idea is to offload the entire rendering task to the server. Basically, when a mobile client connects to the server, the server will initialize a rendering engine and an encoder for the mobile client. All of the models are rendered on the server side and rendered frames are encoded and streamed from the server to the client as a video stream. Moreover, the server receives user interactions from the client. This approach requires no 3D graphics capacity on the client and is able to use advanced encoding, such as H.264/AVC, to adjust the image quality according to the bandwidth availability. Lamberti and Sanna [57], Moimark and Cohen-Or [71] and Lu et al. [64] have proposed such CMR systems. Many cloud gaming services and frameworks have leveraged this technology, e.g., NVIDIA GeForce Now [20], PlayStation Now [63] and Gaming Anywhere [26]. Those services and frameworks use cloud computing infrastructure to render the games and encode the rendered frames, while the clients decode the frames and send the players' interactions to the cloud.

However, there are two main challenges associated with the remote rendering methods. First, video streaming requires large network bandwidth. Therefore, the user experience

heavily relies on network conditions. Second, remote rendering methods introduce additional user interaction latency. The interaction latency consists of four components: transmission time, rendering time, encoding time and decoding time. Researchers have proposed various methods to address these two challenges. However, the existing solutions either only reduce the network bandwidth or cannot handle dynamic scenes well. To address the above described challenges that are associated with remote rendering, we propose a hybrid remote rendering method that leverages both server and client side computational resources. My proposed method reduces network bandwidth requirement and interaction latency at the same time, and can effectively handle dynamic scenes.

2.3.1 Network Bandwidth

Many approaches have been proposed to reduce the network bandwidth requirement. Levoy [59] proposed a method for networked workstations that renders simplified models on clients to reduce bandwidth requirement. According to this approach, the server first renders both complete and simplified models and calculates a difference image between the two. The difference image is transmitted to the client. The client renders the simplified models and applies the difference image to produce a high-quality rendering. The simplification of models can be performed in terms of the textures and the number of polygons. This method reduces the bandwidth requirement as the difference image can be compressed more effectively compared to the complete model. The entire scene is treated uniformly unlike in my selective approach based on object importance. Liu et al. [61] developed an automatic adaption algorithm that changes the rendering quality according to the network bandwidth. They use H.264 video encoding with fixed bitrate mode while adapting the rendering factors (e.g., view distance, realistic effect and texture detail) to improve the user experience. The goal of this approach is to improve the visual quality of encoded frames under a fixed network bandwidth by sacrificing rendering quality. Hemmati et al. [38, 65] proposed a method for cloud gaming, which selectively renders important objects and reduces video bitrate by not rendering unimportant ones. This reduces the network bandwidth requirement as the frames to be encoded contain less details.

The above described methods are targeted at reducing the network bandwidth only. They do not reduce the interaction latency. My proposed method aims at reducing the network bandwidth and latency simultaneously.

2.3.2 Interaction Latency

Many researchers have developed various approaches to reduce the interaction latency. We divide those approaches into two categories according to the "side information" sent from the server to the client: environment maps and image warping.

Environment Map. Boukerche and Pazzi [10] render environment maps on the server and send them to the client. With the environment maps, the client can respond to the user interaction in the form of panning, tilting, and zooming with little latency since the new view can be synthesized using the environment maps. Environment maps are similar to panoramas, which were initially proposed by Chen [19] in their QuickTime VR system, which displayed virtual environments without rendering 3D models. QuickTime VR accomplished moving through the environment by "hopping" to different panoramic points. Boukerche and Pazzi [10] do not simulate movement by "hopping" to another panoramic image, instead, they use a remote server to render panoramic images in real-time. They address the latency in rendering and transmitting panoramic images through a caching design that buffers visited viewpoints.

Image Warping. Shi et al. [87] proposed a method that leverages depth maps to reduce user interaction latency. They take advantage of 3D image warping to synthesize the mobile display from the depth images generated on the server. Chang and Ger [14] proposed building Layered Depth Images (LDIs) on the server. The mobile device uses a 3D warping algorithm to synthesize the frame from a new view point. At the time of the design of LDIs, mobile devices had limited computational capacity to render 3D scenes and hence LDIs were designed as a way around displaying 3D content on a mobile device. These image-based methods incorporating scene depth often have challenges pertaining to visible gaps and holes in the rendered scene. Bao and Gourlay [5, 6] proposed a method that uses a superview to direct the image warping and reduce the gaps and holes. The method is successful in reducing the flaws in the synthetic images. But it is not designed to handle highly dynamic scenes, since a set of new reference depth maps need to be generated and delivered to the client whenever the scene changes. Their method is well suited for environment walkthrough applications but not for games and training applications.

Network bandwidth limitation is typically not a major issue for the two categories of methods described above, as the environment or depth maps are not transmitted very often if the scene is static. However, these methods cannot effectively handle dynamic scenes. If an object in the scene moves, the environment maps or depth maps must be re-generated, which increases the interaction latency. My proposed hybrid remote rendering method aims to reduce the interaction latency for dynamic scenes.

2.3.3 Hybrid Remote Rendering

Some existing methods share several characteristics of the proposed method. As mentioned in Section 2.3.1, the work by Levoy [59] renders different versions of a model on the server and the client, respectively. However, this approach minimizes the bandwidth requirement at the expense of incurring additional latency associated with the rendering of multiple versions of the same model on the server side. Crassin et al. [22] and Liu et al. [60] leverage a remote server to render indirect lighting effect that are too expensive to compute locally. Hence, the client only renders the direct lighting effects and receives the indirect lighting information from the server. The goal of Crassin et al. [22] and Liu et al. [60] differs from ours, as they aim to refine the rendering effect by leveraging a more computationally powerful remote server. Moreover, Liu et al. [60] demonstrate that the interaction latency increases with model complexity, as rendering more complex models takes a longer time. Chan et al. [13] utilize the rendering power of both the cloud server and the client PC or mobile device. They divide the rendering operations to both sides. The rendered frames on the cloud server are sent from the server to the client, while the rendered frames on the client are superimposed with the rendered frames from the server to form complete rendering frames. However, their goal is to relieve the rendering burden of the cloud server. In this way, the cloud server can serve more clients at the same time.

The proposed hybrid remote rendering method aims at minimizing both network bandwidth and latency for dynamic scenes. I also benefit from the increasing rendering capabilities of mobile devices while addressing the fact that mobile devices still fall behind PCs. We believe that the idea of offloading the entire rendering task to the server may not be optimal, given the bandwidth it requires and the recent advances in mobile device hardware.

2.4 Summary

In this chapter, I briefly inspect four aspects of existing AR remote assistance platforms: devices, tracking modules, visualization, and guidance. Then we focus on reviewing existing approaches in the fields of authoring and remote rendering since my proposed platform mainly focuses on these two fields. The bodies of work in offline and online AR systems are independent from each other to a high degree. We observe that the authoring methods in offline AR systems lack the capacity of creating instructions in real-time, while the authoring methods in online AR systems mainly focus on creating 2D instructions. Our

proposed authoring solution leverages an idea from an offline AR platform and uses it in online authoring. This solution enables the expert to create complex instructions in 3D and apply the physical and geometrical constraints more naturally. In the field of remote rendering, many existing approaches cannot reduce network bandwidth requirement and interaction latency at the same time. Some image-based remote rendering approaches can greatly reduce interaction latency but cannot handle dynamic scenes effectively. My proposed hybrid remote rendering leverages both server and client side computational resources so that it can reduce network bandwidth requirement and interaction latency for dynamic scenes.

Chapter 3

An Augmented Reality Remote Assistance Platform for Repair Tasks

In this chapter, I propose an AR platform that assists online communication between remote experts and users for repair tasks. I describe four aspects of the platform: interaction design, authoring procedure, hybrid remote rendering, and other system design issues. Figure 3.1 demonstrates an overview of the proposed AR platform. The proposed authoring method is inspired by the work of Bhattacharya and Winer [8]. However, as opposed to the offline method of Bhattacharya and Winer [8], the proposed approach belongs to the online remote assistance category. My platform tracks the operations of the experts using the Vuforia Engine [75] so that it only requires the experts to demonstrate the operation by simply performing it rather than using a non-intuitive tool. I describe the details of the Vuforia Engine in Section 3.4.2. The advantages of the proposed authoring tool are two-fold: 1) it is efficient to present operations with 6 DoF (Degree of Freedom); 2) since the experts operate on the real products, the physical and geometrical constraints are applied automatically. In the proposed AR platform, I also proposed and adopted a hybrid remote rendering method that aims at relieving the rendering burden of the devices. This remote rendering method uses a client-server architecture, where the server renders the high-fidelity models, and the client renders the low-fidelity ones. The high-fidelity models' rendering frames are encoded and sent to the client as images or video streams. With this configuration, the network bandwidth requirement and interaction latency can be reduced because only key models are rendered in high-fidelity mode.

Figure 3.1: Overview of the proposed AR remote assistance platform. On the user's AR view (left), the virtual component shows where the user should move the real component. The position and orientation of the virtual component are decided by the movements of the real component on the expert's side. On the expert's AR view (right), the virtual component shows where the user has moved the real component. The position and orientation of the virtual component are decided by the movements of the real component on the user's side.

3.1 Interaction Design

As opposed to other remote assistance methods, the proposed approach assumes the existence of two copies of the same product on both the expert's side and the user's side. For example, in a mobile phone repair task, both the expert and the user require a mobile phone of the same type. This way, the AR platform is able to capture the operations that an expert performs by an object tracking method. In this section, I discuss the key issues encountered during the interaction design of the proposed platform.

I designed the instruction schema in my system as a step-based schema, with each step showing the instructions for an atomic task that would be impractical to divide. For example, in a mobile phone repair task, a step may show the user how to plug a battery in. Each step consists of three phases, a) observation, b) identification, c) motion. During the entire process, the expert and the participant are able to communicate by voice. The following describes each phase.

1. Observation: The expert performs the operation on a product of the same type as the one on the user's side. For instance, if the expert plugs a battery in, the camera captures the movement of the battery, and the AR platform displays a virtual 3D model of the battery on the user's view.

2. Identification: The user is prompted to scan the components one by one using a camera to identify the correct component. Once the component is found, the user is notified with a beep and a notation showing the component. Because the object tracking is also implemented on the user's side, it is the AR system that decides if a component is the correct one, instead of the expert.

3. Motion: A virtual component shows the actual trajectory of moving the real component to its destination. The trajectory is generated by capturing the operation of the expert, so the user is supposed to follow this trajectory to perform the operation. A virtual component is also displayed on the expert's view, which enables the expert to know how well the user performs. It is the expert who decides if the operation of the user is correct and if the process should go to the next step. However, if the expert finds an error in the user's operation, the process goes back to the observation phase.

The users should be aware of the three phases. In my implementation, there is a textual display on the top-left corner that shows the current phase. A user is supposed to take corresponding actions in different phases. First, I will introduce the actions that the user needs to take during each phase, see Figure 3.2.

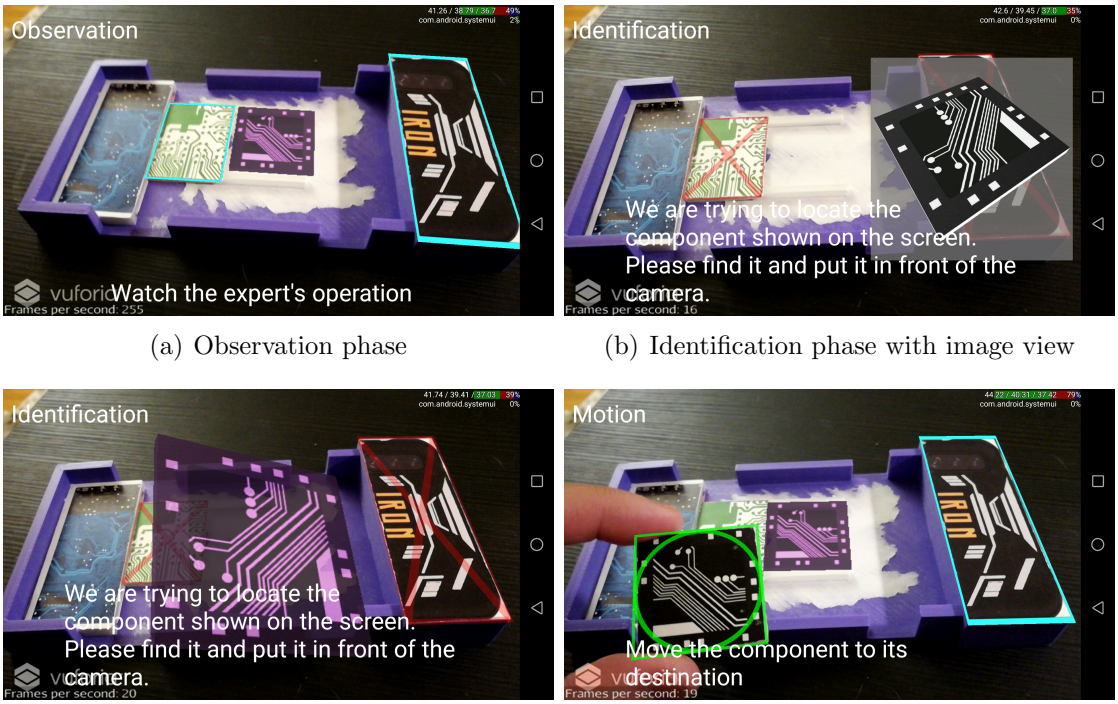
1. Observation: The user should only watch the operations of the expert that is presented on the user's screen by an animation of a 3D virtual component, see Figure 3.2 (a).

2. Identification: An image of the desired component is shown on the screen, and the user needs to bring each potential component into the camera view so that the application can indicate to the user whether it is the correct one. Also, in this phase, the user can switch between the image view and the 3D view of the component, see Figure 3.2 (b) and (c).

3. Motion: The user is supposed to copy the demonstrated motion, see Figure 3.2 (d).

During the entire process, the user only needs to tap a button for switching between the image view and the 3D view of the component in the identification phase. The progress of the task is controlled by the expert.

The expert is also aware of the three phases. Next, I will describe the actions that the expert needs to take during each phase.



(a) Observation phase

(b) Identification phase with image view

(c) Identification phase with 3D view

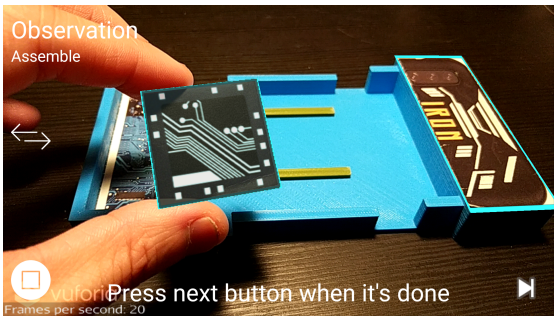
(d) Motion phase

Figure 3.2: Phases on the user side.

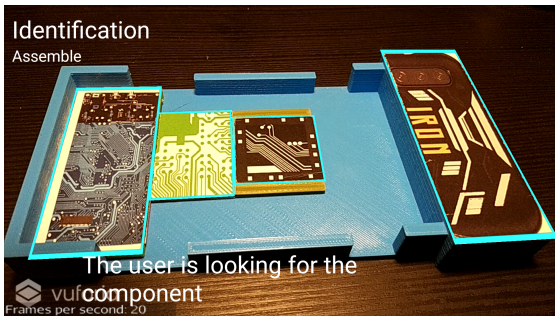
1. Observation: At the beginning of this phase, the expert must tap on the screen to select the currently active component. For example, if the current step is to plug a battery in, the expert should bring the battery into the camera view and tap it on the screen to select it as the active component. The active component is shown as a virtual 3D component on the expert's view. Then the expert is supposed to move the active component to its desired location and make sure that the application has tracked the entire movements, i.e., a blue box that outlines the component indicates that it is being tracked. After demonstrating the desired operation, the expert needs to press the next button to move to the identification phase, see Figure 3.3 (a).

2. Identification: The expert does not need to do anything. After the user finds the correct component, the process goes to the motion phase automatically, see Figure 3.3 (b).

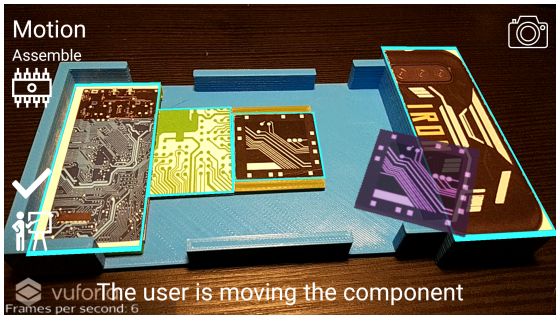
3. Motion: The operation of the user is shown as an animation of a 3D virtual object on the expert's view and the expert needs to decide whether the user has correctly accomplished this step. If the operation is correct, the expert is supposed to press the correct button and start the next step, otherwise, she or he needs to press the feedback button to reset the process back to the observation phase and demonstrate



(a) Observation phase



(b) Identification phase



(c) Motion phase

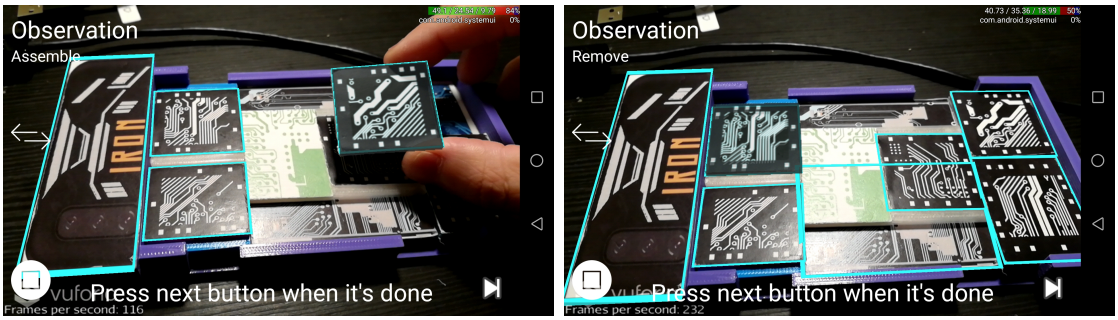
Figure 3.3: Phases on the expert side.

the operation again, see Figure 3.3 (c).

There are three reasons why I designed the instruction schema in this way. First, I only allow a single operation in each step to build a clearer instruction schema. Second, from a remote collaboration perspective, dividing each step into multiple phases results in smoother cooperation between the expert and the user, since it is more apparent to the user what to do in each phase. Last, as will be discussed in Section 3.4.3, my system design supports multiple users, dividing each step into multiple phases facilitates synchronization among the clients.

3.2 Authoring Procedure

In this section, I describe the authoring procedure for a remote expert. My goal of designing the authoring procedure is to enable a remote expert to compose instructions with virtual 3D models in an online manner. In other words, the expert does not need to hardcode any prior knowledge into the software except for the models of the components. There are four steps to create instructions: 1) specify operation type; 2) select the active component;



(a) Assembly operation (b) Disassembly operation

Figure 3.4: Authoring interfaces. The left figure shows the interface for assembly operation, while the right one shows the interface for disassembly operation. The button on the left of the screen is the button of operation type switch. It switches between assembly operation and disassembly operation. The text on the top-left corner is the operation type indicator and shows the current operation type.

3) move the active component; 4) press the next button. Figure 3.4 demonstrates the authoring interfaces.

