Westfälische Hochschule

University of Applied Sciences
Gelsenkirchen Bocholt Recklinghausen

Masterarbeit

Titel der Arbeit // Title of Thesis

Visual Cues: Integration of object pose recognition with an augmented reality system as means to support visual perception in human-robot control

---

Akademischer Abschlussgrad: Grad, Fachrichtung (Abkürzung) // Degree

Master of Science (M. Sc.)

Autorenname, Geburtsort // Name, Place of Birth

Tim Dierks, Bremen

Studiengang // Course of Study

Medieninformatik

Fachbereich // Department

Informatik und Kommunikation

Erstprüferin/Erstprüfer // First Examiner

Prof. Dr. Jens Gerken

Zweitprüferin/Zweitprüfer // Second Examiner

Prof. Dr. Gregor Lux

Abgabedatum // Date of Submission

04.06.2020

Eidesstattliche Versicherung

Dierks, Tim

Name, Vorname // Name, First Name

Ich versichere hiermit an Eides statt, dass ich die vorliegende Abschlussarbeit mit dem Titel

Visual Cues: Integration of object pose recognition with an augmented reality system as means to support visual perception in human-robot control

selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Ort, Datum, Unterschrift // Place, Date, Signature

Abstract

Autonomy and self-determination are fundamental aspects of living in our society. Supporting people for whom this freedom is limited due to physical impairments is the fundamental goal of this thesis. Especially for people who are paralyzed, even working at a desk job is often not feasible. Therefore, in this thesis a prototype of a robot assembly workstation was constructed that utilizes a modern **A**ugmented **R**eality (AR)-**H**ead-**M**ounted **D**isplay (HMD) to control a robotic arm. Through the use of object pose recognition, the objects in the working environment are detected and this information is used to display different visual cues at the robotic arm or in its vicinity. Providing the users with additional depth information and helping them determine object relations, which are often not easily discernible from a fixed perspective.

To achieve this a hands-free AR-based robot-control scheme was developed, which uses speech and head-movement for interaction. Additionally, multiple *advanced visual cues* were designed that utilize object pose detection for spatial-visual support. The pose recognition system is adapted from state-of-the-art research in computer vision to allow the detection of arbitrary objects with no regard for texture or shape.

Two evaluations were performed, a small user study that excluded the object recognition, which confirms the general usability of the system and gives an impression on its performance. The participants were able to perform difficult pick and place tasks with a high success rate. Secondly, a technical evaluation of the object recognition system was conducted, which revealed an adequate prediction precision, but is too unreliable for real-world scenarios as the prediction quality is highly variable and depends on object orientations and occlusion.

Contents

1 Introduction

Autonomy and self-determination is natural for most people, but for physically impaired this freedom can be severely limited. Not being able to perform simple tasks on your own can be frustrating and can make you feel like a burden to others. Even performing a normal job may be impossible without any help. Because of this, research is done in various directions to support people with disabilities and help them to be more autonomous. This project aims to support anyone who can not use their body freely, like tetraplegics, to use a robotic arm for pick and place tasks. It is part of the Human-Robot Interaction at the Workplace (**M**ensch-Roboter **I**nteraktion im **A**rbeitsleben bewegungseingeschränkter Personen) (MIA) project (1.1). We specifically try to enhance the perception of the scene with which users want to interact and give visual cues on alignment and orientation of objects, their relation to one another, occlusion, and more that may be difficult to see from a fixed perspective. In order to achieve this, it was necessary to set up a robotic arm and develop an application that uses hands-free input modalities like speech and head movement to control it. This basis was developed in collaboration with Franziska Rücker as the first part of our master theses. While she will focus in her thesis on visual cues that do not need knowledge of the scene, I come from the other end of the spectrum with full knowledge of the environment through the use of object pose recognition, which enables more sophisticated visual cues. To display and integrate visual information in 3D space we use AR, which is perfectly suited for this task. Especially new AR-glasses integrate many technologies, like speech recognition and different cameras for perception of the environment, which help our project in different ways.

