FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Real-Time Subtitle Visualizations for the Hearing Impaired in Augmented Reality

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieurin

in

Visual Computing

by

Oana-Aurora Moraru, BSc
Registration Number 1108261

to the Faculty of Informatics

at the TU Wien

Advisor: Priv.-Doz. Mag. Dr. Hannes Kaufmann
Assistance: Dipl.-Ing. Mag. Georg Gerstweiler, Bakk.
PhD Student Mohammadreza Mirzaei

Vienna, 7th September, 2018

Oana-Aurora Moraru

Hannes Kaufmann

Erklärung zur Verfassung der Arbeit

Oana-Aurora Moraru, BSc
Rienößlgasse 11/4, 1040 Vienna

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 7. September 2018

Oana-Aurora Moraru

Acknowledgements

This section is dedicated to acknowledge, thank and pay respects to everyone who supported and facilitated the realization and finalization of this master thesis.
The first person who deserves the most appreciation for all the advice, help and especially moral support along the entire duration of the master thesis is Mohammadreza Mirzaei. Thank you for your meaningful counsel as my second assistant, the motivational speeches and for always having my back as a friend.
Next, I would like to show my respects and thank my lovely parents and grandparents, who always believed in me, crossed their fingers for me and supported me from a distance. Thank you for always being there, cheering me on in successful times as well as brainstorming with me for solutions in difficult times.
The next people, who deserve my deepest respects and gratitude, are the voluntary user study participants, without which the implemented system could not have been properly evaluated. Your participation was so meaningful and dedicated and made the results of the work much more valuable for future systems. Furthermore, I would like to thank professor Hannes Kaufmann for the kindness and motivational conversation.
Next, I would like to offer my sincere gratitude to all my friends, who supported me along the way and especially to the ones who helped during the user studies by participating as guest speakers. The experiments could not have been as successful without your help, thank you for your flexibility, your kindness and your time.
I would also like to express my gratitude to all the organizations and institutions working with deaf and hard of hearing persons involved in the testing of the implemented system as part of the user study. Thank you for distributing the information sheet, spreading the word about the user study and supporting the successful executions of the experiment. This would not have been possible without you.
Finally, I would like to show my gratitude to my assistant and advisor Georg Gerstweiler. Thank you for your patience and the valuable lessons you have thought me during the entire duration of this master thesis.

Abstract

This master thesis aims to provide an introduction to the state of the art of some currently very promising technologies such as speech recognition, speech to text, augmented reality, face detection, face tracking, face recognition and sound localization. The thesis provides a brief overview of the aforementioned technologies in the chapter named Technical Basics. This also allows persons from other fields of expertise to read and understand the thesis. Furthermore, the thesis provides an insight to the current situation of deaf and hard of hearing persons in Austria and explains how the aforementioned technologies can be used in order to aid hearing impaired people in their daily lives. The master thesis furthermore comprised the development of a system prototype able to generate and present subtitles in real time in an augmented reality environment. Additionally, multiple visualization possibilities are provided for the generated subtitles. The implemented system's concept is also explained in this thesis, offering an overview of all involved hardware and software components. Then the implementation process of the developed system is explained in detail, while possible encountered limitations are presented and discussed, as well as identified solutions. The design and implementation process of the visualizations are explained and described as well. However, this master thesis consists only in part of the implementation of a system prototype aiming to ease the participation in meetings of deaf and hard of hearing persons. The second part of the thesis comprises the design and execution of a user study, during which deaf and hard of hearing users could test the implemented system prototype. The user study was organized in order to evaluate the implemented system. At the end of the thesis the extracted results are being analyzed and presented, followed by the conclusion as well as some suggestions for future work.

Contents

CHAPTER 1

Introduction

The following chapter presents the motivation behind this thesis. Furthermore, it explains the current situation of deaf and hard of hearing (D/HH) people and presents ideas regarding how they could be provided with the help they need by means of state-of-the-art technology.

1.1 Current Situation

Communication always plays a crucial role in the transfer of information between people and represents one of the basic human needs [NA16]. As humans are social beings, they make use of their communication skills in all sorts of situations during their daily life and often depend on it to solve their problems. Furthermore, as humans we express some of our most important ideas, feelings and thoughts through various languages. Some people are however hindered in their communication possibilities by various disabilities according to Mirzaei et al. [MGM12] and hearing-impaired people represent a large part of the affected persons. They face numerous problems in their daily lives, which leads to the creation of a barrier between them and the rest of the society, as Vishakh and Khwaja [VK15] describe.

In their paper, Toba et al. [THM+15] state, that although the D/HH persons have some alternate communication methods, such as sign language and lip reading, these are only applicable in one-to-one conversations. These methods cannot be applied in one-to-many conversations, such as meetings or conventions according to Toba et al. [THM+15]. Most of these situations would require the presence of a translator, which is not always possible and is rather expensive.
Since deaf people cannot communicate audibly, Mirzaei et al. [MGM12] state, that they are using other aspects of communication, for example visual and physical means. Deaf and hard of hearing people pay much more attention to lip movement, facial expressions and physical gestures.

Sign language is also a very effective means of communication for D/HH people, however it is constrained by the fact, that the conversation partner must also know the sign language. However, learning this requires much effort according to Toba et al. [THM+15] and can take over five years to master. Even though D/HH people have the possibility to learn and use sign language to communicate with each other, the Japanese ministry states that even among all deaf people in Japan only about 14.1% of them can use sign language, as stated by Toba et al. [THM+15]. Furthermore, hearing people without any hearing disabled relatives or acquaintances have no desire to learn sign language. This enforces the communication gap according to Ahmed et al. [AIuA+16].

Written communication represents another communication means for D/HH people, but it requires more time than oral communication and is also not suited for communication between multiple people. Lip reading is only applicable in certain situations, for example when the listener stands close enough to the speaker and the speaker's lips are visible. But this is not always guaranteed, as Toba et al. [THM+15] states. However even under good reading conditions, lip reading only helps to decode about 30% of the observed speech, as one of the user study participants as part of this master thesis explained. Furthermore, if rather complicated, technical or domain-specific terms are used, lip reading and even sign language also become unreliable.

The realization of this master thesis also comprised a meeting with a field specialist, who works with D/HH persons on a daily basis. Thus, the specialist understands their needs and was also able to provide information about the current situation of D/HH children in the educational system in Austria. Apparently the educational system used nowadays in schools consists in the teacher speaking into a microphone. This enhanced speech is then transmitted to the hearing devices placed on the children's ears. This process helps some of the children, especially the hard of hearing (HH) ones, by emphasizing the speech, however it is still error prone, because it is not interactive. For example when there is a discussion in the classroom or when the teacher asks a question, the HH child cannot always hear the answers. Furthermore, this current approach is not really able to help deaf children in any way. Deaf children can solely rely on the presence of a translator, in case they already possess sign language skills at their young age. Otherwise they are left to observe the teacher's lip movements from a distance and read the written material at home, thus not gaining much from participating in classes.

Therefore, there still is room for improvement on this topic. There is a need for technological solutions aiming to help integrate hearing-impaired people in as many social situations as possible. New promising technological advancements such as speech recognition and augmented reality (AR) are available, which may be used to solve this problems for this specific target group in the future. However, there is no concrete solution yet.

1.2 Target User Group

The implemented system is aiming to help people with hearing disabilities. Among these persons there are two groups of people, the ones who are hard of hearing to a certain extent and the ones who experienced complete loss of hearing and are deaf. The system intends to support both of these groups of people with hearing disabilities. However, their situation is a bit different. According to observations the target user group can be categorized as follows.

On the one hand there are hard of hearing people whose symptoms manifest lightly. This means, they can still communicate through speech rather well with other people, even without the aid of any technology.
If their hearing ability does not suffice for regular conversations, their hearing capability can be enhanced by means of a hearing aid. These are represented by various small devices, which are placed around and inside one's ear. These devices emphasize the auditive input the user receives making it more articulate, thus enabling the conversation.
There are also the cases in which the disabled person experience severe to profound sensorineural hearing loss. In these cases, the root cause is lying in the inner ear and a cochlear implant may be necessary. This involves the implantation of an electronic device, which enables the involved person to experience a sense of sound. However, the implants enable an alternative hearing process, as the generated impulses are not interpreted as regular auditive input at first.
This might frustrate the person in the beginning, since they cannot hear right away, as maybe expected. However, by practicing often with a specialist, the person can learn to interpret the generated impulses as sounds and speech.
The last category is represented by the deaf persons, without hearing devices or implants. Their main means of communication is represented by sign language, a skill which requires a few years in order to be perfected.

1.3 Proposed Solution

This master thesis provides an approach on solving the problem of hearing-impaired persons not getting the most out of attending meetings. The proposed approach uses various state-of-the-art technologies, such as speech recognition, face recognition and AR. The solution integrates components, such as the Intel RealSense depth camera, the Microsoft HoloLens AR headset and the Arduino circuit board, with the aim of partially solving the stated problem. The approach aims to provide a solution for the target use case described above.
By using the system the hearing-impaired would gain more from participating in meetings, because they are provided with subtitles of the currently uttered speech in real time. The system furthermore offers visual cues regarding the location of the current speaker, even if that person is out of sight.

Additionally, the system developed as part of this master thesis provides three different subtitle visualization possibilities for the extracted real-time subtitles and analyzes their advantages and disadvantages.
The first visualization is represented by billboard subtitles, which, according to Debernardis et al.[DFG+14], are optimal for AR text visualization. The second and third visualizations represent combined, dynamic approaches, which switch between regular subtitle visualization, when no speaker is in sight, and speech bubble visualizations, placed next to the speaker's head, as soon as a speaker is in sight.

Furthermore, this master thesis includes and presents the results of a conducted user study with eleven participants, attempting to find out how the users receive the implemented system, as well as which of the three visualizations they prefer. Additionally, the user study analyzes the impact, value and perspective of the proposed system for the hearing-impaired community. At the end of the thesis possible ideas and directions for further research are proposed.
However, it is important to state, that the main focus of this master thesis does not lie on the speech recognition, face recognition or face tracking implementation. These are means, which enable the fulfillment of the main focus, which remains the design, implementation and comparison of various real-time subtitle visualizations for the hearing-impaired in AR.

1.4 Contribution

The proposed system is not just a combination of previous works. Some of the most significant contributions of this master thesis are listed below:

- Design and implementation of various subtitle visualizations, including the composite visualization approach
- Design and implementation of a working prototype, which provides real-time subtitle visualization in an AR environment for D/HH persons
- Implementation of the calibration procedure of RealSense and the HoloLens headset
- The use of an Arduino to switch between the various text visualizations - a unique approach, never before used in combination with real-time subtitles in AR
- Design and execution of a conducted user study with eleven recruited D/HH participants, proving that the users could imagine using a similar system in the future
- Analysis of the user study results and accentuation of the key aspects, which D/HH users desire from such systems

1.5 Hypothesis

Master theses usually involve the setting up of a hypothesis, which is then proven or disproven at the end of the thesis. This master thesis represents no exception, as the following hypothesis was set up: D/HH users prefer the composite visualization approach to the regular billboard subtitles.
Thus, the master thesis aimed to investigate, whether the D/HH users prefer the composite visualization approach, switching from speech bubble to subtitles depending on the presence of the speaker's face, to the regular billboard subtitle approach. The conducted user study served the purpose of testing the implemented system and finding answers to the mentioned investigation. The results can be found in the corresponding chapter at the end of the thesis.

CHAPTER 2

Related Work

In this chapter the most relevant state-of-the-art advancements in fields related to the topic of this master thesis are be presented. Furthermore, this chapter explains how this master thesis is related to or differs from the work presented in each mentioned paper.

Toba et al. [THM+15] are proposing a system for supporting the D/HH people, which provides multi-modal visualizations for its users. The system prerequires, that each user, hearing as well as hearing-impaired, utilizes a device such as a laptop or a mobile phone. Among the provided visualization methods the D/HH user can select the lip reading assistant, which offers a close-up of the speakers mouth. This is enabled using the speaker's camera device. Further methods provide either subtitles or additional extensive definitions of technical terms. The authors mention an integration with AR technology, such as head mounted displays, as a possible future work.

The paper of Toba et al.[THM+15] and the system implemented as part of this master thesis have in common the ability to switch between different visualizations and the intention to help D/HH people. This master thesis could be seen as the continuation of Toba et al.'s work, because in this thesis the integration with AR technology has been realized. This comes however with a new set of requirements as far as text visualization is concerned.
Another relevant difference is, that this master thesis focuses mainly on real-time subtitle visualizations and not on lip reading, which was not the aim of the implemented system. For example, in order to enable lip reading, the speaker has to be inside the user's field of view (FOV) at all times. However, the user of the system implemented as part of this master thesis is informed if somebody is speaking, even if the speaker is not inside their FOV.

Rosten et al.[RRD05] are presenting an approach for placing real-time annotations over a video stream, providing information on the scene. The annotations are positioned in such a way, that they do not obstruct the view to important features of the scene or other annotations. This is achieved using feature density analysis. While they also implemented an AR application, they did not use a head-mounted display. They used a hand-held system.
Rosten et al.'s[RRD05] paper is related to this work, because they also provide a real-time system using text visualizations in AR. However, this master thesis provides real-time subtitles instead of annotations and uses a head-mounted display instead of a hand-held one. The application described by Rosten et al. uses feature density analysis to find the optimal placement of the annotations. Even though the system implemented as part of this master thesis does not use the same approach, it provides the user with various possibilities for text visualization. The user can switch between them in real-time using pushbuttons. Additionally, the presented system uses face recognition to place the text next to the speaker's face.

Debernardis et al.[DFG+14] are trying to improve text readability in head-mounted displays. They experimented with different text colors, styles (plain and billboarded) and backgrounds, testing their efficiency on video as well as optical see-through devices. According to their research, the readability is quicker on optical see-through devices. They furthermore state, that the optimal combination for indoor AR applications is white text placed over a blue billboard.
Debernardis et al.'s[DFG+14] paper represents a good foundation for the text visualizations in AR implemented in this master thesis. Their results were taken into consideration and integrated into the implementation of the system. However, this master thesis continues their research by experimenting with further text visualizations in AR and switching between them on button click. Furthermore, this master thesis focuses on what is truly important for D/HH people, who might have different needs than other people.
Another difference between the mentioned paper and this master thesis is, that the paper only analyzes the aspect of text in AR, while this work focuses on real-time subtitles. This means, that the implemented system visualizes the results of an integrated speech recognition engine and not predefined text.

In their paper Kurahashi et al.[KSZ+17] are aiming to develop a real-world captioning system. They explain, that during their use of see-through head-mounted displays they encounter a problem: when users switch their focus between the captions and the faces or objects in the scene, they experience discomfort and stress. They try to solve this problem by positioning the caption next to the user's face with the help of face recognition technology. Figure 2.1 shows an illustration of how the proposed system is supposed to work. The problems stated by Kurahashi et al.[KSZ+17] in their paper also emerged during the implementation of this master thesis. Another similarity is, that the system implemented during this master thesis also uses face recognition in order to solve the

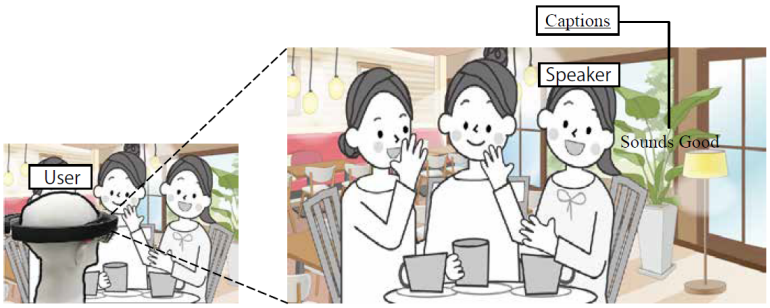


Figure 2.1: Example of real-world captioning system in combination with AR, as proposed by Kurahashi et al.[KSZ+17].

mentioned inconvenience. But the implemented system could provide different and better results, since it uses depth face recognition. Thus, the text can be place exactly next to the speaker's face in 3D. Furthermore, the mentioned paper focuses on real-world captions, while the implemented system in this thesis visualizes real-time subtitles for the hearing-impaired. Thus, besides the visualization aspect, the two works have completely different application scenarios.

The research of Orlosky et al.[OKT13] focuses on solving the text readability problem for see-through head-mounted displays. They aim to solve the problem of changing lighting conditions, while the user is moving. Their approach involves analyzing the scene, trying to find the darkest regions, in order to place the text there for an improved contrast. According to their experiment, the participants preferred this dynamic text management instead of a fixed location, taking lighting changes into consideration.
This master thesis also focuses on text visualization on see-through head-mounted displays. However, because the setting described in the master thesis is an indoor meeting, the lighting conditions are not very likely to change and do not need to be taken into consideration. For this thesis the lighting conditions are assumed to be constant and the main focus lies on the needs of hearing-impaired people. Also, instead of placing the text on the darkest regions, the implementation of this master thesis places the text next to the speaker's face for example, which benefits D/HH users much more. Thus, the target groups are quite different.

Nelson et al.[NRBM00] are claiming, that even though see-through head-mounted displays are very promising because of their various potential applications, they may cause simulator sickness. They studied the subjective ratings of simulator sickness. Their tests analyze time delay, time on task and task complexity. Their results show, that time on task influenced the simulator sickness ratings the most.
Simulator sickness represents one of the major downsides of VR and AR technologies, as Nelson et al.[NRBM00] state. This is why the topic of simulator sickness was also integrated into the user study in this master thesis.

During the user study executed as part of this master thesis a simulator sickness questionnaire is used to evaluate the system's effects on the users.
However, besides the see-through head-mounted displays, there are not many similarities between the mentioned paper and this master thesis. The thesis integrates various other technologies as well, such as an external depth camera and an Arduino controller board.

Dabran et al.[DASD17] are aiming to provide the hearing impaired with the possibility to participate in meetings or lectures, by developing an AR system, which provides real-time subtitles. However, they assert, that the underlying speech recognition is a domain based one. This means, that it needs to be trained in a specific subject, before it can assist the user in a talk on the respective topic.
The main similarity between this master thesis and Dabran et al.'s[DASD17] work is the focus they both share, namely helping the hearing impaired when participating in meetings. However, the system implemented in this master thesis uses an offline speech recognition engine. This is very practical, because it does not require additional training or an internet connection.
While there are many similarities between the two works, the master thesis incorporates some additional functionalities, such as face recognition and speaker awareness.

In their work, Schipper et al.[SB17] are analyzing various placements for captions in one-on-one conversations. They focus on three main factors: whether it is better for the caption to stay in the FOV of the viewer or be placed using world coordinates, whether they should track the view of the viewer or preferably the speaker and what is the most desirable offset from the speaker's face. They conclude, that optimal placement of captions for see-through head-mounted devices differs quite much from stationary screens or hand-held systems. Furthermore, they state, that while others find the locked viewport mode sufficient for their use cases, they could not use it and would rather resort to world coordinates.
The work of Schipper et al.[SB17] is quite related to this master thesis, since they both analyze various placements and visualizations for see-through head-mounted displays. However, they do not focus on the same target group.
Furthermore, the implementation presented in this master thesis also provides a solution for when the speaker is outside the user's FOV.

Garon et al.[GBD+16] aim to enhance the capabilities of the HoloLens headset as well as bypass some of its limitations by pairing it with the RealSense depth camera. They are providing a possibility to enhance the HoloLens device using the mentioned depth camera of a much higher resolution. In order to enable this, they are using a computer stick, which enables the real-time communication between the two devices. In order to present a possible use of the system, they implemented small object detection in 3D.

Even though the work of Garon et al.[GBD+16] has many similarities to the system implemented in this master thesis, such as the intention of enhancing the HoloLens headset using a RealSense camera, there are a number of differences. The first difference is represented by the used RealSense model. The second one is, that the system implemented in this master thesis uses a powerful personal computer as a server, instead of a PC stick, which could not have been able to perform all required operations and calculations per frame. Another difference is the fact, that the mentioned paper focuses on 3D data processing, while the implemented system in this master thesis focuses on subtitle visualization. The system developed as part of this master thesis additionally integrates the speech recognition functionality. The last difference is represented by the application field. In comparison to the system implemented by Garon et al., the implemented system in this master thesis was designed for aiding D/HH people.

The work of Mirzaei et al. [MGM12] is presenting a system resulting from the combination of AR and Automatic Speech Recognition (ASR) among other technologies like Audio Visual Speech Recognition (AVSR) and text to speech (TTS). The work shows, that such a system, using speech recognition in AR, is very helpful and useful for D/HH persons. Mirzaei et al.'s[MGM12] paper represents an important foundation for this master thesis, because it presents the need and interest of D/HH people regarding AR solutions for their problems. Thus, the target group is the same. However, Mirzaei et al.'s [MGM12] paper focuses more on the speech recognition and signal processing approach.
Furthermore, the implementation in this master thesis focuses on the different subtitle visualization possibilities for D/HH people. The users can switch between visualizations in real-time and are informed, when somebody is speaking, even if the speaker is outside the user's FOV. The user is then also pointed in the correct direction of the speaker.

Ohene-Djan et al.[ODS06] are underlining the importance of subtitles for the hearing-impaired, when aiming to learn from film or television. However, they are trying to revolutionize their effectiveness introducing emotional subtitles. These aim to convey more information than the usual subtitles by means of colors and fonts. For example, emotional subtitles are trying to transmit additional information to the user, such as the identity of the speaker, the feelings of the speaker and even information regarding the content of the sentence. Ohene-Djan et al.[ODS06] encourage the user interaction and personalization of the e-subtitles.
While both the work of Ohene-Djan et al.[ODS06] and this master thesis aim to help the hearing-impaired people by providing them with subtitles, this master thesis aims to provide them in real-time, during a meeting. For this reason, further information encoded into the subtitles, such as font, color or size, would not be constructive. Furthermore, the exact emotions of the speakers are not relevant for the targeted setting of a business meeting. The focus in this situation lies more on the content of the speech and less on its delivery.

Another interesting approach for solving the communication problem of D/HH people is presented by Ahmed et al. [AIuA+16]. They propose a system, which can interpret sign language and turn it to speech, using a Microsoft Kinect device for Windows.
Furthermore, the hearing person's speech can also be transformed into sign language, using speech to text methods. The extracted text is then mapped to prerecorded sign language animation sequences, which are then performed by an animated 3D model. They use the AT&T's speech recognition server with the AT&T's SDK and Unity3D for their project.
While Ahmed et al.[AIuA+16] also aim to help the D/HH people and even use Unity3D for their implementation, their main focus lies on sign language. Whereas, in this master thesis it is assumed, that not many people know sign language inside a company and thus, speech recognition is used. Furthermore, for sign language to be interpreted correctly, the person performing the signs has to be inside the user's FOV. However, using a speech recognition engine, like in the implementation of this master thesis, allows the user to not necessarily have to look at the speaker during his entire speech.

Lastly Jain et al.[JFG+15] are also working with transparent head-mounted displays and intend to aid the D/HH person in another way. They focus on helping the hearing-impaired user to notice, when someone, who is currently outside of their FOV, is talking to them. In order to do so, they built a proof-of-concept, which provides various way of visualizing sound location. The authors are using a sound localization algorithm and display visualizations on the Google Glass, indicating to the hearing-impaired user, where the sound is coming from.
Indicating to the user, that someone outside of their FOV is speaking, also represents one of the goals of this master thesis. Therefore, an according visualization has been integrated into the master thesis implementation as well. However, in this thesis the location of the speaker is identified using hardware buttons, which are much more robust and reliable than state-of-the-art sound localization algorithms.
The most important difference between the two works however is, that the mentioned paper only focuses on the direction of the speech, not on the text visualization. The implementation of this master thesis however also interprets the speech, transforms it into text and provides various subtitle visualizations for the subtitles in real-time, while also taking the direction of the speaker into consideration.

CHAPTER 3

Technical Basics

Over the last few years numerous new technologies have emerged as a result of intense research. Among these technologies there is AR, face recognition, speech recognition and sound localization. These new technologies provide various new possibilities for aiding people with disabilities, as stated by Mirzaei et al.[MGM12]. Furthermore, they have also been used in the system implemented as part of this master thesis.
This chapter provides a brief introduction of each of these technologies. This aims to create a minimal, required basis of understanding, which aims to also enable people from other areas of expertise to get the most out of this master thesis.

3.1 Augmented Reality

AR is one of these new technologies, which has awoken much interest in researchers lately, as stated be Mirzaei et al.[MGM12]. Furthermore, the source states that AR, as well as Virtual Reality (VR) can potentially improve the lives of people with disabilities.
The main difference between AR and VR is, that in VR the user is completely separated from the real world. He or she has the impression of being in another, virtual space. In comparison, when using AR, the user can still see the real world through the see-through glasses or head-mounted display. However, Rosten et al.[RRD05] state that AR aims to enhance the user's perception of the real world, by additionally providing virtual or computer-generated items. These virtual items can be for example 2D overlays or 3D objects. Figure 3.1 shows an illustration of how such virtual entities might be presented when wearing an AR headset. The virtual components are visualized in the FOV of the environment. AR also enables the users to interact with the virtual components. This way they can adapt the presented information optimally to simplify their interaction with other people, as stated by Mirzaei et al.[MGM12].
In the case of this master thesis the D/HH users can switch between subtitle visualizations to select the one, which improves their interaction with the speakers the most.

Figure 3.1: Example of virtual interface provided in the context of the real world using Microsoft HoloLens head-mounted display.[ARg].

Among the current problems of AR technology is the fact that the FOV is very limited in comparison to the FOV of the user's eyes. This means that virtual items can only be rendered in a limited rectangular space centered around the optical axis of the user. Other aspects that proved to be problematic, when using AR technology, are the changing lighting conditions and background patterns. These can cause difficulties especially for text visualization.

3.2 Speech Recognition

Natural language processing has been a heavily researched field in the last years, according to Sharma and Sardana[SS16]. While multiple applications of natural language processing exist, speech recognition represents one of its most significant applications. Sharma and Sardana[SS16] define speech recognition as the means by which computers understand human language, also referred to as natural language.
Speech recognition is not to be confused with voice recognition, which has entirely different objectives, as stated by Rocha et al.[RFD+16]. Voice recognition aims to identify or recognize a specific individual's voice and is language independent. Meanwhile speech recognition is highly language dependent. Its purpose is the detection of the individually uttered words within the registered speech. In the paper of Rocha et al.[RFD+16] speech to text software is defined as a type of software, which turns audio content into written words, displaying it on a display destination.

Speech recognition requires an acoustic model as well as a language model for the representation of the statistical properties of speech. The acoustic model's task is to model the relationship between the audio signal and the phonetic units in a language. Meanwhile the language model models the word sequence in a language.

By combining these two models one can get word sequences, which correspond to the input audio segment, as Nouza et al.[NBB+15] state.

Speech to text, as described by Mirzaei et al.[MGM12], can be applied in numerous situations. One of them is the simplified integration and participation of hearing-impaired individuals into conversations with one or more hearing people.
Furthermore, there exist online and offline speech recognition systems. They both have their own advantages and disadvantages.
In the case of this master thesis an offline speech recognition in combination with speech to text are used. They interpret the speech produced by the speakers, and present it to the D/HH user on the AR headset as real-time subtitles. Figure 3.2 visualizes how speech to text works.

3.3 Sound Localization

D/HH persons are using visual signals, as for example lip movement, facial expressions and body language to interpret and understand speech. This process is defined as speech reading, according to Jain et al.[JFG+15]. An important prerequisite for this however, is that the speaker is located in the FOV of the disabled person. Otherwise the person with hearing loss does not know where to place their visual attention.
Even though D/HH persons rely on cochlear implants or hearing aiding technology for help in sound recognition, these technologies do not inform the wearer about the location of the speaker. Thus, a technology is needed in order to inform the D/HH users about the speaker's location. This technology is called sound localization.

As Jain et al.[JFG+15] explain, sound localization helps the user identify the position of the speaker, as soon as the speech starts, regardless of whether the speaker is in the FOV of the D/HH person or not. This enables the D/HH person to turn towards the speaker and notice the necessary visual cues. This can simplify their understanding of the conversation.
Finally, the information regarding the speaker's location has to be presented to the user. Jain et al.[JFG+15] suggest using an AR headset and visualizing the direction of the incoming speech on the head-mounted displays. In the system implemented as part of this master thesis, sound localization is being simulated using pushbuttons. These are used to inform the D/HH user about the current speaker's location during a conversation among multiple oral partners.

3.4 Face Recognition

Taking all pattern processing applications into consideration, face recognition is surely one of the most successful ones. Numerous research efforts have been directed towards it recently, according to Zhao et al.[ZCPR03].

Figure 3.2: Visualization of speech to text.[S2T].