In a repair task, there are assembly operations and disassembly operations. The visualization of the two types of operations differs. For an assembly operation, the instruction should show the start and destination of the operation. For example, when plugging a battery in, there should be an animation starting from the current position of the battery and ending at the proper slot. For a disassembly operation, the instruction should show the start of the operation while prompting the user to remove the component. The destination of a disassembly operation is unimportant. For example, when taking the back cover off, there should be an animation showing which component should be removed and how to remove it. Therefore, the first step of authoring is specifying whether it is an assembly operation or a disassembly operation. The operation type defaults to assembly. Clicking on the switch button switches the operation type between assembly and disassembly. Experts can view the current operation type of the top-left corner of the screen, see Figure 3.4. Note that the interfaces of the assembly operation and the disassembly operation are similar to each other.

The second step is selecting the active component. As shown in Figure 3.4, there may be multiple components in the view, and an expert must select one as the active component, implying that, at each repair step, the expert shows how to assemble or remove one component. The expert taps on the desired component on the screen to designate it as the active component. For the assembly operation, an expert may bring a new component into the camera view and tap on it. There will be a virtual 3D model shown upon the real

component, which indicates that this component has been selected as the active component. For the disassembly operation, since the component is already being tracked, an expert just needs to tap on it. Note that during the observation phase, the expert can change the active component by tapping on another component. In Figure 3.4, the component held by the hand has been selected as the active component in the left figure, while a component docked on the mobile phone has been selected as the active component in the right figure.

The third step is moving the active component. During the movement, the AR platform captures the trajectory of the motion of the active component. In this way, the user will see an animation that is displayed repeatedly, which shows the trajectory of the motion. This gives the user an intuition of how to operate. During this step, the expert must pay attention not to block the active component to make sure the AR platform can capture the entire trajectory.

The last step is pressing the next button to finish the authoring. All the four steps of the authoring procedure are in the observation phase. After finishing the authoring, the process goes into the identification phase.

The main differences between the proposed authoring procedure and the procedure in other AR platforms are two-fold. First, I enable an expert to perform the operations and create instructions in a natural way. There is no complex interface that requires an expert to modify a virtual scene. Second, the instructions shown to users are virtual 3D models instead of 2D sketches.

3.3 Hybrid Remote Rendering

As discussed in the surveys by Syberfeldt et al. [98] and Palmarini et al. [73], the devices for AR applications are relatively weak in capacities: power consumption, processing power, stability of telecommunication, memory, etc. I believe that cloud computing techniques are essential to AR devices with portability, comfort, and long battery life. More specifically, the use of cloud computing decreases the hardware requirements of AR devices, and therefore reduces their weight and increases battery life. With a lighter AR device, users can wear or hold it for a long time without feeling uncomfortable. In my implementation, I used the remote rendering approach described in this section to relieve the rendering burden of the end devices.

My proposed remote rendering method is a client-server architecture and maintains two versions of models: low-fidelity models and high-fidelity models. Low and high fidelity models differ with respect to the number of polygons and resolution of textures.

On the client-side, the mobile device renders low-fidelity models that have fewer polygons and lower fidelity of textures. In contrast, on the server-side, the workstation renders high-fidelity models that have more polygons, higher fidelity of textures and higher quality rendering effects. Hence key models are rendered on the server and captured in images that are sent to the client as images or a video stream. In my implementation, I used FFmpeg [100] to encode and decode the video streams, and used H.264 [110] as the codec. The videos were transmitted over the Transmission Control Protocol (TCP). I define key models as those that the user is interacting with. These models can be identified from interaction information sent to the server. Additional key models may be specified in advance by the developers based on application-specific criteria. It is suitable for AR platforms since only important objects are rendered in AR applications. And for repair tasks, users usually only focus on one or a small number of components that they are operating on.

The proposed hybrid remote rendering approach is designed as a general approach that can be used in various applications, e.g., remote assistance, games, and virtual tours. However, in this thesis, I integrate it with the AR remote assistance platform. The integration details are elaborated in Section 3.4.1.

3.3.1 Client-Server Prototype Design

My proposed system aims at providing high-quality rendering on less powerful mobile devices, while minimizing the amount of data transferred via the network. It has a client-server architecture. Since each expert or user is treated as a "client" in such a client-server architecture, I must enable multi-client cooperation to support remote assistance for repair tasks.

My system is inspired by Levoy [59], Crassin et al. [22] and Liu et al. [60]. It uses a hybrid approach incorporating both local and remote rendering, which stores the low-fidelity key models on the client-side and leverages the local rendering capacity to produce lower quality rendering results. Conversely, the high-fidelity renderings of key models are sent from the server to the client and overlaid upon the locally rendered frames. Accordingly, the data transferred via the network is minimized less information is encoded and sent via the network compared to a CMR solution. An analysis of bitrate reduction of the proposed hybrid remote rendering is in Section 4.1.

There are three challenges in realizing multi-client support within the context of the above described system. First, it is required that the server must not be blocked by any of the clients. Second, each client has its own view that may be different from all of the other

clients; this can result in a performance issue as a scene needs to be rendered multiple times in a single animation loop. Third, since only key models are rendered and sent by the server, occlusion needs to be addressed since the environment models that are closer to the viewpoint may not be rendered on the server but just locally.

I address the first challenge by mandating that the server interacts with each client independently. More specifically, I enable the server to receive commands from and send to every client separately, so that if a client becomes unresponsive, the server is not blocked. The second issue is alleviated by implementing a "render-on-demand" function on the server, implying that the server renders the scene for a client only when the client requires a new frame. I address the third challenge through a two-pass rendering process. In the first pass, the entire scene is rendered in low-fidelity on the server. Then the colour buffer is cleared before the second pass where only the key models are rendered in high-fidelity. In this way, the second pass is rendered with the depth information obtained from the first pass.

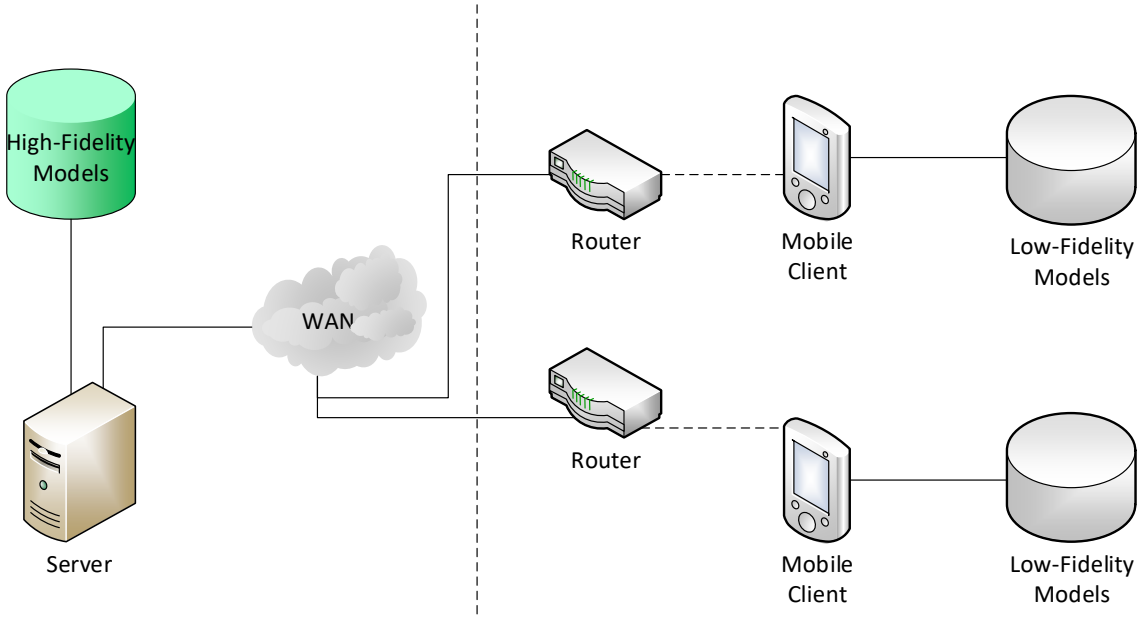


Figure 3.5: System architecture

As depicted in Figure 3.5, in this system, all clients are connected to the rendering server and send interaction commands to the server. On the server side, the application status changes according to the commands received from all clients. The application status changes are synchronized with all clients. In this way, the clients are able to cooperate with each other. In other words, when a remote expert moves a component, the virtual 3D

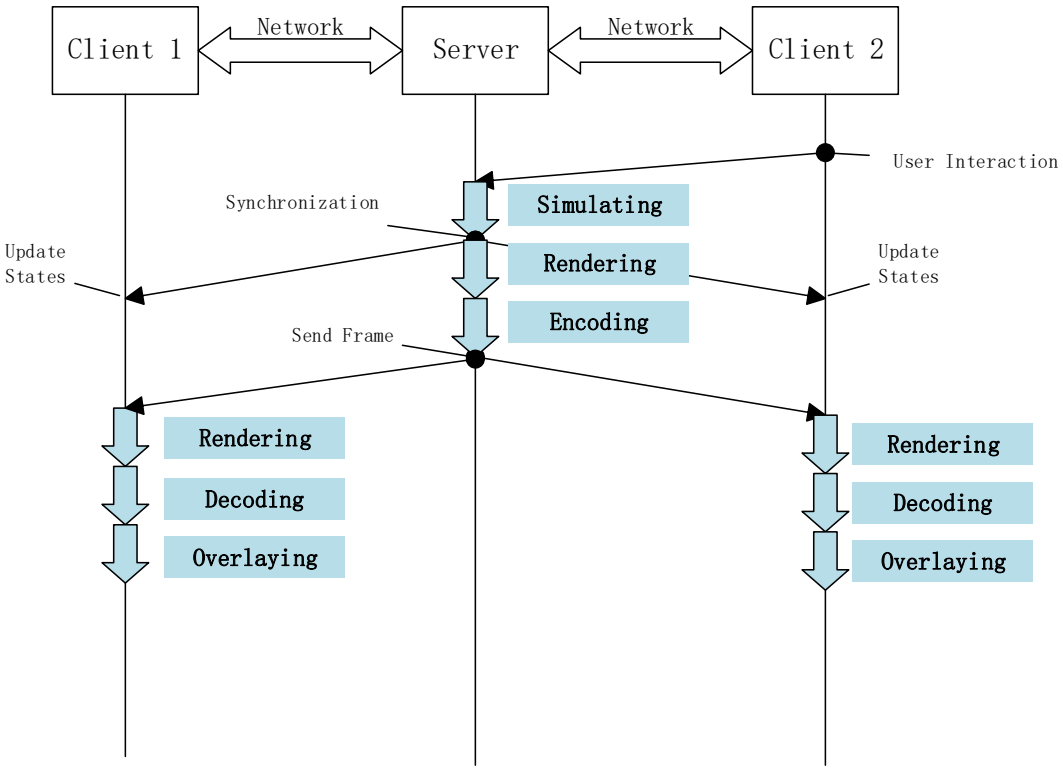


Figure 3.6: Workflow of communication between the server and the clients

model on the user's view is also moved. To minimize the amount of transferred data, only the key models are rendered in high-fidelity and the server only sends the pixels depicting this key model. The other models are still rendered locally. In a repair task, only the active component chosen by the expert is treated as a key model.

Figure 3.6 illustrates the sequence of messages communicated between the server and the client processes. It shows the tasks that are performed by the server and clients to process one frame. If the commands sent from any client influence the simulation states of the server, the latter synchronizes the state changes among all clients to ensure that all of them maintain a consistent state. Moreover, the server renders and encodes the frames for all the clients. Upon receiving the high-fidelity frame, the client decodes and overlays it on the locally rendered frame.

3.3.2 Process Behavior

The proposed system consists of one server and multiple clients. Each client can be either a remote expert or a local user. Their communications are transferred through the server

so that the statuses of all clients are synchronized. Figure 3.7 shows the behavior of the server process. In each loop, the server updates the simulation status of the scene and receives commands from all clients. Next, the server sends the received commands to all clients to ensure synchronization between them. On the server side, each client has its own view that is independent from those of the other clients. The server only renders a new frame upon request from the client.

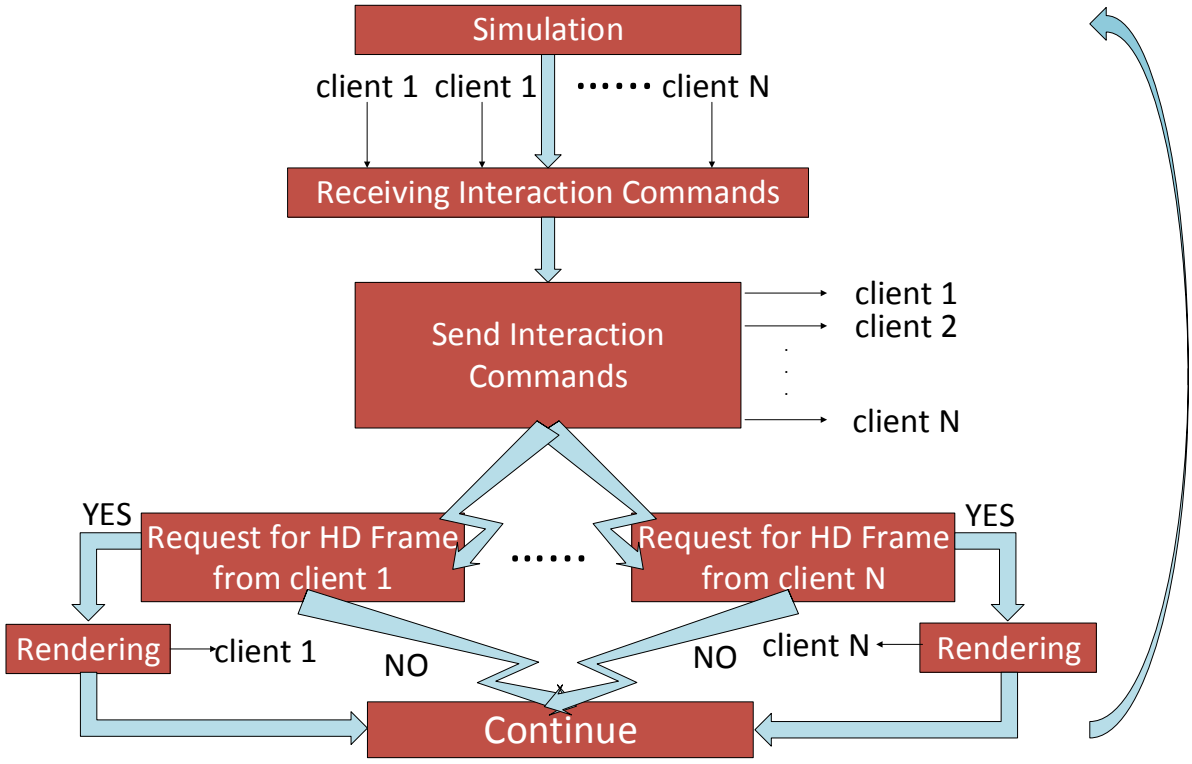


Figure 3.7: Server-side workflow

Figure 3.8 shows the behavior of the client processes. In each loop, the client receives commands from the server and adjusts the simulation status accordingly. Then, it sends its user interaction commands to the server. In each iteration, the client receives the high-fidelity frame from the server that it has requested in the previous iteration. The client has simplified models stored locally, it renders the scene in every iteration. The high-fidelity frames acquired from the server are overlaid upon the locally rendered frames. Once done, it sends the frame request to the server. In this way, the server is not required to render frames for all clients in every loop, instead it renders a frame when it is needed.

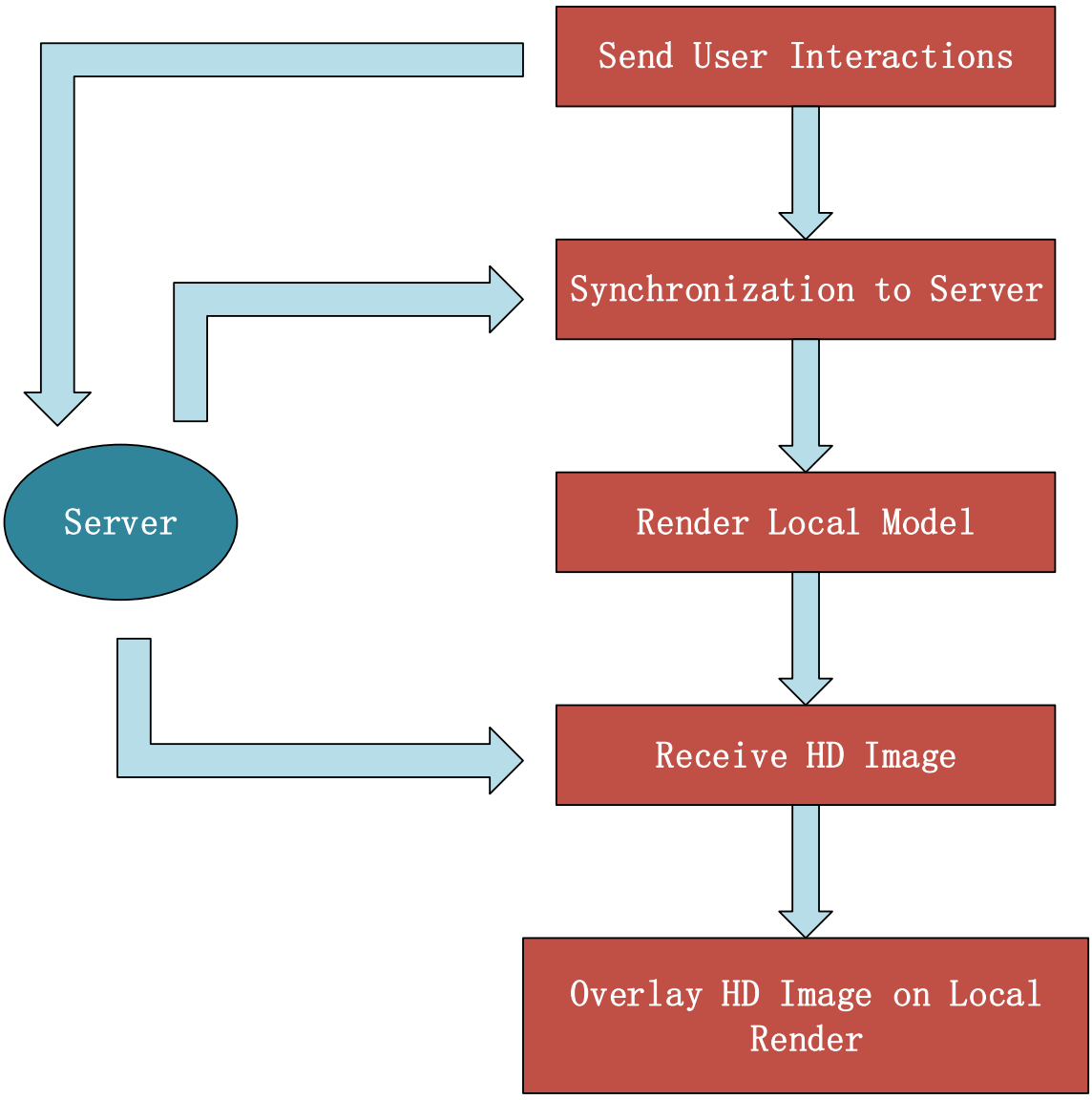


Figure 3.8: Client-side workflow

3.3.3 Two-pass Rendering

As mentioned in Section 3.3.1, I use a two-pass rendering process on the server side. The frames sent to the client depict only high-fidelity key models. However, this is insufficient for a valid view of the 3D scene since other scene content may occlude parts of the key models. I propose a two-pass rendering process to address this issue.