1.1 The MIA Project

The MIA project is a collaborative four-year research project of the Westphalian University of Applied Sciences, the Institute of Automation at the University of Bremen, and the Interactive Systems Group at University Duisburg-Essen, as well as a few industrial partners listed here (Gerken, Jens, 2017). The goal is to use state-of-the-art sensor technology to build workplace environments for the new upcoming *Industry as a Service*. This includes finding novel ways of communication and interaction while the hands of workers are occupied, which extends to people with disabilities. The utilized technologies such as **I**nertial **M**easurement **U**nit (IMU), eye tracking, or **E**lectro**o**culo**g**raphy (EOG), as well as giving feedback through augmented reality, are part of the project proposal. MIA enables two doctorate degrees and a few bachelor and master theses, which this is a part of. Stephanie Arévalo is one of the Ph.D. students and guides this project as an informal supervisor. Additionally, this project is based on, and in support of her research.

1.2 Motivation, context and objective

Having full mental capabilities but being confined to passivity through severe physical limitations is an undesirable situation to be in. Additionally, because of the limited ability of interaction with the world, it is easy to lose one's sense of purpose and feel a disconnect from others and the rest of society as a whole. For many people, their work is a crucial cornerstone of feeling important and needed in life, which is often not possible to have

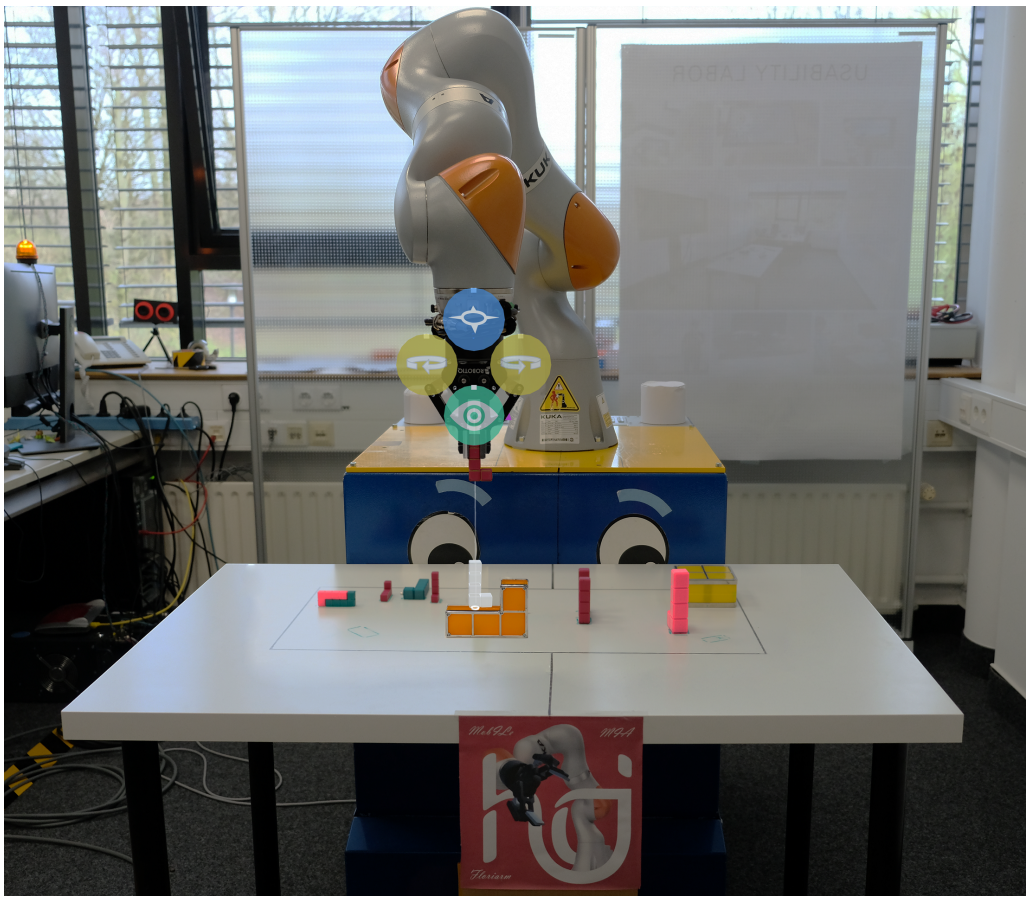


Figure 1: Example of the advanced visual cue *Ghost Object*. When the gripper has grasped an object a copy will be displayed on the surface directly beneath it.