Anggraini et al.[ARL16] describes the face as being the frontal part of the human head, covering the area between forehead and chin. Among the facial features, which can be tracked, for example the eyes, nose, mouth or lips, eyebrows, cheeks or teeth can be taken into consideration. A combination of these features can be used to distinguish a face from others, as mentioned by Anggraini et al.[ARL16].
Even though they are related, face recognition is not to be confused with face detection or face tracking. Face detection is mainly used to identify one or more faces in a static image. Meanwhile, face tracking is used to identify and also track or follow a face among multiple images or within a video stream. Face recognition also involves face tracking, but it additionally categorizes the tracked faces into two categories, namely recognized and unrecognized, as Anggraini et al.[ARL16] state.
If a face is unrecognized, an unique identifier or pattern is assigned to that respective face. This pattern can then be saved in a database, enabling the recognition of that face at a later moment in time.
One can furthermore distinguish among 2D and 3D face recognition. During 2D face recognition the facial features are identified only by means of the 2D image and its changes in intensity. Meanwhile, 3D Recognition also takes the depth image of the face into consideration. The proposed system uses Intel’s RealSense SR300 camera for 3D face recognition and 3D face tracking in order to identify the faces of the speakers.

3.5 Camera Calibration

In computer vision camera calibration often represents a necessary step, as stated by Zhang et al.[Zha00]. This is the case when metric information needs to be extracted from 2D images or video streams. Camera calibration is also required when more than one camera is used in an application and a common coordinate system needs to be agreed upon.

Zhang et al.[Zha00] describe that there are two possibilities for camera calibration. The first possibility uses a three-dimensional reference object, for example a marker, whose exact 3D space geometry is known with very good precision. This object is being tracked by the camera, which needs to be calibrated. Because the object's coordinates and dimensions are known, the camera's exact position can be determined as a result.
The second possibility does not need any kind of marker. Zhang et al.[Zha00] explains, that it is sufficient to move the camera within a static scene and take numerous pictures. Afterward, the images are analyzed, and if correspondences between at least three images are identified, both the extrinsic and intrinsic parameters of the camera can be extracted. The extrinsic camera parameters describe the position and rotation of the camera in world space. Meanwhile, the intrinsic parameters depend on each individual camera and comprise the focal length, the optical center and the skew coefficient.

The system proposed by this master thesis uses two different cameras, the Intel RealSense and the Microsoft HoloLens. Even though the two cameras have the same intrinsic parameters, the extrinsic camera parameters are different. Both cameras have their own coordinate system, but behave differently in relation to their coordinate system's origins. The RealSense camera behaves as the origin of its coordinate system, while the HoloLens camera does not. However, since the two devices need to communicate with each other, a calibration step was introduced. This step is necessary in order to get both cameras into the same coordinate system. The calibration is performed using Vuforia marker tracking.

3.6 Text Visualization

Text visualization represents the way text is presented to the user and is efficient only when it improves visibility and readability. However, text visualization becomes problematic in the context of head-worn AR devices, as stated by Debernardis et al.[DFG+14]. This is caused by the text visualization's sensitivity to the technology of the display, ambient illumination, background and of course the style and color of the text.

One of the most significant characteristics of AR applications is the enhancement of the real world by superimposing digital information. However, in the case of see-through glasses, mostly used in AR, the background affects the readability of the text tremendously. The easy and safe reading of text while using optical see-through devices has been a challenge for numerous years, as stated by Orlosky et al.[OKT13]. The complications emerge due to dynamic, changing environments, lighting conditions and various obstructions in the path of the user. For example a dark text, which might be suited for bright backgrounds, may become unreadable as soon as the user turns his or her head and the text is placed over a darker background. Thus, effective text management is needed. The implemented system as part of this master thesis provides three different subtitle visualizations.

Figure 3.3: The figure represents the Microsoft HoloLens headset.[20118c].

3.7 Hardware Components

The system includes various pieces of hardware components, which were chosen in concordance with the given requirements. The requirements were to provide real-time subtitles in an AR environment in order to aid D/HH people. Furthermore, various subtitle visualizations needed to be tested and the switch between them had to be enabled. The chosen hardware components were integrated in order to work together in the implemented prototype. The hardware components include a RealSense camera, a HoloLens headset, an Arduino Toolkit. Besides those enumerated components, the system also requires a powerful personal computer (PC) and a router.

3.7.1 HoloLens

The first chosen hardware component is the HoloLens. The device has been developed by Microsoft and represents one of the first AR, also called Mixed Reality, headsets available on today's market, as Garon et al.[GBD+16] state. The same source explains, that the HoloLens has had a huge impact on AR application development, due to its intuitive interface, and its availability to developers, who can now implement and deploy AR software applications on it.
Among other remarkable features there is the fact that it allows the user to interact with holograms. Its portability also represents a great advantage, because the device can be seen as a portable computer not needing to be connected to any external machines. Furthermore, it provides a large amount of advanced sensors, which can be used to capture information about the environment, according to [20118c]. Because of all the previously mentioned advantages, the device was chosen for the implementation of the system in this master thesis. Additionally, the HoloLens provides one of the largest fields of view currently available among AR glasses.
Figure 3.3 visualizes the Microsoft HoloLens headset.

3.7.2 RealSense

Even though the HoloLens provides facial and speech recognition services as part of the Microsoft Azure Cognitive Services[Mic18], these services are not free of charge. This does not fulfill the requirements, so that a solution needed to be found. The solution

Figure 3.4: The figure represents the RealSense SR300 camera.[20118b].

involves the use of a second device, which provides similar services, however for free, and can be used in order to enhance the capabilities of the HoloLens.
The chosen hardware component to fulfill this task is Intel's RealSense SR300 camera. The device includes a full 1080p color camera at 30 FPS, depth perception with the optimal distance between 0.2m and 1.5m, as well as a dual microphone audio subsystem[20118b]. In order to enable the communication between the RealSense and the HoloLens, a network was set up. In this network, the RealSense fulfills the role of an input device, being part of the server. Meanwhile, the HoloLens plays the role of the client, visualizing the results to the user. Figure 3.4 visualizes the Intel RealSense SR300 camera.

3.7.3 Arduino

Another hardware component used for the implementation part of this master thesis is an ELEGOO UNO R3 Project Complete Starter Kit[Inc18a]. The kit contains very useful components, such as an UNO R3 Controller Board (perfectly compatible with Arduino UNO R3), Breadboards, buttons and keycaps, various resistors, Female-to-male Dupont Wires, an USB cable and numerous Jumper Wires.
The UNO R3 Controller Board represents a simplified microcontroller board, as described in[Ard15], and can be programmed using the Arduino Software IDE.
The kit, and especially its contained pushbuttons, can be used in order to fulfill the requirement of switching between the subtitle visualization types. This component was chosen because the buttons can be easily programmed using the Arduino IDE and because of the seamless integration with Unity.

The Arduino was used in the implementation of this master thesis, because the speech recognition provided by the Intel RealSense SDK has some limitations. The most significant one being the fact, that there is no possibility to distinguish between speakers. The RealSense takes all sounds as one single audio signal as input. This means, that the system only returns valid results, if the speakers talk one at a time. There is no way of knowing where the speech comes from, as sound localization is not provided.
D/HH people can use their visual abilities to identify the speaker. Visual cues, such as body language or lip movement, are very helpful. However, these cues can only be of use when the speaker is in the FOV of the system. The speaker needs to be actively focusing on these details.

For this reason, a solution had to be developed, in order to let the D/HH user of the system know, if a speaker is currently talking and which speaker it is. These aspects needed to work, even when the speaker was out of sight of the system's user.
In order to do so, and because the main use case of the system was facilitating the participation of D/HH persons in meetings, a button approach was implemented. The approach aims to simulate sound localization using the mentioned Arduino kit.

3.7.4 Personal Computer

Another very important hardware component of the system is a powerful PC, which satisfies the system requirements of the Intel RealSense SR300 camera[Cor18b]. The PC was chosen because it fulfills all necessary requirements in order to communicate with the RealSense camera. The used PC needs to integrate at least a 6th Generation Intel Core Processor, needs to run Windows 10 64-bit as Operating System and lastly needs to include at least an USB 3.0 Port which provide enough power to fuel the RealSense Camera. Previously used laptops did not manage to fulfill this task.

3.7.5 Router

The last but nevertheless very important hardware component of the system is a router. It was used to create a private network and was directly connected to the PC. The network created by the router provided an IP address, which was used in order to connect the HoloLens client over WIFI to the server PC. Thus, the router practically enabled the communication between RealSense and HoloLens, respectively between server and client.

3.8 Software Components

Following the description of the hardware components of the implemented system, the software components, the used toolkits and software development kits (SDKs) are presented. One of the most important requirements regarding the chosen software was the fact, that it needed to be free of charge.

3.8.1 Unity3D

Unity is a popular cross-platform game engine, with an integrated development environment (IDE), which has been developed by Unity Technologies [Tec18c] and can be used for 2D as well as 3D application development.
Unity enables its users to place game objects into the scene, choose certain properties for them and attach scripts to them, in which their behavior can be specified.
Unity was chosen to serve as IDE during the development of the system described in this master thesis, since it is able to communicate with both the HoloLens headset and the RealSense camera. For the implementation of the system in this master thesis Unity 2017.3.1f1 (64-bit) was used, while the used programming language was C#.

3.8.2 RealSense Toolkit

The RealSense Toolkit is needed to develop applications using the Intel RealSense camera. It provides multiple extremely useful features such as face tracking and face recognition, hand gesture recognition, emotion recognition, background segmentation and 3D scanning in order to reconstruct real-world objects as 3D meshes, as [20118b] state.
For all integrated facial recognition algorithms, the Intel's RealSense SR300 camera uses landmark detection on the depth image of a face. For feature extraction the device uses an geometric feature based approach, as explained by Patil et al.[PB16]. The same source states that the RealSense is using the distance between landmarks in order to classify the features.
In order to download the SDK one can go to the following source[Cor18a]. For the implementation part on this master thesis version 10 of the SDK was used, 2016 R2 Release Notes (10.0.26.0396).

3.8.3 Mixed Reality Toolkit

The Mixed Reality Toolkit represents a collection of components and scripts, which aims to encourage and accelerate the development process of applications destined for the Microsoft HoloLens headset, as emphasized by[20118e]. The source furthermore explains, that the MR Toolkit was introduced in order to minimize barriers for developers wanting to build applications for the HoloLens. It is indeed very helpful, because example scenes and HoloLens specific prefab game objects can be reused during development.
For the implemented system in this master thesis MixedRealityToolkit-Unity was used, which utilizes code from the initial MixedRealityToolkit and facilitates the development of AR applications targeting the HoloLens in Unity. The toolkit for Unity can be downloaded from GitHub[20118e].

3.8.4 Arduino Software IDE

The Arduino Software IDE is an open-source software, which enables users to write code and upload it to any Arduino Controller Board, as explained by[Ard18c]. They furthermore mention, that the IDE runs on Windows as well as on Mac OS X and Linux. Once the IDE is opened it looks like a text editor, which enables the writing of Arduino code.
The programming language used by Arduino is quite similar to C++ and can communicate with the Arduino board. However, the communication to numerous other hardware components, included in the ELEGOO UNO R3 Project Complete Starter Kit, is enabled, once these are attached to the board. There are two versions of the Arduino Software IDE: an Arduino Web Editor and an offline desktop IDE. For the development of the system described in this master thesis the desktop version was used. The Arduino Software IDE can be downloaded from here[Ard18c].

3.8.5 Vuforia

Vuforia is a SDK which supports the implementation of AR applications. Using computer vision technology it enables the real-time recognition and tracking of image targets. Using this approach, users can place various virtual 3D objects on said tracked 3D world positions. Vuforia works with any camera, even cameras of mobile devices.
Vuforia provides APIs in C++, Java, Objective-C++ and also offers an extension for Unity, and can be downloaded from the following source[Inc18c]. The marker tracking capabilities provided by Vuforia were used in the implementation of the master thesis for the calibration part. Details are presented in the Integration and implementation chapter.

CHAPTER 4

System Design

In this chapter the most important aspects regarding the design of the implemented system are presented. First, the general concept of the implemented system is explained, including the hardware setup in space and the components' interaction. Then, the data exchange is described, followed by the system's architecture and the visualization designs. After this chapter, the reader should have a good overview of how the implemented system works.
Figure 4.1 shows an abstract representation of the implemented system.

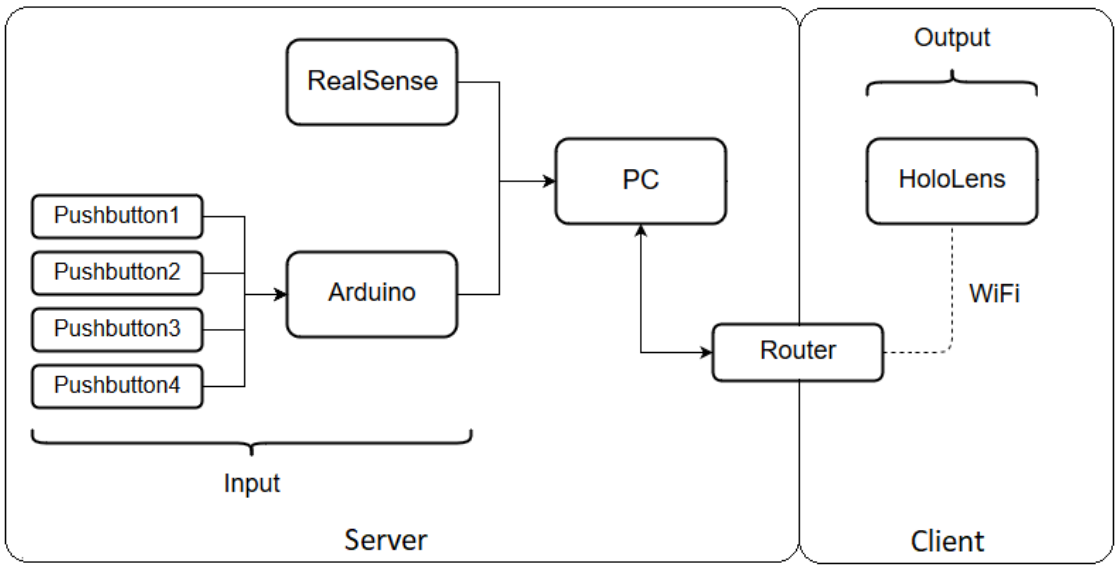


Figure 4.1: The represented graph visualizes the implemented system, its hardware components, as well as the way they interact with each other.

Figure 4.1 not only illustrates the design of the implemented system, but also the interaction of its components. The graphic shows, that the RealSense and the Arduino constructions, including the four pushbuttons, serve as input devices. They send data to the PC, which plays the role of the server. After performing certain calculations, the server sends the result over to the HoloLens client using the wireless connection enabled by the router. The HoloLens serves as output device. It does not make additional calculations, but handles all visualization aspects, presenting the results to the user.

Because the implemented system needs to integrate multiple hardware and software components, the setup of a network architecture became mandatory. Also the decision, regarding which device should play the role of the server and which device should fulfill the role of the client, had to be made.

Server

The role of the server in the implemented system is fulfilled by the PC. The large box on the left of Figure 4.1 comprises all server components, including its input devices. Some of the server's most important tasks are presented below:

- receiving the position and rotation of the HoloLens camera from the client
- receiving the position and rotation of the calibration marker tracked by the HoloLens camera
- receiving the position and rotation of the calibration marker tracked by the RealSense camera
- handling all calculations necessary for the calibration of the two cameras
- handling the game object management, controlling which components are deactivated as well as activated after the calibration is fulfilled
- receiving the speech signal from the Intel RealSense SR300 camera, as soon as a speaker starts talking
- handling the conversion from speech signal to text
- receiving the facial landmarks from the Intel RealSense SR300 camera, as soon as the face of a speaker enters the FOV of the RealSense
- extracting the 3D position from the facial landmarks and modifying it according to the calibration result
- receiving the signals transmitted by the Arduino board, when a speaker is pressing his or her button
- saving all the calculated data on a centralized object, from where the client can access it later

Client

The Microsoft HoloLens plays the role of the Client in the implemented system. The large box on the right of Figure 4.1 shows, that the client comprises one single component, the HoloLens headset. Some of the client's most important tasks are presented below:

- sending the position and rotation of the HoloLens camera over the network to the server
- sending the position and rotation of the calibration marker tracked by the HoloLens camera over the network to the server
- receiving the information regarding the subtitle text and updating the text visualizations accordingly
- handling the deactivation of the subtitles after a specified amount of time passes
- receiving the information regarding the current speaker and updating the sound localization visualizations accordingly
- receiving the information regarding the currently selected visualization and switching between them accordingly
- handling the visualization regarding the network status, informing the user in case the network connection between server and client has been interrupted

4.1 Hardware Setup Design

Following the presentation of the hardware components individually, their hardware setup design, connection and placement in space needs to be described. Starting from left to right in Figure 4.1, the four pushbuttons are connected to the Arduino board using numerous jumper wires and female-to-male dupont wires. The Arduino board is then connected to the PC using an USB cable. This entire Arduino construction can be placed on a table, in the proximity of the server PC. The PC is furthermore connected to the router using an ethernet cable. The router can also be placed on a table, while the PC lays on the ground beneath the table.

The HoloLens headset can function wireless, communicating with the server PC using the router. The only situation in which the HoloLens requires an USB connection to the PC, is when a new build needs to be deployed on the headset. The HoloLens does not have a fixed position in space, as it is placed on the D/HH user's head, which can move and turn.

The placement of the RealSense camera represented a greater challenge. In order to simplify the calibration calculation by performing it only once in the beginning, instead of each frame, the RealSense needed to be fixed upon the HoloLens. In order to ensure this, a rigid connection needed to be constructed, so that the two devices could move together and look in the same direction. Furthermore, it was very important, that this reference offset between the devices did not change over time. This would invalidate the calibration, as explained in the according section.

The first solution involved the use of hook-and-loop fastener stripes, with a sticky side. Suitable shapes were cut out of the hook-and-loop stripes and fixated on the top of the HoloLens as well as on the RealSense. However, after positioning the two devices on top of each other, it became clear, that this solution was not stable enough. The RealSense moved around, which caused much inaccuracy.
The second tried construction design was to flip the RealSense camera upside down and then attach it to the HoloLens. This increased the stability of the construction, however it became clear very fast, that the face tracking and recognition of the RealSense cannot recognize faces upside down and they could not be flipped successfully.
The final and most stable approach was represented by the RealSense being used in its upright position, mounted on a fitting stand. The stand was fixated to the HoloLens using very powerful double-sided adhesive tape. The final construction can be visualized in Figure 4.2.

Arduino Hardware Design

For the execution of the hardware part of the sound localization simulation the ELEGOO UNO R3 Project Complete Starter Kit was used. The wiring diagram of the Arduino construction is presented in Figure 4.3. The kit contains numerous components, however for the hardware design of the Arduino construction the following were used:

- the UNO R3 Controller Board - is the programmable simplified microcontroller board[Ard15], on which the code written using the Arduino IDE is run. The UNO 3 Controller Board controls the interaction of all other connected hardware components.
- Breadboards - the large breadboard is connected directly to the 5V power supply provided by the UNO R3 Controller Board and is used to distribute the electrical power to the four smaller breadboards. The smaller ones were used in order to enable the distributed placement of the buttons. Thus, each button can be moved and placed in the proximity of its corresponding speaker. The system comprises four buttons at the moment, supporting three distinct speakers. However, the number of speakers can easily be incremented.
- Buttons and keycaps - as mentioned, four buttons are currently provided by the system. The hardware button components provided by the kit were very small,

Figure 4.2: The figure shows the final solution developed to mount the RealSense camera on the HoloLens headset in a stable way using a stand.

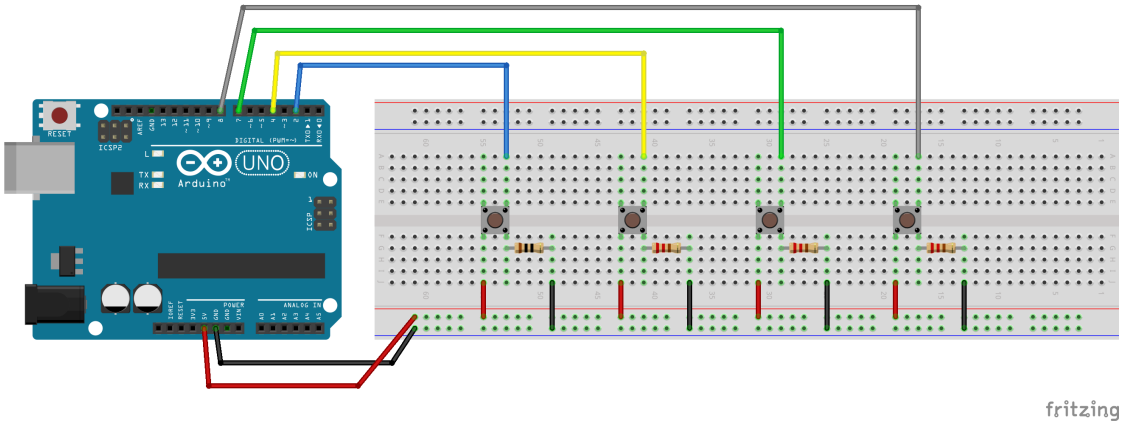


Figure 4.3: The figure visualizes the wiring diagram of the Arduino construction. The diagram was created using the open source software Fritzing[oFf18]. The colors chosen for the wires depicted above the breadboard, are referencing the keycap color of each individual button in the final hardware constellation.

which made their usage difficult. Thus, keycaps in various colors were attached to the buttons, in order to enlarge them and increase their usability. Each of the four augmented buttons are then attached to one of the four small breadboards. Three of the buttons, blue, yellow and green, are destined for the speakers. The forth button, the gray one, is enabling the visualization switch. The following official tutorial provided by Arduino was very helpful during the implementation[Ard18b].

- Resistors - four resistors were used, one for each button. Their role is to absorb some of the excessive electrical power flow. This prevents a short circuit from happening, which would damage the buttons.
- an USB cable - is used to connect the UNO R3 Controller Board to the PC and provides the necessary electrical power.
- Jumper Wires - 26 jumper wires were used for the hardware construction. They were used to establish the connection between the UNO R3 Controller Board and the main breadboard, as well as supply it with electrical power. They were also used to connect the main breadboard with the small breadboards and further distribute the electrical power to all the components. Lastly, they were used to connect the small breadboard units back to the UNO R3 Controller Board.
- Female-to-male Dupont Wires - were used to extend the Jumper Wire connections where necessary. This allows for more liberty of movement and further away placement of the button units.

Figure 4.4 shows the complete Arduino pushbutton hardware constellation.

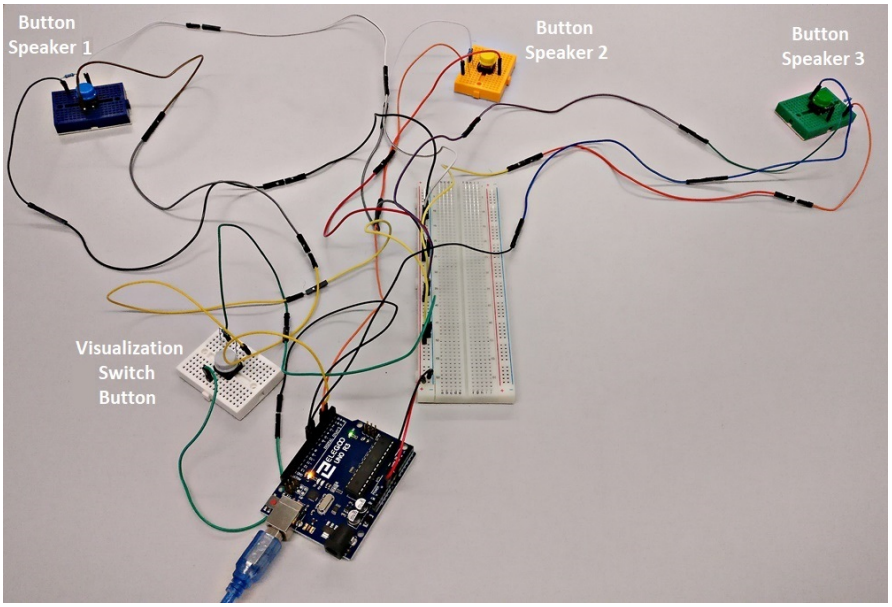


Figure 4.4: The figure shows the final Arduino pushbutton constellation. As specified in the figure, the upper three buttons have a different purpose than the lower, gray one. The colorful buttons are intended to be placed next to each speaker, so that the respective speaker can hold it pressed while speaking. Thus the system is informed about where the current speaker is placed in space. Meanwhile, the gray button fulfills the role of switching among the different subtitle visualizations.

The working principle of the constructed hardware button constellation is similar to the procedure used in teleconferences. When a person wants to speak, they register by pressing the button assigned to them, placed in their proximity. While the speaker is talking, he or she keeps the button pressed. As soon as he or she has finished, the button is released. The input signal from the button, as well as the inputted speech signal from the RealSense, are both sent to the server over USB cables and allow further processing.

4.2 Software Architecture Design

As it was clear from the beginning, that a calibration step was necessary, the optimal execution and integration of the calibration step was designed. Had the RealSense not been mounted on the HoloLens, the calibration would have been necessary in each frame, which would have resulted in an entirely different architecture. However, by fixing the RealSense on top of the HoloLens headset and ensuring that they would move together, one single execution of the calibration step, right after starting the application, was enough. Figure 4.5 shows an overview of the final software architecture including the calibration step.

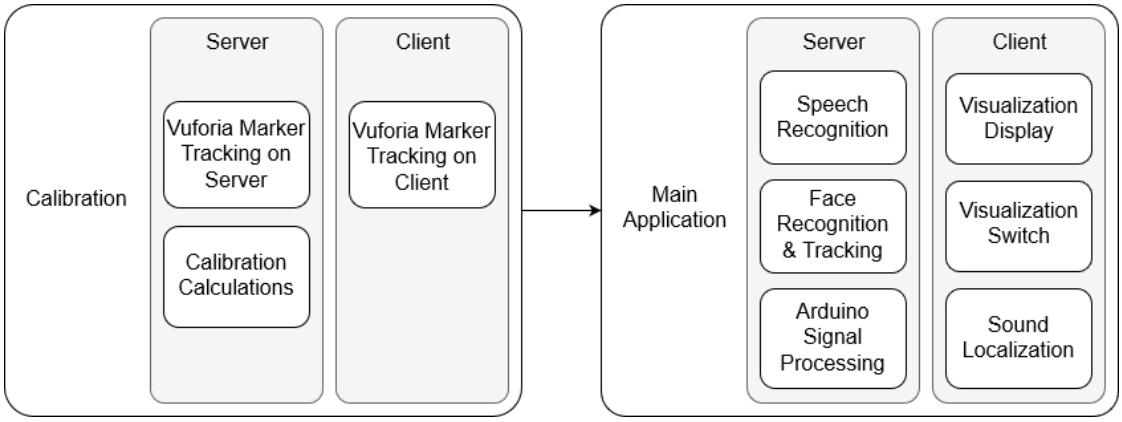


Figure 4.5: The diagram the application's final software architecture. First, when the application is started, only the calibration components are active, so that the calibration can be successfully executed once in the beginning. After the calibration is finished, the components belonging to it are deactivated, while the remaining necessary components for the main application are activated. Afterward, the main application can finally start and includes significant tasks such as speech to text conversion, face coordinates extraction and pushbutton signal processing.

4.2.1 Calibration Design

A calibration step is necessary, since the implemented system incorporates two very different devices, each having their own camera. These cameras have nothing to do with each other. They have completely different extrinsic parameters. Each of the two cameras has its own coordinate system and in addition, each camera manifests a different behavior in relation to the center of their corresponding coordinate system.

After some investigation it was discovered, that the RealSense camera, behaves as if it is the origin of its coordinate system. Its initial coordinates never change in the application. On the contrary, the objects viewed by the RealSense camera, which are actually static in the real world, appear to be moving on the server application. Their coordinates change, when the RealSense camera moves, while the coordinates of the RealSense remain unchanged.
Meanwhile, the HoloLens does not act as the origin of its coordinate system. Its coordinates change as the HoloLens device moves in the real world. Thus, it became clear, that a solution had to be found in order to get both devices in the same coordinate system. This meant, that a geometrical transformation had to be calculated in order to convert an object from one space to the other.