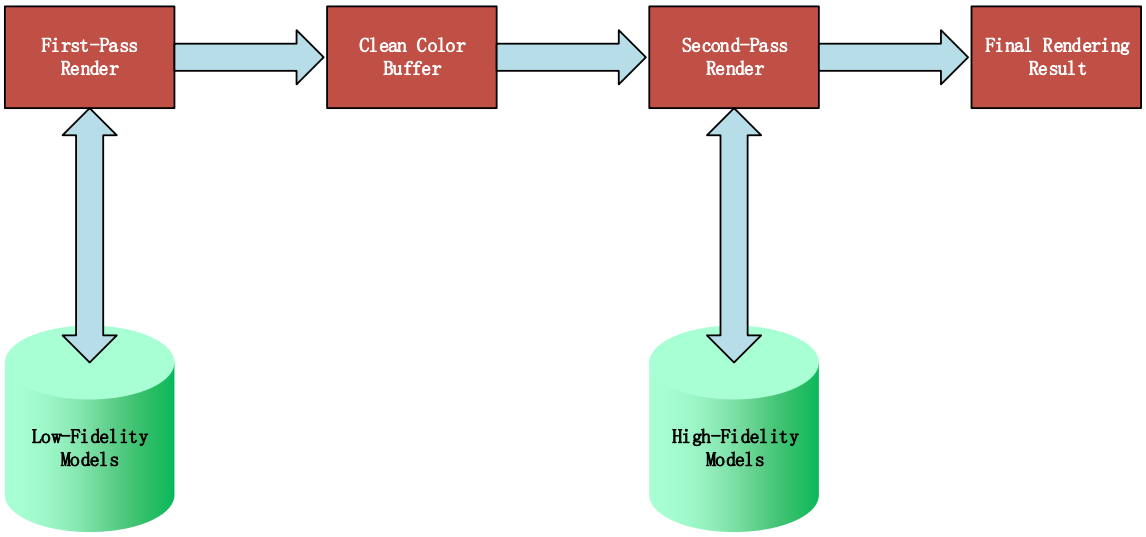


Figure 3.9: Two-pass rendering

As shown in Figure 3.9, in the first pass, all models are rendered in low-fidelity except for the key models. Then the color buffer is cleared and only the depth buffer is preserved. In the second pass, only the high-fidelity key models are rendered, while considering the depth information obtained from the first pass rendering, therefore allowing the occlusions are preserved in the final rendered scene.

3.4 System Design

3.4.1 Communication Schema Design

My implementation is designed as a client-server model where both the expert and the users run as clients of a common server. As shown in Figure 3.10, the system consists of three services: the object tracking service, the rendering service, and the encoding service.

The object tracking service tracks the poses of all the registered components from a live video captured by the camera. Since I use the image target tracking feature of the Vuforia Engine [75] to track the components, before using the proposed method, the users need to register the components of a product. To register the components, the textures of the components need to be stored in the system defining what will be tracked. Although the object tracking service could be implemented on the server-side or the client-side, in the implementation used in my experiment, I implemented it on the client-side, i.e., mobile devices (the cyan dash lines in Figure 3.10), due to the limited compatibility of the Vuforia Engine.

The rendering service decides which virtual components need to be rendered for each client and then renders them. For example, during the motion phase, for each user, the rendering service renders the virtual component showing how a user should move a component, while for the expert, the rendering service renders the virtual components representing the operations of all the users. Note that there is a database that contains the 3D models of all the components connected with the rendering service. This enables us to use a remote rendering method to reduce the rendering demand on the clients, see Section 3.3. Then the rendering results are encoded by the encoding service and sent to the clients. The clients overlap them on their views to create the augmented reality experience. The rendering service and the encoding service were implemented on the server-side (the purple dash lines in Figure 3.10).

The communication schema design of my proposed method extends the hybrid remote rendering approach I propose in Section 3.3. As mentioned in Section 3.3, my proposed remote rendering approach is designed as a general approach. To integrate it with the AR platform, I must implement the following additions or extensions. First, I add an object tracking service that is not present in the original hybrid remote rendering method to track the components. It runs on the client side as a local service and sends the transformation matrices of all components in the camera view to the server. Second, in addition to a user's taps, the interaction commands also come from the object pose tracker. When a component is moved, the object pose tracker sends a series of commands to the server, which indicates the movement trajectory of a component. Third, the clients are divided into two types: expert and user. Depending on the type, a client is limited to send a specific set of interaction commands. For instance, when an expert has finished authoring, the client sends a command "instruction authoring finished" to the server, but a user is not allowed to send such a command. Last, rather than manually defining the key models, the active component is treated as a key model. The rest of the models, i.e., the environment models, are not rendered, since the environment in AR is a real environment.

However, if the network is unstable, and the delay for receiving a rendered frame becomes unacceptable, the key model can be rendered low-fidelity locally. Therefore, the proposed platform is more robust to the fluctuating bandwidth of mobile networks. The server-side and client-side workflows are kept the same as described in Section 3.3.2.

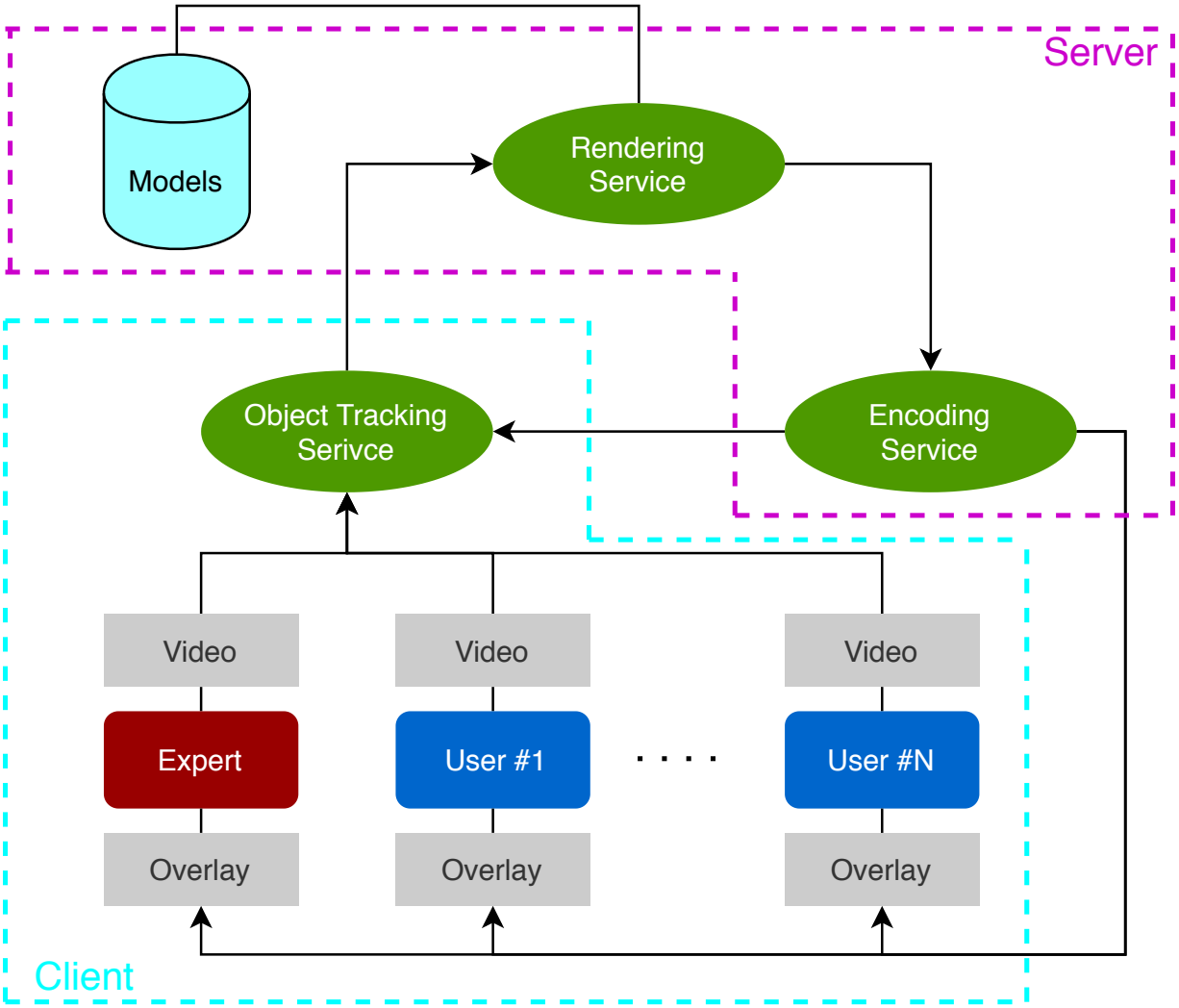


Figure 3.10: Communication Schema. The server consists of the rendering service and the encoding service, while the client contains the object tracking service.

3.4.2 Multi-object Tracking

I used the Vuforia Engine [75] to track the positions and orientations of all the components on the view of the camera. It is a commercial augmented reality development platform that provides various methods for tracking images, 3D shapes, and markers. It is among the most-used AR Software Development Kits (SDKs) [4]. Besides the Vuforia Engine,

there exist many other popular AR SDKs that provide the functionalities of object pose tracking.

Apple provides the developers with its AR SDK, ARKit [47], while Google announced ARCore [33] for the Android platform in 2018. There also exist many third-party AR SDKs in the market and open source community. For example, Dr. Hirokazu Kato created an open-source software library, ARToolKit, in 2000. It is still widely used to complement many augmented reality experiences [48, 49]. DeepAR [24] is a commercial AR programming library that enables the developers to detect faces, emotions, and rigid objects. Those AR SDKs provide developers with various object pose tracking methods that include marker-based methods, image-based methods, and model-based methods. I chose the image-based method of the Vuforia Engine in my implementation since it is more robust than its model-based method. The reason why I did not use the marker-based method of the Vuforia Engine is that I wished to print textures of electronic components and make the components used in my experiment look real.

Blanco-Pons et al. [4] quantitively evaluated the Vuforia Engine and ARToolKit [20]. The results showed that the Vuforia Engine is faster when it comes to tracking time and stable during illumination change, so it is more suitable for indoor scenarios. Blanco-Novoa et al. [3] also conducted a quantitive evaluation on the Vuforia Engine and ARToolKit. The results demonstrated that the Vuforia Engine has a shorter tracking distance than ARToolkit, but confirm that the Vuforia Engine is less affected by illumination changes. In my experiment, the illumination of the components changes due to the user's hands shadowing; therefore, the stability of the Vuforia Engine in illumination change may benefit my application. According to the evaluation conducted by Blanco-Novoa et al. [3], the Vuforia Engine can recognize and track an 8 $cm\times$ 8 cm square image from 1.2 m to 1.7 m under various lighting conditions. This is sufficient for my experiments because the distance between a component and the camera is less than 30 cm and the dimension of the smallest component in my experiment is 3.2 $cm\times$ 2.8 cm. However, there are still some limitations to this object tracking method, which I will discuss in Section 6.2.

3.4.3 Multiple Users

Both, the communication schema and remote rendering method used in my system, enable remote assistance for multiple users. Therefore, an expert is able to concurrently assist more than one user remotely. However, the instructions given to all users are the same. Hence, an expert can assist multiple users concurrently if they are facing the same problem.

To support multiple users, the interaction design should be modified. First, the expert should be able to choose an active user to monitor, since monitoring multiple users at the same time implies a mixture of 3D virtual objects that represent the operations of different users. For example, I can add a drop-down list for the expert to choose an active user and monitor his or her operations. Second, when a user makes a mistake, the expert should be able to give feedback to this user individually. Third, the expert clicks the correct button and proceeds to the next step only if all the users have completed this step correctly.

3.5 Summary

In this chapter, I describe the online AR-based remote assistance platform for repair tasks. This platform requires the expert and the user to operate on the same product for repair simultaneously. The platform tracks the components of the product on the expert's side so that the operations of the expert can be displayed and overlaid as 3D virtual objects on the user's view. It enables the expert to create 3D representations that demonstrate 6 DoF movements in real-time. It also enables the expert to apply geometrical and physical constraints naturally. By using a remote rendering schema, we can relieve the rendering burden of the AR end devices. First, the details of the interaction design was provided. I described the three phases of each assembly or disassembly step and listed the differences of each phase for the expert and the user. Second, a new authoring procedure for remote experts was described. I designed the authoring procedure as four steps and two modes: assembly mode and disassembly mode. Third, a new hybrid remote rendering method was proposed and integrated into the AR platform. This remote rendering method was first published in 2019 [95]. I integrated it in my proposed AR platform. Last, the details of the system design was provided. I described its three aspects: communication schema, multi-object tracking, and multiple users.

Chapter 4

Experiments for the Hybrid Remote Rendering Method

In this chapter, I evaluate the proposed hybrid remote rendering method of Section 3.3 with two experiments. The first experiment aims to analyze the effectiveness of the method in reducing network bandwidth requirement and interaction latency. The second experiment is a user study that investigates the influence of various factors of the hybrid remote rendering method on user experience.

4.1 Quantitative Analysis

The proposed hybrid remote rendering method offloads a part of the rendering task to the remote server. More specifically, only key models are rendered remotely to minimize network bandwidth requirements and interaction latency. In this section, I demonstrate the effectiveness of the proposed method in reducing interaction latency and network bandwidth requirement by selectively rendering high-fidelity models.

To overcome the limitations of mobile devices in rendering high-fidelity models, designers often use one of two approaches:

1. Local-only rendering: Reduce the fidelity of the models and render them locally on mobile devices.

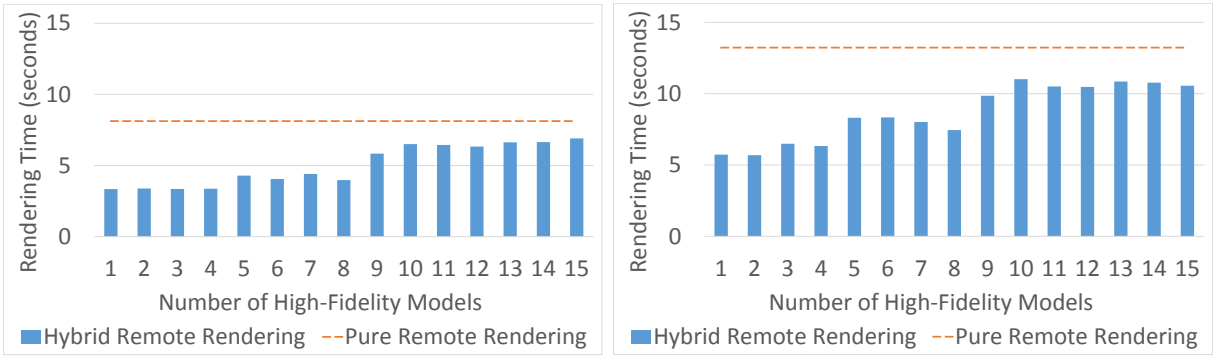
2. Server-only rendering: Render the scene remotely on a server and transfer to the mobile device as a video stream.

I designed and conducted four sub-experiments to evaluate the effectiveness of the proposed method. In all sub-experiments, I use the same models: one environment model and fifteen object models. I created a low-fidelity and a high-fidelity version of each. I developed a simulation program that depicts a 3D scene where all the object models shuffle randomly in front of the camera. This allows us to conduct a quantitative analysis on the interaction latency and network bandwidth requirement of the program while minimizing the effect of object positions and movements on the results. I ran each sub-experiment five times and used the average of the measured metrics for the analysis. I compare my results to the server-only approach, which renders every model in high-fidelity. This is the approach most widely applied when the client is incapable of computationally handling the rendering task of an application. In this experiment, I considered two resolutions for the encoded video: 640×480 and 1440×900. I chose these two resolutions because they were the minimum and maximum resolutions supported by my experiment platform, respectively. I would like to assess whether the proposed method reduces the interaction latency and the network bandwidth requirement.

4.1.1 Interaction Latency

The interaction latency consists of four components, i.e., the network transmission latency, rendering time, encoding time and decoding time. These components are influenced by different factors. The network transmission latency is mainly influenced by the network condition and the geographical distance between the server and the client. This can be relieved by distributing servers in different geographical locations [86]. However, this is out of the scope of this paper. The rendering time is affected by the complexity of the scene, while the encoding and decoding time are affected by the complexity of the rendered frames. Moreover, the rendering time and encoding time are affected by the computational capacity of the server. To minimize the effect of the computational capacity on the results, I ran all experiments on the same machine (CPU: Intel i5 4950 - four 3.3 GHz cores, GPU: NVidia GTX 960).

Figure 4.1 shows the comparison of the rendering time between the hybrid and server-only remote rendering methods. For the server-only rendering approach, I render all of the models in high-fidelity. However for the proposed method, I only render a certain number of models in high-fidelity, while I render the rest models and the environment in low-fidelity. I recorded the total rendering time for 10000 frames. The dash lines in Figure 4.1 shows the rendering time of 10000 frames with the server-only method.



(a) Resolution 640 × 480

(b) Resolution 1440 × 900

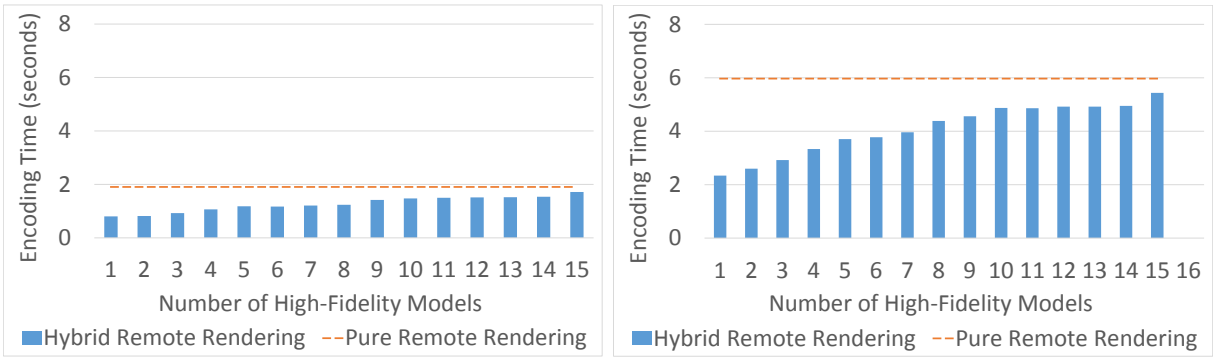
Figure 4.1: Rendering time comparison. The dash line shows the rendering time of 10000 frames with the server-only method, while the bars demonstrate the rendering time with the proposed hybrid remote rendering method with different number of high-fidelity models. The vertical axis shows the rendering time in seconds. Figure (a) shows the results for resolution 640 × 480, and Figure (b) shows the results for resolution 1440 × 900. The measures reflect the average of five runs of the experiment.

As we can observe from the results of Figure 4.1, the proposed method is able to save rendering time, e.g., more than half with one high-fidelity model. By increasing the number of high-fidelity models, the rendering time of the proposed method gradually increases. By rendering more models in high-fidelity, the complexity of the scene increases. However, the proposed method does not result in a rendering time as high as the server-only approach even when all models in the scene are rendered at high-fidelity. That is because the environment model is still rendered in low-fidelity.

Note that in Figure 4.1, sometimes the rendering time drops when the number of high-fidelity models increases. For example, the rendering time decreases from 5 to 6 and from 7 to 8. This is caused by the random selection of high-fidelity models. As mentioned in Section 4.1, I used fifteen object models with two versions (low-fidelity and high-fidelity). The number of polygons in a model is different from other models in both the low-fidelity and the high-fidelity versions. In each experiment run, a certain number of high-fidelity models were randomly selected. Therefore, it is possible that the number of polygons will decrease with an increase in high-fidelity models

I use FFmpeg [100] to perform the encoding and decoding tasks. More specifically, I use H.264 [110] as the codec. The default settings of FFmpeg are used. However, I disable the lookahead feature to minimize the latency of the network video streaming.