with severe disabilities as shown in the short video from Hannappel, Philipp (Hannappel, Philipp, 2015). New possibilities are created with novel and more advanced technologies. Extending the range of interaction and opening the possibilities of working, for example at an assembly style workplace, are the key motivations for this work. The baseline for working with a robotic arm is to pick and place objects precisely and over an extended period of time. As previously mentioned, with restricted movement it may not always be possible to have an adequate overview of the scene, which then requires higher concentration and creates more cognitive strain. Especially the inability to see object relations and occlusion of objects makes the task harder. To be able to provide visual cues that help with such problems, information about the scene, specifically the objects, is necessary. Figure 1 shows an example of a 3D-projection of the picked object shown on top of the surface it would be placed at that moment. This reduces the need to theorize where exactly the object will be placed. Knowledge about the precise orientation and position of the object is required to show such a projection. Acquiring these necessitates robust and precise object pose recognition. Secondly, visualizing this projection in 3D space is another requirement which is perfectly suited for AR, especially modern AR-glasses like the *Microsoft HoloLens* or the *Magic Leap One*. These are hands-free and create a rough 3D-model of the environment for various kinds of interactions.

One crucial aspect is the fact that, with *perfect* knowledge of the environment in which these tasks are performed, it would be possible to automate the whole process and leave

the human out completely. Important to note is that this thesis is focused on supporting humans in need and not building a highly sophisticated automated system that can perform exceedingly difficult object-manipulation tasks totally autonomously. Additionally, the fact that humans are capable to perform mentally complex tasks and have higher-level planning skills is a valuable benefit, which can be leveraged for highly variable or unpredictable environments. These are not easily replaced with information about the environment. This is further explained in section 2.2.2.

The core of this thesis is the enhancement of visual cues, called *advanced visual cues*, through robust 6D-pose recognition of objects in the working area of the robotic arm. The idea is to employ and adapt a state-of-the-art pose recognition method and integrate it into an AR-based application to control a robotic arm. Allowing the user to perform pick and place tasks without moving around, through the support of their visual perception with advanced visual cues.
The resulting contributions are:

- An AR-based robot control scheme
- Support of visual perception through AR- and pose recognition
- General pose recognition for/with an AR-HMD

1.3 Structure

Following this introduction, the thesis will deal with the basics of robotic arms and introduce the required concepts for the rest of the thesis in chapter 2.1. Additionally, it gives a broad overview of the scientific work that was already done with robotics supporting people with disabilities (2.2), going into different input modalities (2.2.1) and the importance of trust and autonomy (2.2.2). The focus here will be specifically on literature without the use of AR, which will be discussed in the third chapter.
Chapter 3.1 will first define the term AR, **V**irtual **R**eality (VR) and **M**ixed **R**eality (MR) and discuss their relationship to one another, as well as the confusion surrounding them. After that it will briefly introduce modern AR technologies and discuss their respective advantages and disadvantages in chapter 3.2 and 3.3. Followed by a literature review about enhancement of robotic systems with AR in different scenarios (3.4).
Object and pose recognition will be discussed in chapter 4. Beginning with a definition of the different terms and their relevance for research in 4.1, followed by dealing with the question of what the current difficulties in the field are (4.2). 4.3 will deal with the three main approaches which dominated the field at different points in time and discuss their respective advantages and disadvantages. Lastly, 4.4 is a literature review about bridging the gap of the three research areas. Robotics and object-pose recognition in 4.4.1, AR and object-pose recognition in 4.4.2 and at the end a combination of all three (4.4.3).
Beginning with chapter 5 the practical parts of this thesis will be described. This chapter discusses the taken decisions for all aspects of the constructed prototype based on the research and initial constraints. Chapter 6 describes the whole system beginning with a