Thus, before the system could execute its main and already implemented operations, the insertion of another step, the calibration, was needed. The calibration step is intended to take place before everything else, only once in the beginning, right after the start of

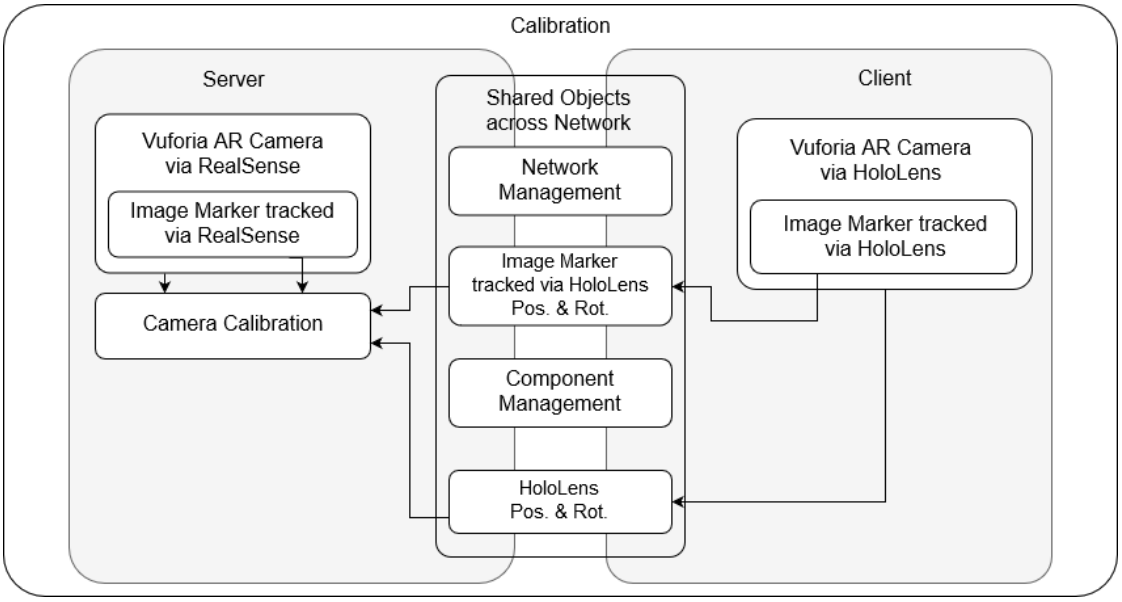


Figure 4.6: The diagram visualizes object components as well as the data flow in the calibration part of the application. The box on the left represents the server with all its components. The box on the right visualizes the client components. The box in the middle connecting the server to the client represents the network, as well as the objects which are shared across it.

the application. It was designed to be triggered by pressing the Space button on the keyboard attached to the PC acting as server.
To facilitate the calibration of the two cameras an object is required, which can be tracked by both the server and the client application. This results in two very different positions and rotations. The goal is to afterward find the transformation between these two tracked positions and rotations. Finally, the tracked object's position and rotation from server and client should overlap in the HoloLens client application.

Figure 4.6 visualizes a schematic overview of the calibration part of the application, as well as the components which are present in the scene as soon as the application is run. Regarding the data flow, one can notice, that the client is sending important position and rotation information to the server. The camera calibration component of the server then handles all necessary calibration calculations.

4.2.2 Main Application Design

After a successful calibration, the calibration components which are not needed anymore are deactivated. At the same time, the components of the main application are initiated. This main application represents the principal focus of the master thesis, as it provides the actual subtitles in the form of various visualizations in the AR environment.

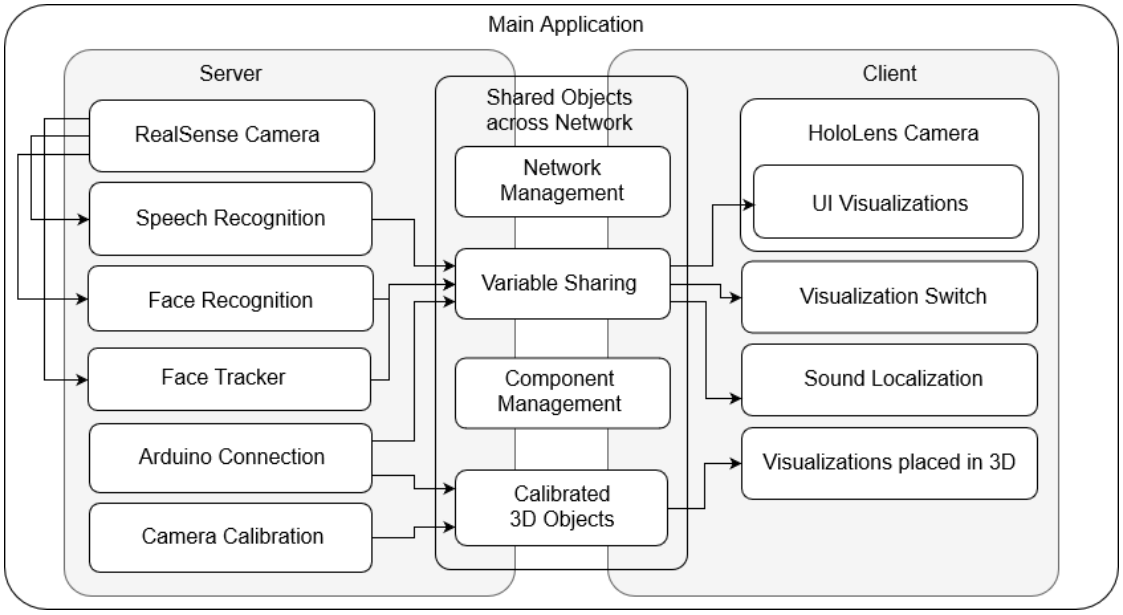


Figure 4.7: The figure visualizes the components of the main application. Furthermore, it aims to provide an overview of their interaction.

For a better understanding of the of the main application, first an overview of the application's components is provided, followed by the presentation of the data flow.

Figure 4.7 provides an overview of the most important components of the main application. Similarly to the structure of the calibration application implementation, the client-server architecture is maintained. On the server, the Vuforia AR camera is however replaces by the RealSense AR camera component and a few other new components are instantiated by the component manager. The new components on the server side handle the input data from the RealSense and the Arduino and save the results on the variable sharing component.
On the client, the Vuforia functionality is also deactivated and some new components, which handle the visualization tasks are created. The client components can access the data from the variable sharing component and use it to select one of the visualizations and present it to the user.
The network management component in the center of figure 4.7, maintains the client-server connection. The component manager fulfills a very important role, although its connection to other components has not been visualized in the diagram. It is responsible for the transition between the calibration application and the main application, once the calibration is done.

Figure 4.8 represents a diagram, which aims to explain the data flow in the main application of the implemented system. As shown in the figure 4.8, the input data

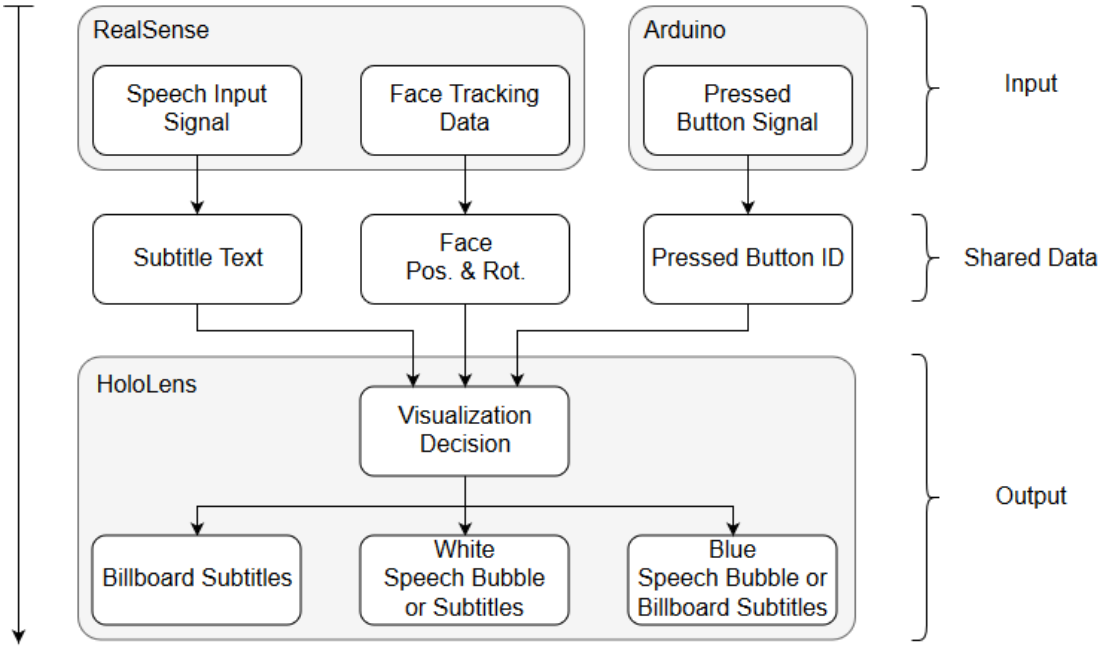


Figure 4.8: The diagram visualizes the data flow in the implemented system. The first row of the figure depicts all input devices as well as the data they provide to the server PC. The middle row shows the data transformed by the server, which is shared across the network, so that the client application can access it. The lower part of the diagram visualizes how the calculated data is used.

is provided by the Intel RealSense SR300 camera and the Arduino pushbuttons. The RealSense receives the speech signal as well as the facial landmarks and transmits them to the Unity application on the server.
Furthermore, the Arduino board serves as another input device, transmitting signals regarding the currently pressed pushbutton to the server PC. After having received its necessary data from various input devices, the server handles all necessary calculations. During these calculations the input data is transformed into the data needed by the visualizations. The calculated output data is then shared across the network, so the HoloLens client can access it. The client then uses the received data to handle the visualization switch. The resulting visualization is then presented to the user.

4.3 Visualization Design

One of the main goals of this master thesis focuses on the subtitle visualizations in real time. This section provides much detail and thorough explanations regarding the visualization design, as part of the system developed in the practical part of this master thesis.

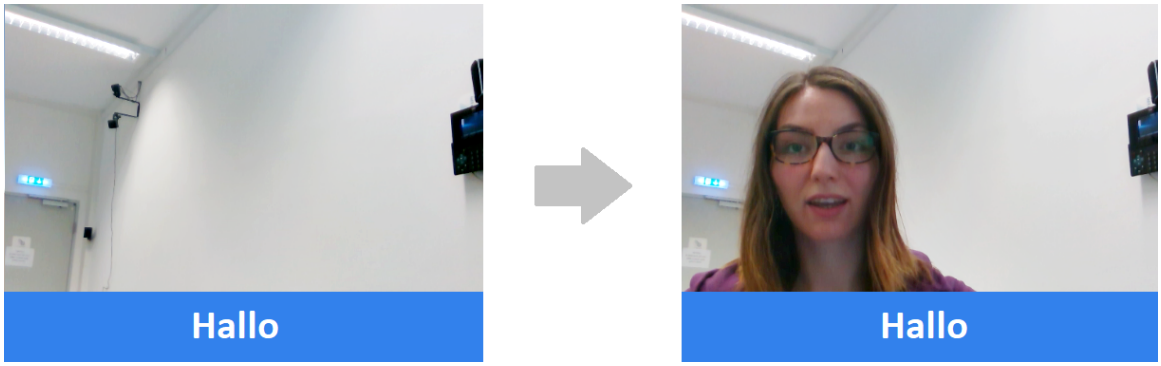


Figure 4.9: The figure shows the first visualization type: white subtitles on a blue billboard, regardless of the presence of the speaker inside the FOV.

First, the text visualization designs will be presented, followed by the sound localization and network connection visualization designs.

4.3.1 Text Visualization

As explained in the data flow diagram, the text strings, needed for the subtitles as well as for the speech bubble visualizations, are extracted using the speech recognition component of the server application. When new subtitles are available, they are stored in the variable sharing component. This enables the client's components to access the extracted subtitle text and visualize it in various ways. During the implementation of the system many visualization types were proposed and implemented, however in the end some of them were combined to form new ones, resulting in the following three visualizations.

Subtitles on Billboard

The first implemented visualization type consists of usual static subtitles, as Figure 4.9 shows. The subtitle text is placed at the bottom of the HoloLens FOV. The used font color is white, as also used by Microsoft for their HoloLens menus. The used font is called Univers_45_Light_Bold. The font style is set to normal. Line spacing is set to 1 and a central alignment of the text has been chosen. A fitting font size is chosen, which works well with the scaling. Furthermore, a blue billboard is used, similarly to the solution proposed by Debernardis et al.[DFG+14]. This is done in order to ensure a good contrast between text and background at all times. This is enabled using the billboard. This is especially important in the case of AR applications, where the real world is also visible through the display of the headset or glasses. This can result in changing lighting conditions and heavily structured background, which often cause readability issues, when a text with no billboard is used. Furthermore, the choice of colors, blue billboard and white text, were not only chosen because of the good contrast they provide, but also because they have a soothing effect.

Figure 4.10: The figure shows the second visualization type: normal subtitles in the absence of a speaker's face and emergence of a white speech bubble in the presence of a face.

Another design decision for this visualization was keeping the billboard on the display, empty, when no speaker is currently talking. This is intended to inform the user about the location of the subtitles, once a speaker starts talking. Since the FOV of the HoloLens is very limited, this appeared to be a good idea. When no billboard is used, even though the subtitles are placed at the lowest possible point visible to the user, they still do not appear where the user would expect them, namely even lower below. This is caused by the FOV of the HoloLens being much more limited than the FOV of the user's eyes. Therefore, the billboard is used and remains visible even in silent times. This not only gives the user a clue about where the text is going to appear when it does, but also the assurance, that nobody outside of his or her FOV is currently talking. Finally, the subtitle on billboard visualization remains unchanged at all times, as long as it is active, and regardless of whether the face of a speaker appears in front of the user or not.

White Speech Bubble and Regular Subtitles

The second implemented visualization is presented in Figure 4.10 and represents the first combined visualization. It is defined as a combined visualization, because it comprises two different visualization types. This text visualization switches from the first type to the second type, depending on the presence or absence of a speaker's face in the system's FOV. When no speaker is currently visible, but somebody is talking, regular subtitles are visible without any billboard. The font is also Universal_45_Light_Bold. The font style is normal. The font size depends on the scaling. The line spacing is 1 and the paragraph alignment is set to center. The font color of the text is still white, which is important in the absence of a billboard. This still provides good contrast with the background, in case of indoor real world environments most of the time. This is not the case for outdoor scenarios, where the background can suddenly become very light. However, the implemented system was intended for indoor use.

Figure 4.11: This figure shows the third implemented visualization type: billboard subtitles in the absence of a speaker's face and emergence of a blue speech bubble in the presence of a face.

Furthermore, the subtitles are also provided with a back outline. Thus, even in the case of a very light background color, the subtitles are still readable. A similar situation can be noticed in the left image presented in Figure 4.10.
The white subtitles with black outline are furthermore intended to provide the users with a certain sense of familiarity. The same type of subtitles are often used in movies or series on Netflix for example.

As soon as the face of the speaker appears in the FOV, this combined visualization switches to the speech bubble visualization type. The speech bubble is placed on the right side of the speaker's face, from the point of view of the user wearing the headset. The speech bubble is aiming to provide the user with excitement and make him feel, as if he or she is being part of a comic book adventure. The speech bubble is white with a black margin. It was manually designed using GIMP, a cross-platform image editing software. The font used for the text in the speech bubble is KOMIKASK and the font color is black. This is also meant to provide a good contrast and increase readability.

Blue Speech Bubble and Billboard Subtitles

The last implemented visualization is also a composite one, as illustrated in Figure 4.11. It combines two visualization types as well, depending on the presence of a speaker's face. However, it represents a combination of the first two already presented visualizations. In case a speaker is talking but not inside the system's FOV, the billboard subtitles are visible. The billboard subtitles were already explained during the presentation of the first visualization type. They are intended to inform the speaker about the exchanged words, even if the user is currently not looking at the speaker. This was intended to help deaf persons, who can only determine if somebody is talking by means of visual cues, such as moving lips, facial expression or hand gestures of the talking person.
However, all these visual cues require attention, concentration and eye contact.

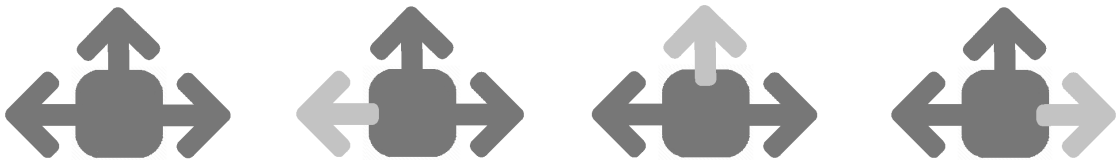


Figure 4.12: This figure shows the design of the implemented sound localization visualization. The left graphic shows, how the visualization looks, when nobody is currently speaking. One of the following three graphics presents the situation, when a speaker is talking. This informs the seated deaf user about the location of the current speaker.

Once the user turns towards the speaker and the speaker's face enters the system's FOV, the a speech bubble appears to the right of the speaker's face, as described in the previous visualization.
However, the speech bubble emerging this time is blue, while the presented text is white. This represents a further development of Debernardis et al.[DFG+14]'s solution. The speech bubble's color was set to the same shade of blue used for the billboard subtitles. All used fonts were acquired from Font Squirrel[Squ18] and are free for personal use.

Sound Localization

As previously mentioned, when the speaking person is outside of the FOV of a deaf person, all visual cues such as lip movement, facial expressions and hand gestures become useless. This is caused by the fact, that visual contact is not established in these situations. The implemented system aims to provide support for this specific target group. The provided help does not only include the subtitles, but also the information regarding the direction, where the speech is coming from. This especially targets the case, in which the speaker is outside of the FOV of the user.

The graphics used for the sound localization visualization can be observed in Figure 4.12. They were drawn manually using the GIMP image editing software. The square present in all icons represents the conference table, around which the speakers as well as the D/HH user are seated. The three arrows are aiming to inform the deaf user about the direction, where the speech is coming from.
When an arrow is highlighted, it suggests that the corresponding speaker is currently talking. When no arrow is highlighted, all speakers are currently silent. The choice of colors aims to provide a good contrast for the highlighted arrow. Meanwhile, the darker shade of gray was intended to blend in with the background to a certain amount, not attracting too much of the user's attention.

Network Connection Feedback

In order to improve the usability of the system, a further UI text was added to the Canvas. Its aim is to inform the user about the current status of the network, specifically if the server started effectively. At the same time, it informs about whether the client managed to connect to the server successfully or not. If the connection succeeds, the UI text is set to Connected. However, if the connection fails, the UI text is set to Disconnected. Either way, the user is provided with feedback.
The network status UI text is of color white, with no outline and is smaller than the subtitle text. It is placed in the upper left corner of the HoloLens' FOV, in order to not attract much of the user's attention.

CHAPTER 5

Implementation and Integration

This chapter goes into much detail regarding the implementation of the system, which was developed in the practical part of this master thesis. All implemented components are explained, as well as their integration into the entire system.

5.1 Software Setup

The implemented solution incorporates various software pieces and libraries, which needed to be embedded into it in the beginning. This section explains their setup and configuration.

RealSense Setup

In order to start implementing software applications using the Intel RealSense camera, one first needs to download the RSToolkit from the following source[Cor18a]. For the implemented application version 10 of the SDK was used. It can be found on the cited source by the following name: 2016 R2 Release Notes (10.0.26.0396). The Intel RealSense Depth Camera Manager (DCM) driver also needs to be downloaded and installed. This enables the use of the RealSense camera.
In order to use the RealSense with Unity, a new project needs to be created, in which the RealSense Unity package is imported. After the RSToolkit has been integrated into the project, one can use several out-of-the-box prefabs provided by the toolkit, such as the SenseAR prefab. This prefab depicts the RealSense device as a game object in the Unity scene. It contains merely two components. The first one is the Main Camera, which represents the actual RealSense camera. The second one is the Real Camera, which handles the video feed received as input from the device.

HoloLens Setup

The Mixed Reality Toolkit for Unity .zip file can be downloaded from Github[20118e]. After opening the contained project in Unity, one can notice three MRToolkit related subfolders in the Assets folder. The first one, the HoloToolkit-Examples, contains a few example scenes, which demonstrate various functionalities, when ran on the HoloLens. The other two folders are named HoloToolkit-UnitTests and HoloToolkit. The latter contains many useful components, which are necessary, when developing for the HoloLens.
Some of the most important HoloToolkit prefabs, which have been used during the implementation, are listed below:

- the MixedRealityCameraParent - contains the MixedRealityCamera prefab. These two combined contain various child game objects and scripts, which enable the use of the HoloLens camera.
- the InputManager and the DefaultCursor - both are necessary for a successful build targeting the HoloLens. Within the DefaultCursor's child game objects certain Skinned Mesh Renderers have been deactivated, since the implemented system does not require a visible cursor.

All three mentioned prefabs were gathered underneath a common parent, an empty game object named HololensObj. The HololensObj, and especially the MixedRealityCamera game object within, contains some more important game objects. However, these are explained in the coming sections. The HololensObj was saved as a new prefab and then deleted from the scene.

Some other aspects, which need to be handled when developing for HoloLens, are some specific settings. These need to be adjusted in the Player Settings. In the Windows tab of the Player Settings, in the Other Settings, the Scripting Backend needs to be set to .NET instead of IL2CPP.
The Publishing Settings also require adjustment. A few more capabilities need to be added to the application targetting the HoloLens. Therefore, under Capabilities: InternetClient, InternetClientServer, PrivateNetworkClientServer and SpatialPerception need to be enabled. These play an important role for the communication between the HoloLens client and the server.
Finally, the XR Settings need to be adjusted too, by checking the Virtual Reality Supported checkbox.

Arduino Setup

In order to develop applications, which involve programming the UNO R3 Controller Board, the Arduino Software IDE is needed. The offline desktop IDE can be purchased from the following source[Ard18c]. After installing the IDE, a programming language similar to C++ can be used to write code.

Vuforia Setup

In the beginning of the implementation Unity 2017.2.0f3(64-bit) was used and the Vuforia SDK was downloaded separately and planned to be integrated additionally, as the RSToolkit and the HoloToolkit were. However the importing of the external Vuforia SDK caused numerous problems, existing components did not function anymore and the build was invalidated by a long list of error messages. Finally, in order to facilitate the integration of the Vuforia SDK libraries into the application, Unity 2017.3.1f1 Personal64*bit* was used, because Vuforia is already integrated into it. One simply needs to enable the Vufora functionality by going to Edit/Project Setting/Player/Windows tab/XR Settings and checking the Vuforia Augmented Reality checkbox.

This enables the use of the game objects provided by Vuforia. For example, the server application uses the AR Camera prefab provided by Vuforia. This represents the Vuforia camera object, which handles the marker tracking. The Vuforia settings on the AR Camera game object also had to be adjusted. The most important script, which comes with the object, is the VuforiaBehaviour.cs script. When selecting the game object, one can notice a large button called Open Vuforia configuration in the Inspector. Once this button is pressed, the editing of Vuforia's configuration settings is enabled.
The first thing one needs to handle is the setting of the App License Key. The key can be purchased from the Vuforia Developer Portal[Inc18b]. After the key is created, it can be copied and pasted into the Unity application, in the App License Key input field of the Vuforia Configuration.
As a next step the Digital Eyewear needs to be set. An important realization during the implementation of the system was, that there can only be one Vuforia Configuration for an application. This means, that all game objects requiring it within the same application, no matter if they are server or client components, need to access the same configuration. Since the objects, transformed after the calibration step, need to be visualized on the HoloLens, the settings corresponding to this device are prioritized. Therefore, the Device Type is set to Digital Eyewear and the Device Config to HoloLens. If this is not set correctly, the build and deploy on the HoloLens fails.
The next thing which needs to be configured are the Databases. Vuforia enables its users to choose an arbitrary image as a marker, which is then tracked. In the case of this implementation an image target provided by Vuforia was chosen. The image is depicting stones and can be visualized in Figure 5.1.

The chosen image target should be printed using either Actual size or Scale: 100%. Then, a new database can be created on the Vuforia Developer Portal using the chosen image target. For the database creation process, the width, chosen for the image target, can have a great impact on the quality and result of the marker tracking. One should keep in mind, that Unity uses meters as its default unit. For this implementation, the actual width of the printed image target on an A4 sheet of paper was measured in millimeters using a ruler. That exact measured width, converted into meters, was then input as the Width of the Target during definition.

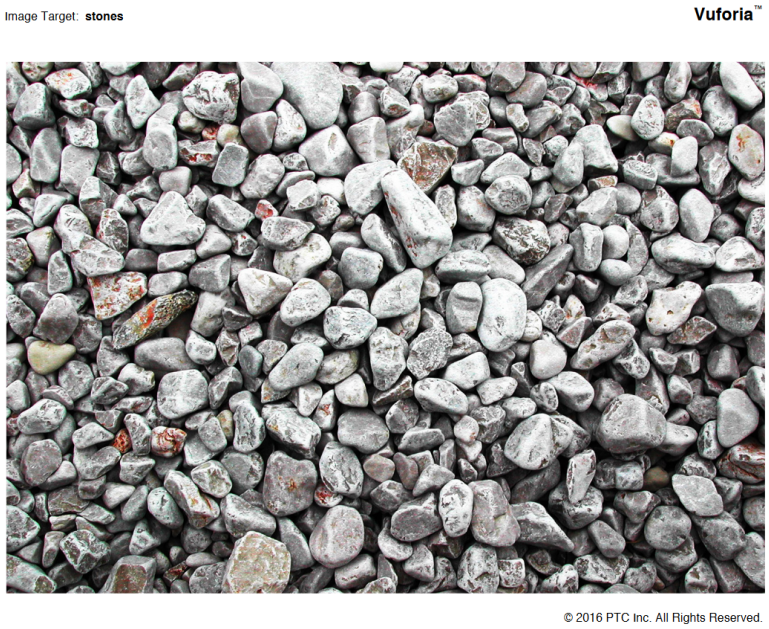


Figure 5.1: The figure represents the image target used as marker for the calibration implementation using Vuforia. The image visualizes various stones. This provides numerous features, which are useful for the Vuforia tracking algorithm. This image target, as well as other ones, can be downloaded from the following source[Inc18d].

After the target definition is completed, the database can be downloaded for the Unity Editor. After downloading the database package, it can be imported into Unity. The newly created database, containing the new marker, should be listed beneath the Databases section of the Vuforia Configuration file. The two checkboxes corresponding to the created database can be checked. One checkbox loads the database, while the other one activates it. The remaining checkboxes from other databases can be unchecked.
The last step of configuring Vuforia is represented by the checking of the webcam. The Intel RealSense RGB should be selected as camera device on the server.
As a next step the image target game object was attached to the AR Camera prefab, by right clicking on the AR Camera object and selecting Vuforia/Image from the visualized context menu. The image target object contains a component called ImageTargetBehaviour.cs script, where a database needs to be selected as well. For the implementation of the system, the StoneMarker database was chosen, which was previously created and activated in the Vuforia Configuration file. Then, the corresponding image target with the correct width should be chosen for the Image Target field.
In order to enable a better visualization of the correctness of the calibration, another object can added as a child of the image target. The chosen object during the implementation was a cylinder. The cylinder's role was to provide a better visualization regarding where the marker was perceived on the server, as well as on the client application. The cylinder attached to the image target on the server received a red shaded material.

After attaching the red cylinder, the VuforiaServer_ARCamera object is ready for use and was saved as a prefab. After that is was deleted from the scene, since the SetupNetwork game object's script instantiates the prefab directly at the start, but only in case the application is started as server.

For the client application a similar approach was tried at first and a VuforiaClient_ARCamera prefab was created, however this did not work in combination with the HoloLens camera. Thus, another solution had to be found. The developed solution involved the HoloLens camera remaining the only used camera on the client side and the Vuforia tracking capabilities were enabled as follows: the VuforiaBehaviour.cs script was added directly to the MixedRealityCamera on the HololensObj prefab, while the Vuforia settings remained the same as already mentioned above.
However, it was still required, that an image target object was added to the HoloLens camera and the correct database and image target had to be set in the ImageTarget-Behaviour.cs script as well. Lastly, a cylinder was added to the image target of the MixedRealityCamera too and given a blue shaded material, so that the tracked marker could also be visualized better on the client application.
Furthermore, a script was added to the cylinder, which fetches the previously mentioned MarkerClientCube object and sets the position and rotation of the MarkerClientCube to the current position of the blue cylinder, which is tracking the Vuforia marker image on the client. The MarkerClientCube is shared across the network and is provided with client authority for this exact use case, of the client application being able to transmit the current position and rotation of the tracked marker to the server.
The HololenObj prefab was then updated and removed from the scene again.

5.2 Software Architecture Implementation

After the Software Architecture Design section in the System Design chapter provided an overview of the implemented system, the following section provides detailed explanations regarding implementation. First, the component management solution as part of the implemented software architecture is presented. Then, the network implementation, as well as object synchronization solutions are explained. Lastly, the calibration application implementation, as well as the implementation of the main application are presented.

After various approaches for the component management have been tried, the final and working solution has been selected. The solution involves starting both the server and the client application from the same Unity application, but with different network setups. However, the game objects of the one application always seemed to throw errors in the other and vice versa. They could not be in the same scene at the same time. Thus, some changes were needed in order to solve the problem of the server or client game objects interfering with the game objects of the other.