Figure 4.2 and Figure 4.3 demonstrate the comparison of the encoding and decoding time between the two methods, and I recorded the total encoding and decoding time for



(a) Resolution 640×480

(b) Resolution 1440×900

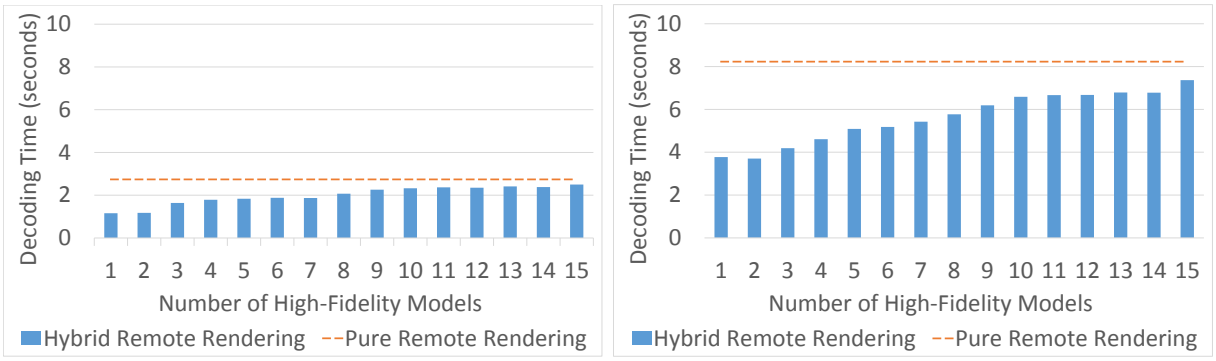
Figure 4.2: Encoding time comparison. The dash line shows the encoding time of 100 frames generated by the server-only method, while the bars demonstrate the encoding time with the proposed hybrid remote rendering method with different number of high-fidelity models. The vertical axis shows the encoding time in seconds. Figure (a) shows the results for resolution 640×480, and Figure (b) shows the results for resolution 1440×900. The measures reflect the average of five runs of the experiment.

100 frames. The dash lines in Figure 4.2 and Figure 4.3 show the encoding and decoding time of 100 frames generated by the server-only method. The proposed method is able to minimize the encoding and decoding efficiently. With resolution 640×480, it saves 11 ms encoding time and 16 ms decoding time for every frame with one high-fidelity model. For resolution 1440×900, it saves 36 ms encoding time and 45 ms decoding time. With the larger frame resolution, the proposed method saves more encoding and decoding time. The encoding and decoding time approaches that of the server-only approach as the number of high-fidelity models increases.

4.1.2 Network Bandwidth

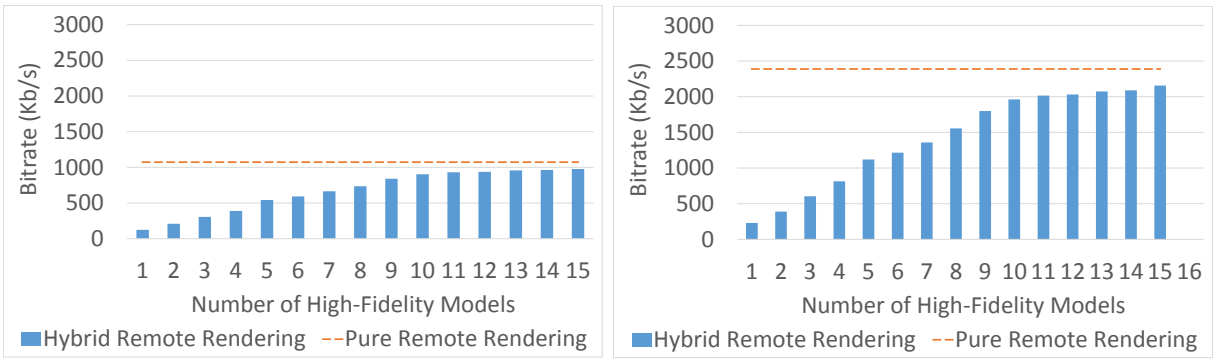
I conducted an experiment to show the amount of network bandwidth saved by the proposed method. In each run of the experiment, I recorded the bitrate of 100 frames. In Figure 4.4, we can see that with one high-fidelity model, the proposed method saves $949Kb/s$ of the network bandwidth ($124Kb/s$ vs. $1073Kb/s$) for the resolution 640×480, and saves $2159Kb/s$ for the resolution 1440×900 ($230Kb/s$ vs. $2389Kb/s$). Similar to the experiments discussed above, the bitrate also increases alongside the number of high-fidelity models. But the bitrate with all of the models rendered in high-fidelity is still less than that of the server-only method. That is because the environment is still rendered in low-fidelity.

The proposed method is effective in reducing the encoding time, decoding time and bitrate with the same encoding parameters as the frames sent from the server to the client



(a) Resolution 640×480 (b) Resolution 1440×900

Figure 4.3: Decoding time comparison. The dash line shows the decoding time of 100 frames generated by the server-only method, while the bars demonstrate the decoding time with the proposed hybrid remote rendering method with different number of high-fidelity models. The vertical axis shows the decoding time in seconds. Figure (a) shows the results for resolution 640×480, and Figure (b) shows the results for resolution 1440×900. The measures reflect the average of five runs of the experiment.



(a) Resolution 640×480 (b) Resolution 1440×900

Figure 4.4: Bitrate comparison. The dash line shows the bitrate of the video generated by the server-only method, while the bars demonstrate the bitrate of the video generated by the proposed hybrid remote rendering method with different number of high-fidelity models. The vertical axis shows the bitrate in Kb/s. Figure (a) shows the results for resolution 640×480, and Figure (b) shows the results for resolution 1440×900. The measures reflect the average of five runs of the experiment.

contain less high frequency components, which results in a more effective compression.

In summary, I have conducted four experiments to quantitatively evaluate the effects of the proposed method on interaction latency and network bandwidth. The experiments show that my method reduces interaction latency and network bandwidth. The reduction of the interaction latency is achieved by decreasing the rendering time, encoding time and decoding time. The reduction of the bitrate is achieved by decreasing the complexity of the frames. If the number of high-fidelity models increases, the interaction latency and network bandwidth approach the server-only rendering method. However, in a practical application, the number of objects that the user needs to pay attention to or interact with is not typically large. For instance, in a repair task, the user typically concentrates on the component she or he is currently operating on. Therefore, I conclude that the proposed method reduces the interaction latency and network bandwidth.

4.2 User Study

Local-only and server-only methods risk reducing the user's Quality of Experience (QoE). For the local-only rendering approach, the user is interacting with models that are not rendered at their optimal quality. Fine details in the model may be missing, thus reducing the aesthetics of the scene. Also, for some applications, these details may carry important information. For instance, it is important for users in a repair task to quickly distinguish between similar components. Objects rendered as low-fidelity models might not be immediately discernible. For the server-only approach, QoE might be affected by a long latency between the user's actions and application's response. This would be mostly caused by the interaction latency between client and server. Furthermore, bandwidth limitations can cause interruptions in the video stream. The proposed approach strikes a balance between local-only and server-only rendering. It renders only key models remotely at high-fidelity while other models are rendered locally in low-fidelity.

Previous work has explored how various factors affect the user experience. Hong et al. [44] conducted a subjective user study to quantify the effect that video bitrate and frame rate have on QoE in cloud gaming. They concluded that, with the bitrate fixed, the QoE increases when the frame rate decreases, while, with the frame rate fixed, the QoE increases when the bitrate increases. Slivar et al. [91] use the Steam In-home streaming platform [21] as a case to study how the frame rate and bitrate influence the QoE. They model the QoE as a combination of two aspects: perceived graphics quality and perceived game fluidity. In the work by Suznjevic et al. [97], five factors (i.e., latency, jitter, packet

loss, frame rate and frame resolution) are studied in terms of their effects on the QoE of NVIDIA GeForce Now platform [20]. Hemmati et al. [38] developed an algorithm that minimizes bitrate by not rendering unimportant models. My method leverages a similar idea by rendering key models in high-fidelity and environment models in low-fidelity. But the difference is that my method renders the environment models locally instead of not rendering them at all.

My method mainly targets at AR remote assistance for repair tasks, but it can also be applied in other types of applications, such as gaming and virtual tours. The expectations of users in terms of graphics quality and application fluidity may depend on the different types of applications. For instance, a user of an AR remote assistance platform may benefit from a clear rendering of the active component, while the players of a game may focus on the quality of special effects. Thus, I measure how difficult it is to accomplish a given task. I am interested in the effects of the key model fidelity and environment fidelity on the perceived difficulty of task completion. I name this measure the Difficulty of Task (DoT). I measure DoT in two ways: Objective DoT (ODoT) and Subjective DoT (SDoT). The ODoT is a measure of how well the users can accomplish a task. It might be measured by a score or a level calculated by the application. The SDoT is a measure of the users' opinion on the task difficulty and can be answered by a subjective questionnaire.

The user study was approved by the Research Ethics Board at the University of Ottawa (File Number: H12-16-05). The effectiveness study was informed by a pre-trial study discussed next.

4.2.1 Pre-Trial Study

A major goal of the effectiveness study is to analyze the effect of 3D model fidelity on the DoT. However, in the proposed method, there are two types of models, high-fidelity models and low-fidelity models. Those models will be used in the effectiveness study (see Section 4.2.2). I have prepared 15 high-fidelity models and simplified them to obtain low-fidelity models using QSlim [28].

Objective

The objective of the pre-trial study is to decide on the number of polygons for the low-fidelity models. If it is too low, the participants will not be able to recognize the object that the model represents. Hence, I want to find out at which point the object represented by the model becomes discernible to the majority or all participants

Participants

I recruited five participants, four males and one female, for the pre-trial study, a sufficient number of participants given the simple objective. None of the participants were visually impaired.

Independent Variables and Dependent Measures

The experiment was conducted with one independent variable: the number of polygons of the surface mesh in the 3D model. The independent variable has 50 levels. For the first level, the model is composed of 20 polygons. For each of the 49 subsequent levels, the number of polygons is increased by 10%. I measure the number of polygons necessary for a subject to recognize the object for each 3D model.

Procedure

The experiment was conducted using a program that shows 15 models at different levels of fidelity. I created 50 levels of fidelity for each model by starting with 20 polygons and increasing its polygon count by 10% for every level. I generated the model of each level by simplifying the same complex mesh using QSlim [28]. The program showed the participants a model on the screen and asked them to select the object it represents from a side menu by choosing among dog, cat, horse, or unknown. Participants were instructed to choose the unknown option if they were unsure. I started with a model with a very low polygon count (20 polygons). Every time the participant selected an answer, I increased the polygon count to the next level adding more details to the model. The order through which the models were shown to the participants was randomized to avoid biasing the results.

Results

Table 4.1 contains the data collected from the pre-trial study. The number of polygons above which a participant ceases to make mistakes (i.e., the participant chooses the correct answer consistently at all further levels of fidelity). The data are used to decide on the number of polygons for the low-fidelity models in the effectiveness study. To eliminate the outliers, I select the second highest number among all participants for each model.

Table 4.1: Data collected from the pre-trial study, which represents the number of polygons above which a participant ceases to make mistakes are shown. However, in one case, one of the participants does not recognize the model even at full resolution (marked by a dash). The numbers in bold are those selected as the final number of polygons of the low-fidelity models for the effectiveness study.

	Participant #1	Participant #2	Participant #3	Participant #4	Participant #5
Model #1	148	22	57	51	**111**
Model #2	162	57	**422**	22	422
Model #3	122	383	134	22	**238**
Model #4	75	**148**	-	69	134
Model #5	162	51	32	22	**134**
Model #6	**1095**	179	464	1940	464
Model #7	995	1603	**1325**	32	422
Model #8	75	162	148	91	**148**
Model #9	75	83	216	122	**196**
Model #10	464	262	748	216	**562**
Model #11	422	1204	**422**	288	383
Model #12	47	101	**134**	22	134
Model #13	22	47	510	29	**238**
Model #14	238	22	348	**317**	162
Model #15	69	83	**75**	75	51

4.2.2 Effectiveness Study

The effectiveness study is designed to address the following research questions:

1. What is the effect of key model fidelity on the DoT?
2. What is the effect of environment model fidelity on the DoT?
3. What is the effect of interaction latency on the DoT?
4. What is the effect of network jitter on the DoT

I ask the participants to interact with a 3D mobile application that prompts the user to respond rapidly to visual stimuli. This is a stand-in for a variety of applications that require the user to react swiftly to changing aspects in a 3D environment. Examples of such applications are games, flight simulators, military training systems, etc. I compare my results to conventional server-only and local-only rendering approaches. In my user study, I do not consider any network-aware adaption strategies of the video stream performed by the remote server.

(a) Sub-task of navigation (b) Sub-task of recognition

Figure 4.5: Screenshots of the test application. During one run of the test application, the two sub-tasks are repeated ten times with different animal models. (a) Sub-task of navigation. In the test application, users need to pan and tilt to align the camera viewfinder with the orange plate by following the direction of the compass. If there exist interaction latency and jitter, users will feel them when panning and tilting. (b) Sub-task of recognition. After aligning the camera viewfinder with the plate, an object appears. The object is not static, but rotating and moving around instead. As similar to the sub-task of navigation, the movement and rotation of the object is also influenced by interaction latency and jitter.

In this section, I only want to evaluate the effectiveness of the proposed remote rendering method without the influence of the other parts of the AR platform. I designed the test application using an independent scenario. However, it is closely related to remote assistance for repair tasks. The test application I use in my experiment depicts a 3D environment representing a virtual room. Guided by a compass, the user must navigate (pan and tilt) the environment to reach a destination at which an object will appear and disappear within a short period of time. A menu prompting the user to specify the type of that object is brought up after an object appears, as shown in Figure 4.5. This test application consists of two sub-tasks: navigation and recognition. These two sub-tasks are very common in an AR remote assistance platform for repair tasks. For instance, when performing a repair task, users often need to change the viewpoint to inspect a product from different directions and distances. Furthermore, fast and accurate recognition of a component is the key to performing an assembly or disassembly operation correctly, since there may be many components involved and some of them may have a similar appearance. Therefore, the test application should reflect the influence of various factors of the remote rendering method in a repair task.

Hypothesis

I devise eight null hypotheses to test in the experiment, see Table 4.2. The null hypotheses are defined for each measurement factor. In those null hypotheses, I assume all four factors do not have any effect on the ODoT nor the SDoT. Rejecting a null hypothesis indicates that the corresponding factor has a statistically significant effect on the dependent measure.

Table 4.2: Hypotheses.

H_{01}	Key model fidelities do not affect the ODoT
H_{02}	Environment model fidelity does not affect the ODoT
H_{03}	Interaction Latency does not affect the ODoT
H_{04}	Jitter does not affect the ODoT
H_{05}	Key model fidelities does not affect the SDoT
H_{06}	Environment model fidelity does not affect the SDoT
H_{07}	Interaction latency does not affect the SDoT
H_{08}	Jitter does not affect the SDoT

Participants

Fifteen volunteers participated in the effectiveness study, twelve males and three females. None of the participants were visually impaired.

Independent Variables and Dependent Measures

The experiment was conducted with four independent variables: key model fidelity, environment model fidelity, latency and jitter. Key model fidelity and environment model fidelity have two levels: high and low, as described in Section 4.2.1. The variable latency is set to introduce interaction latency. This factor summarizes the effects of network transmission, remote rendering and encoding. In this experiment, the variable latency was simulated locally. Similar to the work by Zhu et al. [115], to model jitter, I use the normal distribution of (4.1) for settings with non-zero latency.

$$p(j|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}^{-\frac{(j-\mu)^2}{2\sigma^2}} \tag{4.1}$$

where j is the value of jitter, $\mu = 77$ ms and $\sigma^2 = 1$.

Combining the four variables results in 20 configurations. Table 4.3 details the test application configurations. The order of the 20 configurations is randomized across participants to avoid biasing the results.

Table 4.3: Test application configurations.

Configuration	Key Model Fidelity	Environment Model Fidelity	Latency	Jitter
1	low	low	0 ms	not applicable
2	low	high	0 ms	not applicable
3	high	low	0 ms	not applicable
4	high	high	0 ms	not applicable
5	low	low	80 ms	not applied
6	low	high	80 ms	not applied
7	high	low	80 ms	not applied
8	high	high	80 ms	not applied
9	low	low	80 ms	applied
10	low	high	80 ms	applied
11	high	low	80 ms	applied
12	high	high	80 ms	applied
13	low	low	120 ms	not applied
14	low	high	120 ms	not applied
15	high	low	120 ms	not applied
16	high	high	120 ms	not applied
17	low	low	120 ms	applied
18	low	high	120 ms	applied
19	high	low	120 ms	applied
20	high	high	120 ms	applied

In this experiment, I collect two types of data: ODoT and SDoT. The ODoT is measured by the number of objects that participants incorrectly recognize. The SDoT is measured using a questionnaire directly integrated within the test application. The questionnaire contains one question: "Please score how difficult it was to complete the task". A 5 point Likert scale is used to collect the answers. A score between 1 to 5 is associated with each question, where 1 represents the option "Very Easy" and 5 represents the option "Very Difficult".

Procedure

I ran the test application 20 times for each subject. I varied the key model fidelity, environment model fidelity, latency and jitter across these runs. Table 4.3 details the test

application configurations for the 20 runs. The order of the 20 configurations was randomized across participants to avoid biasing the results. For each configuration, ten 3D objects appeared. So the maximum score of the ODoT for each configuration is 10 and the minimum score is 0. As mentioned in Section 4.2.2, I also collected SDoT during the experiment. After each run of the test application, the participant was asked to fill the SDoT questionnaire.

As shown in Table 4.3, I prepared four types of scenes: a scene with only low-fidelity key models and low-fidelity environment models, a scene with low-fidelity key models and high-fidelity environment models, a scene with high-fidelity key models and low-fidelity environment models and a scene with high-fidelity key models and high-fidelity environment models. I simulated three different interaction latencies: 0 ms, 80 ms and 120 ms and I also consider network jitter.

Results

To understand how each factor affects the SDoT and the ODoT, I use analysis of variance (ANOVA) to interpret the experimental data.

I use a four-way ANOVA test to investigate the effect of all four factors. However the factor combination of latency and jitter is not complete, since jitter is not applicable when the latency is 0 ms. So I exclude the latency level 0 ms in the four-way ANOVA test. As shown in Table 4.4, the four-way ANOVA considers the four factors and their levels, except for 0 ms of latency.

I also analyze the effect of the latency factor and its interaction with key model fidelity and environment model fidelity. For this analysis, I resort to a three-way ANOVA for the combination of latency, key model fidelity and environment model fidelity. Table 4.4 shows the factors of the three way ANOVA.

Table 4.5 shows the grand means and standard deviations of the ODoT measure for all factors, respectively. The ODoT refers to the number of objects the user did not recognize, hence higher scores represent higher objective difficulty of task. Since I employ two tests of significance (four-way and a three-way ANOVA), I calculate two grand means of the ODoT for each factor level. Four-way ANOVA does not consider the results of configurations 1 to 4 of Table 4.3 (since it does not consider 0 ms latency as previously explained). Table 4.5 shows that the ODoT varies with different levels of each factor. However, not all factors have an effect on the ODoT. We can see that in both, the four-way and the three-way ANOVA, participants are able to reduce the number of incorrectly recognized objects with

Table 4.4: Factors and levels of the ANOVA tests.