general overview (6.1), the intended workflow (6.2) and the kind of interactions that where utilized (6.3). Following this, the three main parts of the prototype are introduced: the control scheme (6.4), the advanced visual cues (6.5) and finally the recognition system (6.6). Chapter 7 deals with the evaluation of the prototype. Two separate evaluations were performed, a small user study which is discussed in section 7.1 and a technical evaluation in 7.2. Concluding the evaluation with a discussion of the results in the context of the whole system (7.3). The last chapter is the conclusion (8) in which the system is shortly summarized and reflected upon and finally, possible future work on this project will be discussed in 8.1.

2 The robotic arm

There are several different ways to give a disabled person more autonomy in their personal life and workplace like a motorized wheelchair, but enabling physical interaction is dependent on a robotic device. May it be in humanoid form or other, it will most likely have some form of arm with a gripper (or other tools) to interact with the world. Even controlling a standalone robotic arm, fixed to a stationary mount, is already a difficult task. The cause of this difficulty and how a robotic arm works in general, will be discussed in the following chapter. Followed by a review of advancements to help physically impaired with robotics.

2.1 What exactly is a robotic arm?

Robotic arms are multifunctional mechanical devices originally developed for industrial use and highly hazardous or repetitive tasks. They are inspired by the versatility of actual human arms and are comprised of several joints for freedom of movement and a hand-like gripper or a tool for use in different tasks. In addition, they are programmable by an operator to adapt them for varying scenarios. Garcia, Jimenez, De Santos, and Armada (2007) describes the robotic arm as the biggest workforce in modern production lines which enables the almost unlimited mass production that started during the 1960s in the automotive industry. Garcia et al. additionally name their versatility as one reason why robots became a popular field of research for **H**uman-**C**omputer **I**nteraction (HCI) and sparked the field of **H**uman-**R**obot **I**nteraction (HRI) (Garcia et al., 2007).

2.1.1 Composition and movement

A robotic arm is a series of links and joints connected to one another, also called a kinematic chain. Links are rigid connections between joints similar to bones in a human arm. The end of a robotic arm is called end-effector and has a gripper or tool attached to it, with which the arm can perform its designated task, as described by Saha (Saha, 2008). The center point of the end-effector attachment is called **T**ool **C**enter **P**oint (TCP), it is the point where the highest force is applied when the arm is working. Knowledge of how strong the force at this point is, is important for the regulation of pressure applied to the work-piece. Joints are usually rotational in one axis and have limits on how far they can rotate. The joint motion is produced by a motor that can be powered through hydraulic, pneumatic, or electric energy. Electric step motors are used most often for smaller sized robotic arms because they ensure high precision and can easily be controlled programmatically (Harris Tom, 2002).

For each joint that a robotic arm features, it gains one **D**egree **O**f **F**reedom (DoF). Most robotic arms have three to six DoF. Six DoF are already enough to reach every position and orientation in 3D space with the end-effector (within the arms reach). When a kinematic chain has more DoF than necessary it is called redundant, because in this case it has many different poses in which it can reach a specific position and orientation in 3D space. The term redundant is relative to the work space of the robotic arm, which means for work on a 2D plane it only needs 3 DoF to be redundant and for reaching a point in 3D space (rotation excluded) 4 DoF would be enough. Shamir (1990) describes the robotic arm as