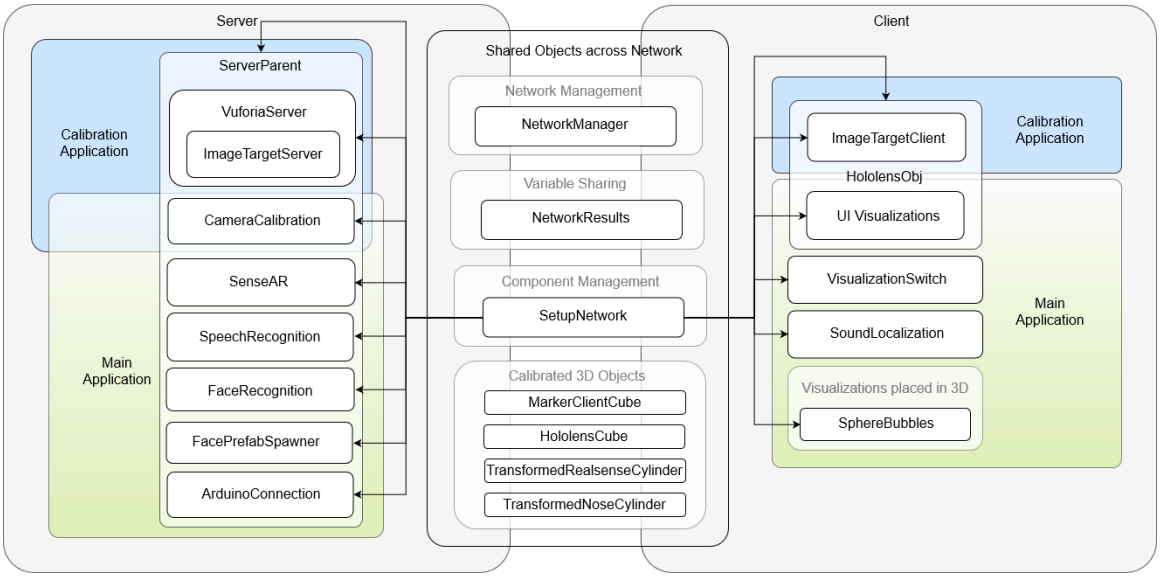


Figure 5.2: The diagram shows all game objects, which are linked and managed by the SetupNetwork game object.

The solution involved leaving only game objects, which aimed to be shared across the network, such as Network, NetworkResults, MarkerClientCube and HololensCube, in the scene at the start of the application. Meanwhile, all other game objects, which were only relevant for the server or the client, were saved as prefabs and removed from the scene. Additionally, another game object, which was also allowed to remain in the scene, was created and named SetupNetwork. While both the server and client need access to this object, it was not of importance, that they were accessing the same instance. Thus, the SetupNetwork game object did not need to be actually shared across the network. It was sufficient for both the server and the client to have an individual instance of the entity. This game object received a single script as a component, which plays a crucial role.
The script inherits from the NetworkBehaviour parent class, in order to be able to access the server or client of the network directly. The SetupNetwork game object plays the role of a linker. It either gets or searches for references to all game objects, which are needed in the scenes of both the server and the client. It only instantiates the game objects needed for the specific case. Thus, the one application is not disturbed by unneeded game objects of the other. Figure 5.2 visualizes how SetupNetwork interacts with and manages all other game objects.

All objects on the left side of Figure 5.2 are saved as prefabs and instantiated, when the application is started as server. All entities on the right side of Figure 5.2 represent the game objects instantiated, when the application is started as a client. The object comprised by the light blue boxes, are active in the calibration application. Meanwhile, the object comprised by the light green boxes, are active during the main application. The

SetupNetwork object handles the transition between calibration and main application. The objects above and below the SetupNetwork entity represent game objects, which are already present in the scene for both the server and the client application. The ones above SetupNetwork take care of various networking aspects. Meanwhile, the objects below SetupNetwork represent shared game objects across the network. The client has authority over the first two entities of the ones below. The lowest two are explained in the calibration section. The boxes with transparent name are not actual game objects in the scene. They are just meant to provide further explanations for a better understanding of the diagram in Figure 5.2.

The fact, that two cameras are never allowed in the same Unity scene, became clear rather early. Thus, solutions were brainstormed, in order to enable the VuforiaServer_ARCamera (calibration application) to coexist with the SenseAR camera (main application) as part of the same project. The working solution involved splitting the instantiations taking place in SetupNetwork in two phases:

1. the calibration phase
2. the main functionality phase, which would be initiated as soon as the calibration phase ended.

The solution developed in order to enable the seamless transition from the calibration phase to the main application phase on the server involves the following actions: first, the ServerParent prefab is instantiated in the Start() method of SetupNetwork. The ServerParent contains a child game object named Server, which is aimes to transforms all of its children by means of the calculated offset position and rotation. Next the instantiation of the VuforiaServer_ARCamera prefab follows as well as its attaching to the already existing child of ServerParent, Server. Then the CameraCalibration and the ArduinoConnection are instantiated and attached to Server too, since they do not require any information from the RealSense.
Meanwhile on the client a ClientParent game object is created which contains all following instantiated game objects necessary to the client application. Next, the MixedReality-CameraParent prefab is instantiated, which also contains the VuforiaBehaviour script on the MixedRealityCamera child and all other Vuforia components needed for the calibration step. Then, the VisualizationSwitch and the SoundLocalization prefabs are instantiated, however they are not used yet. They are used in the main application phase. Both of these objects are described in much details in the Visualization implementation section of this chapter.

After the initialization of the calibration phase and as soon as the image target is visible to both cameras of server and client and the Space key is pressed, the calibration process is triggered. This means, that all necessary data is collected and the offset between RealSense and HoloLens is calculated.

However, after finalizing all calculations, some changes in the structure of both applications take place once more. On the server the VuforiaServer_ARCamera game object is deactivated, datasets belonging to the Vuforia tracking are deactivated, destroyed and the Vuforia TrackerManager is deinitialized. Instead of the Vuforia camera, the SenseAR camera game object is instantiated and attached to the ServerParent's child, Server. Then follows the instantiation of all RealSense SDK related components, namely the FaceRecognition prefab, the FacePrefab_Spawner, the SpeechRecognition and the Speech prefab. After all of this is accomplished, the server is marked as complete.
Meanwhile on the client side, in order to shut down the Vuforia functionality, merely the VuforiaBehaviour script attached to the MixedRealityCamera child of the HololensObj needs to be deactivated. A new prefab is instantiated, the TestPosition game object, which proceeds to be attached to the ClientParent and will be explained in the visualization implementation section. Next, before the NetworkResults can be informed, that the client is complete, the deactivation of all 3D calibration visualization helpers takes place. This means, that the cylinders which were used to check the correctness of the calibration phase can are now disabled. This concludes the initialization of the implemented system's second and main phase.

The SetupNetwork game object instantiates all mentioned Prefabs during runtime. In order to be found and successfully instantiated or retrieved by SetupNetwork, all game objects were provided with unique tags. Besides all the mentioned references to the numerous game objects, SetupNetwork's script also contains an important variable named isClient, which stores whether the currently running application is the client or not. The mentioned variable is set to true, only when the Unity application is compiled and ran for the Windows Universal Platform, which represents the client.
This variable represents the flag, telling SetupNetwork, which game objects need to be instantiated: the ones for the server or the ones of the client. The instantiations take place in the Start() method of SetupNetwork's script, so that when the network connection between server and client is set up, all necessary game objects are already present in the corresponding scenes. In the Update() method further queries of the isClient boolean variable take place, as visualized in Figure 5.3.
The code segment only runs the first time the Update() method is executed. It plays a crucial role, since it sets up the connection between client to server by specifying the correct IP address and port for the client. The client is then started, followed by the server.

5.2.1 Network Implementation

In order to establish the communication between the Intel RealSense camera and the Microsoft HoloLens headset, the setup of a network was necessary. First, an empty game object named Network was created. Then, the NetworkManager.cs script provided by Unity was added to this game object.
All settings were left as they were, except the Player Prefab field. Since the network in

```
if is_client && is_at_startup {
    set network address;
    set network port;
    start the client;
    is_at_startup = false;
}
if !is_client && is_at_startup {
    reset the network server;
    start the server;
    is_at_startup = false;
}
```

Figure 5.3: The code segment shows what happens at the application start up, including the launch of the server or the client.

Unity requires a player object, one was created and important network components were attached to it. The first attached component is NetworkIdentity, also provided by Unity. This component is mandatory for all objects, which need to be shared across the network. None of the checkboxes provided by the script needs to be checked, as both server an client should notice the presence of the player prefab game object in the scene. However, it does not need to be controlled by either one.
The next network component attached to the Player Prefab is NetworkTransform. This component is usually needed, when other complex aspects of the game object need to be synchronized over the network as well, such as its Transform, its Rigidbody or its Character Controller. However, this was not the case for this object, so the settings were left as they were.

The Player Prefab was then attached to the NetworkManager Script's Player Prefab field. Lastly, the Player Prefab was also added to the Registered Spawnable Prefab field. Besides the NetworkManager, an additional network component was added to the Network game object, namely the Network Manager HUD script. This one is also provided by Unity. It enables the visualization of a small network user interface in the corner of the game view in the Unity editor, as soon as the application is run. This user interface is only visible on the server. It informs the user of whether the application is running as server or as client. Furthermore, the IP address and port can be chosen. The HoloLens does not visualize this HUD.
The Network game object does not need to be saved as a prefab. It can remain in the scene, as a regular game object. It is used by both the server and the client application.

Network Results

The second important network component is the NetworkResults game object. For its creation an empty game object with the same name was created. Two further components were attached to it. The first necessary component is the NetworkIdentity script, provided by Unity.

This one is needed, because the NetworkResults game object is aiming to be shared across the network. This means, that both server and client have access to it. Thus, no special authority of client or server is chosen. The second attached component to the NetworkResults game object is a script named NetworkResults_Script.cs. When creating a new Unity C# script, it is usually inheriting all methods from the MonoBehaviour class. However, since this object and its attached script aims to be shared over the network, the class needs to be inheriting from NetworkBehaviour instead. In this case, the UnityEngine.Networking library needs to be imported as well.

The purpose of the NetworkResults game object and script is, that it contains certain variables, which also need to be available for both server and client. Thus, certain components of the system can place variables in the NetworkResults script, while other components can access these variables at a later time. In order to enable the sharing of simple data type variables across the network, SyncVar is used. SyncVar is used in this script to synchronize string, int as well as bool variables between the server and the client.

Among the most important synchronized variables, there is stringText, which is set by the Speech game object's TextOnBaloon.cs script. It represents the currently recognized speech segment. It is set in the NetworkResults_Script of the NetworkResults object, which is shared across the network. Thus, the HoloLens client can access and visualize it for the D/HH user.
The next shared variables are numFacesDetected and currentFaceID. Both are set in the FaceRecognition game object's FaceRecognition_Script.cs. The first one is provided by the Intel RealSense face detection functionality. It informs about the number of human faces currently identified in the FOV of the RealSense camera. It is also used later on for the visualization calculations on the client. The second one, currentFaceID, is provided by the RealSense FaceRecognition, after having registered at least one face.
The pressedButton variable is set by the ArduinoConnection game object's ArduinoConnection_Script.cs. It informs other components if and which pushbutton is currently pressed by the speaker. This also aids the visualization implementation. The string variable activeVisualization informs about the currently selected type of visualization.
The last two shared variables are useful for the calibration part, which is explained in great detail further below.

The NetworkResults game object provides no other additional functionality, besides representing a centralized location, where variables to be shared across the network can be placed and accessed from. Therefore, is plays a crucial role for the implemented system. The NetworkResults object also does not need to be saved as a prefab, it can remain as regular object in the scene and both the server and client application will use it from there.

Checking the Network Connection

As mentioned before the connection between the server and the HoloLens client is only possible, if they are both in the same network. In order to enable that, a router was used, which provides a fixed IP address. One can check if the server is in the correct network by opening a console window and inserting the command ipconfig.
This presents all information related to the network the machine is currently connected to. If the resulting IP address corresponds to the one written on the router, the server is successfully connected.
While the connection with the server can be checked as described, the connection with the HoloLens is more problematic to check. The HoloLens can not be pinged. This means, that the ping from the PC console can not reach the HoloLens. However, this does not necessarily mean, that the HoloLens is not connected to the network. Another way needed to be found in order to verify the connection between server an client. The solution involves starting the application once as server and once as client. If the Player Prefab object is instantiated correctly and visible on both server and client, the connection is successful.

Synchronized Objects

Besides sharing simple variables over the network as explained above, UNET, Unity's client/server architecture solution for networking, also provides the possibility to share whole game objects across the network. In order to do so, one needs to attach both a NetworkIdentity component, without checking any of the two checkboxes, and a NetworkTransform component to it. This suffices, in the case in which their position is defined by the server. This approach was used for two of the game objects during the implementation, namely TransformedRealsenseCylinder and TransformedNoseCylinder. Their role is explained in much detail in the Calibration section.
As mentioned, if the position of the shared game objects is set by the server, the procedure is easier. However, another change has to be made, in order to enable the successful sharing of game objects across the network. They also need to be saved as prefabs and added to the list of Registered Spawnable Prefabs. The list can be found in the NetworkManager script of the previously described Network game object.

Client Authority

The implementation contains two additional game objects, which are shared over the network. They contain the same two NetworkIdentity and NetworkTransform scripts. Furthermore, they also need to be saved as prefabs and need to be registered as Spawnable Prefabs in the NetworkManager script of the Network object. These game objects are called MarkerClientCube and HololensCube. However, the main difference is, that the position and rotation of these game objects is set by the client application. This requires a few additional steps.
First of all, in the NetworkIdentity script of each of the two game objects, the Local Player Authority checkbox needs to be checked.

```
CmdMarkerCube {
  fetch marker client cube network identity;
  assign authority over marker client cube network identity
    to the client;
}

CmdHololensCube {
  fetch hololens cube network identity;
  assign authority over hololens cube network identity
    to the client;
}
```

Figure 5.4: The code segment shows how client authority can be assigned to certain game objects.

However, this does not suffice. In order to truly enable the client to set the pose of these game objects, a new script has been added to the Player Prefab object. The newly added script is called AssignClientAuthority_Script and inherits from NetworkBehaviour. Then, in the Start() method, the two game objects are fetched using GameObject.Find().

In the Update() method, the isLocalPlayer variable provided by NetworkBehaviour is queried. Thus, the following code is only executed on the client application. The code in Figure 5.4 calls two command methods, which are explained next. There is a method for each of the two game objects, whose pose needs to be defined by the client.
The two mentioned methods each receive the according game object as an input parameter. Both require the Command attribute. This attribute is can be set on methods belonging to NetworkBehaviour classes. It enables their invocation on the server, when the client sends the command, as specified by the official Unity documentation[Tec18a].

Thus, the NetworkIdentity component of each game object is fetched and the client authority is assigned manually. The client then has the capability of also moving game objects on the server. This is required for example in the case of the HololensCube game object, which depicts the location and rotation of the HoloLens device on the server. The HololensCube is a simple game object with the described network components attached to it. However, its position and rotation are set each frame to the current position and rotation of the HoloLens camera. The SetHololensPose_ClientScript is attached to the MixedRealityCamera game object, in the HololensObj on the client.
Thus, the server can see and use the position of the HoloLens headset for further calculations, as explained in the Calibration section.

Firewall

Another thing, which caused issues during the implementation of the system and especially during the network connection setup phase, was the firewall. The network connection

was only established successfully, after the firewall was deactivated. However, the firewall does not need to be deactivated completely, leaving the computer at risk. It can merely be deactivated for the specific Unity version, which is used. The finally used Unity version for the system's implementation was Unity 2017.3.1f1 Personal (64bit).

5.2.2 Calibration Implementation

Before the insertion of a calibration step, the position of the nose from the RealSense 3D face tracking was sent over the network using the NetworkResults game object. The client application then fetched the nose position from NetworkResults and tried to visualize it in HoloLens space by means of a sphere. However, the sphere was either nowhere to be found or very hard to find considering the limited FOV of the HoloLens. As the location of the sphere was arbitrary, it became clear, that a calibration step was mandatory.
For the implementation of the calibration step Vuforia SDK was used, because of its marker tracking capabilities targetting AR applications.
After having thoroughly described the setup and configuration of Vuforia and all necessary prefabs for the calibration in the Vuforia Setup section, the actual implementation can begin. At this point, when starting the server and the client with their enabled Vuforia Behaviour components, both the cameras of server and client were tracking the marker, once it entered the FOV of both devices. The two cylinders, which visualize the the marker in the virtual space, were visible too. The red cylinder shows where the marker is identified on the server, while the blue cylinder visualizes where the marker is spotted by the client.
Vuforia however, does not provide the actual calibration implementation. It merely provides the implemented system with the necessary tracking tools, which enable an implementation of a calibration step. The two cylinders visualizing the same market were still rather far apart, underlining the necessity of a calibration step.
Another aspect to take into consideration, when using Vuforia next to other cameras, is the fact that it takes control over all cameras at runtime. It does so by automatically attaching the VuforiaBehaviour script to all other cameras and interfering with their intended functionality. In order to fix this, the VuforiaBehaviour script needs to be attached to the affected camera objects manually and deactivated beforehand. This allows for their intended functionality to take place during runtime again.

Synchronized Camera Movement

As mentioned, the HoloLens moves according to the head movements of the user. The RealSense camera, being attached to the HoloLens, moves together with the head in real world too.
This meant, that this real world movement had to be recreated in the application as well. However, as previously mentioned the devices each have their own coordinate system and their own behavior in relation to it. The most important discovery was, that the RealSense's coordinates remained at the initialization point. Even when moved around the real world, the RealSense acts as its coordinate system's origin.

Meanwhile, the HoloLens moves around the virtual space, according to the real world movements of the head it is placed on. The developed solution for this issue represents moving the RealSense camera by means of the code too according to the HoloLens. This was implemented by the insertion of a new, initially empty, game object prefab called ServerParent.
The ServerParent prefab is also referenced in the SetupNetwork game object and represents the first prefab to be instantiated, when running the application as server. It is the parent of another game object called Server, who's aim is to contain all other game objects further instantiated, which are needed by the server application. They are both needed. ServerParent manages the following of the HoloLens' position and rotation, while the Server child applies the calculated offset, which will be explained below, to all of its child game objects. Further on, in the Update() method of the SetupNetworkScript, the ServerParent's position and rotation are always set to the position and rotation of the HoloLens. This is accomplished by means of the HololensCube game object, which is shared across the network. This enabled the moving of all server components all at once. However, the most important server components, which need to be moved according to the HoloLens, are the SenseAR and the VuforiaServer_ARCamera game object. They both represent the RealSense camera at different moments in time.

As already mentioned, the VuforiaServer_ARCamera's child game object ImageSererTarget has a further child game object, namely the red RealSense cylinder. This cylinder plays the role of visualizing where the image marker is tracked by the RealSense camera on the server. However, since the correct results need to be displayed on the HoloLens, and in order to be able to examine the correctness of the calibration implementation, the tracked image marker's cylinder on the server needed to be sent over the network to the client. In order to do so, a new game objects with the aim of being shared across the network was created and named TransformedRealsenseCylinder. This new game object is a cylinder shaded with a green material, is controlled by the server and has the the role of sending the position and rotation of the red RealSenseCylinder, the image tracking visualization on the RealSense, to the client. After the position and rotation of the RealSense cylinder have been successfully sent to the client by means of the green cylinder named TransformedRealsenseCylinder, both Vuforia Image Tracking results, visualized by two cylinders, can be seen on the client, when the image marker is in the FOV of both cameras.

Offset Calculation

After having managed to move the RealSense according to the HoloLens in the virtual space too, and having sent the RealSense cylinder position and rotation to the client application, an additional thing was needed: the geometrical calculation of the offset between the two cylinders. The calibration would only be successful if a transformation was found, which transformed the RealSense cylinder onto the HoloLens cylinder.

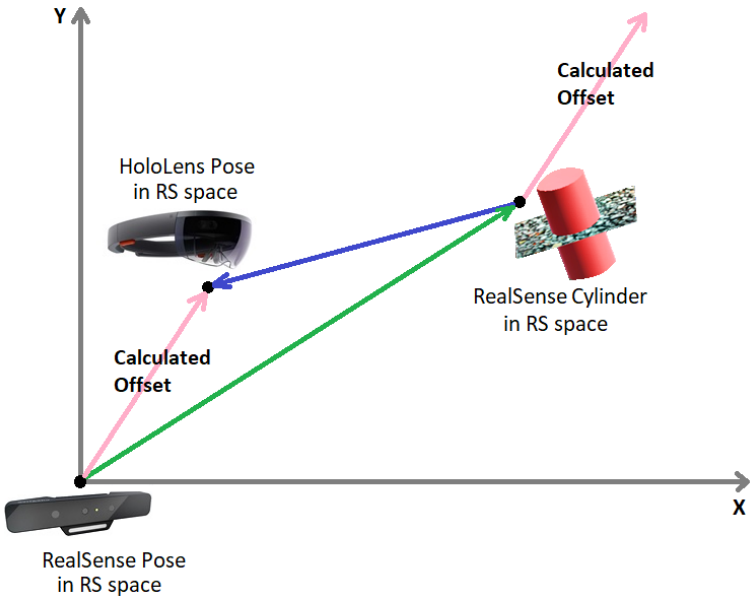


Figure 5.5: The figure provides a graphical representation of the executed calculations in the calibrationCalculation() method, in order to calculate the offset between the RealSense and the HoloLens cameras.

While the correctness of the calibration implementation can be evaluated by checking whether the two cylinders overlap on the client application, the offset calculation still needed to be handled on the server. This is the case because the software architecture requires the server to take care of all calculations and merely send the result over to the client.
For the second part of the calibration implementation, the offset calculation, a new empty game object was created and named CameraCalibration. Then, a new script was created and attached to the CameraCalibration object. Since the offset calculation is taking place on the server, the CameraCalibration script needs to gathers all needed game objects in RealSense space first. The necessary game objects for the calculation were:

- the HololensCube - representing the position and rotation of the HoloLens camera on the server
- the VuforiaServer_ARCamera - representing the position and rotation of the RealSense camera on the server
- the RealsenseCylinder - the red cylinder, visualizing the position and rotation of the tracked stone image marker by the VuforiaServer_ARCamera on the server.

In the Update() method of the CameraCalibration script the check occurs of whether the Space key has been pressed on the keyboard. The pressing of the Space key triggers

```
calibration_calculation {
  RS_coordinateSystem:
    RS_to_RSMarker_Transl =  RSMarker_Pos - RS_Pos;
    RS_to_RSMarker_Rot =  RSMarker_Rot * Inv(RS_Rot);

    RSMarker_to_HL_Transl =  HL_Pos - RSMarker_Pos;
    RSMarker_to_HL_Rot =  HL_Rot * Inv(RSMarker_Rot);

    HL_offset_Transl = RS_to_RSMarker_Transl + RSMarker_to_HL_Transl;
    HL_offset_Rot = RS_to_RSMarker_Rot * RSMarker_to_HL_Rot;

  return(HL_offset_Transl, HL_offset_Rot);
}
```

Figure 5.6: The code segment shows the necessary calculations executed in the calibrationCalculation() method. The result of this method is represented by the offset of the HoloLens in RealSense space. In order to keep the code short, abbreviations were used: RS stands for RealSense and HL stands for HoloLens.

the offset calculation process. The offset calculation takes place one single time, in the beginning, after the application has been started. Figure 5.5 visualizes all gathered game objects and necessary vectors for the offset calculation. Furthermore, the visibility of the entire image target is required in the FOV of both the RealSense and the HoloLens cameras.

The first thing that happens after the Space bar is pressed is, that all positions vectors and rotation quaternions are extracted from the three fetched game objects mentioned above. All of this gathered data is then sent as input parameters of the implemented calibrationCalculation() method, where the geometrical offset calculation takes place. The code segment in Figure 5.6 illustrates the calibrationCalculation() method.
In the calibrationCaluculation() method two vectors are calculated in the beginning:

- the vector pointing from the RealSense to the RealSenseCylinder, visualized as the green vector in Figure number 5.5
- the vector pointing from the RealsenseCylinder to the HoloLens in RealSense space, visualized as the blue vector in Figure 5.5

These vectors can be calculated using basic vector subtraction. For the calculation of the rotations from RealSense to RealsenseCylinder and from the RealsenseCylinder to the HoloLens, the calculations were a bit more complicated, since quaternions were involved. In order to calculate the addition of two quaternions, their multiplication is required. For the calculation of their difference, the multiplication of the rotation of the vector's starting point with the inverse of the rotation of the vector's terminal point has to be carried out.

Once these two vectors and these two quaternions have been calculated, the calculation of the offset vector, visualized as the pink vector in Figure 5.5, is possible by their addition, as dictated by the parallelogram law. Since the addition of quaternions is also required in order to extract the offset rotation, the previously calculated quaternions can be multiplied.

After having successfully calculated the offset translation and rotation between the HoloLens and the RealSense, the NetworkResults game object is informed. First, the offset translation and rotation are saved in the NetworkResults and then the boolean variable calibrationDone is set to true.
After having figured out the offset, it can be applied to various game objects in the form of subtraction, in order to correct their position and rotation in HoloLens space. The transformation is also applied to the TransformedRealsenseCylinder by subtracting the calculated offset from the RealsenceCylinder's position and rotation. This should cause the two cylinders to overlap in HoloLens space, which concludes the calibration step.

5.2.3 Main Application Implementation

As soon as the calibration part is done, the component management performed by the SetupNetwork game object sets the scene up for the main application. This means, that all necessary game objects for the main application phase are instantiated. Meanwhile, the calibration phase objects, which are not needed anymore, are deactivated.
This section goes into much detail regarding some of the main features of the implemented system: the speech recognition, face recognition, face tracking and Arduino implementation.

Speech Recognition

The implementation of speech recognition is split into two parts. First, the perception of the speech by means of the RealSense device is handled. Then, the processing of the registered speech takes place. For the speech perception a game object named SpeechRecognition was created and a script called SenseInputRecognition.cs was attached to it.
In the initialization phase of the script various aspects regarding the RealSense device are specified, such as the device's name and the Recognition Type. As recognition type one can choose between Dictation and Command Control. Dictation can be used, when speech to text is needed. This means, that all spoken words of a user are registered and turned into text. Whereas, Command Control can be used when the application needs to react to certain commands defined by the user. Dictation was chosen in the case of this master thesis.
The RSToolkit provides a very useful entity called PXCMSpeechRecognition, which handles the speech recognition. PXCMSpeechRecognition also provides the users with the possibility of selecting a preferred Language Type.

During the conception phase of the master thesis, English was the agreed upon language. However, as the development proceeded and actual D/HH users were imagined, the Language Type was changed to German. This change was made, because German much rather fits the target group and facilitates their usage of the system.
For the processing part of the registered speech another empty game object, called Speech, was created. A script called TextOnBaloon.cs was attached to this one. This script contains a sentence queue object, which receives data from the SenseInputRecognition. As soon as the queue contains sentences, the script casts them to a String form, which makes them available for further use.

The speech recognition provided by the Intel RealSense also has some limitations. For example the fact, that it can not locate the direction from which the speech is coming. Furthermore, it cannot distinguish between speakers. Which means, that if more than one person were speaking at the same time, their sentences would get mixed up by the speech recognition.
However, this is a problem with which many current speech recognition systems are confronted. A solution for this would be the additional integration of voice recognition and sound localization algorithms. However, since the speech recognition part did not represent the main focus of this master thesis, the speech recognition provided by the Intel RealSense did suffice.
Finally, the SpeechRecognition and Speech game objects were both saved as Prefabs and then deleted from the scene.

Face Recognition

For the implementation of the face recognition a new game object, named FaceRecognition, was created. A script called FaceRecognition_Script.cs was attached to this object. Two of the most important game objects, which are initialized in the Start() method of the script, are the FacePrefabSpawner and NetworkResults, which was explained in the network implementation section. The Start() method furthermore handles the configuration of the RealSense. Some objects needed for the configuration are enumerated below:

- the PXCMSenseManager - used to enable the color stream received as input from the RealSense. Among the parameters needed for the creation of this object, one needs to set the frames per second (FPS). Intel recommends using 30 FPS for FaceRecognition according to the following official source[Cor18c].
- the PXCMFaceModule - is necessary for the PXCMFaceConfiguration
- the TrackingTypeMode - is part of the faceConfig. During the implementation it was set to both color and depth.
- the the PXCMFaceData - can be extracted from the faceModule. It is used in every frame to query, if any faces are identified.

- the recognition database - is either created or loaded, depending on whether it already exists.
- the enabling of the facial recognition - called using PXCMFaceConfiguration. The result is stored into an PXCMFaceConfiguration.RecognitionConfiguration object.
- the PXCMFaceConfiguration.RecognitionConfiguration - allows for the setting of a RecognitionRegistrationMode. One can choose between Continuous or OnDemand registration. For the implemented system continuous mode was selected.

The Update() method contains an implementation destined for the recognition of a single face. In order to do this, the current frame is extracted by means of the senseManager. The faceData is updated and informs about the presence of a user in the received feed. In case there is a face in the FOV of the RealSense camera, the faceData object is used in order to retrieve the information related to the face. This information is used in order to retrieve the PXCMFaceData.RecognitionData.
Then, using the recognitionData extracted from the current face, the system checks whether the face has already been registered or not. In case the face is identified as already registered, the user ID of the face is extracted. In case the face is not identified in the existent database, the user is registered and the FacePrefabSpawner object is informed about it.
The face recognition implemented in the system is an automatic one. This means, that no button click is necessary in order to register a new face. Instead, as soon as a new face, which can not be identified in the existing database entries, appears in front of the camera for longer than 40 frames, the registration is triggered automatically. No user input is required.