	Key Model Fidelity		Environment Model Fidelity		Latency			Jitter	
	Low	High	Low	High	0 ms	80 ms	120 ms	applied	not applied
Four-way ANOVA	✓	✓	✓	✓	✗	✓	✓	✓	✓
Three-way ANOVA	✓	✓	✓	✓	✓	✓	✓	✗	✗

high-fidelity key models and in low-fidelity environment models. The grand means for the factor latency do not agree in the four-way ANOVA and the three-way ANOVA, so I consider the factor latency as not having an effect on ODoT.

Table 4.6 shows the grand means and standard deviations with respect to SDoT. As in Table 4.5, I show the grand means calculated for the four-way ANOVA and three-way ANOVA tests in Table 4.6. The means are measured through a 5 point Likert Scale, where larger values indicate higher subjective difficulty of task.

From Table 4.6, we can see that three factors have effects on the SDoT. The SDoT is reduced for high-fidelity key models and low latency. The grand means for the factor environment model fidelity do not agree in the four-way ANOVA and the three-way ANOVA, so I cannot conclude that the factor environment model fidelity has an effect on SDoT.

Table 4.5: Grand means (GM) and their standard deviations (SD) of the ODoT for each factor.

	Key Model Fidelity				Environment Model Fidelity				Latency						Jitter			
	low		high		low		high		0 ms		80 ms		120 ms		applied		not applied	
	GM	SD	GM	SD	GM	SD	GM	SD	GM	SD	GM	SD	GM	SD	GM	SD	GM	SD
Four-way ANOVA	4.6	2.2	3.5	2.2	3.9	2.1	4.2	2.4	-	-	4.0	2.3	4.1	2.3	3.8	2.1	4.2	2.4
Three-way ANOVA	4.3	2.0	3.5	2.3	3.8	2.1	4.0	2.0	4.0	1.9	3.9	2.4	3.8	2.6	-	-	-	-

However, the conclusions drawn from Table 4.5 and Table 4.6 may not be significantly different. Hence, I analyze the variance for both metrics, i.e., ODoT and SDoT to determine whether any of the differences between the means are statistically significant, I compare the p-values to a significance level to decide whether the p-value suggests accept or reject on the null hypothesis. I choose a significance level of 0.05.

Table 4.7 and Table 4.8 demonstrate the ANOVA results in terms of the ODoT, where

Table 4.6: Grand means (GM) and their standard deviations (SD) of the SDoT for each factor.

	Key Model Fidelity				Environment Model Fidelity				Latency						Jitter			
	low		high		low		high		0 ms		80 ms		120 ms		applied		not applied	
	GM	SD	GM	SD	GM	SD	GM	SD	GM	SD	GM	SD	GM	SD	GM	SD	GM	SD
Four-way ANOVA	2.7	1.0	2.5	1.1	2.7	1.1	2.6	1.0	-	-	2.5	1.0	2.8	1.1	2.5	1.1	2.7	1.0
Three-way ANOVA	2.4	1.1	2.2	1.2	2.2	1.2	2.3	1.1	1.9	1.2	2.4	1.0	2.6	1.1	-	-	-	-

Table 4.7 shows the results of the four way ANOVA while Table 4.8 shows the results of the three way ANOVA.

Only the key model fidelity factor exhibits a p-value less than 0.05 for both ANOVA tests for the ODoT measure. To explore the interaction between a factor pair or among multiple factors, I also calculate the p-values for those interactions. Such calculations are performed for every combination of factors, including the combination of two factors, combination of three factors, and combination of four factors if plausible. From Table 4.7 and Table 4.8, we can see that there is no combination whose p-value is smaller than 0.05. This indicates that these factors are independent from each other in terms of the ODoT.

Table 4.7: Four-way ANOVA of the ODoT.

Source	F-statistics	p-value
Key Model Fidelity	14.9	0.0001
Environment Model Fidelity	1.28	0.2589
Latency	0.14	0.7063
Jitter	1.71	0.1929
Key Model Fidelity × Environment Model Fidelity	0	0.9769
Key Model Fidelity × Latency	0.14	0.7063
Key Model Fidelity × Jitter	0.24	0.6223
Environment Model Fidelity × Latency	0.24	0.6223
Environment Model Fidelity × Jitter	1.03	0.3109
Latency × Jitter	0.53	0.4689
Key Model Fidelity × Environment Model Fidelity × Latency	2.93	0.0882
Key Model Fidelity × Environment Model Fidelity × Jitter	0.37	0.5429
Key Model Fidelity × Latency × Jitter	0.04	0.8392
Environment Model Fidelity × Latency × Jitter	3.13	0.0781
Key Model Fidelity × Environment Model Fidelity × Latency × Jitter	1.28	0.2589

Table 4.9 shows the results of four-way ANOVA for the SDoT measure, and Table 4.10

Table 4.8: Three-way ANOVA of the ODoT.

Source	F-statistics	p-value
Key Model Fidelity	6.33	0.0128
Environment Model Fidelity	0.12	0.7342
Latency	0.12	0.8826
Key Model Fidelity × Environment Model Fidelity	004	0.8385
Key Model Fidelity × Latency	0.28	0.7544
Environment Model Fidelity × Latency	0.48	0.6217
Key Model Fidelity × Environment Model Fidelity × Latency	0.6	0.5517

shows the results of three-way ANOVA test. As opposed to the tests of significance for the ODoT measure, for the SDoT, the key model fidelity is not significant. Instead, the latency factor is statistically significant (with a p-value of 0.0281 for the four-way ANOVA test and 0.0018 for the three-way ANOVA test).

Furthermore, I analyze the interaction for factor combinations (just like what I did for the ODoT measure). The results demonstrate that there is no interaction among the four factors.

Table 4.9: Four-way ANOVA of SDoT.

Source	F-statistics	p-value
Key Model Fidelity	2.95	0.0871
Environment Model Fidelity	0.96	0.3271
Latency	4.88	0.0281
Jitter	2.95	0.0871
Key Model Fidelity × Environment Model Fidelity	0.38	0.54
Key Model Fidelity × Latency	0.02	0.9024
Key Model Fidelity × Jitter	0.38	0.54
Environment Model Fidelity × Latency	1.22	0.2704
Environment Model Fidelity × Jitter	1.82	0.1783
Latency × Jitter	0.74	0.3911
Key Model Fidelity × Environment Model Fidelity × Latency	0.24	0.6239
Key Model Fidelity × Environment Model Fidelity × Jitter	0.54	0.4622
Key Model Fidelity × Latency × Jitter	0.54	0.4622
Environment Model Fidelity × Latency × Jitter	2.95	0.0871
Key Model Fidelity × Environment Model Fidelity × Latency × Jitter	3.39	0.0669

Table 4.10: Three-way ANOVA of SDoT.

Source	F-statistics	p-value
Key Model Fidelity	1.2	0.2743
Environment Model Fidelity	0.34	0.5623
Latency	6.55	0.0018
Key Model Fidelity × Environment Model Fidelity	0.04	0.8468
Key Model Fidelity × Latency	0.7	0.4964
Environment Model Fidelity × Latency	0.16	0.8503
Key Model Fidelity × Environment Model Fidelity × Latency	0.46	0.6309

Given the results obtained with the tests of significance, I conclude that the key model fidelity factor affects the ODoT significantly, while the latency factor affects the SDoT significantly. I could not show any influence of other factors beyond what may occur by chance. I summarize my results in Table 4.11. H_{01} and H_{07} are rejected since the key model fidelity and latency factors have a significant effect on ODoT and SDoT respectively.

Table 4.11: Hypotheses. The third column shows if a hypothesis is accepted or rejected according to the ANOVA analysis.

H_{01}	Key model fidelities do not affect the ODoT	reject
H_{02}	Environment model fidelity does not affect the ODoT	accept
H_{03}	Network latency does not affect the ODoT	accept
H_{04}	Jitter does not affect the ODoT	accept
H_{05}	Key model fidelities do not affect the SDoT	accept
H_{06}	Environment model fidelity does not affect the SDoT	accept
H_{07}	Network latency does not affect the SDoT	reject
H_{08}	Jitter does not affect the SDoT	accept

Now I can answer the four questions asked at the beginning of this section.

1. The key model fidelity has a significant effect on the ODoT. Greater key model fidelity helps the user to achieve better recognition. But I could not show a significant effect on the SDoT.

2. I could not show a significant effect of environment fidelity on the ODoT nor the SDoT.

3. The latency factor has a significant effect on the SDoT. Longer latency results in higher SDoT. It did not have a significant effect on the ODoT.

4. I could not show a significant effect of jitter on the ODoT nor the SDoT in the study.

In conclusion, the user study supports the idea of only rendering and sending key models in remote rendering applications. Based on an analysis of the variance of both, ODoT and SDoT, I found that key model fidelity is the only factor that has a statistically significant effect on ODoT, and that latency is the only factor that has a statistically significant effect on SDoT. Thus, my application design is well informed to address overall DoT by addressing the factors key model fidelity and latency which are the two factors that have significant effects. To minimize the network bandwidth requirement and interaction latency, I reduce the environment model fidelity. The user study indicates that this reduction does likely not affect the SDoT. Moreover, since the factor key model fidelity has a significant effect on ODoT, my method maintains the fidelity of the key model.

4.3 Summary

In this chapter, I analyzed the effectiveness of the proposed hybrid remote rendering in reducing network bandwidth requirement and interaction latency quantitatively. The results indicate that we can reduce video transmission bitrate and interaction latency by lowering the number of high-fidelity models.

To investigate the influence of the proposed method on user experience, I conducted a user study with two measurements: the objective difficulty of task (ODoT) and the subjective difficulty of task (SDoT). I developed an experimental application in which the user is asked to recognize objects in different configurations pertaining to the identified factors. My user study shows that the key model fidelity affects the ODoT and that interaction latency plays an important role in the SDoT. When streaming over non-dedicated networks, the conditions often do not satisfy the requirements of remote rendering applications. Therefore, trading off between rendering quality of less important objects and the two factors, network bandwidth requirement and interaction latency, is essential but existing remote rendering applications cannot address this issue in a satisfactory way. The most important contribution of the proposed method is enabling such a trade-off.

As to the remote assistance for repair tasks, the results reveal that offloading rendering burden to a server should not affect the performance of users significantly. Since the active component is rendered in high-fidelity, the objective difficulty of tasks (ODoT) should not be increased greatly. In other words, a user's capacity for component recognition will be preserved. Furthermore, by reducing network bandwidth requirement and interaction

latency, my proposed remote rendering method aims at minimizing the subjective task difficulty of the AR remote assistance platform.

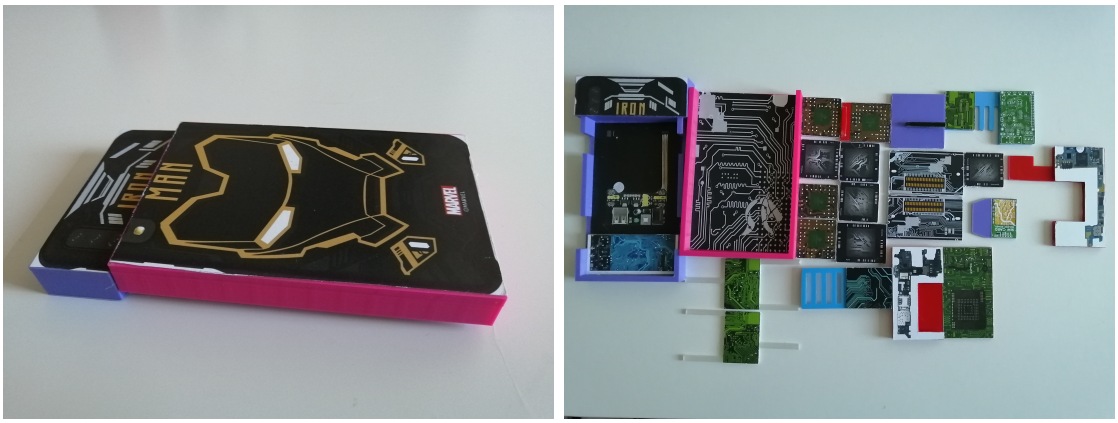
Chapter 5

Experiment for the Augmented Reality Remote Assistance Platform for Repair Tasks

In Chapter 3, I propose an AR online assistance platform for repair tasks, where I describe the overall architecture of the platform, the authoring method, and the remote rendering method. In Chapter 4, I evaluated the effectiveness of the remote rendering method.

In this chapter, I describe the design and execution of a user study to demonstrate the usability of the authoring method and investigate the overall performance of the AR online assistance platform. The user study was approved by the Research Ethics Board at the University of Ottawa (File Number: H11-18-1092). The participants were asked to complete a mobile phone repair task that involved disassembling a mobile phone, replacing one presumably faulty component, and re-assembling the phone.

Some mobile phone components, e.g., the screen, the camera module, wires, etc., are either fragile or can only be disassembled once. Given that I need two identical mobile phones that can be disassembled and assembled repeatedly for the user study, I designed and printed mobile phone models using 3D printers. Each printed mobile phone is composed of 21 components. The disassembly and assembly of the printed mobile phone model is complex enough so that a user is most likely not able to perform the task without any instructions. The textures were printed with paper and glued on the components. Figure 5.1 shows the mobile phone model. Note that in the user study for the proposed hybrid remote rendering, I have used the models of dogs, cats, and horses to evaluate the participants' performance on recognizing different models. I did not use the same models in the two



(a) Assembled status of the mobile phone model (b) Components of the mobile phone model

Figure 5.1: Mobile phone model used in the repair task.

user studies because the models of mobile phone components used in the user study for the AR platform have low numbers of polygons (less than 20 for most of them). This is not enough to distinguish between low-fidelity models and high-fidelity models in terms of polygon numbers.

There were 30 participants with each being assigned to one of the three groups:

- AR group: participants were assisted using the proposed AR method.
- sketching group: participants were assisted using a sketching method.
- video group: participants were assisted using an instructional video.

Sketching methods are among the most studied categories in remote assistance [32, 11, 35, 77]. The implementation that I used in this experiment is very similar to the work of Gurevich et al. [35]. However, the difference is that the instructions drawn by the expert are displayed on a mobile phone screen in my implementation, while the instructions are displayed by a projector in the work of Gurevich et al. [35].

In order to compare with the proposed method, I implemented the instruction schema of the sketching method as step-based. In every step, the expert gives the instruction for only one component. Each step consists of two steps: a) observation, b) motion. The following describes each phase.

1. Observation: The expert draws the instructions by lines that show the user how to complete this step.

2. Motion: The user performs the movement. After this phase, the expert decides if the movement from the user is correct, and restarts the step from the observation phase if not.

Compared with the proposed AR method, the sketching method that I used has only two phases in each step instead of three. Since it is the expert who identifies the correct component instead of the AR platform, the identification phase is not needed. During the entire process, the expert and the participant are able to communicate by voice.

The participants in the video group used an instructional video as the assistance method, which is probably the most popular method used by consumers. In order to compare with the proposed method, I implemented it also as step-based. Each step contains two phases: a) observation, and b) motion, as same as the sketching method. However, the difference is that there is no remote expert who assists the participant.

In the experiment, the experimenter represented the remote expert and the participant represented the local user. Both of them used a mobile phone to run the AR client application and operate on a real product. They were co-located and sitting back to back, so they could not see the operations of each other but can talk to each other. The server application was run on a Macbook Pro with a 2.4 GHz Dual-Core Intel Core i5 CPU, 4GB RAM, and an Intel Iris Graphics. The experiment setup for the sketching group was similar to the AR group. The differences were that the mobile phones ran the sketching program instead and there was no server for the sketching group. The experiment setup for the video group was simpler. There was no remote expert and the mobile phone was used to play the instructional video.

5.1 Procedure

5.1.1 Introduction

The study was conducted with one participant at a time. The experimenter described to the participant the purpose and the procedure of the experiment. The experimenter also informed the participant that a video of her or his operations would be recorded for further analysis of the performance after the experiment. Participants were instructed that if they felt tired, or did not wish to continue, they could always have a rest or completely stop the experiment. Then the participants were asked to read and sign a consent form. The experimenter also introduced the remote assistance method to the participant. Note that

the participant was not told about the other remote assistance methods or other groups. This was to minimize the potential biases such information may engender.

5.1.2 Familiarization with the Method

For all three groups, participants were asked to familiarize themselves with the corresponding method in a pre-task. The pre-task was a task that could be treated as a simplified version of the main mobile phone repair task. I did not limit the time of the familiarization step. The experiment proceeded to the next step when the participant felt confident about using the corresponding method.

Since the implementation of the proposed method sometimes loses the tracking of a component, the participants in the AR group were also informed of how to resume the tracking manually. It involved adjusting the pose of the camera and moving the components closer to the camera.

5.1.3 Testing

During testing, the experimenter served as the expert, and stayed in the same room with the participant. However, the experimenter and participant were not facing each other. Therefore, the expert and the participant could only see each other's actions through the mobile phone screen. However, they could talk to each other during the testing process. A video of the participant's workspace was recorded, which I used for the analysis of the performance after the experiment. A timer was also set up to measure the duration of the testing process.

5.2 Participants

30 participants were recruited in the experiment with 10 assigned to each group. The number of participants is comparable with similar studies [39, 30, 106, 108]. All participants were assigned randomly to one of the three groups. Ten of them were female (3 in AR group, 2 in sketching group, and 5 in the video group) and twenty were male. The participants' ages ranged from 20 to 39, where 57% were between 20-29 and 43% were between 30-39. For the 17 participants between 20-29, 6 of them were assigned to the AR group, 7 of them were assigned to the sketching group, and 4 of them were assigned to the video group. Three participants had repaired their mobile phones at least once. One of them was in the

AR group and two of them were in the sketching group. Seven of them had repaired other types of electronic devices. Two of them were assigned to the AR group, three of them were assigned to the sketching group, and the others were assigned to the video group.

5.3 Independent Variables and Dependent Measures

The experiment was conducted with one independent variable: the remote assistance method. It has three levels: AR method, sketching method, and video method.

During the experiment, I collected three types of data:

- Time: the time in seconds that a participant has spent to accomplish the task,
- NCR: the number of steps that a participant has completed with corrected errors, and
- NUR: the number of steps that a participant has completed with uncorrected errors.

The measured NCR represents the number of steps that were corrected by the expert during the task. It reveals the capacity of a method to represent the movements. If a participant's movement needs correction from the expert, it reflects that the participant has not understood the instructions well. A corrected error is a potential uncorrected error. The correction of an error relies on the expert, which may not notice the error depending on their view angle, concentration, and skill. Moreover, an incorrect operation of an electronic component may lead to hardware damage even if it is corrected by the remote expert. For instance, if a participant is plugging storage into the slot, he/she may not be aware of the orientation, which could damage the slot. But with the AR platform, an animation shows the participant how to plug the storage in by rotating and flipping it properly. A smaller number of corrected errors may reflect uncomplicated instructions that enable new users to follow them more easily, and less chance to damage the hardware. The measured NUR is the number of uncorrected errors that a participant has made. The number of uncorrected errors refers to the number of steps that were performed incorrectly and remained uncorrected after task completion.

There are two issues we need to pay attention to. First, for the AR and sketching groups, the NUR should be zero or very close to zero, because the expert is likely to find the errors and correct them. Therefore, I compared the pair of AR-video and the pair of sketching-video in terms of NUR, but I did not compare the AR-sketching pair. Second,

in the video group, because there is no intervention from the expert, the NCR should be always zero. Therefore, I only compare the AR and sketching groups in terms of NCR.

5.4 Questionnaires

To conclude the experiment, the participants are presented with two questionnaires. The first questionnaire focused on the evaluation of the user experience of the corresponding remote assistance method, while the second questionnaire focused on the task. Each question used a five-level Likert scale.