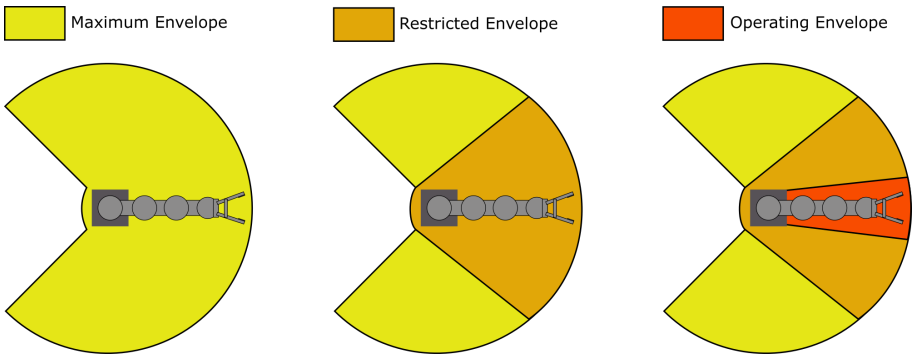


Figure 2: Safety envelopes of a standard robotic arm.

redundant when $n > m$, where n = number of dimensions the robotic arm moves in and m = number of workspace dimensions. The human arm, for example, has seven DoF, which makes it redundant and gives it a very high maneuverability (Shamir, 1990).
The versatility gained by a high number of DoF comes with a caveat. Control of the robotic arm quickly becomes very complex and hard to perform for a human operator. Generally, the movement paradigms can be divided into two categories, controlling the robot in joint space and Cartesian space (also called world space). The first is very simple and involves moving each joint of the robotic arm individually one after another. This has the advantage that the operator always knows how the arm will move, but its tediousness and slowness are the reason this movement control scheme is not often utilized. Cartesian space movement control, on the other hand, is quite complex but makes the control comparatively straight forward. The operator directly controls the end-effector of the robotic arm and can move/rotate it along each dimensional axis, while all other joints rotate accordingly by themselves to accommodate for the effector movement. Unexpected movement of the arm and the accompanied risk of collisions are the most prominent disadvantages of this scheme. It additionally has a high computational cost, which further increases with more DoF. The underlying method to calculate each DoF, dependent on their position in the kinematic chain, is called inverse kinematics. There are many other different algorithms and mathematical approaches with drastically varying computational effort, complexity, and precision (Aristidou & Lasenby, 2011; Buss, 2004; Tolani, Goswami, & Badler, 2000).

2.1.2 Safety concerns

Working with a robotic arm, especially an industrial one, requires attention to safety. Different from normal machines a robotic arm has the capabilities of powerful, fast, and broad movement that can be unexpected for the operator. Occupational Safety and Health Administration (2019) describes different kinds of *envelopes* which represent areas with varying safety concerns (Occupational Safety and Health Administration, 2019). In figure 2 the three zones are denoted as *Maximum Envelope* which represents the furthest reach of the arm, the *Restricted Envelope* which is the zone to which the arms movement is restricted to by software or limiting devices and the *Operating Envelope* is the area used by the active program and the robots intended area of motion. These *envelopes* should only be entered with appropriate measures, like cutting the power to the machine.

As robotic arms and robots in general spread more into the domain of assistive robotics for human needs, the safety aspect becomes a crucial concern as industrial arms are mainly teleoperated or automated and, as highlighted, need restrictive *envelopes* that need to be fenced off. The main approach to solving this problem is to give the robotic arm either awareness of collisions and let its motion stop in place, or awareness of its surroundings to avoid collisions overall. Much research has been done in either direction, for example, (Lumelsky & Cheung, 1993) created a skin for robotic arms that contains about 500 small infrared proximity sensors evenly distributed in a grid around all parts of the robotic arm, where a collision could occur. In real-time their system collects the sensor data, processes it, and calculated the size and direction of the next step. Their setting is for teleoperated systems, where the operator gives movement commands and the system tries to move as close to the given path without jeopardizing its safety, respectively avoiding collisions.
A different approach is the depth space collision avoidance system (Flacco, Kröger, De Luca, & Khatib, 2012) developed. With the use of a *Microsoft Kinect*, they can calculate the distance between the robotic arm and an obstacle. This method produces a repulsive vector (a vector in the opposite direction of the obstacle) to control movement and can adjust the joint positioning in case of a redundant arm, to still perform the given task.
A system that tries minimizing the force of a collision to conform to the ISO 10218-1 standard (ISO/TC 299 Robotics, 2011) for human-robot collaboration was developed by (Lauzier & Gosselin, 2011). They mechanically enhanced the safety of the robotic arm by placing an electronically adjustable torque limiter in between each actuator (joint). They present a method to calculate the optimal limit for each torque limiter to preserve as much force as possible, without compromising safety. If a collision occurs that exceeds the torque limits the arm will just stop automatically by triggering an emergency stop.
S.-D. Lee, Kim, and Song (2013) also approach the collision detection of a redundant robotic arm. They present a collision detection index calculated from the data of joint torque sensors build into each joint of the robotic arm. Remarkable is the ability to detect human body collisions even when force is simultaneously applied from the end-effector and the collision, which normally interferes with reliable detection (S.-D. Lee et al., 2013).