The FacePrefabSpawner has also been created as an empty game object and the script FacePrefabSpawner_Script.cs has been attached to it. As soon as the FacePrefabSpawner is informed by the FaceRecognition_Script, that a new face has been registered, and in case it is the first face to be registered during the current run of the system, a new FacePrefab object is instantiated. This represents the interaction point between face recognition and face tracking in the implemented system.
The FacePrefab game object resulted from the modification of the FaceTracking prefab provided by the RSToolkit. The RealSense facial tracking capability can identify the presence, as well as facial features of a user. Furthermore, it can support up to 78 Landmark points. These can be used for accurate recognition of expressions, according to Patil et al.[PB16].
The initial FaceTracking prefab contains numerous child components, which all aim to track different facial elements, such as the eyes, the lips, the nose and the eyebrows. While those can be used for facial expression recognition as mentioned above, the implemented system required merely a centrally positioned element of the face, so that the nose was chosen. Therefore, the FacePrefab game object, contains merely one child game object, called Nose.

The Nose contains some child game objects, which are plain 3D objects. These are used to visualize the tracked object in the Unity editor. However, the most important component of the Nose is the TrackingAction.cs script, which provides the tracking functionality.
Among the discovered limitations of the facial recognition provided by the RealSense SDK, is the fact, that the recognition works successfully only when there is a single person in the FOV of the camera. As soon as a second person joins in the FOV, the new person can not be recognized anymore. The algorithm treats the second face, as if they never were in the database, even if it was registered before. However the second face can also not be registered again, at least not as long there is still more than one person in front of the camera. As soon as only one person remains in the FOV, the recognition returns to working as expected.

In order to solve this, another approach has been tried, namely splitting the face recognition process into two halves. The first half was intended to handle the registration of each person individually. In the second phase, called recognition phase, multiple persons in the FOV of the RealSense were expected to be recognized successfully. However, this approach also failed. Even if the persons had been individually and successfully registered before, they still could not be recognized in the second phase, if more than one person was identified by the camera. This led to the conclusion, that even though the RealSense SR300 can track up to 4 faces, according to [Cor18c], the recognition only works well for one face at a time. The implementation was adapted accordingly.
Finally, the FaceRecognition, FacePrefabSpawner and FacePrefab game objects were all saved as prefabs and deleted from the scene. They would be called into action when needed by the SetupNetwork, component managing game object.

Face Tracking

As already described, when the face of a speaker enters the FOV of the RealSense, the face recognition game object informs the FacePrefabSpawner to instantiate the FacePrefab game object or update it, if it already exists. The FacePrefab game object is also added as a child of the ServerParent's child Server, so that is should also move according to the HoloLens movement and already be transformed by means of the calculated offset. The FacePrefab furthermore incorporates the face tracking functionality and its child component, Nose, tracks the position and rotation of the speaker's nose. In order to send the tracked nose position over to the client a game object shared over the network is used, called TransformedNoseCylinder. The TransformedNoseCylinder is shaded yellow and has an attached script component named GetNosePose_Script. This cylinder fulfills the role of adopting the position and rotation of the Nose tracker object on the client side.
Another thing, which needs to be taken into consideration, is the fact that the TrackingAction script provided by the RSToolkit, which is attached to the Nose Tracking object of the FacePrefab, defines the dimensions of two objects: a Real World Box and a Virtual World Box.

Their dimensions are both set to (100, 100, 100) and they signify the space within which the nose can be detected in the real world. However, even though the box dimensions seem to be representing centimeters, Unity interprets these units as meters. This means, that in the TransformedNoseCylinder's GetNosePose_Script, the fetched position needs to be divided by 100, in order to enable a movement on the same scale as the movement of the cameras in the real world.
After fetching the position and rotation of the Nose successfully, the TransformedNose-Cylinder sends it over to the client to be visualized as a virtual object on the nose of the speaker in HoloLens space.

RealSense and HoloLens Integration

The first attempt to integrate the RealSense libraries with the HoloLens ones, by simply importing the RSToolkit into a project already containing the HoloToolkit, failed miserably. Multiple hundreds of errors were caused. After much investigation, it became clear, that the scripts from the RSToolkit caused numerous problems, when intending to build and deploy the client application. This happened, even if the scripts were deactivated on the client. Their mere presence seemed to raise the errors.
The successful solution was represented by using Unity's Platform Dependent Compilation feature. This feature provides certain preprocessor directives. These enable the partitioning of scripts, which allows for compilation and execution of code sections exclusively for the specifically defined platforms. More details on this topic can be read in the Unity documentation[Tec18b].

This means, that each RSToolkit script causing errors was handled individually, by inserting the mentioned directives. For code, which should only be compiled and executed on the server application, the directive UNITY_EDITOR was used. Meanwhile, for code, which should only run on the client, WINDOWS_UWP was used as a directive. These directives were placed in all scripts, where it was necessary.

After inserting the directives in all RSToolkit scripts, the Platform Dependent Compilation was enabled. This meant, that the application could finally be build and deployed to the HoloLens. The RSToolkit had been successfully integrated into the HoloLens project.

Arduino Implementation

In the following subsections, the steps, which were necessary in order to program the Arduino pushbuttons, as well as their integration into the Unity project are explained.
In the beginning of the Arduino script, implemented using the the Arduino IDE, various declarations are done, starting with the pins. The numbers of the pins, which were used to establish the connection between the controller board and the button components, have to be declared first. The number of each pin is depicted on the controller board. In the case of the described system, pins number 2, 4, 7 and 8 were used.

The setup() method handles further initialization aspects and has the same role as the Start() method in Unity. In the setup() method the pinMode for each button pin is set to INPUT_PULLUP. The pinMode handles the configuration of the specified pin, defining its behaviour either as input or output. INPUT_PULLUP furthermore enables the internal pullup resistors, as specified by[Ard18a]. Then, the serial port used to connect the UNO R3 Controller Board needs to be defined. The serial port has been set to 9600 in the implementation.
The loop() method fulfills a similar role as the Update() method in Unity. Each time the loop is run, the state of each button is read. Then, dependent on the read button states, arbitrary signals are flushed over the serial port. For example, when the first button is pressed, 101 is flushed and printed out. For the second button 201 was chosen, for the third 301 and for the fourth 401. At the end of the loop() method the previous status variables are updated.

Later on, after having integrated the Arduino script with Unity, a very drastic decrease in the frame rate could be noticed. The application was running very slowly, with 20 frames per second (FPS) or less. As a solution, the Arduino code was adjusted, so that it would stop sending continuous signals to Unity each frame. Instead, only two signals were sent, one when a button was pressed, and one when the button was released, regardless of how long the button remained pressed. This allowed the frame rate to return to normal.
The final implementation flushes 101, when the first button is pressed, and 110 when it is released. Similarly 201 and 210 were used for the second button, 301 and 310 for the third and 401 and 410 for the fourth button. The final loop() method implemented in Arduino code can be visualized in Figure 5.7. In order to run the code written in Arduino IDE on the UNO R3 Controller Board, one needs to fulfill two steps: the code verification and its upload to the controller board.

Arduino Integration

For the intergration of the Arduino functionality into the Unity project, first of all a game object, named ArduinoConnection, was created. To this object a script called ArduinoConnection_Script.cs was attached. After importing the System.IO.Ports package, the serial port is set. The port can be found on the bottom right corner of the Arduino IDE interface. The band rate has to be set to the same one set in the Arduino code. In the Start() method of the Unity ArduinoConnection script a ReadTimeout for the created stream is specified and the serial stream is opened.
In the Update() method, the information regarding the currently pressed button, sent over from the UNO R3 Controller Board by means of the stream, can be recalled. The received input data is stored in the NetworkResults game object, which was previously explained in the network implementation section. From the NetworkResults script, the input data can be accessed by other Unity scripts. Thus, certain actions can be triggered depending on which button is pressed.

```
void function loop {
  if 1st_button pressed {
    send 101 to the server PC;
  } else if 1st_button released {
    send 110 to the server PC;
  }
  else if 2nd_button pressed {
    send 201 to the server PC;
  } else if 2nd_button released {
    send 210 to the server PC;
  }
  else if 3rd_button pressed {
    send 301 to the server PC;
  } else if 3rd_button released {
    send 310 to the server PC;
  }
  else if 4th_button pressed {
    send 401 to the server PC;
  } else if 4th_button released {
    send 410 to the server PC;
  }
}
```

Figure 5.7: The code segment shows the implemented logic, which handles the pushbutton signals in the loop() method of the Arduino script.

For a successful integration of the Arduino components in the Unity server application, another important change needs to be executed in the Player Settings. The tab with the Settings for PC, Mac & Linux Standalone needs to be selected this time in the Player Settings. Then, in the Other Settings section, the Api Compatibility Level needs to be adjusted to .NET 2.0.
Further information on how to connect the Arduino with Unity can be found on the following source[20118a].

5.2.4 Encountered Limitations

During the implementation of the face recognition part and especially during the nose position synchronization with the client part, some limitations of the used RealSense SDK have been noticed. The face tracking is not always very accurate or stable, which can cause problems, since the resulting tracking position is sometimes quite unpredictable. However, as Zhao et al.[ZCPR03] explains, despite the fact that current machine recognition systems, such as face tracking and recognition, have reached the highest level of maturity they ever had before, their results are still very much dependent on the conditions, such as lighting for example, which real applications often enforce. As these aspects still represent a widely unsolved problem, currently existing systems can not compete yet with the human perception system's capability, according to Zhao et al.[ZCPR03].

5.3 Visualization Implementation

Subtitle visualizations in real time represent one of the main goals of this master thesis. The design of the visualizations has already been described in the System Design chapter. This chapter provides thorough explanations regarding the visualization implementation in the implemented system. All implemented components playing a role in the visualization concept are presented and explained, as well as their integration into the entire system. First, the text visualization implementation is presented, followed by implementation of the sound localization visualization and the network connection visualization.

5.3.1 Text Visualization

As explained before, the text strings, needed for the subtitle and the speech bubble visualizations, are extracted by means of the speech recognition game objects. These are components of the server application. When new subtitles are available, they are stored in the NetworkResults game object's script. The script provides the functionality of sharing them across the network by means of the SyncVar attribute. This enables the client's game object to access the extracted subtitle text and visualize it in various ways. During the implementation of the system many visualization types were proposed and implemented, however in the end some of them were combined to form new ones, resulting in the final three visualizations.

For the visualization of text Unity provides two different possibilities: Text Mesh and UI Text. During the implementation both of them were tested. While both of them worked fine, both induced different kinds of complications regarding the implemented system's requirements. During the comparison of Text Mesh and UI Text a difference in quality was noticed. The text provided by Text Mesh seemed to have much worse resolution than the UI Text. UI Text provided the overall impression of a better quality.
The next issue noticed with Text Mesh is, that no outlines could be added to the subtitles. Meanwhile, adding outlines to UI Text game objects was very easy, by simply attaching a additional Outline.cs script to the text object.

Furthermore, when testing Text Mesh, which works by actually generating 3D geometry for the display of string text, the main encountered problem were the line breaks. They were not facilitated. Text Mesh did not provide automatic line breaks after a certain amount of words is reached in a sentence. Instead, the text was presented as one single long row. Since the line break functionality was absolutely needed for the system's textual visualizations, alternative solutions were researched. A possible solution would have been to download an additional asset from the Unity Asset Store called Text Mesh Pro. However, this additional complication combined with all the other drawbacks, resulted in UI Text being adopted as final text visualization modality.

Therefore, UI Text in addition to further UI game objects provided by Unity were used for the implementation of all visualizations. However, even though UI Text comes with much better quality, easier line break capabilities and the possibility to attach outlines to the illustrated text, it still caused some issues. When the first UI game object is created in the Unity scene, an additional game object named Canvas is created as well. The Canvas acts as the created UI game object's parent, as well as the parent of all further created UI game objects.
However, for the implementation of the visualizations, the HololensObj game object needed to be the parent of the created UI Text game object. This was the case, because the text visualization elements needed to be attached to the display of the HoloLens camera. Thus, they could move according to the head movements of the user. But the attachment of the MixedRealityCamera to the Canvas, in order for it to be the parent of the UI Text element, was not possible. The Canvas parent only allows UI game objects as its children. The developed solution consisted in the attachment of the whole Canvas game object, with all its UI children components, underneath the MixedRealityCamera game object within the HololensObj. This works fine, as all UI elements are moving as they should, together with the HoloLens MixedRealityCamera.
The HololensObj prefab was updated accordingly. This allowed for the generation and attachment to the Canvas of further needed UI game objects, which are explained next. Figure 5.8 shows the structure of the final HololenObj prefab.
For the implementation of the billboard subtitles, an empty UI game object named Visualization1_BillboardSubtitle was created and attached underneath the Canvas. Then, a new UI image game object called Billboard_Image was created and attached under the previously generated parent.

The second child attached to the Visualization1_BillboardSubtitle parent was an UI Text game object named BillboardSubtitle_UIText. This UI Text's Outline script was deactivated. The position and scale of the text were adjusted, in order to place it in the center of the previously created billboard image. Finally, a script was attached to the BillboardSubtitle_UIText, which fetches the NetworkResults game object at runtime and retrieves the information it needs for its functionality. The needed information includes the currently pressed button and the text string extracted by the speech recognition unit. If a new subtitle text is detected in the NetworkResults and one of the buttons corresponding to the speakers is pressed, the string text is set as the text of the BillboardSubtitle_UIText. Thus, the subtitles are displayed on the HoloLens display, in order to be seen by the D/HH user.
For the implementation of the second visualization, first of all regular subtitles were needed. In order to facilitate this a further empty UI game object named Visualization2_Subtitle was created and attached to the Canvas. This object contains merely one UI Text component named Subtitle_UIText. The implementation resembles the implementation of the billboard subtitles. However, no billboard UI Image is needed and the Outline script of the Subtitle_UIText was set to active, to provide better readability even on light backgrounds.

Figure 5.8: The figure shows the final structure of the HololensObj prefab. It contains all components needed in order to use the camera of the HoloLens, as well as all UI components needed for all three subtitle visualizations.

These subtitles make use of the same script as the billboard subtitle visualization for the setting and updating the subtitle text. The second needed component for this visualization is an UI Image named Visualization2_SpeechBubble_Image, which fulfills the role of containing the speech bubble graphic. This speech bubble UI Image has a further child component, SpeechBubble_UIText, whose Outline script was deactivated too. Its text is fetched and managed using the same script, which the subtitles use. Then, a new script was created and attached to the speech bubble UI Image, which checks the status of the child game object SpeechBubble_UIText's text. If the string is currently empty, the whole bubble is deactivated. Once a new subtitle text is detected, the speech bubble appears again showing the text.

For the third visualization's implementation a new UI Image named Visualization3_Blue Bubble_Image was attached to the Canvas within the HololenObj prefab. The image also contains a script, which deactivates the speech bubble as well as its containing text, if all speakers are silent. The contained text is an UI Text named BlueBubble_UIText, which uses the same script as the other subtitles to retrieve its text from the NetworkResults. The Outline script was deactivated. The third visualization additionally requires the blue billboard subtitles.

```
define arbitrary duration;
timer += deltaTime;
if timer >= duration {
  timer = 0;
  reset text to empty string;
}
```

Figure 5.9: The code segment enables the subtitles to still remain visible for an arbitrary amount of seconds after the speaker's button is released.

Thus, the same implemented component, Visualization1_BillboardSubtitle, which has been described above, is used when needed. In the first implementation version of the script fetching the subtitle texts, the subtitle text disappeared as soon as the speaker stopped pushing his or her corresponding button.
However, this was presumed to be insufficient for the D/HH user to be able to read the subtitle text. Thus, the implementation was changed, so that the subtitle text still remains visible for two additional seconds, after the speakers release their buttons. This is shown in the code snippet in Figure 5.9.

Speech Bubble Placement

As previously mentioned, the position of the speech bubbles was not final. Another game object is used in order to provide them with their position during runtime. For the creation of the said game object an empty new game object was created, which contains a single child game object, namely a 3D sphere. The Mesh Renderer of the sphere has been deactivated, so it would not distract the user during the runtime. The role of the sphere object is to mark the 3D position, where the speech bubbles are visualized in the HoloLens space.
The sphere contains two further child game objects, named Visualization2_SpeechBubble and Visualization3_BlueSpeechBubble, which contain a single script each. The scripts fetch the SpeechBubble and BlueSpeechBubble components of visualization 2 and 3's UI Image items in the HololensObj. The scripts furthermore handle the visibility in relation to the pressed buttons. If any of the speaker's buttons is pressed, the speech bubbles are activated. However, if the visualization switch button is pressed, which is be explained in the following section, the speech bubbles are deactivated.
Each of the sphere's children contains a further child game object. These are however empty and represent mere placeholder game objects positioned in relation to the sphere. Their role is to indicate to the Canvas' children, where the speech bubbles should be placed. Both previously mentioned game objects in the HololensObj prefab, Visualization2_SpeechBubble_Image and Visualization3_BlueBubble_Image, contain an additional script. These scripts fetch the placeholder objects attached to the sphere, retrieve their position and update their own position according to it.

In order to set the sphere's position, a new script has been attached to the sphere object. The initial plan was to set the sphere position to the position of the TransformedNose-Cylinder. The TransformedNoseCylinder represents the position of the speaker's nose, as detected by the face tracker functionality of the RealSense and is sent over the network. However, even after applying the transformation resulting from the calibration phase, the tracked nose position was still too unrealiable and inaccurate to be used for the user study. The speech bubble jumped around and had random offsets, which translated the sphere outside of the FOV of the HoloLens. While the RealSense SDK's face tracking could be replaced with a more solid and dependable one in a future work, a solution had to be found for the user study. This was necessary in order to enable the users to evaluate the speech bubble visualization.

As a solution, the TransformedRealsenseCylinder was used instead of the Transformed-NoseCylinder. The TransformedRealsenseCylinder is necessary in the calibration phase in order to visualize the Vuforia image target tracked by the RealSense on the HoloLens. Since the image target is tracked in real time, during the calibration, the position of the TransformedRealsenseCylinder changes accordingly.
However, after the finalization of the calibration phase, its position remains to its last updated one and its Mesh Renderer is deactivated. This provides the advantage, that it follows the user's head movements, always remaining in the user's FOV.
By setting the speech bubbles to the position of the TransformedRealsenseCylinder instead of the unreliable TransformedNoseCylinder, the user studies could be carried out successfully. This solution ensured, that the bubble was always visible to the users, as soon as a user's face appeared in the system's FOV.

5.3.2 Sound Localization

In order to implement the described sound localization visualization, a new UI Image called SoundLocalizationVisualization_Image was created and added to the Canvas object of the HololensObj as a child. The image is initially set to the graphic representing silence, where all arrows are inactive.
Furthermore, a new empty game object was created in the scene and named SoundLocal-ization. A script called SoundLocalizationVisualization is attached to it and contains a reference to the sound localization visualization UI image created before. The previously explained pushbuttons were very useful for the implementation of the sound localization visualization. The SoundLocalization game object furthermore contains a reference to the NetworkResults game object, in order to fetch the information regarding the currently pressed button. This script fulfills the role of switching the sprite in the Image component of the sound localization visualization, depending of the button currently pressed. If button 101, 201 or 301 are pressed, the image is set to the one, where the corresponding arrow is highlighted. In all other cases, the image is set back to the image visualizing the silence scenario.

5.3.3 Visualization Switch

The system's implementation comprises the possibility to switch between the three presented visualizations. In order to enable this, a new empty game object named VisualizationSwitch was created. Also a new script was generated and attached to the game object. The VisualizationSwitch script first gathers all the needed UI game objects, the blue billboard subtitles, the speech bubble and the regular subtitles, as well as the blue speech bubble, in order to enable their management.
NetworkResults is also needed in order to inform about, whether a speaker's face is detected by the RealSense or not. NetworkResults additionally informs about the currently pressed button. The VisualizationSwitch script defines and makes use of another important variable named clickCounter. This variable is incremented each time the fourth, gray button is pressed. The counter goes from 0 to 2, and at its next incrementation it is set to 0 again. The whole visualization switch logic is based on this variable.

Before the gray button is pressed, clickCounter is set 0. This means, that only the components of the first visualization, the blue billboard subtitles, are visible. Meanwhile all components of the other visualizations are deactivated. The only other visualization, which is left activated, is the sound localization visualization. This is done, so that the user is informed about the location of the speaker at all times.
Once the gray button has been clicked, the VisualizationSwitch script deactivates the previously active components of the first visualization. At the same time it activates the components of the second visualization, the composite visualization comprising the speech bubble and the regular subtitles.
Components of different visualization types are never active at the same time. The nrFaces variable extracted from the NetworkResults game object is used to decide, which one is shown at a given time. If no faces are detected, the subtitle visualization is presented. Thus, the user is not obliged to establish visual contact with the speaker all the time. The user can take notes instead for example. If nrFaces is greater than 0, the visualization switches to speech bubble.

Even within the same visualization, the number of faces can change. This requires updates in the visualizations. These changes are also handled in the VisualizationSwitch script. When the gray button is pressed a second time, the visualization script deactivates all members of the second visualization. At the same time it activates the components of the third visualization instead. The third visualization is the composite visualization comprising the blue speech bubble or the blue billboard subtitles. However, they are also not visualized at the same time. Instead, their activation also depends on the nrFaces variable. If the gray button is pressed a third time, the elements of the first visualization are displayed again. Meanwhile all elements belonging to the other two visualization types are deactivated. This procedure can be repeated over an arbitrarily long period of time.

5.3.4 Network Connection Feedback

As mentioned in the System Design chapter, an additional text is visualized to the user, in order to inform about the status of the network connection. This was implemented by adding an additional UI Text game object to the Canvas. A script was created and added as a component of the network status UI Text game object. The script fulfills the task of trying to fetch the PlayerPrefab game object. In case this game object can be detected in the scene, the script can access it. This means, that the client has connected successfully and the UI network status variable's text is set to Connected. In the opposite case in which the PlayerPrefab cannot be detected, it cannot be fetched. In this case the UI Text is set to Disconnected, so that the user is provided with feedback regardless of the situation.

CHAPTER 6

User Study

This thesis also includes the execution of a user study to analyze the results of the implemented system. The user study design, its execution, as well as various other related factors are explained in this chapter. On the one hand the user study aimed to gather data on the participating users. Some examples of questions, which the users were asked to answer, are: questions regarding their gender, age, skills and experience. On the other hand the user study comprised questions regarding the implemented system. The users answered questions like: did they have enough time to read the subtitles, were the subtitles easily readable, was the sound localization visualization helpful, could they imagine using the system in the future and of course which visualization type was their favorite. The answers to these questions were extremely valuable for the evaluation of the system, which is presented in the next chapter.

6.1 Recruiting of Participants

For the execution of the user study, participants belonging to the mentioned targeted user groups needed to be recruited. In order to enable that, a Doodle poll was created, in which numerous one hour slots were listed in the course of four weeks. Persons belonging to the targeted user group willing to participate, could register for the user study participation by means of the Doodle link.
First, the user study was intended to be executed with the help of a public institution, such as a school. However, this required certain approval processed, which would have taken months to be carried out. Since this represented a setback at that time, solutions were brainstormed. The optimal solution for the recruiting issue of user study participants was to search for people with hearing disabilities, who would voluntarily participate in the experiment in their own free time.
However, since such persons are rather hard to find and their willingness to participate was also questionable. Thus, an alternative backup solution was developed.

In the case in which no D/HH participants would have been recruited, hearing participants would have been invited to join. However, since they did not represent the main target group, their sense of hearing would have had to be invalidated somehow.
This could be achieved for example by either plugging in-ear headphones in their ears or wearing noise canceling headphones. Another solution would have been for the hearing participant to wear headphones, which would play loud music or generate loud noise. Thus, they would not have been able to hear the carried out conversation during the experiment. These alternative solutions however, would not have been optimal, because of many reasons. The most important reason being, that regularly hearing persons, which suddenly do not hear anymore, are phased and distracted by this fact. They would not be able to focus on the experiment.
Luckily enough participants from the intended target user group could be recruited. In order to facilitate the recruiting process, an information sheet has been prepared. This sheet quickly informed about the system's context and purpose, as well as about the fact, that voluntary participants were needed. The information sheet furthermore contained the exact address, where the implemented system was located and the contact information of the contact person, for the case in which more details were inquired. Finally, the sheet also contained the previously mentioned Doodle link, which enabled the registration for the participation during a preferred time slot.
The recruitment process involved personal visits to many organizations and institutions working with and for people with hearing disabilities in the Vienna region on the one hand. On the other hand the recruiting of the participants also involved many phone calls, exchanged emails and social media posts on platforms, such as Facebook and Instagram. All exchanges involved the distribution of the prepared information sheet as well as friendly exchanges on the subject. Furthermore, the exchanges were accompanied by the provision of many details on the system and the user study. The efforts finally resulted in 11 D/HH persons agreeing to test the implemented system and participating in the user study.

6.2 User Study Design

The user study design consisted in the brainstorming of how the user study experiment should be carried out optimally. All features of the implemented system needed to be taken into account, which would enable their evaluation. All important features of the system had to be tested by the users in order to enable the evaluation of the system in an optimal and intelligent way.
In order to facilitate the execution of the user study, as well as the evaluation of the system, a number of documents were designed and supplied. All documents, which were used during the user studies, can be found at the end of this thesis, in Appendix A.
This section explains the steps, which were necessary for the execution of the experiment. This includes the user study setup, as well as the chronological sequence of events executed during each user study appointment. Furthermore, this section explains where all previously explained user study documents were applied.

User Study Setup

In this section the setup of the experiment will be explained, starting with the location. The system included a powerful server PC, which was located in a laboratory of the TU Wien. Furthermore, the laboratory was spacious enough to enable the execution of the experiment. Thus, it was decided, that the user studies would take place at the TU Wien. The first thing, which was taken care of, was the clearing up of the unnecessary furniture items in the room, as well as the placement of a rectangular table in its center. A few wooden chairs were required as well. Three of the chairs were placed on three different sides of the table. Two additional chairs were placed outside the laboratory for eventual accompanying persons, who would have to wait.
The next important test consisted in checking, if the application would run successfully on the HoloLens. This test could be carried out after the build and the deployment processes were completed. The test was positive, which meant that the application was running successfully, without any connecting cable. This simplified the user studies of course, since it meant that the users would have to deal with one less cable. The client application could then either be started from the HoloLens website in the web browser on the PC or from the HoloLens itself. Both of these cases were handled by the user study organizer.

The next thing which was taken care of was the elongation of the USB 3.0 cable of the RealSense camera. This was handled by using an USB 3.0 Hub and a five meter elongation cable. The cable was connected to the back of the PC, which fulfills the role of the server. After this was established, the USB cable connecting the UNO R3 Controller Board to the PC had to be elongated as well. A further extension cable was engaged for this matter. The Arduino hardware components were then placed on the mentioned table using the elongated cable. In order to prevent any problems, the cable was fastened to the table using duck tape.
For the case in which the user would have agreed to being filmed for scientific purposes during the execution of the experiment, a tripod with a corresponding holder were strategically placed in front of the table. This enabled the filming of the experiment by means of a smartphone. A smartphone was used, since they currently generate videos of even better quality than camera devices. Figure 6.1 shows a picture of the described setup.

User Study Execution Steps

1. **Instructions and Explanations** - as soon as the user arrived at the agreed upon appointment, they were greeted cordially, as well as the person(s) eventually accompanying the user. Then, they were politely asked to sit, as well as whether they desired a glass of water. After this, they were handed the first three pages of the instructions and explanations document. Eventual questions were answered right away. The instructions and explanations document welcomes the participants and gives them a brief introduction on the context of the experiment.