The first questionnaire consisted of six questions, as follows:

Q1. The instructions provided by the assistance method during the execution of the repair task are clear.

Q2. The use of the assistance method can save time during the execution of repair tasks.

Q3. It is easy to follow the instructions presented through the assistance method.

Q4. I was satisfied with the help I received through the assistance method.

Q5. If it is available, I would like to use this assistance method to repair my mobile phone in the future.

Q6. I recommend this assistance method for the repair task I executed.

The second questionnaire consisted of four questions, as follows:

Q7. I am confident that I have completed the repair task correctly.

Q8. With the help of the assistance method, I was clearly made aware of the errors I made (if applicable).

Q9. When I made an error during the repair task, I was able to recover very quickly with the help of the assistance method (if applicable).

Q10. I have not experienced any difficulty in carrying out the repair task.

5.5 Hypotheses

Based on the analysis of the task and the methods to be evaluated, I predicted:

H1. The AR method will achieve a comparable performance time to the sketching method. Since instructions with 3D virtual objects will be easier to follow and my proposed authoring tool is intuitive to use.

H2. The mean of NCR of the AR group will be smaller than that of the sketching group. This prediction is based on the fact that the proposed method has a more intuitive presentation of the 6DoF operations compared to the 2D presentations of the sketching method.

H3. The participants who use the AR method will make fewer errors than those who use the video method resulting in a smaller NUR. I expect this because the remote expert is able to correct the errors, but the participants who use the video method have to find the errors and correct them themselves.

H4. The participants who use the sketching method will make fewer errors than those who use the video method, hence it will have a smaller NUR. The reason is the same as for H3.

H5. The participants will rate the AR method with a higher score than the sketching method in the questionnaires.

H6. The participants will rate the AR method with a higher score than the video method in the questionnaires.

H7. The participants will rate the sketching method with a higher score than the video method in the questionnaires.

5.6 Results

The experimental results consist of two parts: the objective performance and the subjective evaluation questionnaires. In this section, I will present both of them. To understand how the results differ between the groups, I used the analysis of variance (ANOVA) measure to interpret the experimental data. I chose a significance level of 0.05. In order to investigate which pairs of groups are significantly different, I further performed Tukey's Honestly Significant Difference Procedure [102].

5.6.1 Objective Performance

Table. 5.1 shows the means and standard deviations for each performance measure. Since there are no interventions from the expert for the video method, the mean of NCR is missing for the video group.

Table 5.1: Summary of objective performance results: means and standard deviations for each measure and each group. The standard deviations are shown in the brackets.

Group	Time (seconds)	Mean of NCR	Mean of NUR
AR	871.3 (94.86)	2.2 (1.14)	0.1 (0.32)
sketching	518.4 (83.88)	7.8 (2.74)	0.1 (0.32)
video	397.6 (125.41)	-	3.9 (3.76)

Figure 5.2 shows the box plot of time for all the three groups. The differences are significant ($p = 8.96 \times 10^{-}8$ for the AR-sketching pair, $p = 1.19 \times 10^{-}9$ for the AR-video pair, and $p = 0.036$ for the sketching-video pair). The participants in the video group had spent the least time. The average time of the AR group is longer than the sketching group, and is the longest among the three methods, rejecting hypothesis H1.

When comparing the groups in terms of the measured NCR, I only compare the AR and sketching groups, since there is no intervention for the video group. Figure 5.3 demonstrates the box plot of the measured NCR. The number of corrected errors of the AR group is less than that for the sketching group. The difference is significant ($p = 2.16 \times 10^{-}7$), validating hypothesis H2. It reveals that the participants in the sketching group had not understood the instructions as accurately as in the AR group and needed corrections from the expert. In other words, the instructions of the AR method convey the movements better than those of the sketching method. Because the instructions of the AR method provide more informative clues on 6DoF movements, such as rotations and flips. Moreover, the NCR of the AR group is more centralized than that of the sketching group, which indicates that the performance of the participants is more stable with the AR method.

Figure 5.4 demonstrates the box plot of the measured NUR. The bottom and top of the groups AR and sketching are identical to 0, while they spread from 0 to 8 for the video group. This reveals that the AR and sketching methods lead to a more reliable remote assistance experience. The differences are significant ($p = 0.002$ for the pair of AR-video and $p = 0.002$ for the pair of sketching-video), supporting hypotheses H3 and H4. Note that there is one plus symbol that indicates the outlier in the AR group and the sketching group, respectively. This was caused by one case in each group where the expert failed to recognize an error and did not correct it. It is worth pointing out that there were also

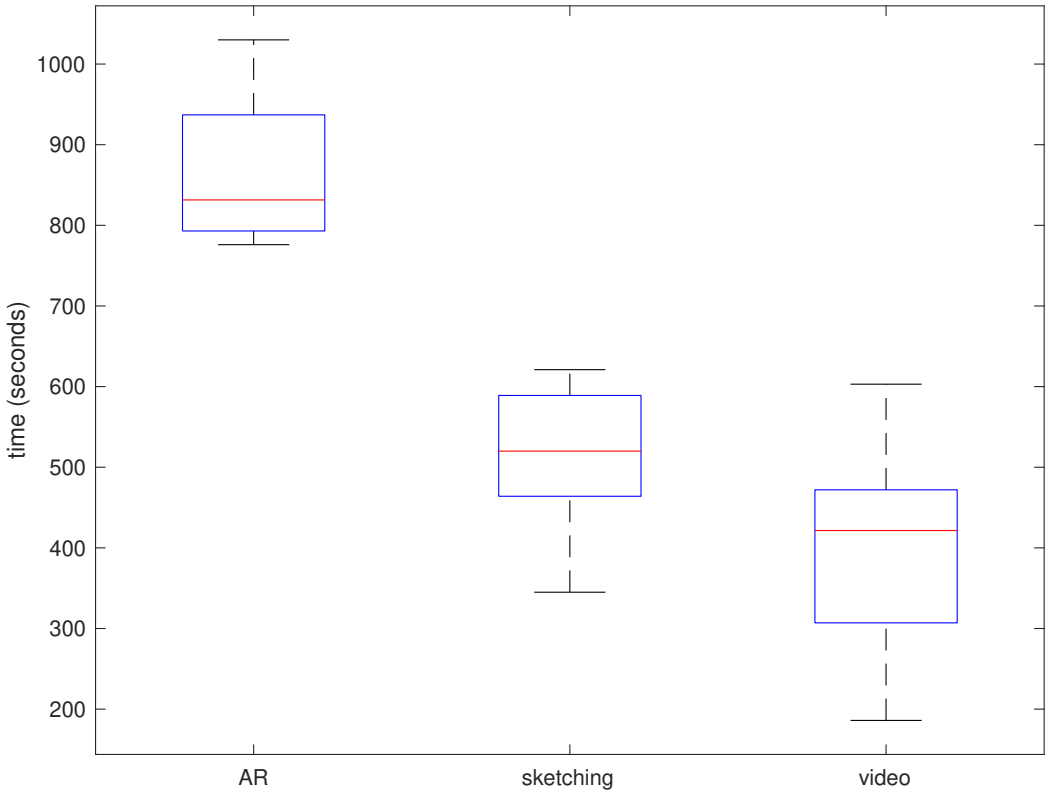


Figure 5.2: Box plot of the time in seconds for all the three groups. It shows the median, the bottom, the top, the 25th, and the 75th percentiles.

participants in the video group who have accomplished every step correctly. However, there were only two such cases. It further reveals the stability of the performance of the participants in the AR and sketching groups.

From the analysis of objective performance, we can see that each method has its advantage. First, the video method is the fastest method, while it performs the worst in terms of the NUR. Second, the sketching method can eliminate the uncorrected errors and remains relatively fast, but there are more corrected errors. Last, the AR method outperforms the other methods in terms of NCR and NUR, but it is the slowest method among the three. In practice, one needs to compare the advantages of the three methods and make the choice according to the requirements.

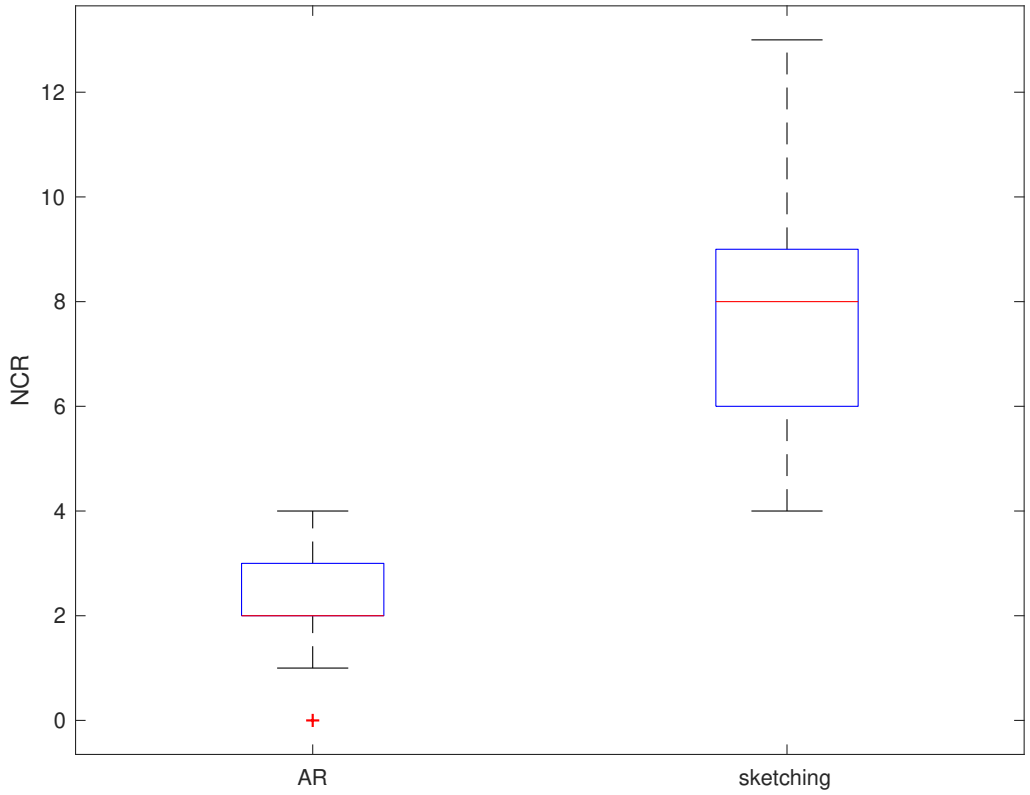


Figure 5.3: Box plot of the measure NCR for the AR and sketching groups. It shows the median, the bottom, the top, the 25th, and the 75th percentiles. The plus symbol indicates the data locating outside of the 1.5 Interquartile range (IQR).

5.6.2 Subjective Questionnaires

The results for the questionnaire focusing on the corresponding remote assistance method are shown in Figure 5.5. The AR method outperforms the other two groups in the first questionnaire. But the differences in the means are relatively small for some questions. In contrast to the AR method, the sketching method does not exhibit clear advantages over the video method (outperforms the video method in three of the six questions).

From Figure 5.6, it is shown that in the second questionnaire, the AR method and the sketching method have overall advantages over the video methods (outperforms in three of the four questions). But the differences are also relatively small. Meanwhile, in two questions, the AR method outperforms the sketching method.

Considering that the differences of the means of all the three groups are relatively

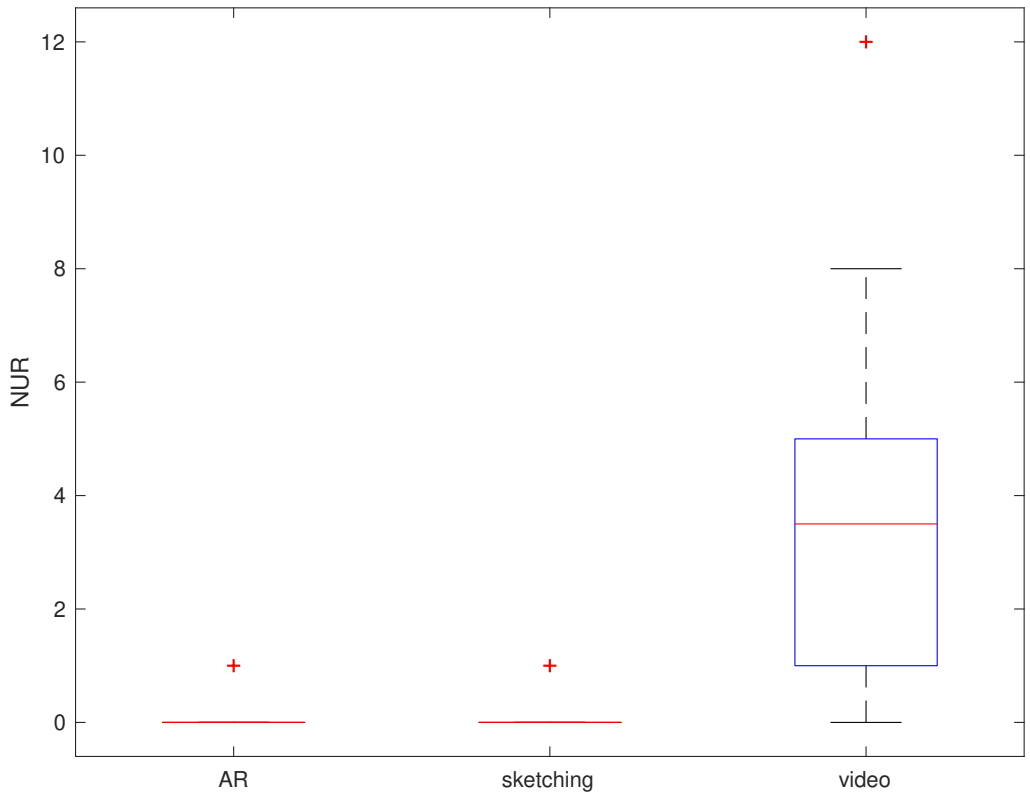


Figure 5.4: Box plot of the measure NUR for all the groups. It shows the median, the bottom, the top, the 25th, and the 75th percentiles. The plus symbol indicates the data locating outside of the 1.5 Interquartile range (IQR).

small, I further performed statistical tests on the results. Table 5.2 lists the p-values from the statistical tests. For each question, three numbers are shown, which are the p-values for the AR-sketching, AR-video, and sketching-video group pairs, respectively. Eight of them are less than the selected significance level (0.05). I have highlighted them using bold texts. For hypothesis H5, I can only validate it on question Q6, which means that, the participants rated the AR method with a significant higher score than the sketching method only on one question. For hypothesis H6, it is validated on questions Q3, Q5, Q6, Q7, and Q9. This implies that the participants rated the AR method with a significant higher score than the video method on the half of the questions. For hypothesis H7, I can accept it on questions Q3 and Q7. This indicates that the participants rated the sketching method with a significant higher score than the video method on two questions.

Note that not all the differences between two groups are significant for every question.

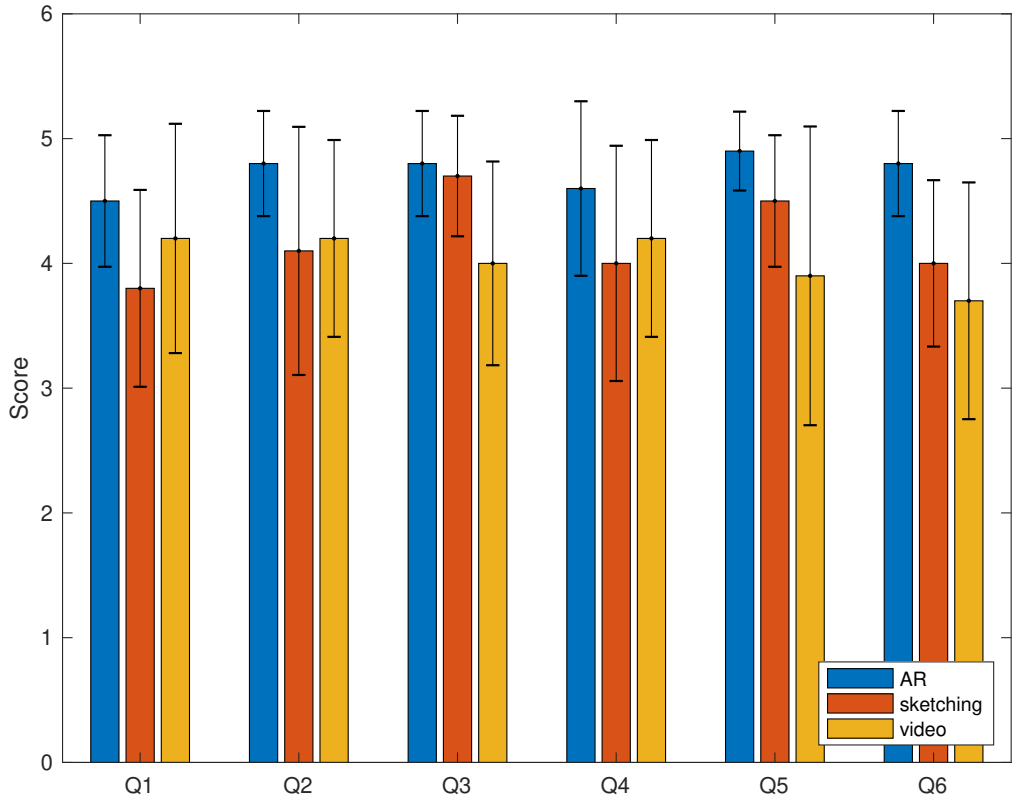


Figure 5.5: Mean scores of the questions in the first questionnaire for all the three groups. The standard deviations are shown as error bars.

Table 5.2: ANOVA analysis of the subjective questions. The numbers are the p-values. The columns represent various group pairs, while the rows indicate the ten questions. Bold numbers are those below the significance level I chose (0.05).

	AR-sketching	AR-video	sketching-video
Q1	0.23	0.68	0.68
Q2	0.13	0.21	0.96
Q3	0.93	**0.02**	**0.04**
Q4	0.25	0.53	0.85
Q5	0.49	**0.02**	0.21
Q6	**0.047**	**0.01**	0.62
Q7	1	**0.007**	**0.007**
Q8	0.75	0.052	0.093
Q9	0.76	**0.02**	0.11
Q10	0.75	0.97	0.88

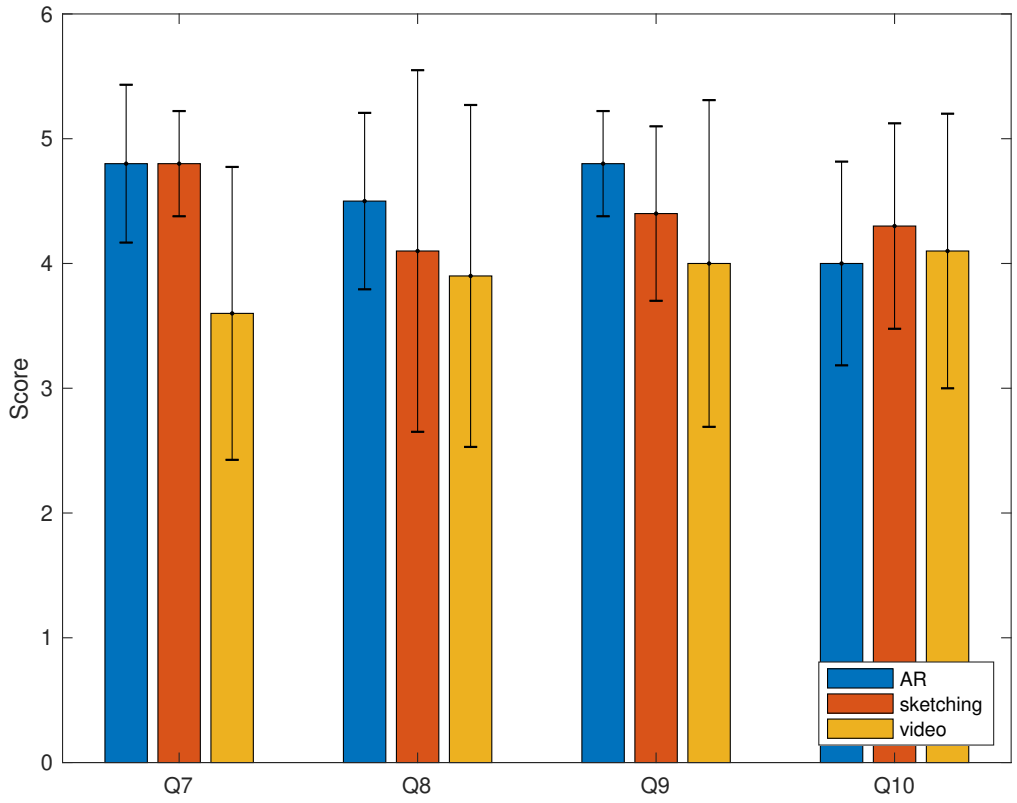


Figure 5.6: Mean scores of the questions in the second questionnaire for all the three groups. The standard deviations are shown as error bars.