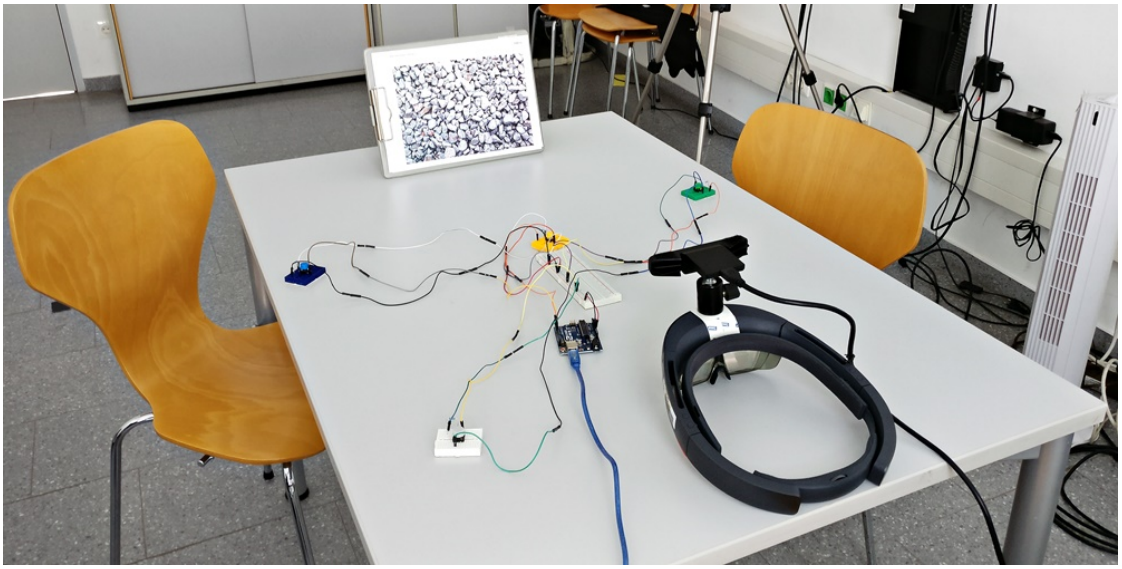


Figure 6.1: The figure shows the constructed setup for the user studies. The Arduino components can be noticed in the middle of the table. As two speakers were going to take part in each user study, the blue and the green button are placed to the sides, near the seating positions of the speakers. The HoloLens headset with the RealSense camera mounted on top of it, is placed at the front end of the table. This shows where the participating user will be seated. On the picture one can also notice the stone image target. This is used by the organizer of the user study in the beginning of the experiment for the calibration step. However, the image target did not maintain its showcased placement during the experiment.

First, the implemented system is described briefly, then short introductory explanations of AR and speech recognition are provided. This enables users to learn a bit more about the context of the master thesis and deepens their understanding regarding the impact of their contribution.
The second page of the instructions document informs the participants again about the fact, that the implemented system represents a scientific prototype. The document furthermore informs, that the used technology is not ready yet for everyday usage. However, the user study participants are assured, that their contribution would impact the progress on this topic for future systems in a positive way.

Furthermore, the participants are informed about the fact, that the user study does not focus on the speech recognition as much as on the subtitle visualizations. The participants are informed about where to focus their attention during the experiment. At the end of this document, the users are reminded, that they can ask questions at any time and that not they, as users, are tested, but the system.

2. **Safety Warning and Consent Form** - after the user finished reading the three pages they were provided with, they were asked to read and sign the safety warning and consent form. This document was designed with respect to the legal aspects. For the document's design a template was used, which is often used by the TU Wien for user studies in the fields of Virtual and Augmented Reality. In the beginning of the document, the user study participants are informed about the fact, that the system to be tested is a scientific prototype, not a final product. The document informs the participants about the fact, that they are required to take full responsibility for their part taking in the experiment, as well as in the case of the emergence of possible consequences.
   The next part of the document investigates the state of health of the participants. The participants are asked questions about certain conditions or diseases, that would affect the experiment's execution. For example the user study participants should not suffer from any heart disease, circulatory disease, epileptic seizures, panic attacks or other similar conditions. In the case in which the user shows any of the enumerated system, their participation is denied by the TU Wien as a safety measure.

   The next part of the document instructs the participants to follow the instructions of the user study operator. They are furthermore asked to inform the operator as soon as there is a problem, case in which the experiment would be interrupted at once. Furthermore, the participants are advised to not drive a car or carry out any physically strenuous activities for at least one hour after taking part in the experiment. This is required, so that eventual aftereffects would pass.
   The last part of the document asks the participants, whether they agree with being filmed or photographed during the experiment for research purposes. The participants are also asked, whether they agree with the fact, that the results of the user study will be published in papers, presentations or publications. Provided that their participation would remain anonymous, of course. At the end of the safety warning document the signature of the participant is required in order to prove their consent.
   After fulfilling the legal aspects comprised in this document, and if no special case arises, which prohibits the participation of the user, the user study can be carried on.

3. **User Questionnaire** - the next step was represented by the completion of the user questionnaire. This document has the purpose of gathering information on the participating users. It comprises various single choice and Likert scale questions. The questions are related to the characteristics, skills and experiences of the users, including for example their gender, age, lip reading and sign language knowledge level as well as their disability type. The goal hereby, is to enable the examination of the randomness of the participants.

Furthermore, the questionnaire collects information on the participants' experience with technology and VR or AR. The participants are also asked, whether the participants wear glasses, whether they tried speech to text systems before and whether they are excited to be part of the project.
Another important question inquires, whether they believe, that persons with hearing disabilities would benefit from technological aid or not. The gathered information is merely collected for the anonymous statistical evaluation and will not be distributed or misused in any way.

4. **The First Simulator Sickness Questionnaire (SSQ)** - right after completing the questionnaire regarding their own person, the user is asked to complete the first SSQ, inquiring their current state. The SSQ represents another important document which was used in order to investigate eventual simulator sickness manifestations. According to Nelson et al.[NRBM00], AR headsets may also cause simulator sickness, similarly to VR glasses. Thus, the usage of this type of questionnaire has been decided, in order to assess this aspect for the implemented system. Even though it is known, that AR induces less simulator sickness than VR equipment, the extent of simulator sickness induced by the implemented system needed to be tested. This was necessary in order to evaluate, if simulator sickness would constraint the system's use.

   In order to examine the system's effect on the users, the SSQ needs to be filled out twice: once before and once after the execution of the experiment. The SSQ consists of a table composed out of symptoms, which the user study participants have to rank by means of Likert scales. If a certain symptom does not apply, the user is instructed to check the None checkbox on the corresponding Likert scale. The other possible outcomes of a scale are: Slight, Moderate and Severe. The list includes symptoms such as general discomfort, fatigue, headache, sweating, nausea, concentration difficulty, dizziness and eye strain. The SSQ can also be regarded as part of the user questionnaire, since it also refers to the user.
   The state of the user as reported in the SSQ questionnaire does not have to be perfect, for example it is possible, that the user may already sweat or have a headache. The initial state of the user merely needs to be registered, in order to be compared to their state after the experiment's completion.

5. **Experiment Execution** - the entire execution of the experiment is explained in detail in the Testing the System section.

6. **The second SSQ** - after the experiment ends, the organizer helps the user take off the headset and stops the system. Right after the user is asked to fill out the second SSQ.

7. **System Questionnaire** - after the user completes the second SSQ, he or she is asked to fill out the system questionnaire. The user is given as much time as they

need to fulfill this task, discussing eventual emerging questions and improvement suggestions. The system questionnaire focuses on the examined system. Most of the questions in this questionnaire are designed using Likert scales.
The first two questions target the speech recognition. The users are asked, if they understood the carried out discussion, while they were wearing and using the system. Then, it is inquired, whether the users had enough time to read the displayed subtitles.
The next three questions aim to gather information on the three different kinds of displayed subtitle visualizations. In order to enable this evaluation of each visualizations, the corresponding questions are accompanied by a sketch of the according visualization. The sketches are meant to help the users remember, which visualizations they tested, as well as enable a deeper insight and a well thought out answer. Each of the three mentioned questions ask the users to rate each visualization type. The user are furthermore asked to provide a justification of the given answer.

The next question refers to the readability of the subtitles, implying size and contrast. The users can rate the readability using a Likert scale too. Furthermore, the system questionnaire presents a sketch of the sound localization visualization and asks the users about its usefulness. The users are also asked, whether they noticed it or not. The following question requires the user to rate the user-friendliness of the system.
The next two questions aim to evaluate the usefulness of the combined visualization types, the ones where the speech bubble visualization switches to the billboard subtitles, when the face of the speaker is spotted. Question number 11 inquires, whether the user could imagine using such a system in the future. This aims to ensure, that the researchers are on the correct path and invest time and energy into something, that the people can really see themselves using in a few years.
The next question requests the users to enumerate application scenarios, where they can imagine, that the implemented system would be most useful. A few suggested application fields are already enumerated in the form of checkboxes. However, the users are encouraged to suggest more situations. Finally, the users are asked if they enjoyed taking part in the experiment and if they can provide any improvement suggestions.

The final page of this questionnaire comprises 51 concepts presented in the form of checkboxes. The concepts comprise positive as well as negative attributes, which might or might not characterize the tested system. The users are asked to choose five of the 51 concepts, which describe the system best in their opinion. This summation of all chosen adjectives as well as their frequency can then be used for the generation of a word cloud. The resulting word cloud would describe the implemented system in an honest way. This kind of procedure is known under the name of Microsoft Reflection Cards.

All input provided by the participating users in this questionnaire are used to test and evaluate as many features and aspects of the implemented system as possible. This enables their scientific evaluation, which is presented in the next chapter.

8. **The Farewell** - this represents the final stage of the user study, after the user finished completing the system questionnaire with regards to the implemented and tested system. The user is cordially thanked for his or her support and given a small participation gift.

6.3 Testing The System

As mentioned before, the setup provides three chairs. As soon as the experiment begins the D/HH participant sits at one end of the table wearing the headset. On each of the table's lateral sides a speaker is seated. For each of the user studies two speakers were present. This enabled a conversational exchange. The organizer of the user study played the role of the first speaker, sitting at the left of the user study participant. The organizer also fulfilled the role of switching among visualizations by pressing the gray button, also placed on the left side of the participant.
The second speaker, seated to the right of the participating D/HH user, was always represented by an additional guest speaker. The guest speakers joined the experiment as helpers in their free time. The guest speakers were very diverse, as the role was fulfilled by seven different people. Furthermore, the guest speakers were not only both male and female with accordingly different voices, but also multicultural. This means, that some of them spoke German with various accents. This made the user study even more interesting.

The two speakers necessary, since this underlines the importance of the sound localization visualization. This specific visualization informs the user when another user is speaking from another direction. Thus, a minimum of two speakers was required in order to evaluate this visualization. During the experiment, the speakers engage in an alternating conversation. Each time a speaker intends to talk, he or she registers to speech by pressing their entitled button. The button is placed in front of them on the table. While speaking, the speaker hold their button pressed, releasing it soon after they finish their sentence. Meanwhile, as the buttons are alternately pressed and speech is exchanged, the D/HH participant receives corresponding subtitles in real time. The subtitles are visualized on the display of the HoloLens.

The users were furthermore told, that they had to fulfill a task during the experiment. Their task was to observe the conversation by means of the subtitles. They were asked to try to understand the conversation, as well as take notes of the exchange using the provided paper and ballpoint pen. The users received this task of writing everything down, that they could extract from the conversation, because of multiple reasons.

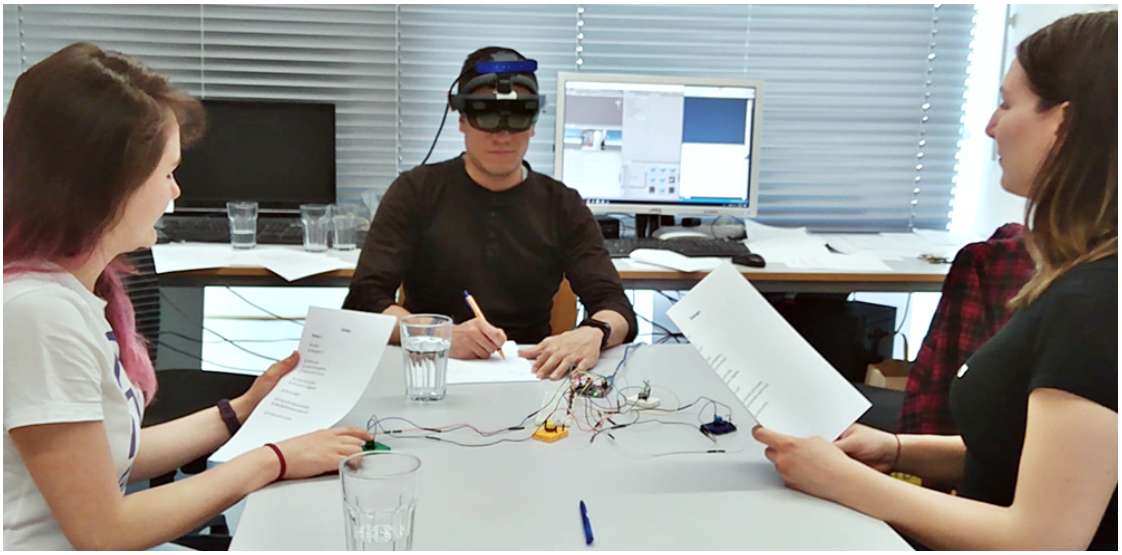


Figure 6.2: The figure shows a picture taken during one of the user studies. One can notice the constructed setup for the user studies, as well as how the speakers and the participating D/HH user are sitting. The user wearing the headset is seated at the front side of the table, right next to the provided paper and pen. They need these in order to fulfill their assigned task of taking notes, while the two speakers engage in a conversation. The two speakers are sitting on both sides of the participating user. Each of them has a button in their proximity, which they press while speaking. The provided subtitles in real time help the user follow the conversation, even when he is not looking directly at the speakers.

First of all, this enabled a good measurement of the speech recognition accuracy. Secondly, it required the user to look down on the paper. This aimed to urge the user to move their head, so that the faces of the speakers would exit the FOV of the system. This was necessary in order to trigger the visualization switch from speech bubble to subtitles during the first and second visualizations. The introduced task therefore aims to activate all implemented visualization features integrated in the system. Figure 6.2 provides an impression of the user study's final setup, as well as of the user fulfilling the assigned task.

The conversational exchange between the two speakers was based on three predefined dialogs. All three dialogues comprise sentences of similar complexity. The sentences are relatively short and not very difficult. Their content involves regular everyday discussions. The speakers are greeting each other and asking how they have been. They then proceed to inquire the other about their vacations. In another dialogue the speakers talk about their favorite dishes, as well as about their university degrees. The dialogues were predefined in order to optimize the results of the offline speech recognition provided

by the RealSense SDK on the one hand. On the other hand the predefined dialogues enabled a fair evaluation of the speech recognition performance during all of the user studies. Had the dialogues not been predefined, the speakers would have been required to communicate freely allowing for non-deterministic results and discarding the chance to compare the speech recognition results from a user study to the others.

The three different implemented visualizations were tested in turn. During each visualization type the speakers exchanged one of the three predefined dialogues. The testing of each visualization took about two or three minutes. As soon as a dialogue was completed, the organized pressed the gray button construct, causing the visualization to change to the next type. After the second dialogue exchange was done and the user had taken the required notes, the organizer pressed the gray button one more time. This enabled the testing of the third visualization. As soon as the testing of the third dialogue was completed, the organizer helped the user to take off the headset and then stopped both client and server applications.

6.4 Miscellaneous

Besides the already mentioned aspects, some other ones needed to be taken into consideration for the user study. These are listed as follows.

Pilot User Study

After having prepared all the documents and having defined the sequence of events, which were meant to take place during each user study, a pilot user study was organized. During the pilot user study a person, who has not seen the system before, went through the whole user study process step by step. The pilot user study is necessary in order to test the whole designed procedure including all of its steps one final time, before the actual user tests take place. This fulfills the role of ensuring the designed process' applicability and faultlessness. This furthermore helps to identify critical moments or aspects, which could be further improved.
For the pilot user study, the selected user is required to be an external person, who was not involved in the project or the implementation of the system in any way. This is recommended, since it allows for an objective and unbiased point of view, providing the organizer with valuable improvement suggestions regarding the designed user study process.
Furthermore, the pilot user did not necessarily need to be D/HH, since they would only test the user study process and provide feedback. Even though the execution of a pilot user study is not absolutely required, it is highly recommended. In the case of this master thesis a number of improvements of the user study process emerged from its execution. For example, the pilot user suggested, that water and opening the door of the laboratory would be a great improvement to the users' comfort.

Additionally, the experiment execution was tested and improved and some small ambiguities in the designed documents were identified and clarified.

Demographics Inquiry

The demographics represent the persons participating in the user study and should optimally be representatives of the target user group, comprising D/HH persons. Luckily, in this user study all participants were indeed recruited D/HH persons. These persons were kind enough to participate in the evaluation of the system in their own free time. This of course enabled the extraction of more significant and accurate results.
Some other aspects needed to be taken into consideration regarding the participating users, is the variety of the participants. The mentioned variety refers to the participant's age, gender, their skill sets or their interaction and experience with technology. The variety of all these aspects guarantees a certain randomness of the participants, which improves the quality of the user study evaluation. Fortunately the chosen recruiting process ensured this randomness of the participants.
The randomness of the participants, as well as other interesting results, were extracted from the user questionnaire.

Language

The chosen language, in order to ensure a successful communication with the D/HH participants, was the German language. The motivation is the same as the one for choosing German for the language setting of the speech recognition, which is the fact, that German would most likely be the mother tongue of D/HH persons living in Vienna, Austria. Since the system was aiming to simplify the life of the target user group, no additional requirements, as for example the knowledge of a foreign language, such as English, were desired. Therefore, all provided written documents, designed for the user study, were written in German, in order to simplify the interaction with the users.

Repetition

The number of times the user study needed to be repeated is directly proportional to the number of D/HH participants. However, in order to enable significant results, enough user study participants were needed. At least ten participants were required, so that the statistical evaluation of the user study results were meaningful and could be generalized to a larger amount of people. Luckily, the recruiting efforts payed off and eleven voluntary participants offered to take part in the user study, fulfilling this criteria.
Furthermore, all experiments in the user study had to be carried out in the same way for each user. This represents the only way, which guarantees correct and significant statistical results. This meant, that the system's implementation needed to be and was completed before the start of the user study.

Duration

The duration of a single user study appointment needed to be optimized. All relevant features of the implemented system needed to be tested and all relevant information, feedback and answers needed to be gathered from the users in a minimal amount of their time. This required very well-thought planning and optimal communication methods. Therefore, the time slots in the shared Doodle link were restricted to one hour each. This was intended to comprise everything from explanations, to experiment execution and answering questions. It also turned out to have been a very good approximation, since some of the user studies were finished in less than one hour.
As mentioned before, the user study took place over the period of four weeks, at the beginning of summer 2018.

Communication

Another very important aspect was represented by the fact, that the communication needed to be ensured in some way. This was not always easy, taking into consideration, that some of the participants in the user study were deaf and usually communicated using sign language. This was handled by taking the safety measure of providing all necessary explanations and instructions also in written form, with much detail, even sketches and pictures for better understanding. This aimed to assure, that even if the deaf person came with no translator or attendance, the communication would not represent a problem. As a matter of fact, also in the cases in which the verbal communication was not possible, the communication by means of writing or drawing was also encouraged and carried out in a friendly environment.

Thanksgiving

At the end of the user study, after the participants finished completing the system questionnaire, the users were cordially thanked for their kindness and participation. Furthermore, they were assured, that their contribution is meaningful and highly appreciated. The thanksgiving was provided in written and verbal form, as well as the assurance, that the organizer would further on be available for questions. Finally, a small departure gift was prepared for the users. This consisted in a small food item, which was aiming to illustrate the gratitude for their participation. In order to take into consideration as many personal tastes and diets as possible, various food items were provided. Thus, the participants could choose between two healthy and two sweet food items.

CHAPTER 7

Results

This chapter comprises the results, which were extracted after the execution of the user studies. Furthermore, this chapter includes significant feedback received from the participating D/HH users.

7.1 Feedback from a specialist

The first feedback, which was received before the execution of the user studies, emerged from meeting with a specialist, who works with D/HH persons on a regular basis. The field specialist manifested quite much interest in the prototyped system and believed in the importance and usefulness of such research.
The specialist explained, that technologies such as speech recognition, after having reached a certain maturity and reliability, would represent a tremendous improvement in the lives of D/HH people. The specialist furthermore underlined, that there is a large request for a system like this. However, another valuable feedback received from the specialist was the fact, that D/HH people highly value their integration among hearing people. Thus, D/HH persons would not like to wear a large headset on a daily basis, according to the specialist. This problem will probably be solved in the near future, as AR systems are getting smaller and less conspicuous. Thus, the size of the headset was not taken into consideration during the user studies for the system's evaluation. The feedback of the field specialist on what is important for D/HH persons was however very useful.

7.2 Safety Warning and Consent Form

One of the first documents handed to the participating users during the user study was the safety warning and consent form document. The evaluation of the forms, after being filled out by the users, shows, that 81.82% of the users did not manifest any of

the enumerated symptoms. The symptoms represent cases, in which the users are not recommended to take part in the experiment for safety reasons. The remaining 18.18% manifested high blood pressure as a symptom, however still agreed to take part in the user study of their own free will.
Furthermore, 90.91% of the participants agreed to being filmed or photographed during the execution of the experiment for scientifical purposes. Lastly, 100% of the participating users agreed to the publication of the evaluated results of the user study.

7.3 Simulator Sickness Questionnaire

For the evaluation of the simulator sickness possibly caused by the system, the SSQ was used. However, the user studies took place in summer inside a laboratory, which may have influenced some symptoms, such as sweating. The simulator sickness pre- and post- questionnaires were evaluated by gathering the answers of all participants and weighting them with certain predefined weights. The resulting total score describes the manifestation percentage of the simulator sickness symptoms. The total score is calculated twice, once for the pre-questionnaire and once more for the post-questionnaire. This allows for the comparison of the user's status before and after the execution of the experiment. After the evaluation of the pre-questionnaire, a total score of 12.58% was obtained. However, after calculating the total score of the post-questionnaire, a percentage of 9.86% was acquired.

This means, that simulator sickness did not represent an obstacle during the execution of the experiment. On the contrary, the general condition of the users seems to have improved after using the system. This means, that they experienced an improvement in their general state and some of their symptoms decreased after the experiment. An explanation for this might be the user's excitement regarding the participation in the experiment. Their concentrated efforts might also have impacted the results. These may have distracted the users from the symptoms, resulting in an improvement.
However, after the execution of all user studies, the expected conclusion can be drawn, namely that the simulator sickness caused almost no problems during the use of the implemented AR system. Consequently, simulator sickness may most probably remain a bigger issue for VR systems.

7.4 User Questionnaire

The assessment of the user questionnaire showed that 54.55% of the participants were male, while the other 45.45% were female. Furthermore, Figure 7.1 shows the distribution of age among the participants. According to the statistics 27.28% of the participants were between the age of 19 and 25 years old. 18.19% of them were between 26 and 30 years old. The next 27.28% ones were of ages between 31 and 40. The following approximately 9% were between 41 and 50 years old, while the remaining 18.19% of the participants

Figure 7.1: The figure visualizes the distribution of age among the user study participants. As can be noticed from the box plot diagram, the distribution is quite even, there are no outliers and users from five different age categories took part in the experiment. The median lies at the 31-40 age category. The lower side of the box represents the first quantile, while the upper side of the box stands for the third quantile. The lower whisker visualizes the minimum age, while the upper whisker shows the maximum age of the participants.

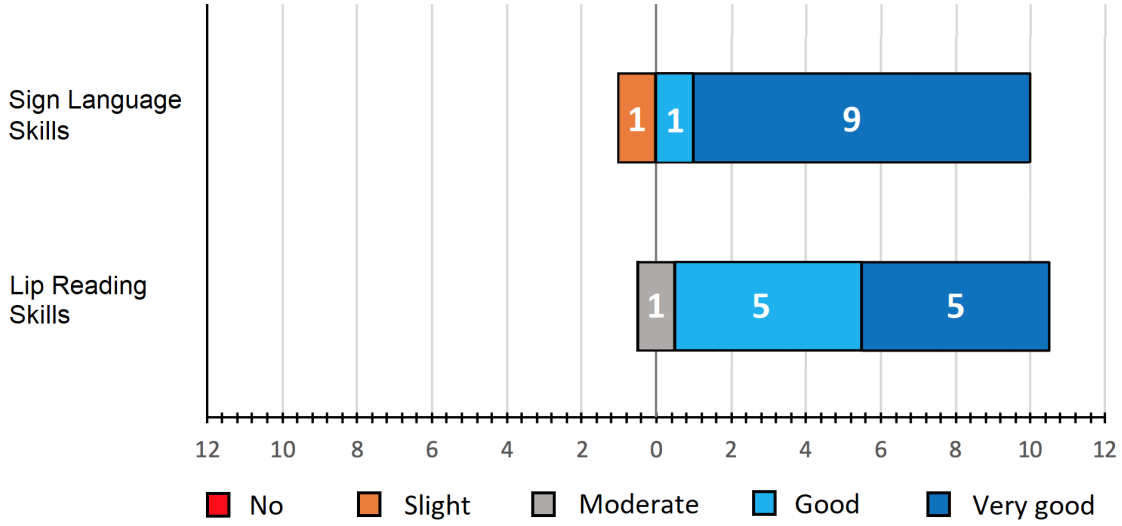


Figure 7.2: In the upper row of the diagram the distribution of sign language skills among the participating users is visualized. The lower row presents the distribution of lip reading knowledge levels among the user study participants.

were between 51 and 60 years old. When visualizing the diagrams, one can notice, that a very diverse and well spread demographics took part in the user study. This leads to the conclusion, that the recruiting process was a success.

The next evaluated category was referring to the skills of the participants. As shown in the lower row of Figure 7.2, all participants possessed relatively good lip reading skills. Only one of the participants rated themselves as moderately good. The remaining ones estimated their lip reading capabilities as either good or very good.
In the upper row of Figure 7.2, the results regarding the sign language skills of the participants are shown. These are of course very much related to the type of their disability. Hard of hearing persons, who can still hear in comparison to deaf persons, may need the sign language less than a deaf person, who relies on sign language as their main means of communication. Nevertheless, only 18.19% of participants rated their sign language skills as slight or good. All remaining 81.81% defined their sign language knowledge as being very good.

The results of the evaluation also show the distribution of the disability types among the user study participants. As presented by the diagram on the left side of Figure 7.3, the distribution was very well balanced. 45.45% of the participants were deaf, while the remaining 54.55% of the participating persons were hard of hearing. This allowed for very useful and interesting feedback from both sides.
The next inquiry referred to the participants' experience regarding computers and technology.

These results were also quite evenly distributed. 18.18% of participants claimed to be computer experts. 27.27% stated, that they use the computer on a daily basis for work. 36.36% defined their interaction with technology as being moderate. The remaining 18.19% stated, that they never use computers or do so very rarely. When asked if they ever experienced VR or AR before however, most of the participants, namely 72.73%, claimed, that they have never tried these technologies before. Only the remaining 27.27% had tried it before and therefore had a notion of what they could expect.

The next inquiry asked the participants, whether they wear glasses of not. This question was included, as this might also play a role in their acceptance or rejection of the new, showcased technology. The majority of participants, 72.73%, were wearing glasses. The remaining 27.27% either did not need glasses or were wearing them only in certain situations. The question regarding the wearing of glasses is interesting, because of two reasons. First of all, AR currently presumes the use of a headset or glasses. This means, that user study participants, who are already accustomed to wearing these items, would possibly react more accepting to the system's use during the user studies, as well as in the future. Secondly, there was the question of comfort during the experiment. Since glass wearing persons might need to choose between wearing their glasses or the HoloLens headset. Even though the HoloLens explicitly provides the possibility of wearing one's glasses underneath the headset, some users still preferred to take their glasses off. They did so in order to simplify the experiment and increase their comfort.

Next, the users were asked, whether they have tested any speech recognition systems before. The majority of 63.63% answered, that they have never tested speech recognition before. This meant, that this represented a further novelty for them during the execution of the experiment. The other 36.37% had already tested such a system, so they were better prepared to a certain extent.
When asked, if they were eager to try new things, 72.73% of the participants declared, that they absolutely are. This could of course also be concluded from the fact, that they decided to take part in the scientific user study during their free time. 18.19% also stated, that they try new things quite often, while the remaining 9.08% were undecided.
Furthermore, the diagram on the right side of Figure 7.3 shows the evaluation of the participant's answers, when asked if they believed, that D/HH people needed more technological support in order to gain more from taking part in meetings. The results were very interesting and show, that almost three out of four participants, namely 72.72%, believe that D/HH people absolutely need more technological support. 18.18% gave a positive answer too and the remaining 9.1% were undecided. Finally, in the end of the user questionnaire the users were asked, if they were excited about the experiment. All of them answered extremely positively.

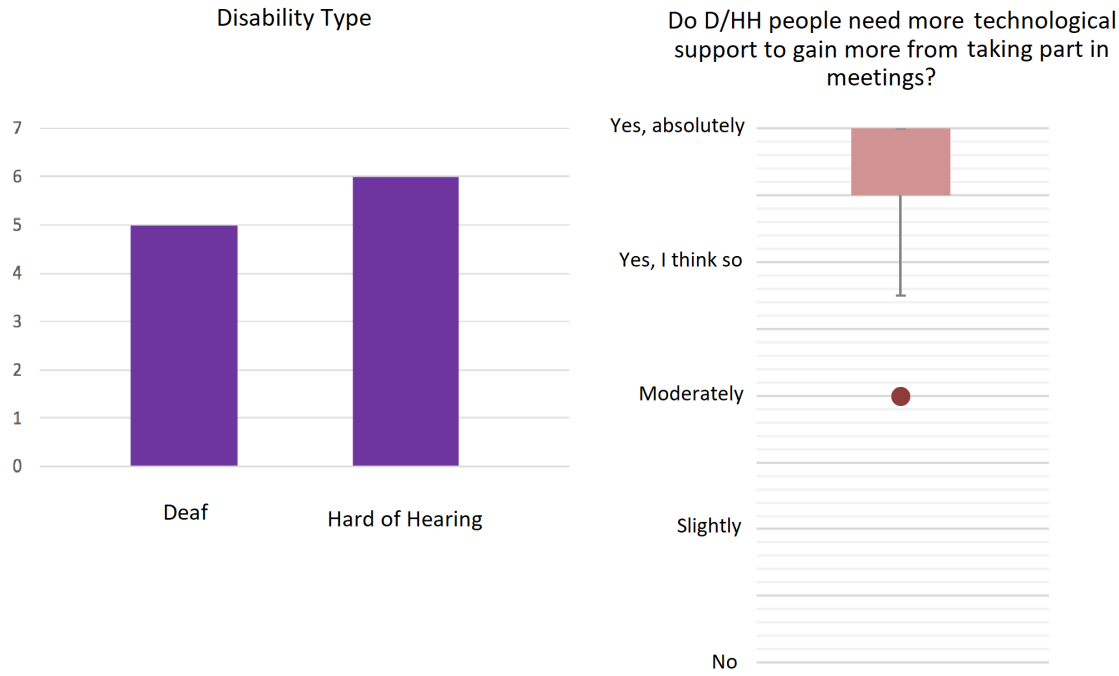


Figure 7.3: The left side of the figure shows a diagram representing the hearing disability type distribution among the participating users. The box plot diagram on the right side visualizes the distribution of the users' answers, when asked if they believed, that D/HH persons would benefit from more technological aid. There is one outlier at M̈oderately, as one person was not completely sure about the need for such a system, however the median of the plot lies at Ÿes, absolutely.

7.5 System Questionnaire

The first questions in the system questionnaire regarded the used offline speech recognition algorithm. When asked, if they had enough time to read the subtitles, the majority, meaning about 90.91% of the participants, replied positively. The exact results can be inspected on the upper, left side of Figure 7.4.
When asked, whether they understood what the speakers were talking about, the participating users responded slightly more diverse. However, the majority of 72.73% still answered, that they understood the content of the witnessed conversation. The exact results can be seen on the upper, right side of Figure 7.4.

The lower diagram in Figure 7.4 shows the satisfaction of the participants regarding the text readability. All of them confirmed, that the text readability was ensured, in means of chosen font, font size, selected colors and provided contrast. The next questions, which were answered by the users, regarded the important aspect of rating the visualizations. Therefore, the users were asked to state, how much they liked each visualization in turn.

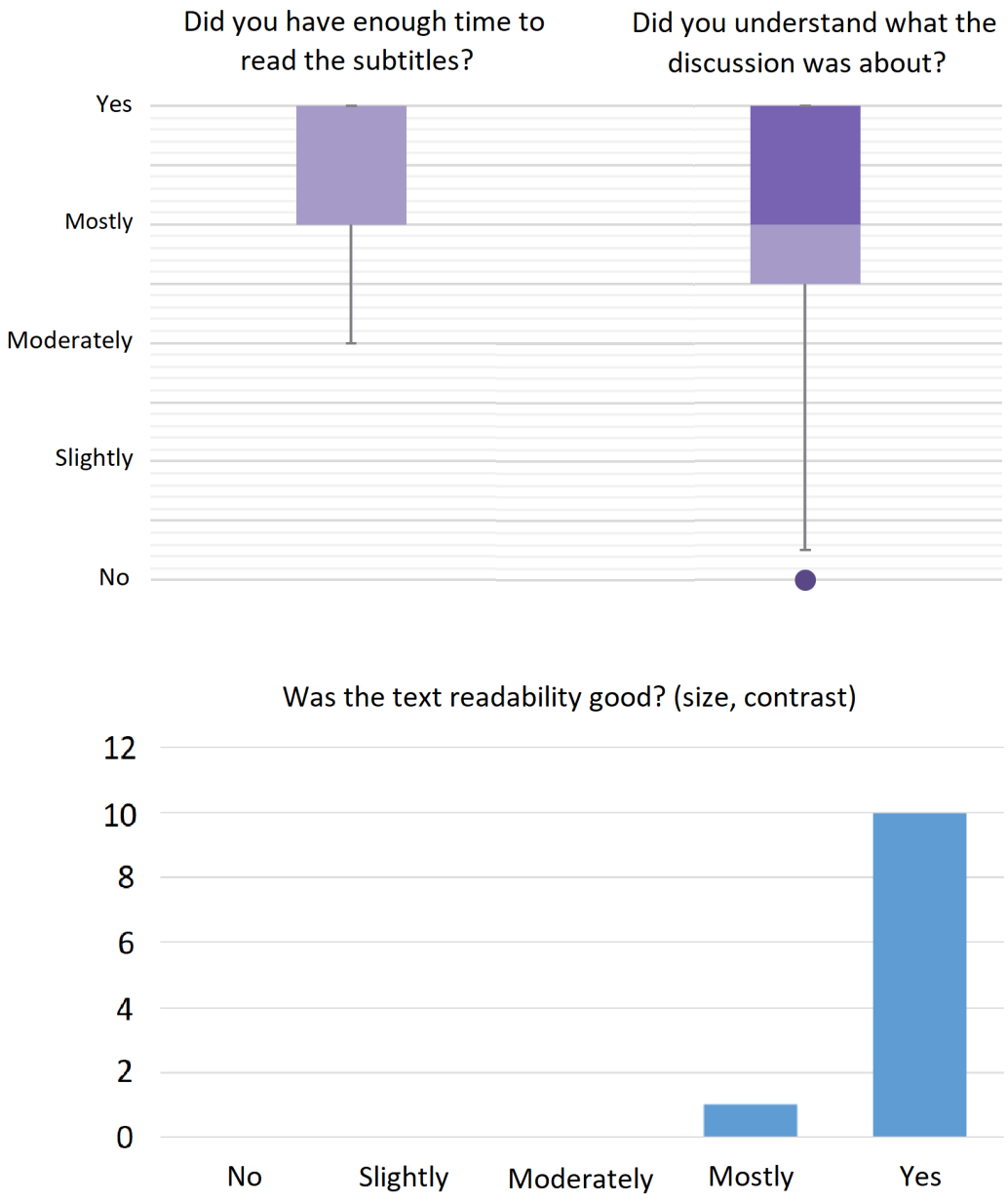


Figure 7.4: The figure shows the evaluation results regarding the speech recognition (upper diagram) and text readability (lower diagram). The diagram in the upper left corner of the figure shows the users' answers regarding whether they had enough time to read the subtitles. In this diagram the median, as well as the third quantil and the maximum lie at Ẏesẏ The second diagram, in the upper right corner, shows the evaluation of their answer regarding if they understood, what the speakers were talking about. The median in this diagram lies at M̈ostlÿ The lower and third diagram in the second row shows, that the implemented system enabled very good readability of the subtitles, referring to chosen font, size and contrast.

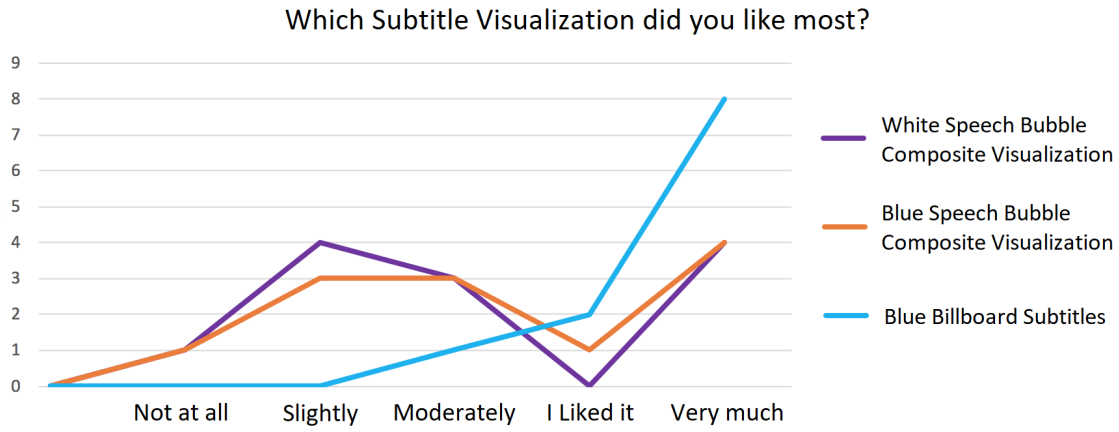


Figure 7.5: The visualized graphic shows the answers of the participants, when asked which visualization type they liked most. As one can notice from the diagram, the clear majority preferred the blue billboard subtitles.

The results are presented in Figure 7.5. Summing up, even though about 30% percent of the users did like the composite visualizations combining the speech bubble with the subtitle visualizations, the clear majority of about 70% definitively preferred the simple subtitle visualization. When asked to explain their choice, the majority of the participating users produced the same argument. They prefer the simple subtitle visualization, because of their being used to it for many years, due to movies and series which they watch on television or the internet. Less than 18% of the users liked the speech bubble visualization best. This percentage of the users found the combined speech bubble visualization to be more interesting. They also stated, that it reminded them of comic books and they believed it fitted the smalltalk.

However, even though a few of them liked the speech bubble visualization too, the majority still defined it as being rather confusing, since it appears rather unexpectedly. In comparison, the billboard subtitles are implemented in such a way, that the billboard still remains visible to the user, even when there is silence. The users seem to appreciate this very much, since they know exactly where to expect the subtitles, as soon as they are available. This visualization type does not produce any surprises. When asked about the speech bubble visualization, one participant even stated, that she startled when it appeared, because it appeared so suddenly, when a face entered the FOV of the system.

Another important argument brought by the user study participants is, that the simple subtitle visualization can fit much more text into it. Meanwhile, the speech bubble provides a limited space for textual display. Between the regular white subtitles with black outline and the blue billboard subtitles, the users found the blue billboard subtitles to

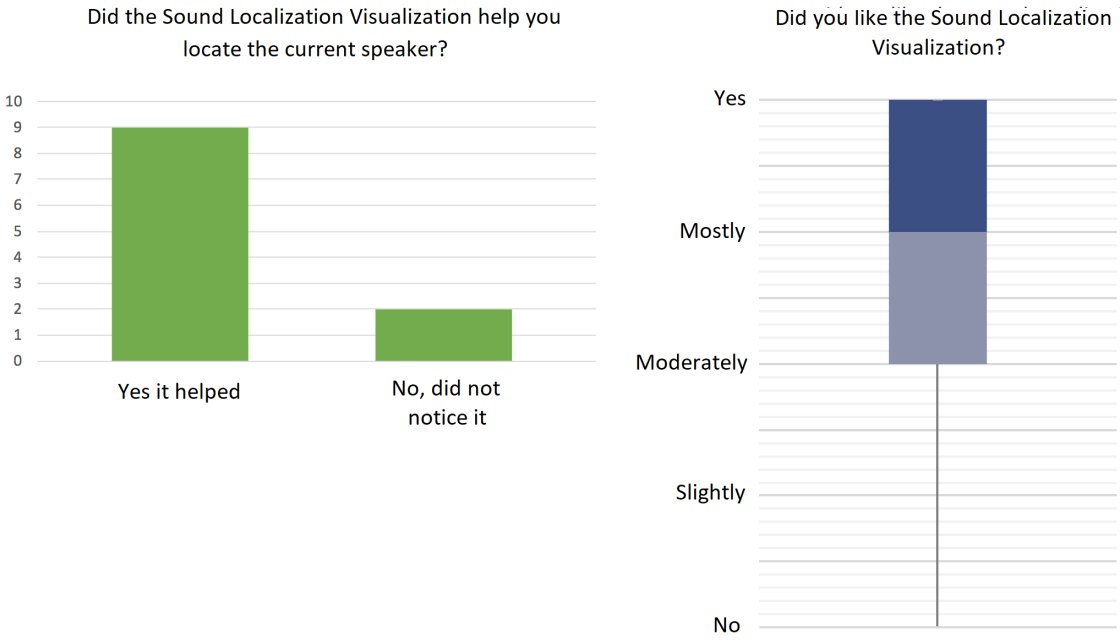


Figure 7.6: The diagram on the left presents the extent, to which the participants found the sound localization simulation visualization helpful. The feedback was mostly positive. Meanwhile, the diagram on the right side visualizes the answers of the participants, when asked if they liked the mentioned visualization. The answers were various, however the median lies at M̈ostly.̈ Meanwhile, the maximum and the third quartile lie at Ÿes.̈

be more readable. They explained, that it provides better contrast and at the same time shows where the subtitles will appear, once they did. Another very valuable argument provided by the users was, that the speech bubble's appearance might cover important other aspects of the scene or other faces. This is undesirable. Again, the users preferred the billboard subtitles. When using this visualization type the billboard also covers a part of the scene, however continuously. Also its placement is rather low, so that the user can adjust his or her FOV accordingly.

The next important inquiry regarded the sound localization visualization. When asked if this specific visualization was helpful, 81.82% of the users answered, that it did help them locate the current speaker. The remaining percentage did not notice it. This is probably the case for hard of hearing users, who do not completely rely on their eyes for the identification a speaker, as they can also make use of acoustic signals. Regardless, the majority of participating users liked this visualization and found it useful. These results are presented in Figure 7.6.

When asked, whether the users liked the switch between speech bubble and subtitles more than the regular subtitles, 90.01% responded, that they preferred the regular subtitles because of habit. Merely the remaining 9.09% preferred the speech bubbles.

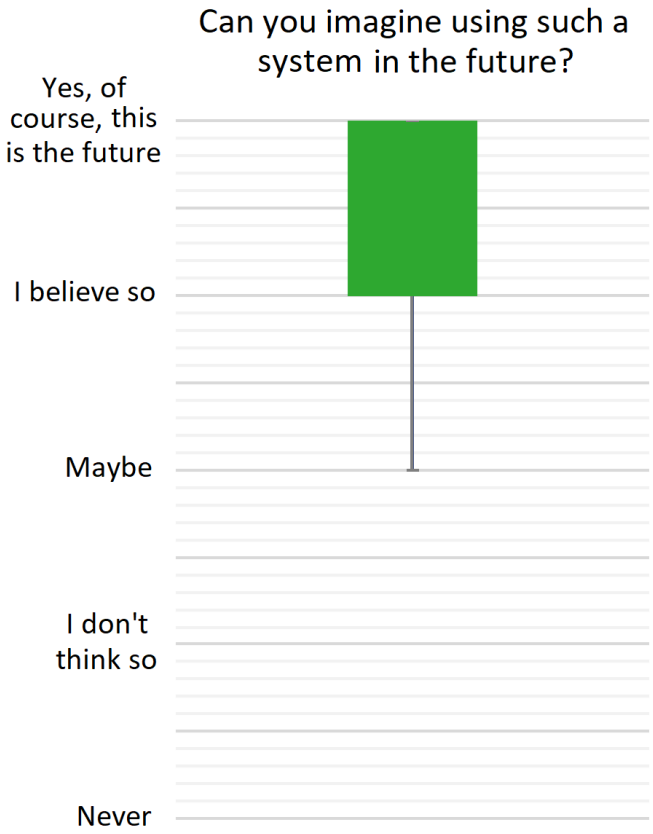


Figure 7.7: The diagram visualizes the answers of the participants, when asked whether they could imagine using such a system in the future. As one can see in the diagram, the results were extremely positive. The median lies at Ï believe so.

When asked to provide arguments as to why they liked the speech bubble more, they said it was more fun and made them feel like part of comic book. However, the majority clearly preferred the regular subtitles and believed, that the switch caused confusion. This demonstrates, that simplicity represents the most elegant solution.
When asked if they liked the fact, that the visualization changes back to subtitles, when no speaker is present in the FOV, 63.63% of the users did reply, that they likes this. They explained, that it enabled them to see the subtitles, even while they are doing something else, such as writing notes, instead of looking at the speaker. However, the users explained, that they would still prefer having regular subtitles, regardless of the presence of a speaker's face.

When asked about the system's ease of use, 36.36% responded, that it is very easy to use. 45.46% answered, that it is not that hard to use. The remaining 18.18% said either that the system is moderately or relatively hard to use. When asked to justify their answer, the users said, that the system's headset takes some time to get used to.

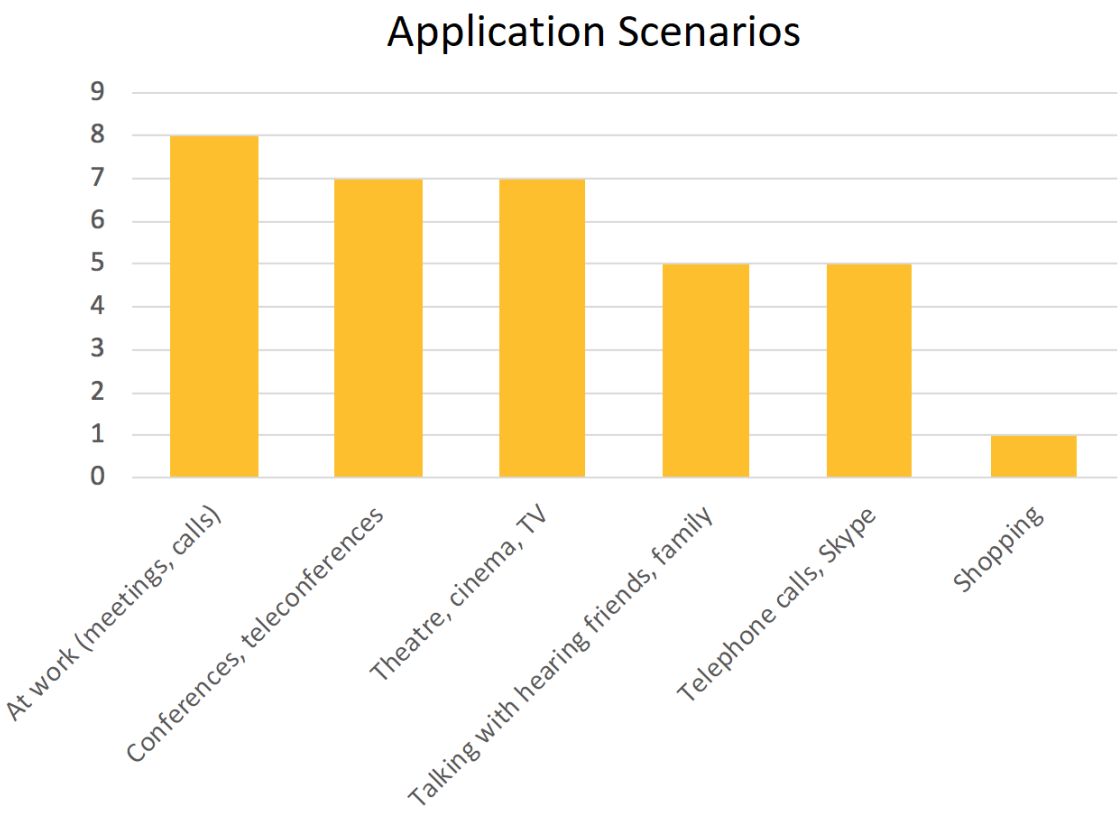


Figure 7.8: The diagram shows the most important hypothetical application scenarios for the system, as chosen by the participants. The user study participants could imagine the system being of great use in these scenarios in the future.

When asked, if they would use such a system in the future, 90.91% of the participants responded, that they would most probably use it. They believe, that this technology is going to establish itself. Only 9.09% answered, that they were not sure if they would use it. These results can be seen in the diagram shown in Figure 7.7.

The users were furthermore asked to enumerate application fields, in which they could imagine such a system being of great use. Most of the participants could imagine using such a system for work, in order to communicate with colleagues. They believe, that the system would be of great use, especially when the colleagues are not in one's FOV or in order to take part more efficiently in meetings or calls.
The users also thought, that the system would be very useful during the participation at conferences or teleconferences. The system could provide a solution for the cases, in which the speaker is too far away or not visible. The users also believe, that the system would be of use in the entertainment industry, for example when going to the theater, cinema or simply watching movies without subtitles on the television. 45.46% of the users also could imagine using it, when talking to hearing friends or family members,

Figure 7.9: The figure shows a word cloud resulting from the Microsoft Reflection Cards evaluation. The word cloud contains numerous words, which have all been chosen by the users. They represent the main characteristics of the system in the user's eyes. The size of the displayed words is directly proportional to the number of times the users have chosen the respective word as optimally describing the system.

as well as during telephone calls or Skype calls. These results are visualized in the bar chart presented in Figure 7.8. When asked to enumerate further application fields, the users also mentioned, that they would use it during doctor's appointments, interviews or shopping. 18.19% of the users said, that the system would be of use in every situation during everyday life, where a translator would normally be required.
When asked if they enjoyed taking part in the user study and testing the system, 100% of the participants replied very positively. When asked to justify their answer, they said that they enjoy trying out new technologies, which they believe to be the future and that they know, that progress is only possible, if people like them support the people implementing such systems.

7.6 Microsoft Reflection Cards

As mentioned, on the last page of the system questionnaire the users were provided with 51 checkboxes. Each contains an attribute. The participating users were asked to select five attributes, which describe the implemented and tested system best from their point of view. This procedure is known as the Microsoft Reflection Cards.

The answers of all participants were collected and evaluated as soon as the user studies were terminated. The evaluation was done using the following website[20118d], which enables the generation and customization of a word cloud using the input words as well as their frequencies. Figure 7.9 shows the resulting word cloud. It contains all words, which describe the system best in the eyes of the users.

CHAPTER 8

Conclusion and Future Work

Summing up, this master thesis provided an introduction to the state of the art of promising new technologies, such as speech recognition, speech to text, AR, face detection, face tracking, face recognition and sound localization. The overview provided in the beginning also enables persons from other fields of expertise to optimize their understanding of the thesis. Furthermore, the thesis provided an insight to the current situation of D/HH persons in Austria and explained, how the mentioned technologies can be used in order to aid hearing-impaired people in their daily lives.

During this master thesis the mentioned hardware and software components were successfully combined, offline speech recognition was used in combination with depth face recognition, tracking and Arduino. A working system prototype was designed and implemented, which is able to provide subtitles in real time, in an AR environment and in the form of three different visualization types. The switching between the visualization types was facilitated by means of pushbuttons. The implementation of each part was described in great detail, in order to help further researchers, when aiming to perform a similar task or build up on the existing prototype. The encountered limitations and problems have also been described and possible solutions have either been implemented or presented.

However, this master thesis also included the design and execution of a conducted user study with eleven D/HH participants. The user study participants tested the system and provided valuable feedback, which has been analyzed and presented in the Results chapter. Even though the hypothesis of the user study design aimed to demonstrate, that the composite visualizations work better than regular subtitles and the majority of users clearly preferred the regular subtitles instead of the composite visualization types, this still represents a very valuable result, because it provides clear guidance for the implementation of future systems. Now it is known, what the target user groups desire and need.

The users not only preferred the simple subtitle solution, which they were used to, but also gave a lot of arguments in order to explain, why they did so. Some of the most important arguments in favor of the regular billboard subtitles were the fact, that they always appear in the same spot, marked by the billboard, even during silence, the fact that they could visualize longer sentences, the fact that they did not cover important aspect of the scene and the fact, that they were not unpredictable, startling or hard to find.
The users also stated, that using the system, which provides subtitles in real time, which can be read while the speaker talks, is very valuable for the users. It allows for easier memorizing, as the users do not need to focus on decoding the speaker's speech using various complex cues. Instead, they can relax and read the provided subtitles.

Even though the implemented system successfully combines numerous technologies and provides good results according to the user study evaluation, there are a few aspects, which require future work. All of the provided feedback was analyzed, presented and transformed into suggestions for future work.
The first aspect, which needs improvement, is represented by the face tracking algorithm. The used face tracking and face recognition are provided by the RealSense SDK. However, several limitations have been discovered during the implementation. The most important one is the inaccuracy of the face tracking and the fact, that the face recognition stops working as soon as more than one face enter the FOV of the camera. These aspects would require either an improvement by the development of a solution or a replacement with another face recognition or tracking library, which may provide more accurate results.

The next aspect, which should be improved in order to make the system more usable in the future, is the speech recognition. As mentioned, the currently used algorithm provided by the RealSense SDK is free and offline. This certainly provides some advantages, however also implies less accurate results in the speech recognition than online services, such as IBM Watson or Alexa. The speech recognition unit could be replaced with an online algorithm in a future work. However, not even online systems are completely accurate at the moment. This will improve a lot in the near future too.
Another thing, which could be improved in the existing implemented system prototype, is the RealSense camera holder. Ideally, a customized holder should be modeled and 3D printed, which could fulfill the role of securing the RealSense camera in a stable way on the HoloLens headset.

Besides the so far enumerated aspects, the users who participated in the user study provided some additional ,valuable feedback on what could be improved in the system. The users highly recommend the use of regular billboard subtitles. They advised against the addition of any further complications, such as additional speech bubble visualizations, as this might cause distractions or obstructions in an already limited FOV.

About 20% of the users not only found the blue billboard subtitle's contrast strong enough, but even too strong and thus uncomfortable. They suggested the introduction of a possibility to allow the users to customize their own subtitle preferences. They wished to be able to make adjustments, such as setting their own optimal color for the billboard and subtitles, as well as selecting the font size and style. This appears to be a great idea, because everyone has an unique way of seeing. A contrast, which would be comfortable for somebody, might disturb another person. However, this feature was suggested by the users with rather much computer knowledge. Perhaps the more inexperienced users would not enjoy the liberty of configuration as much.

Another helpful feedback involved the sound localization implementation. A user suggested the mere introduction of some arrow characters at the beginning of the subtitles. Thus, no additional sound localization visualization would be needed. This would mean, that two visualizations would be combined into one. The billboard subtitles would already contain the arrows before the text. This way even more space would be provided in the FOV of the HoloLens.
One of the users even proposed a very interesting improvement to the provided subtitles, which would allow the users more time to read long phrases. The user study participant suggested, that the text rows should slide upward, when a longer text follows. The sliding up of the first row would make room for a second row. In the case in which more text would follow, the existing two rows should slide up, making space for a third row. The user suggested, that this should continue as described, until a certain temporal threshold was reached. This would cause each row to disappear after a certain predefined period of time.

An additional important aspect, which needs to be adjusted in the future, is the weight and size of the used headset. Also, the FOV of the HoloLens needs expansion, as it is rather limited currently and cannot compete with the FOV of the human eye. These aspects have not been taken into consideration within the evaluation of this master thesis, because of the fact, that this is expected to improve a lot in the future.
However, the users who participated in the user study were very glad that they did, since they knew how important their contribution is for future technological advancements. Finally, this master thesis was written in the hope of representing a contribution to improving the world we currently live in.

Appendix A

This appendix contains all documents, in German language, which were designed and used for the user study, namely:

1. the instructions and explanations document
2. the safety warning and consent form
3. the user questionnaire
4. the simulator sickness pre-questionnaire
5. the dialogues used during the experiment execution
6. the simulator sickness post-questionnaire
7. the system questionnaire

Einführung

Willkommen zu unserer Benutzerstudie!
Wir sind sehr glücklich und dankbar, dass Sie sich Zeit genommen haben dafür! ☺

Worum geht's?

➔ Wir haben ein System entwickelt, das Untertitel in Echtzeit in verschiedenen Darstellungen auf einer Augmented Reality Brille zur Verfügung stellt.

Was ist eine **Augmented Reality** (AR) Brille?

➔ Es ist eine Brille, durch die man durchschauen kann, aber die uns auch ermöglicht verschiedene virtuelle (in der Wirklichkeit nicht vorhandene) Dinge zu sehen: in unserem Fall – den gesprochenen Text.



➔ Diese Untertitel werden erzeugt durch Spracherkennung (Sprache zu Text).

Was ist **Spracherkennung** / Sprache zu Text?

➔ Es ist eine Technologie die die Sprache eines Sprechers aufnimmt und sie in Text umwandelt.

➔ Der Text wird dann auf der Brille angezeigt.

Bitte haben Sie Verständnis:

➔ Es handelt sich um ein **Forschungsprojekt**, das heißt: Das System ist noch NICHT bereit für die Benutzung im alltäglichen Leben

➔ Die Resultate dieser Benutzerstudie stellen einen Beitrag für die Wissenschaft dar, und sollen Fortschritte in der Entwicklung eines Systems erlauben, der hörbeeinträchtigte Menschen in der Zukunft unterstützen soll.

➔ Wir sind uns bewusst, dass die Brille noch sehr groß und schwer ist, aber das soll sich in den nächsten paar Jahren ändern. Bitte bei der Evaluierung nicht berücksichtigen.

➔ Wir wissen auch, dass die Spracherkennung noch nicht perfekt ist, und noch Fehler produzieren kann, aber das wird in der nächsten Jahren auch mit Sicherheit deutlich verbessert werden

➔ Dieses Experiment heute konzentriert sich auf **die Darstellungen des Textes/der Untertitel** auf der Brille (Schriftart, Größe, Funktionalität, Dimension, Farben, usw)

➔ Wir haben 3 verschiedene Text Darstellungen implementiert und möchten mit Ihnen testen, welche der 3 Text Darstellungen hilfreicher ist.