However, there are some points I would like to highlight. First, for question Q3, both the AR group and the sketching group have a significant advantage over the video group, which points out that both of them have clearer instructions than the video method does. Second, for question Q6, the AR method gets a higher recommendation over the other two methods for the repair task of this experiment. Third, for question Q7, the participants who were assisted with the AR method and the sketching method felt much more confident about the correctness of their operations.

I want to point out an observation during the task that the participants in the AR group and sketching group have a more "fluid" performance, namely, they appear rarely stuck in a step. In contrast, the participants in the video group often seem to be held up by a step. For example, they might not be able to find a component, be confused about why a component did not fit a slot, or struggle on whether a step was done correctly. Sometimes, they were able to solve this by playing the video back, yet, at other times they just continued the assembly by leaving the errors there. I believe this to be the reason why the participants in the video group felt less confident about their operations and rated question Q7 much lower than those in the other two groups.

Note that for some questions, the p-values are 1 or very close to 1 for the comparison between the two groups, since for those questions, every participant rated them with high scores, which means that the participants in all groups had a very positive attitude towards the corresponding method for those questions. They could not rank these methods since the participants were only introduced a single method.

As mentioned in Section 5.4, I divided the questions into two questionnaires. The first questionnaire consists of the questions related to the user experience of the corresponding remote assistance method, and the second questionnaire consists of the questions related to the task. I calculated the average score for the questions in each questionnaire per participant. Figure 5.7 depicts the mean scores and the standard deviations of both questionnaires. As to the first questionnaire, the participants gave a significantly higher score to the AR method over the video method ($p = 0.01$). There are no significant differences between the AR method and the sketching method nor between the sketching method and the video method ($p = 0.06$ for the AR-sketching pair and $p = 0.74$ for the sketching-video pair). The standard deviation for the AR method is smaller than those for the other two methods. It indicates that the opinions from the participants in the AR group are more consistent than those from the participants in the other two groups. As to the second questionnaire, the differences in each group pair are not significant ($p = 0.93$ for the AR-sketching pair, $p = 0.06$ for the AR-video pair, and $p = 0.13$ for the sketching-video

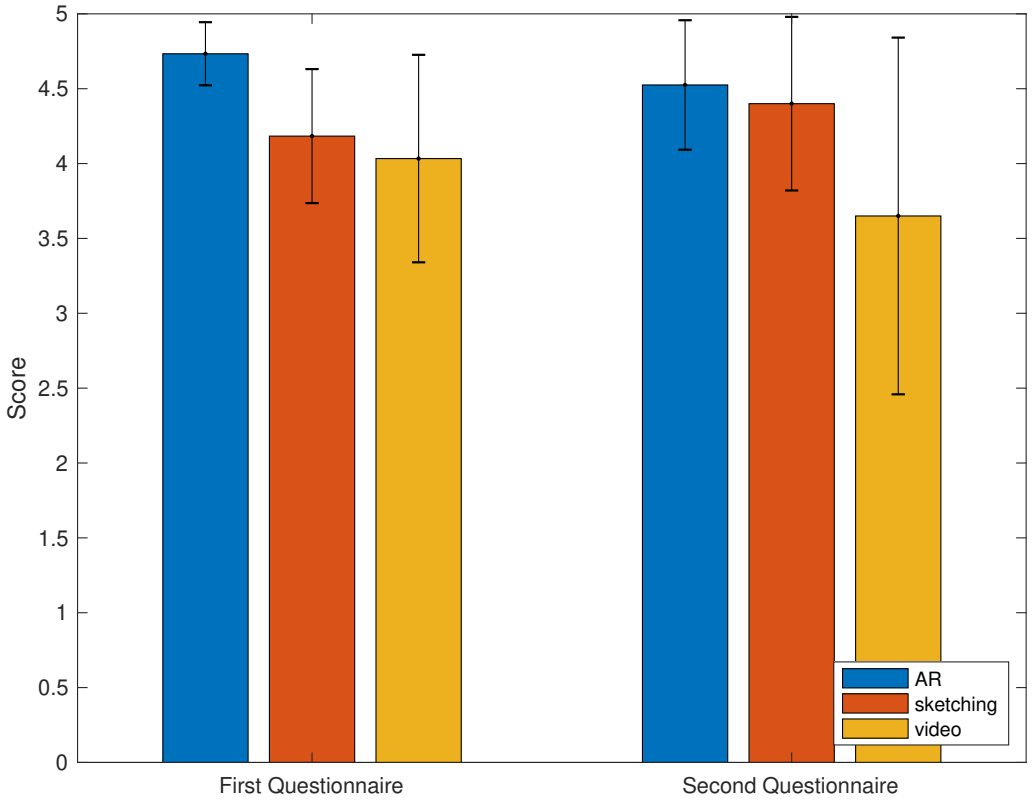


Figure 5.7: Mean scores of the question averages of both questionnaires for all the three groups. The standard deviations are shown as error bars.

pair).

5.7 Discussion

I propose a method that allows remote experts to author AR content easily in a repair task. The user study provides some objective and subjective evidence that supports the benefits of showing 3D virtual objects to users in an AR platform. However, I could not find evidence to validate the hypothesis, H1, to claim that the AR method will achieve a comparable performance time to the sketching method. The potential reason is two-fold. First, the participants often encounter the problem of losing track, so they needed to spend time to resume the tracking. Second, with the AR method, the participants needed to bring each component into the camera view for the AR platform to decide if it was the correct component. However, with the sketching method, it was the remote expert who found

the correct component for the participants. Two potential optimizations may improve the performance time of the AR method. First, with a robust object pose tracking method, the participants may spend less time tracking the components. Second, using a wearable AR device, e.g., AR glasses, that has a large field of view, may reduce the performance time. As all the components would be in the AR device's view, the AR platform may find the correct component faster.

Furthermore, the participants in the AR group made less corrected errors than the participants in the sketching group. The reason is that, with the sketching method, the remote expert used sketches to instruct the user where and how to move a component. If an operation only involves translational movements, the sketches may be enough. However, if an operation involves rotations and flips, it is difficult for the user to understand the sketches clearly. Therefore, the user is more likely to make mistakes. In contrast, the AR method displayed the desired movements in a 3D animation. It can express the rotational movements more intuitively. This suggests that the AR remote assistance method may benefit from rich media formats, especially the 3D representations, since the 3D virtual objects and animations enable the user to understand the desired movements more intuitively.

For the AR and the sketching method, the remote expert inspected the user's operations and corrected the errors if he/she noticed them. However, for the video method, there was no feedback from the remote expert. This is the main reason why the participants in the video group made more uncorrected errors than the participants in the other groups.

The difficulty of a task can affect the participants' performance. In the user study's familiarization step, the participants were asked to accomplish a simplified version of the repair task, which involves only three components. Although the purpose is for the participants to become familiar with the corresponding method, every participant in each group quickly and correctly accomplished the task.

Although the implementation has the capacity to support multiple users, in the experiment, I tested the proposed method for one user instead of multiple users. This choice is motivated because I wanted to limit the number of independent variables. It would have created challenges in conducting the experiments including increasing the required number of participants. However, the capacity of supporting multiple users is still useful in some cases. For instance, an expert from a mobile phone customer service could arrange the customers who encountered the same problem to the same time slot and assist them at the same time. In this way, the productivity can be greatly increased.

5.8 Summary

To evaluate the performance of the proposed system, I conducted a user study to compare it to the sketching method and the video method. 30 participants participated in the user study and accomplished a mobile phone repair task with one of the remote assistance methods that I compared. The results show that the proposed method and the sketching method have the same capacity in helping the participants avoid uncorrected errors in an assembly task, with both outperforming the traditional instructional video. In terms of the number of corrected errors, the proposed method has a clear advantage over the sketching method. Since it uses 3D virtual objects to show 6DoF movements, which may help the participants understand the intended movements better. However, the participants who used the sketching method and the video method performed significantly faster. From the subjective questionnaires filled by the participants, I can conclude that the proposed method has a significantly better overall user experience than traditional instructional videos.

Chapter 6

Conclusions

In this thesis, I discussed the current status of AR technology for repair tasks and the existing challenges in this domain. Then I presented a platform that addresses two of the existing challenges: device limitations and authoring. I conducted two user studies to evaluate the platform. In this chapter, I summarize the work of this thesis in Section 6.1, and list the limitations of our work and suggest future work in Section 6.2.

6.1 Summary

There are various AR applications for repair tasks that have been developed, but few of them have been deployed outside of laboratories [107, 73]. I summarize the limitations of current prototypes into four main challenges:

1. Device limitations: Computing capacity, sensory capacity, weight, and battery life are the main limitations of current AR devices.

2. Object pose tracking: Existing methods have relatively weak capacity of handling small, untextured, nonrigid components and suffer from severe occlusions.

3. Human-computer interaction: As an emerging technology, researchers are seeking the best way of visualization and to present instructions. Also, horizontal evaluation and design principles are needed to guide researchers and developers on HCI design.

4. Authoring: Authoring of instructions with AR content is not a trivial task since it involves solving spatial correspondence and richer multimedia formats than any traditional instructions.

The vision of this thesis is to move the AR technology for repair tasks towards being widely used in the real world. Under this vision, I have proposed two solutions to address two challenges: device limitations and authoring.

In the AR platform, I first proposed a more convenient authoring tool for remote experts. Our method enables remote experts to create instructions in 3D formats in an online manner by capturing the operations of them with an object pose tracking method. Moreover, with the proposed method, the geometric and physical constraints can be applied naturally.

Second, I proposed a remote rendering method for mobile applications. It helps reduce the rendering burden of mobile devices with a network connection. The advantages of the method are two-fold. On the one hand, it enables a trade-off between rendering quality and the network requirements. On the other hand, it can reduce bandwidth requirements and, in the meantime, reduce interaction latency. I achieved this by using a client-server model. The server renders high-fidelity models, encodes and sends the rendering results to the client. The client renders the low-fidelity models and overlays the rendering frames received from the server onto its local rendering frames.

I performed a quantitive analysis to examine the effectiveness of the method in reducing the network bandwidth requirement and interaction latency. The results show that the method can effectively reduce video transmission bitrate and interaction latency by lowering the number of high-fidelity models. Besides, I conducted a user study to evaluate the effects of different factors on user experience. The user study indicates that the key model fidelity affects the Objective Difficulty of Task (ODoT) and that the interaction latency has a main effect on the Subjective Difficulty of Task (SDoT). I argue that, when using a non-dedicated network, trading-off between rendering quality of less important objects and the network requirements is essential.

To analyze the performance of the AR remote assistance platform in repair tasks, I conducted a user study and compared the proposed method to a sketching method and a traditional method using an instruction video. The results suggest that our method can effectively help a remote expert to author accurate and clear instructions in an online manner. Although it is slower than the sketching method and the traditional method, the instructions with 3D virtual objects help reduce the number of errors made by the local user. From the questionnaires, I can conclude that users has a significant preference on using the proposed AR platform over the traditional video method.

6.2 Limitations and Future Work

I have noticed two limitations of the proposed AR platform. First, the users frequently encountered the problem of losing track of the components. This was caused by fast movements of the components or a long distance between the camera and some components. Therefore, the users spent a lot of effort to resume track of the components. Second, in the user study, I have compared three methods, i.e., the proposed AR platform, the sketching method, and the video method, on the same task to evaluate the performance of the users and their subjective attitudes towards those methods. It is also valuable to evaluate the effects of different methods on the users as remote experts. Third, I only conducted the user study of the proposed AR platform on smartphones. Since the user experience on smartphones is different from that of wearable devices like AR glasses, the results might be different with wearable AR devices. For instance, AR glasses usually have a larger field of view than smartphones, which might benefit users when they are looking for a specific component among many other components.

For future work, I plan to employ a better object tracking method to improve our proposed AR platform. During the experiment, I noticed that users required extra effort to maintain tracking, and it affects the user experience. Moreover, the tracking method that I used is not able to handle small or untextured components well. It is reasonable to expect a smoother and faster user experience. In recent years, deep-learning-based methods achieved promising results for object pose estimation. For example, Wang et al. [104] proposed a deep-learning-based method that uses an RGB-D camera and works with untextured objects. The experiment results show that it is robust towards heavy occlusion that was the main cause of tracking failures in our experiment. It also has a real-time performance.

I used mobile phones as the AR devices in our implementation. However, it would be interesting to use head-mounted devices. For example, Microsoft HoloLens 2 [67] and Google Glass [34] are good options. HoloLens 2 is equipped with multiple types of sensors, e.g., eye-tracking, RGB camera, depth camera, and heading tracking. So I can employ those object tracking solutions that require a depth camera. In the meantime, it provides a larger field of view than mobile phones. Moreover, with an eye-tracking solution, our proposed method can only display the object that a user is looking at in high-fidelity to minimize the bandwidth usage. Google Glass (46g) is much lighter than HoloLens 2 (566g) and more suitable for long time use. Although it has less computing power than HoloLens 2, our remote rendering method can help it render complicated components.

Finally, I also plan to loosen the step-by-step interaction and enable a more flexible collaboration. For instance, I can enable the expert to demonstrate multiple steps at a time which may result in shorter completion time.

References

[1] Matt Adcock and Chris Gunn. Using projected light for mobile remote guidance. *Computer Supported Cooperative Work*, 24(6):591–611, 2015.

[2] Matt Adcock, Dulitha Ranatunga, Ross Smith, and Bruce H. Thomas. Object-based touch manipulation for remote guidance of physical tasks. In *Proceedings of the 2nd ACM symposium on Spatial User Interaction*, pages 113–122, 2014.

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

[4] Keven Baird and Woodrow Barfield. Evaluating the effectiveness of augmented reality displays for a manual assembly task. *Virtual Reality*, 4(4):250–259, 1999.

[5] Paul Bao and Douglas Gourlay. A framework for remote rendering of 3-D scenes on limited mobile devices. *IEEE Transactions on Multimedia*, 8(2):382–389, 2006.

[6] Paul Bao, Douglas Gourlay, and Youfu Li. Deep compression of remotely rendered views. *IEEE Transactions on Multimedia*, 8(3):444–456, 2006.

[7] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346 – 359, 2008.

[8] Bhaskar Bhattacharya and Eliot Winer. A method for real-time generation of augmented reality work instructions via expert movements. In *Proceedings of the Engineering Reality of Virtual Reality*, volume 9392, pages 109–121, 2015.

[9] Sébastien Bottecchia, Jean-Marc Cieutat, and Jean-Pierre Jessel. T.A.C: Augmented reality system for collaborative tele-assistance in the field of maintenance through internet. In *Proceedings of the 1st Augmented Human International Conference*, pages 1–7, 2010.

[10] Azzedine Boukerche and Richard Werner Nelem Pazzi. Remote rendering and streaming of progressive panoramas for mobile devices. In *Proceedings of the 14th Annual ACM International Conference on Multimedia*, pages 691–694, 2006.

[11] Pierre Boulanger. Application of augmented reality to industrial tele-training. In *Proceedings of the 1st Canadian Conference on Computer and Robot Vision*, pages 320–328, 2004.

[12] Thomas P. Caudell and David W. Mizell. Augmented reality: an application of heads-up display technology to manual manufacturing processes. In *Proceedings of the 25th Hawaii International Conference on System Sciences*, volume ii, pages 659–669, 1992.

[13] Kar Long Chan, Kohei Ichikawa, Yasuhiro Watashiba, Uthayopas Putchong, and Hajimu Iida. A hybrid-streaming method for cloud gaming: To improve the graphics quality delivered on highly accessible game contents. *International Journal of Serious Games*, 4(2), 2017.

[14] Chun-Fa Chang and Shyh-Haur Ger. Enhancing 3D graphics on mobile devices by image-based rendering. In *Proceedings of the IEEE Pacific Rim Conference on Multimedia: Advances in Multimedia Information Processing*, pages 1105–1111, 2002.

[15] Kuo-En Chang, Chia-Tzu Chang, Huei-Tse Hou, Yao-Ting Sung, Huei-Lin Chao, and Cheng-Ming Lee. Development and behavioral pattern analysis of a mobile guide system with augmented reality for painting appreciation instruction in an art museum. *Computers & Education*, 71:185–197, 2014.

[16] Jeff Chastine, Kristine Nagel, Ying Zhu, and Mary Hudachek-Buswell. Studies on the effectiveness of virtual pointers in collaborative augmented reality. In *Proceedings of the IEEE Symposium on 3D User Interfaces*, pages 117–124, 2008.

[17] Dimitris Chatzopoulos, Carlos Bermejo, Zhanpeng Huang, and Pan Hui. Mobile augmented reality survey: From where we are to where we go. *IEEE Access*, 5:6917–6950, 2017.

[18] Kuan-Ta Chen, Yu-Chun Chang, Hwai-Jung Hsu, De-Yu Chen, Chun-Ying Huang, and Cheng-Hsin Hsu. On the quality of service of cloud gaming systems. *IEEE Transactions on Multimedia*, 16(2):480–495, 2014.

[19] Shenchang Eric Chen. Quicktime VR: An image-based approach to virtual environment navigation. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, pages 29–38, 1995.

[20] NVIDIA Corporation. Your games. Your devices. Play anywhere | NVIDIA GeForce NOW. https://www.nvidia.com/en-us/geforce/products/geforce-now. Accessed: 2020-05-05.

[21] Valve Corporation. Steam remote play. http://store.steampowered.com/streaming. Accessed: 2020-05-05.

[22] Cyril Crassin, David Luebke, Michael Mara, Morgan McGuire, Brent Oster, Peter Shirley, Peter-Pike Sloan, and Chris Wyman. Cloudlight: A system for amortizing indirect lighting in real-time rendering. *Journal of Computer Graphics Techniques*, 4(4):1–27, 2015.

[23] Francesca De Crescenzio, Massimiliano Fantini, Franco Persiani, Luigi Di Stefano, Pietro Azzari, and Samuele Salti. Augmented reality for aircraft maintenance training and operations support. *IEEE Computer Graphics and Applications*, 31(1):96–101, 2011.

[24] DeepAR. DeepAR - Snapchat face filters and lenses augmented reality SDK. https://-www.deepar.ai. Accessed: 2020-09-06.