Generelle Infos:

- Behalten Sie im Sinn, dass nicht Sie als Benutzer getestet werden, sondern das System und die Text Darstellungen ☺
- Sie können jeder Zeit Fragen stellen!
- Während des Ablaufs der Experiments werden wir begleitende Personen, ausgenommen den Übersetzern, bitten draußen zu warten, um Ablenkungen zu vermeiden.

Benutzerstudie :
Allgemeiner Ablauf

1. Wir werden Sie zuerst bitten ein Dokument (die Zustimmungserklärung) durchzulesen und zu unterschreiben.

2. Wenn alles passt, fangen wir mit dem ersten Fragebogen an.

3. Dann setzen wir Ihnen die Augmented Reality Brille auf und führen das Experiment durch, dauert nur einige Minuten (maximal 10).

4. Nachher werden wir Sie bitten ein weiteres Fragebogen auszufüllen, wo wir Ihnen Fragen stellen zum getesteten System, bzw zu den Text Darstellungen.

5. Fertig! ☺

Ablauf im Detail

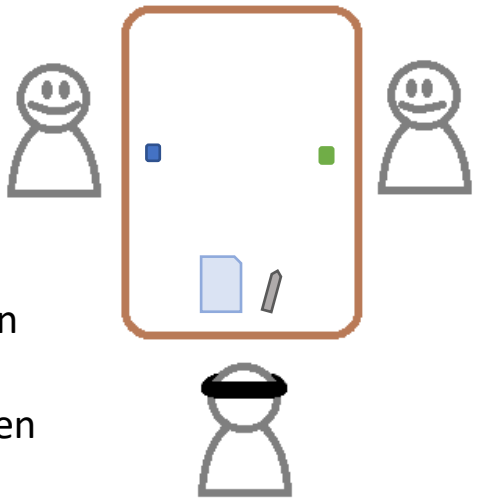
Wir erklären Ihnen nun den genauen Ablauf:

1. Zustimmungserklärung unterschreiben
   - wenn Sie das System testen wollen/sich an der User Study beteiligen wollen: unterschreiben Sie bitte zuerst eine Zustimmungserklärung
   - die gesammelten Daten werden anonym für Forschungszwecke verwendet.
   - Sie können jederzeit aufhören oder austreten wenn Sie wollen oder sich nicht wohl fühlen.

2. Fragebogen 1
   - Vor dem Experiment bitten wir Sie ein paar Fragen schriftlich zu beantworten: die Fragen beziehen sich auf ihre Person, und auf Ihre jetzige Gemütslage

3. Experiment:

   <u>Aufbau:</u>

   - Es gibt einen Tisch
   - Die hörbeeinträch-tigte Person sitzt an einem Ende, mit der AR Brille am Kopf
   - 1 oder 2 Sprecher sitzen an anderen Tischseiten
   - Die Sprecher unterhalten sich (Dialog/Monolog)
   - Während sie der Reihe nach sprechen, halten sie ihre entsprechenden Knöpfe gedrückt.
   - Die hörbeeinträchtigte Person verfolgt die Diskussion mit Hilfe der Untertitel an der Brille und macht Notizen.

- Ich werde das System starten und Ihnen die Brille dann geben
- Setzen Sie sich bitte am angewiesenen Platz am Tisch
- Wir helfen Ihnen die Brille aufzusetzen
- Wir werden 3 verschiedene Text Darstellungen hintereinander testen (2-3 Minuten pro Darstellung, max 10 Min insgesamt)
    - 1-2 Sprecher werden der Reihe nach reden
    - Sie haben Papier und Kugelschreiber auf dem Tisch
    - Ihre Aufgabe ist: Notizen zu nehmen von der Rede der Sprecher:
        - Schreiben Sie bitte alles auf, was Sie aus dem dargestellten Text entnehmen können

4. Fragebogen 2
    - Nach dem Experiment werden wir Sie bitten ein weiteres Fragebogen schriftlich zu beantworten:
        - Fragen zu ihrer Gemütslage nach dem Experiment
        - Fragen zum System und zu den 3 verschiedenen Text Darstellungen

Verabschiedung

Wir bedanken uns herzlichst, dass Sie an unserer Benutzerstudie und Systemexperiment teilgenommen haben!

Und wünschen Ihnen noch einen wunderschönen Tag!

Falls Sie Fragen haben stehen wir Ihnen selbstverständlich jederzeit zur Verfügung ☺

Sicherheitshinweise – Zustimmungserklärung

Der Echtzeit Untertitel Augmented Reality Setup (EUARS) ist ein Forschungsprototyp innerhalb einer Masterarbeit der Arbeitsgruppe der Interactive Media Systems an der TU Wien (IMS). Die Person, die diese Zustimmung unterschreibt, ist sich im Klaren darüber dass das System nicht ein zertifiziertes Produkt ist (zum Beispiel, laut TÜV Verordnung) und willigt ein auf eigenes Risiko und Verantwortung an dem Experiment teilzunehmen.

Die IMS und die TU Wien sind nicht verantwortlich für mögliche kurz- oder langfristige gesundheitliche Nachwirkungen der Benutzung des EUARS Systems.

Vor der Teilnahme am Experiment sind folgende Fragen, von der teilnehmenden Person zu beantworten:

- Leiden Sie an erhöhtem Blutdruck, haben Sie eine Herz- oder Kreislaufsystem Erkrankung?
- Leiden Sie oft an Übelkeit?
- Leiden Sie an epileptische Anfälle?
- Leiden Sie an Übelkeit verursacht durch 3D Fernseher?
- Leiden Sie an Panikattacken?
- Sind Sie unter Alkohol-, Drogen- oder Medikamenteneinfluss?
- Leiden Sie an einer anderen wichtigen Erkrankung von der IMS wissen sollte?

Falls sie eine oder mehrere Fragen mit "Ja" beantwortet haben, ist die Teilnahme am Experiment nicht erlaubt! Mit der Unterschrift bestätigt die teilnehmende Person die Richtigkeit der abgegebenen Aussagen.

Während der Benutzung des EUARS Systems sind folgende von IMS spezifizierten Benutzungsrichtlinien einzuhalten:

1. Folgen Sie den Anweisungen des Veranstalters in allen Situationen.

2. Falls Übelkeit, Panik oder andere ähnliche Reaktionen auftreten - sofort dem Veranstalter/ Experimentator Bescheid geben.

3. Nachdem Sie am Experiment teilgenommen haben, ist es nicht empfohlen, dass sie ein Auto, Fahrrad fahren, andere Maschinen betätigen oder sich in physisch anstrengende Aktivitäten

beteiligen welche ernsthafte Folgen haben könnten. Daher ist eine Pause von mindestens einer Stunde zwischen der Teilnahme am Experiment und einer solchen Aktivität notwendig.

4. Falls Sie jedwelche Nachwirkungen später bemerken, bitte verschieben Sie die oberhalb anstrengenden erwähnten Tätigkeiten.

Erlauben Sie IMS Sie zu filmen/fotografieren während dieses Experiments und das gefilmte Material für wissenschaftliche Zwecke zu benutzen?	☐ Ja ☐ Nein
Erlauben Sie IMS die aufgenommenen Materialen von Ihnen für die Präsentation der Resultate des Forschungsprojektes in der Öffentlichkeit zu benutzen? (wissenschaftliche Papers, Präsentationen, Pressemitteilungen und andere Publikationen).	☐ Ja ☐ Nein

Mit der Unterschrift, stimmt die teilnehmende Person den Verhaltensrichtlinien zu und übernimmt die Haftung.

_____
____

Ort, Datum Name und Vorname in Blockbuchstaben Unterschrift

SIMULATOR SICKNESS (PRE-) QUESTIONNAIRE

Bitte geben Sie an, wie stark jedes Symptom in diesem Moment auf Sie zutrifft.

Symptom	Nein	Leicht	Moderat	Stark
1. Generelles Unbehangen				
2. Erschöpfung				
3. Kopfschmerzen				
4. Augenbelastung				
5. Schwierigkeiten zu fokussieren				
6. Zunehmender Speichelfluss				
7. Schwitzen				
8. Übelkeit				
9. Schwierigkeiten zu konzentrieren				
10. « Voller Kopf »				
11. Verschwommene Wahrnehmung				
12. Schwindelgefühl bei offenen Augen				
13. Schwindelgefühl bei geschlossenen Augen				
14. Schwindel in Bezug auf aufrechte Haltung				
15. Den Magen bewusst/unangenehm wahrnehmen				
16. Rülpsen				

STOP HERE PLEASE!

Fragebogen 1

1. Sie sind:

   ◯ Mann

   ◯ Frau

2. Ihr Alter:

   ◯ zwischen 13 – 18

   ◯ zwischen 19 – 25

   ◯ zwischen 26 – 30

   ◯ zwischen 31 – 40

   ◯ zwischen 41 – 50

   ◯ zwischen 51 – 60

   ◯ zwischen 61 – 70

   ◯ zwischen 71 - 80

   ◯ über 81

3. Können Sie Lippen lesen?

   Nein ☐ ☐ ☐ ☐ ☐ Ja

4. Können Sie die Gebärdensprache?

   Nein ☐ ☐ ☐ ☐ ☐ Ja

5. Würden Sie sich als gehörlos oder schwerhörig bezeichnen?

   ◯ Gehörlos

   ◯ Schwerhörig

6. Ihre Erfahrung mit Technik/Computer:

①	②	③	④	⑤
Gar nicht	Selten	Mäßig	Ich arbeite oft am PC	Ich bin ein Profi

7. Ihre Erfahrungen mit Virtual Reality / Augmented Reality?

1	2	3	4	5
Keine	1 mal	Mäßig	Oft	Viel Erfahrung

8. Tragen Sie eine Brille?

- ◯ Ja
- ◯ Nein
- ◯ Manchmal

9. Haben Sie schon einmal ein Speech to Text (Sprache zu Text) System ausprobiert?

- ◯ Ja
- ◯ Nein

10. Sind Sie neugierig, möchten Neues ausprobieren?

1	2	3	4	5
Gar nicht	Selten	Mäßig	Oft	Ja sehr, bin immer offen für Neues

11. Finden Sie gehörlose/schwerhörige Leute bräuchten mehr Unterstützung (von der Technik) um an Meetings/Treffen teilnehmen zu können?

1	2	3	4	5
Nein	Ein wenig	So und so	Schon	Ja, auf jeden Fall

12. Freuen Sie sich auf das Experiment?

1	2	3	4	5
Nein	Ein wenig	Es geht	Schon	Ja sehr

Dialoge:

Dialog 1:

A: Hallo!
A: Wie geht’s?

B: Danke, gut.
B: Lange nicht gesehen.
B: Wie war dein Urlaub?

A: Es war sehr schön.
A: Wir waren in Ägypten.

B: War es heiß?

A: Es war sehr heiß und trocken.
A: Aber die Pyramiden waren toll.

B: Es freut mich zu hören.

Dialog 2:

A: Es war sehr entspannend.
A: Und wie war dein Urlaub?

B: Es war auch sehr schön.
B: Wir waren klettern in Kroatien.
B: Da gibt es einen Kletterpark.

A: Hat es dir gut gefallen?

B: Ja
B: Kann ich empfehlen.

A: Danke für die Infos.
A: Bis bald!

Dialog 3:

A: Ich liebe Käsekuchen.
A: Was ist dein Lieblingsessen?

B: Ich liebe Apfelstrudel.
B: Palatschinken sind auch lecker.

A: Auf jeden Fall.
A: Kannst du programmieren?

B: Ja, ich habe Informatik studiert.
B: Und du?

A: Maschinenbau.

B: Klingt kompliziert.
B: Wünsche noch einen schönen Tag.

SIMULATOR SICKNESS (POST-) QUESTIONNAIRE

Bitte geben Sie an, wie stark jedes Symptom in diesem Moment auf Sie zutrifft.

Symptom	Nein	Leicht	Moderat	Stark
1. Generelles Unbehangen				
2. Erschöpfung				
3. Kopfschmerzen				
4. Augenbelastung				
5. Schwierigkeiten zu fokussieren				
6. Zunehmender Speichelfluss				
7. Schwitzen				
8. Übelkeit				
9. Schwierigkeiten zu konzentrieren				
10. « Voller Kopf »				
11. Verschwommene Wahrnehmung				
12. Schwindelgefühl bei offenen Augen				
13. Schwindelgefühl bei geschlossenen Augen				
14. Schwindel in Bezug auf aufrechte Haltung				
15. Den Magen bewusst/unangenehm wahrnehmen				
16. Rülpsen				

STOP HERE PLEASE!

Fragebogen 2

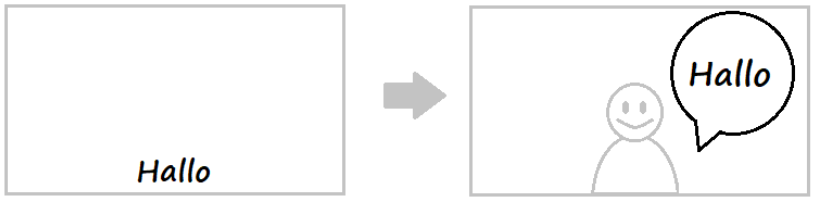
1. Hatten Sie genug Zeit die Texte zu lesen?

   Nein ☐ ☐ ☐ ☐ ☐ Ja

2. Haben Sie verstanden worum es in den Gesprächen ging?

   Nein ☐ ☐ ☐ ☐ ☐ Ja

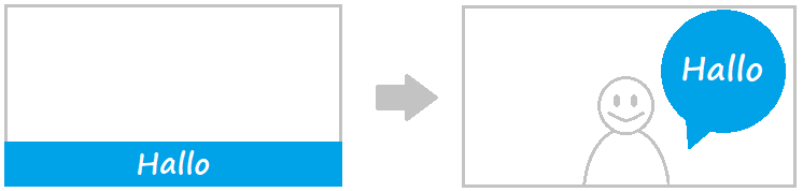
3. Wie sehr hat Ihnen die Darstellung: *Weiße Untertitel und, beim Auftritt einer Person im Sichtfeld, weiße Sprachblase* gefallen?



   Nicht gefallen ☐ ☐ ☐ ☐ ☐ Sehr gut gefallen

   Bitte begründen:

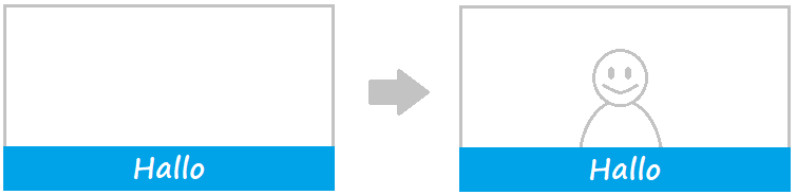
4. Wie sehr hat Ihnen die Darstellung: *Blaue Untertitel und, beim Auftritt einer Person im Sichtfeld, blaue Sprachblase* gefallen?



   Nicht gefallen ☐ ☐ ☐ ☐ ☐ Sehr gut gefallen

   Bitte begründen:

5. Wie sehr hat Ihnen die Darstellung: *Nur die blauen Untertitel, unabhängig von Person im Sichtfeld* gefallen?



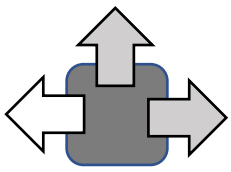
Nicht gefallen ☐ ☐ ☐ ☐ ☐ Sehr gut gefallen

Bitte begründen:

6. War die Schrift gut lesbar für Sie (groß genug, guter Kontrast)?

Nein ☐ ☐ ☐ ☐ ☐ Ja

7. Hat Ihnen die Visualisierung mit den Pfeilen geholfen zu wissen welche Person gerade spricht?



- ○ Ja, wenn ich weggeschaut habe (beispielsweise wenn ich geschrieben habe) und jemand anfing zu reden, machte mich diese Visualisierung aufmerksam darauf, in welcher Richtung der Sprecher sich befand.
- ○ Nein, ich habe sie nicht bemerkt/sie hat mir nicht viel geholfen.

Wie sehr hat Ihnen diese Anzeige gefallen?

Nicht gefallen ☐ ☐ ☐ ☐ ☐ Sehr gut gefallen

8. Finden Sie, dass das System benutzerfreundlich/einfach zu benutzen ist?

1	2	3	4	5
Nein, es ist sehr schwer zu benutzen	Relativ schwer	Es geht	Nicht so schwer	Ja, einfach Brille aufsetzen und auf Untertitel warten

9. Fanden Sie die Kombination **Untertitel und Sprachblase** (sobald die sprechende Person im Blickfeld ist) **besser als** die Visualisierung mit **nur Untertitel**, egal ob Personen im Blickfeld sind oder nicht?

- ◯ Ja, dann ist klarer welche Person gerade spricht
- ◯ Ja, ist sympathischer und man fühlt sich wie in einem Comicheft
- ◯ Nein, nur Untertitel ist besser, weil:

10. Finden Sie es hilfreich, dass die Visualisierung von Sprachblase zu Untertitel wechselt, wenn der Sprecher nicht im Blickfeld ist?

- ◯ Ja, so kann man weiterhin mitbekommen was gesprochen wird und gleichzeitig etwas anderes machen (zB. Notizen nehmen).
- ◯ Nein, weil:

11. Können Sie sich vorstellen so ein System in der **Zukunft** zu **benutzen**? (Natürlich angenommen die Brille wird <u>unauffälliger</u> und die Spracherkennung <u>besser</u> in den nächsten Jahren)

1	2	3	4	5
Nein, niemals	Unwahr-scheinlich	Vielleicht	Ich glaube schon	Ja, mit Sicher-heit, solche Syste-me werden sich früher oder später durchsetzen

12. Was wären andere **Anwendungsszenarien**, wo Sie glauben, dass ein solches System in der Zukunft das Leben vereinfachen würde?

(Sie können gerne mehrere Antworten ankreuzen ☺)

- ☐ In der Arbeit (Meetings, Calls)
- ☐ Konferenzen, Telekonferenzen
- ☐ Theater, Kino, Fernseher
- ☐ Gesprächen mit Freunden, Familie
- ☐ Telefongespräche, Skype
- ☐ Einkaufen

Bitte andere Anwendungsbereiche aufzählen, die Ihnen einfallen:

13. Hat es Ihnen Spaß gemacht das System zu testen?

- ◯ Ja
- ◯ Nein

Bitte begründen:

14. Hätten Sie Verbesserungsvorschläge für uns, wie wir das System noch weiter verbessern könnten?

15. Welche 5 Begriffe treffen aus Ihrer Sicht am ehesten auf das System zu?

- ☐ anpassbar
- ☐ auf den neusten Stand
- ☐ aufregend
- ☐ außergewöhnlich
- ☐ bedeutsam
- ☐ beeindruckend
- ☐ befriedigend
- ☐ effizient
- ☐ einfach zu verwenden
- ☐ einschüchternd
- ☐ erwartungsgemäß
- ☐ fortgeschritten
- ☐ frustrierend
- ☐ high-tech
- ☐ hilfreich
- ☐ hochwertig
- ☐ innovativ
- ☐ komplex
- ☐ kreativ
- ☐ langsam
- ☐ langweilig
- ☐ lustig
- ☐ mächtig
- ☐ motivierend
- ☐ neuartig
- ☐ nicht sicher
- ☐ nicht wünschenswert
- ☐ nützlich
- ☐ optimistisch
- ☐ praktisch
- ☐ professionell
- ☐ revolutionär
- ☐ sauber
- ☐ schnell
- ☐ schwer
- ☐ schwierig zu benutzen
- ☐ sicher
- ☐ stabil
- ☐ störend
- ☐ stressig
- ☐ überwältigend
- ☐ überzeugend
- ☐ unfein
- ☐ unkonventionell
- ☐ unpraktisch
- ☐ unterhaltsam
- ☐ verlässlich
- ☐ verwendbar
- ☐ wertvoll
- ☐ zeitsparend
- ☐ zu technisch

Vielen vielen Dank

dass Sie an unserer

Benutzerstudie

teilgenommen haben!

☺

Bibliography

[20118a] Yifei Yin 2017. *Communication between Arduino and Unity, Tutorial*, 2018 (accessed July 1, 2018).

[20118b] Intel Corporation 2018. *Intel RealSense Technology, Intel Software Developer Zone*, 2018 (accessed June 28, 2018).

[20118c] Microsoft 2018. *Microsoft Hololens, Mixed Reality: your world is the canvas*, 2018 (accessed June 28, 2018).

[20118d] Zygomatic 2018. *Word Clouds, Generator*, 2018 (accessed July 13, 2018).

[20118e] Inc. 2018 GitHub. *Github, Microsoft, Mixed Reality Toolkit for Unity*, 2018 (accessed June 28, 2018).

[AIuA+16] Mateen Ahmed, Mujtaba Idrees, Zain ul Abideen, Rafia Mumtaz, and Sana Khalique. Deaf talk using 3d animated sign language: A sign language interpreter using microsoft's kinect v2. In *SAI Computing Conference (SAI), 2016*, pages 330–335. IEEE, 2016.

[Ard15] SA Arduino. Arduino. *Arduino LLC*, 2015.

[Ard18a] 2018 Arduino. *Arduino Reference, pinMode*, 2018 (accessed July 1, 2018).

[Ard18b] 2018 Arduino. *Arduino Tutorials, button*, 2018 (accessed July 1, 2018).

[Ard18c] 2018 Arduino. *Arduino, Software, Download the Arduino IDE*, 2018 (accessed June 28, 2018).

[ARg] Reality technologies, augmented reailty, guide navigation, graphic. `www.realitytechnologies.com/augmented-reality`. Accessed: 2018-06-27.

[ARL16] N. Anggraini, N. F. Rozy, and R. A. Lazuardy. Facial recognition system for fatigue detection using intel realsense technology. In *2016 6th International Conference on Information and Communication Technology for The Muslim World (ICT4M)*, pages 248–253, Nov 2016.

[Cor18a] Intel Corporation. *Intel Software, Developer Zone, Intel RealSense SDK for Windows*, 2018 (accessed June 28, 2018).

[Cor18b] Intel Corporation. *Intel, Support, System Requirements for the Intel® RealSense™ Camera SR300*, 2018 (accessed June 28, 2018).

[Cor18c] Intel Corporation. *Intel Software, Developer Zone, Intel RealSense SDK for Windows, Face and Head Tracking using the Intel RealSense SDK*, 2018 (accessed June 30, 2018).

[DASD17] I. Dabran, T. Avny, E. Singher, and H. Ben Danan. Augmented reality speech recognition for the hearing impaired. In *2017 IEEE International Conference on Microwaves, Antennas, Communications and Electronic Systems (COMCAS)*, pages 1–4, Nov 2017.

[DFG+14] Saverio Debernardis, Michele Fiorentino, Michele Gattullo, Giuseppe Monno, and Antonio Emmanuele Uva. Text readability in head-worn displays: Color and style optimization in video versus optical see-through devices. *IEEE transactions on visualization and computer graphics*, 20(1):125–139, 2014.

[GBD+16] Mathieu Garon, Pierre-Olivier Boulet, Jean-Philippe Doironz, Luc Beaulieu, and Jean-François Lalonde. Real-time high resolution 3d data on the hololens. In *Mixed and Augmented Reality (ISMAR-Adjunct), 2016 IEEE International Symposium on*, pages 189–191. IEEE, 2016.

[Inc18a] [2011-2017] Elegoo Inc. *ELEGOO, ELEGOO UNO R3 Project Complete Starter Kit*, 2018 (accessed June 28, 2018).

[Inc18b] 2011-2018 PTC Inc. *Vuforia Developer Portal*, 2018 (accessed July 4, 2018).

[Inc18c] 2011-2018 PTC Inc. *Vuforia, Developer Portal, Home*, 2018 (accessed June 28, 2018).

[Inc18d] 2016 PTC Inc. *Vuforia Developer Portal, Sample Image Targets*, 2018 (accessed July 4, 2018).

[JFG+15] Dhruv Jain, Leah Findlater, Jamie Gilkeson, Benjamin Holland, Ramani Duraiswami, Dmitry Zotkin, Christian Vogler, and Jon E Froehlich. Head-mounted display visualizations to support sound awareness for the deaf and hard of hearing. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 241–250. ACM, 2015.

[KSZ+17] T. Kurahashi, K. Suemitsu, K. Zempo, K. Mizutani, and N. Wakatsuki. Disposition of captioning interface using see-through head-mounted display for conversation support. In *2017 IEEE 6th Global Conference on Consumer Electronics (GCCE)*, pages 1–4, Oct 2017.

[MGM12] M Mirzaei, S Ghorshi, and M Mortazavi. Helping deaf and hard-of-hearing people by combining augmented reality and speech technologies. In *Proc. 9th Intl Conf. Disability, Virtual Reality & Associated Technologies*, pages 149–158, 2012.

[Mic18] 2018 Microsoft. *Microsoft Azure, Cognitive Services*, 2018 (accessed August 11, 2018).

[NA16] Ashish S Nikam and Aarti G Ambekar. Bilingual sign recognition using image based hand gesture technique for hearing and speech impaired people. In *Computing Communication Control and automation (ICCUBEA), 2016 International Conference on*, pages 1–6. IEEE, 2016.

[NBB+15] Jan Nouza, Karel Blavka, Marek Boháč, Petr Červa, and Jiří Málek. System for producing subtitles to internet audio-visual documents. In *Telecommunications and Signal Processing (TSP), 2015 38th International Conference on*, pages 1–5. IEEE, 2015.

[NRBM00] W Todd Nelson, Merry M Roe, Robert S Bolia, and Rebecca M Morley. Assessing simulator sickness in a see-through hmd: Effects of time delay, time on task, and task complexity. Technical report, AIR FORCE RESEARCH LAB WRIGHT-PATTERSON AFB OH, 2000.

[ODS06] J. Ohene-Djan and R. Shipsey. E- subtitles: Emotional subtitles as a technology to assist the deaf and hearing-impaired when learning from television and film. In *Sixth IEEE International Conference on Advanced Learning Technologies (ICALT'06)*, pages 464–466, July 2006.

[oFf18] 2018 Friends of Fritzing foundation. *Fritzing App, electronics made easy*, 2018 (accessed September 9, 2018).

[OKT13] Jason Orlosky, Kiyoshi Kiyokawa, and Haruo Takemura. Dynamic text management for see-through wearable and heads-up display systems. In *Proceedings of the 2013 International Conference on Intelligent User Interfaces*, IUI '13, pages 363–370, New York, NY, USA, 2013. ACM.

[PB16] J. V. Patil and P. Bailke. Real time facial expression recognition using realsense camera and ann. In *2016 International Conference on Inventive Computation Technologies (ICICT)*, volume 2, pages 1–6, Aug 2016.

[RFD+16] Ricardo Sousa Rocha, Pedro Ferreira, Inês Dutra, Ricardo Correia, Rogerio Salvini, and Elizabeth Burnside. A speech-to-text interface for mammoclass. In *Computer-Based Medical Systems (CBMS), 2016 IEEE 29th International Symposium on*, pages 1–6. IEEE, 2016.

[RRD05] Edward Rosten, Gerhard Reitmayr, and Tom Drummond. Real-time video annotations for augmented reality. In George Bebis, Richard Boyle, Darko

Koracin, and Bahram Parvin, editors, *Advances in Visual Computing*, pages 294–302, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[S2T] Telestream, speech to text, graphic. `http://www.telestream.net/company/press/2017-08-22-TimedTextSpeech.htm`. Accessed: 2018-06-27.

[SB17] Chris Schipper and Bo Brinkman. Caption placement on an augmented reality head worn device. In *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '17, pages 365–366, New York, NY, USA, 2017. ACM.

[Squ18] 2018 Font Squirrel. *Font Squirrel, Free Font Utopia*, 2018 (accessed July 6, 2018).

[SS16] Neha Sharma and Shipra Sardana. A real time speech to text conversion system using bidirectional kalman filter in matlab. In *Advances in Computing, Communications and Informatics (ICACCI), 2016 International Conference on*, pages 2353–2357. IEEE, 2016.

[Tec18a] 2018 Unity Technologies. *Unity Documentation, Command Attribute*, 2018 (accessed July 3, 2018).

[Tec18b] 2018 Unity Technologies. *Unity Documentation, Platform Depedent Compilation*, 2018 (accessed July 3, 2018).

[Tec18c] 2018 Unity Technologies. *Unity3D, official website*, 2018 (accessed June 28, 2018).

[THM+15] Yusuke Toba, Hiroyasu Horiuchi, Shinsuke Matsumoto, Sachio Saiki, Masahide Nakamura, Tomohito Uchino, Tomohiro Yokoyama, and Yasuhiro Takebayashi. Considering multi-modal speech visualization for deaf and hard of hearing people. In *Information and Telecommunication Technologies (APSITT), 2015 10th Asia-Pacific Symposium on*, pages 1–3. IEEE, 2015.

[VK15] BV Vishakh and Mohammed Kamal Khwaja. Wearable device for hearing impaired individuals using zigbee protocol. In *Modelling Symposium (AMS), 2015 9th Asia*, pages 181–184. IEEE, 2015.

[ZCPR03] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Comput. Surv.*, 35(4):399–458, December 2003.

[Zha00] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, Nov 2000.