[25] Timo Engelke, Jens Keil, Pavel Rojtberg, Folker Wientapper, Sabine Webel, and Ulrich Bockholt. Content first - A concept for industrial augmented reality maintenance applications using mobile devices. In *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality*, pages 251–252, 2013.

[26] GamingAnywhere.org. GamingAnywhere - An open source cloud gaming system. http://gaminganywhere.org. Accessed: 2020-05-05.

[27] Liz Gannes. Google unveils project glass: Wearable augmented-reality glasses. http://allthingsd.com/20120404/google-unveils-project-glass-wearable-augmented-reality-glasses/. Accessed: 2020-09-03.

[28] Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, pages 209–216, 1997.

[29] Michele Gattullo, Giulia Wally Scurati, Michele Fiorentino, Antonio Emmanuele Uva, Francesco Ferrise, and Monica Bordegoni. Towards augmented reality manuals for industry 4.0: A methodology. *Robotics and Computer-Integrated Manufacturing*, 56:276–286, 2019.

[30] Nirit Gavish, Teresa Gutiérrez, Sabine Webel, Jorge Rodríguez, Matteo Peveri, Uli Bockholt, and Franco Tecchia. Evaluating virtual reality and augmented reality training for industrial maintenance and assembly tasks. *Interactive Learning Environments*, 23(6):778–798, 2015.

[31] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, 2012.

[32] RE'FLEKT GmbH. Augmented reality platform for maintenance operations and training. https://www.re-flekt.com/reflekt-one. Accessed: 2019-05-22.

[33] Google. Build new augmented reality experiences that seamlessly blend the digital and physical worlds. https://developers.google.com/ar. Accessed: 2020-09-06.

[34] Google. Glass – glass. https://www.google.com/glass/start. Accessed: 2020-05-05.

[35] Pavel Gurevich, Joel Lanir, Benjamin Cohen, and Ran Stone. Teleadvisor: A versatile augmented reality tool for remote assistance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 619–622, 2012.

[36] Matthias Haringer and Holger Regenbrecht. A pragmatic approach to augmented reality authoring. In *Proceedings of the International Symposium on Mixed and Augmented Reality*, pages 237–245, 2002.

[37] Roger Harrabin. EU brings in 'right to repair' rules for appliances. https://www.-bbc.com/news/business-49884827, 2019. Accessed: 2020-03-22.

[38] Mahdi Hemmati, Abbas Javadtalab, Ali Asghar Nazari Shirehjini, Shervin Shirmohammadi, and Tarik Arici. Game as video: Bit rate reduction through adaptive object encoding. In *Proceedings of the ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, pages 7–12, 2013.

[39] Steven J. Henderson and Steven K. Feiner. Evaluating the benefits of augmented reality for task localization in maintenance of an armored personnel carrier turret.

In *Proceedings of the 8th IEEE International Symposium on Mixed and Augmented Reality*, pages 135–144, 2009.

[40] Steven J. Henderson and Steven K. Feiner. Augmented reality in the psychomotor phase of a procedural task. In *Proceedings of the 10th IEEE International Symposium on Mixed and Augmented Reality*, pages 191–200, 2011.

[41] Stefan Hinterstoisser, Stefan Holzer, Cedric Cagniart, Slobodan Ilic, Kurt Konolige, Nassir Navab, and Vincent Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 858–865, 2011.

[42] Tomáš Hodaň, Pavel Haluza, Štepán Obdržálek, Jirí Matas, Manolis Lourakis, and Xenophon Zabulis. T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pages 880–888, 2017.

[43] Tomáš Hodaň, Xenophon Zabulis, Manolis Lourakis, Štěpán Obdržálek, and Jiří Matas. Detection and fine 3D pose estimation of texture-less objects in RGB-D images. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4421–4428, 2015.

[44] Hua-Jun Hong, Chih-Fan Hsu, Tsung-Han Tsai, Chun-Ying Huang, Kuan-Ta Chen, and Cheng-Hsin Hsu. Enabling adaptive cloud gaming in an open-source cloud gaming platform. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(12):2078–2091, 2015.

[45] Adrian H. Hoppe, Kai Westerkamp, Sebastian Maier, Florian van de Camp, and Rainer Stiefelhagen. Multi-user collaboration on complex data in virtual and augmented reality. In *Proceedings of the HCI International - Posters' Extended Abstracts*, pages 258–265, 2018.

[46] Xueshi Hou, Yao Lu, and Sujit Dey. Wireless VR/AR with edge/cloud computing. In *Proceedings of the 26th International Conference on Computer Communication and Networks*, pages 1–8, 2017.

[47] Apple Inc. Augmented reality - Apple developer. https://developer.apple.com/-augmented-reality/. Accessed: 2020-09-06.

[48] Hirokazu Kato. ARToolKit home page. http://www.hitl.washington.edu/artoolkit/. Accessed: 2020-09-06.

[49] Hirokazu Kato and Mark Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings of the IEEE and ACM International Workshop on Augmented Reality*, pages 85–94, 1999.

[50] Bui Minh Khuong, Kiyoshi Kiyokawa, Andrew Miller, Joseph J. La Viola, Tomohiro Mashita, and Haruo Takemura. The effectiveness of an AR-based context-aware assembly support system in object assembly. In *Proceedings of the IEEE Virtual Reality*, pages 57–62, 2014.

[51] Yu-Doo Kim and Il-Young Moon. E-training content delivery networking system for augmented reality car maintenance training application. *International Journal of Multimedia and Ubiquitous Engineering*, 8(2):69–80, 2013.

[52] Christian Koch, Matthias Neges, Markus König, and Michael Abramovici. Natural markers for augmented reality-based indoor navigation and facility maintenance. *Automation in Construction*, 48:18–30, 2014.

[53] Jan Kolkmeier, Emiel Harmsen, Sander Giesselink, Dennis Reidsma, Mariët Theune, and Dirk Heylen. With a little help from a holographic friend: The OpenIMPRESS mixed reality telepresence toolkit for remote collaboration systems. In *Proceedings of the 24th ACM Symposium on Virtual Reality Software and Technology*, pages 1–11, 2018.

[54] Hideaki Kuzuoka. Spatial workspace collaboration: A sharedview video support system for remote collaboration capability. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 533–540, 1992.

[55] Jin Uk Kwon, Jae-In Hwang, Jaeyoung Park, and Sang Chul Ahn. Fully asymmetric remote collaboration system. *IEEE Access*, 7:54155–54166, 2019.

[56] Philipp Ladwig, Bastian Dewitz, Hendrik Preu, and Mitja Säger. Remote guidance for machine maintenance supported by physical leds and virtual reality. In *Proceedings of Mensch und Computer*, pages 255–262, 2019.

[57] Fabrizio Lamberti and Andrea Sanna. A streaming-based solution for remote visualization of 3d graphics on mobile devices. *IEEE Transactions on Visualization and Computer Graphics*, 13(2):247–260, 2007.

[58] Kangdon Lee. Augmented reality in education and training. *TechTrends*, 56(2):13–21, 2012.

[59] Marc Levoy. Polygon-assisted jpeg and mpeg compression of synthetic images. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, pages 21–28, 1995.

[60] Chang Liu, Jinyuan Jia, Qian Zhang, and Lei Zhao. Lightweight WebSIM rendering framework based on cloud-baking. In *Proceedings of the ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, pages 221–229, 2017.

[61] Yao Liu, Shaoxuan Wang, and Sujit Dey. Content-aware modeling and enhancing user experience in cloud mobile rendering and streaming. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 4(1):43–56, 2014.

[62] Alfredo Liverani, Giancarlo Amati, and Gianni Caligiana. Interactive control of manufacturing assemblies with mixed reality. *Integrated Computer-Aided Engineering*, 13(2):163–172, 2006.

[63] Sony Interactive Entertainment LLC. PlayStation Now – Online streaming services on PS4 or PC - PlayStation. https://www.playstation.com/en-ca/explore/playstationnow. Accessed: 2020-05-05.

[64] Yan Lu, Shipeng Li, and Huifeng Shen. Virtualized screen: A third element for cloud-mobile convergence. *IEEE MultiMedia*, 18(2):4–11, 2011.

[65] Yao Lu, Yao Liu, and Sujit Dey. Asymmetric and selective object rendering for optimized cloud mobile 3d display gaming user experience. *Multimedia Tools and Applications*, 76(18):18291–18320, 2017.

[66] Zhan Ma, Tao Yue, Xun Cao, Yiling Xu, Xin Li, and Yongjin Wang. Interactive screen video streaming-based pervasive mobile workstyle. *IEEE Transactions on Multimedia*, 19(10):2322–2332, 2017.

[67] Microsoft. Microsoft HoloLens | mixed reality technology for business. https://-www.microsoft.com/en-us/hololens. Accessed: 2020-05-05.

[68] Antonija Mitrovic, Brent Martin, Pramuditha Suraweera, Konstantin Zakharov, Nancy Milik, Holl, Jay, and Nicholas Mcguigan. ASPIRE: An authoring system and deployment environment for constraint-based tutors. *International Journal of Artificial Intelligence in Education*, 19(2):155–188, 2009.

[69] Peter Mohr, Denis Kalkofen, Bernhard Kerbl, Michael Donoser, and Dieter Schmalstieg. Retargeting technical documentation to augmented reality. In *Proceedings of*

the 33rd Annual ACM Conference on Human Factors in Computing Systems, pages 3337–3346, 2015.

[70] Katharina Mura, Nils Petersen, Markus Huff, and Tandra Ghose. IBES: A tool for creating instructions based on event segmentation. *Frontiers in Psychology*, 4:994, 2013.

[71] Yuval Noimark and Daniel Cohen-Or. Streaming scenes to MPEG-4 video-enabled devices. *IEEE Computer Graphics and Applications*, 23(1):58–64, 2003.

[72] Ohan Oda, Carmine Elvezio, Mengu Sukan, Steven Feiner, and Barbara Tversky. Virtual replicas for remote assistance in virtual and augmented reality. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, pages 405–415, 2015.

[73] Riccardo Palmarini, John Ahmet Erkoyuncu, Rajkumar Roy, and Hosein Torabmostaedi. A systematic review of augmented reality applications in maintenance. *Robotics and Computer-Integrated Manufacturing*, 49:215–228, 2018.

[74] Thammathip Piumsomboon, Arindam Dey, Barrett Ens, Gun Lee, and Mark Billinghurst. The effects of sharing awareness cues in collaborative mixed reality. *Frontiers in Robotics and AI*, 6:5, 2019.

[75] PTC. Vuforia - Engine. https://www.ptc.com/en/products/vuforia. Accessed: 2020-05-05.

[76] Jason Rambach, Alain Pagani, Michael Schneider, Oleksandr Artemenko, and Didier Stricker. 6DoF object tracking based on 3D scans for augmented reality remote live support. *Computers*, 7(1):6, 2018.

[77] Dulitha Ranatunga, Matt Adcock, David Feng, and Bruce Thomas. Towards object based manipulation in remote guidance. In *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality*, pages 1–6, 2013.

[78] Philipp A. Rauschnabel, Alexander Rossmann, and M. Claudia tom Dieck. An adoption framework for mobile augmented reality games: The case of Pokémon Go. *Computers in Human Behavior*, 76:276–286, 2017.

[79] Guido Maria Re and Monica Bordegoni. An augmented reality framework for supporting and monitoring operators during maintenance tasks. In *Proceedings of the*

International Conference on Virtual, Augmented and Mixed Reality, pages 443–454, 2014.

[80] Dirk Reiners, Didier Stricker, Gudrun Klinker, and Stefan Müller. Augmented reality for construction tasks: doorlock assembly. In *Proceedings of the International Workshop on Augmented Reality: Placing Artificial Objects in Real Scenes*, pages 31–46, 1999.

[81] Nobuchika Sakata, Takeshi Kurata, Takekazu Kato, Masakatsu Kourogi, and Hideaki Kuzuoka. WACL: Supporting telecommunications using wearable active camera with laser pointer. In *Proceedings of the IEEE International Symposium on Wearable Computers*, pages 53–56, 2003.

[82] Tapio Salonen and Juha Sääski. Dynamic and visual assembly instruction for configurable products using augmented reality techniques. In *Proceedings of Advanced Design and Manufacture to Gain a Competitive Edge*, pages 23–32, 2008.

[83] Gerhard Schall, Erick Mendez, Ernst Kruijff, Eduardo Veas, Sebastian Junghanns, Bernhard Reitinger, and Dieter Schmalstieg. Handheld augmented reality for underground infrastructure visualization. *Personal and Ubiquitous Computing*, 13(4):281–291, 2009.

[84] Michael Schneider, Jason Rambach, and Didier Stricker. Augmented reality based on edge computing using the example of remote live support. In *Proceedings of the IEEE International Conference on Industrial Technology*, pages 1277–1282, 2017.

[85] Javier Serván, Fernando Mas, José Luis Menéndez, and José Ríos. Using augmented reality in AIRBUS A400M shop floor assembly work instructions. In *Proceedings of the AIP Conference*, volume 1431, pages 633–640, 2012.

[86] Shu Shi and Cheng-Hsin Hsu. A survey of interactive remote rendering systems. *ACM Computing Surveys*, 47(4):1–29, 2015.

[87] Shu Shi, Klara Nahrstedt, and Roy Harold Campbell. A real-time remote rendering system for interactive mobile graphics. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 8(46):43–56, 2012.

[88] Zhenghao Shi, Hao Wang, Wei Wei, Xia Zheng, Minghua Zhao, and Jinwei Zhao. A novel individual location recommendation system based on mobile augmented reality. In *Proceedings of the International Conference on Identification, Information, and Knowledge in the Internet of Things*, pages 215–218, 2015.

[89] Sanni Siltanen. *Theory and applications of marker based augmented reality.* PhD thesis, Aalto University, 01 2012.

[90] Pieter Simoens, Bojan Joveski, Ludovico Gardenghi, Iain James Marshall, Bert Vankeirsbilck, Miha Mitrea, Francoise Prêteux, Filip De Turck, and Bart Dhoedt. Optimized mobile thin clients through a mpeg-4 bifs semantic remote display framework. *Multimedia Tools and Applications*, 61(2):447–470, 2012.

[91] Ivan Slivar, Mirko Suznjevic, and Lea Skorin-Kapov. The impact of video encoding parameters and game type on QoE for cloud gaming: A case study using the steam platform. In *Proceedings of the 7th International Workshop on Quality of Multimedia Experience*, pages 1–6, 2015.

[92] Maximilian Speicher, Kristina Tenhaft, Simon Heinen, and Harry Handorf. Enabling industry 4.0 with holobuilder. In *Proceedings of INFORMATIK*, pages 1561–1575, 2015.

[93] Mengu Sukan, Carmine Elvezio, Ohan Oda, Steven Feiner, and Barbara Tversky. ParaFrustum: Visualization techniques for guiding a user to a constrained set of viewing positions and orientations. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, pages 331–340, 2014.

[94] Hongling Sun, Yue Liu, Zhenliang Zhang, Xiaoxu Liu, and Yongtian Wang. Employing different viewpoints for remote guidance in a collaborative augmented environment. In *Proceedings of the 6th International Symposium of Chinese CHI*, pages 64–70, 2018.

[95] Lu Sun, Hussein Al Osman, and Jochen Lang. A hybrid remote rendering method for mobile applications. *Multimedia Tools and Applications*, 79(5):1–26, 2019.

[96] Ivan E. Sutherland. A head-mounted three dimensional display. In *Proceedings of the Fall Joint Computer Conference, Part I*, pages 757–764, 1968.

[97] Mirko Suznjevic, Ivan Slivar, and Lea Skorin-Kapov. Analysis and QoE evaluation of cloud gaming service adaptation under different network conditions: The case of NVIDIA GeForce NOW. In *Proceedings of the 8th International Conference on Quality of Multimedia Experience*, pages 1–6, 2016.

[98] Anna Syberfeldt, Oscar Danielsson, and Patrik Gustavsson. Augmented reality smart glasses in the smart factory: Product evaluation guidelines and review of available products. *IEEE Access*, 5:9118–9130, 2017.

[99] Anna Syberfeldt, Oscar Danielsson, Magnus Holm, and Lihui Wang. Visual assembling guidance using augmented reality. *Procedia Manufacturing*, 1:98–109, 2015.

[100] FFmpeg Team. FFmpeg. https://www.ffmpeg.org. Accessed: 2020-05-05.

[101] Franco Tecchia, Leila Alem, and Weidong Huang. 3d helping hands: A gesture based mr system for remote collaboration. In *Proceedings of the 11th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*, pages 323–328, 2012.

[102] John W. Tukey. Comparing individual means in the analysis of variance. *Biometrics*, 5(2):99–114, 1949.

[103] Sebastian Utzig, Robert Kaps, Syed Muhammad Azeem, and Andreas Gerndt. Augmented reality for remote collaboration in aircraft maintenance tasks. In *Proceedings of the IEEE Aerospace Conference*, pages 1–10, 2019.

[104] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martin-Martin, Cewu Lu, Li Fei-Fei, and Silvio Savarese. DenseFusion: 6D object pose estimation by iterative dense fusion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[105] Junfeng Wang, Yaqing Feng, Cheng Zeng, and Shiqi Li. An augmented reality based system for remote collaborative maintenance instruction of complex products. In *Proceedings of the IEEE International Conference on Automation Science and Engineering*, pages 309–314, 2014.

[106] Ming-Jen Wang, Chien-Hao Tseng, and Cherng-Yeu Shen. An easy to use augmented reality authoring tool for use in examination purpose. In *Proceedings of the IFIP Human-Computer Interaction Symposium*, pages 285–288, 2010.

[107] Xiangyu Wang, Sohkhim K. Ong, and Andrew Y. C. Nee. A comprehensive survey of augmented reality assembly research. *Advances in Manufacturing*, 4:1–22, 2016.

[108] Sabine Webel, Uli Bockholt, Timo Engelke, Nirit Gavish, Manuel Olbrich, and Carsten Preusche. An augmented reality training platform for assembly and maintenance skills. *Robotics and Autonomous Systems*, 61(4):398–403, 2013.

[109] Giles Westerfield, Antonija Mitrovic, and Mark Billinghurst. Intelligent augmented reality training for motherboard assembly. *International Journal of Artificial Intelligence in Education*, 25:157–172, 2015.

[110] Thomas Wiegand, Gary J. Sullivan, Gisle Bjøntegaard, and Ajay Luthra. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):560–576, 2003.

[111] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. In *Proceedings of Robotics: Science and Systems*, 2018.

[112] Robert Xiao, Julia Schwarz, Nick Throm, Andrew D. Wilson, and Hrvoje Benko. Mrtouch: Adding touch input to head-mounted mixed reality. *IEEE Transactions on Visualization and Computer Graphics*, 24(4):1653–1660, 2018.

[113] Jürgen Zauner, Michael Haller, Alexander Brandl, and Werner Hartmann. Authoring of a mixed reality assembly instructor for hierarchical structures. In *Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 237–246, 2003.

[114] Jiahong Zhu, Sohkhim K. Ong, and Andrew Y. C. Nee. An authorable context-aware augmented reality system to assist the maintenance technicians. *The International Journal of Advanced Manufacturing Technology*, 66(9):1699–1714, 2013.

[115] Wenwu Zhu, Yiwei Thomas Hou, Yao Wang, and Ya-Qin Zhang. End-to-end modeling and simulation of MPEG-2 transport streams over ATM networks with jitter. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(1):9–12, 1998.

[116] Zhiwei Zhu, Vlad Branzoi, Michael Wolverton, Glen Murray, Nicholas Vitovitch, Louise Yarnall, Girish Acharya, Supun Samarasekera, and Rakesh Kumar. AR-mentor: Augmented reality based mentoring system. In *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality*, pages 17–22, 2014.