As shown by these examples, research feature many different approaches which may still have some disadvantages, like enhanced complexity and cost of the robot in the case of additional sensors or blind spots and high computational costs with vision-based collision detection. None the less they bring the field of research further along and possibly enabled *KUKA* to design the first series-produced **H**uman-**R**obot **C**ollaboration (HRC) compatible robotic arm, called *LBR iiwa*, that conforms to the aforementioned ISO standard (ISO/TC 299 Robotics, 2011) (KUKA-AG, 2018).

2.2 Research on how to support the physically impaired with robotics

Creating a robotic system that is capable of helping physically impaired is not an easy task. Research in a few crucial areas is still very important and several problems need to be addressed to produce a broadly usable system. Here I want to highlight two of them, which are the different input modalities and how much autonomy for a robotic arm is beneficial and what the implications of too much autonomy are.

2.2.1 Input modalities

One important aspect is the input modalities for the user controls. As aforementioned a robotic arm with only a few DoF is already difficult to control. In 3D-space even with inverse kinematics there are still six axes to handle (three for movement and three for rotation). In a situation of severe physical impairment, the standard method of input does not apply anymore. Hand input is used for almost everything and has great versatility, fast response times, and can be very precise. Only in recent years did other forms of input, like speech, eye-tracking, and head-tracking, start to become more usable. All these methods still have some major disadvantages, mostly caused by technological limitations. Following I will give a brief introduction to these three methods and discuss some of their remaining disadvantages and additionally give a glimpse into the kinds of research done for the niche field of robotics for physically impaired.

Speech input

Speech inputs' biggest problem is that the actual recognition of words is dependent on many factors like word similarity or accent of the speaker which make it unreliable. Additionally, background noise has a negative impact that can be alleviated to a certain degree by directional short-range microphones. Lv, Zhang, and Li (2008) for instance developed a speech-based control system for a small mobile robot that can be driven around and has a small arm that can be extended. Their system is based on pattern matching of single words. As they point out, for a small vocabulary their pattern matching approach is highly efficient and pretty robust to errors, which is shown by their test results which had a perfect score in a normal environment but got significantly worse with background noise. It fell from 100% accuracy to about 82%. An additional disadvantage is the vocabulary size which needs to be small in order to work reliably (Lv et al., 2008).

In their work another problem with speech input becomes apparent. The direct control as used by Lv et al. (2008) is not practical in a real scenario. An exemplary command they use is *go forward* but for how long or how much the robot moves is not clear, like pressing a button for the duration of the movement would be. This extends to controlling an end-effector which could not be easily controlled with simple voice commands, even when they are recognized perfectly. Judging distances or angles for rotation to add into speech commands (i.e. *rotate 23° to the left*) works only well for discrete, well-known values (90°, 180°, etc.). One approach would be to make the robot more autonomous and therefore not require such low-level commands. Only higher-level commands as *pick object* would then be needed and the robot figures out how to move

and rotate its end-effector to pick up the object by itself.
Kurnia, Hossain, Nakamura, and Kuno (2004) used simple object recognition as a basis for selecting an object with higher-level speech input. They developed a system in which the robot can detect all the objects in the scene and tries to deduce which one the user means when giving a command to for example *pick up the apple*. By asking about certain features the robot can narrow down remaining possibilities. This dialog that arises is similar to the *Akinator* [1] application which tries to identify a figure (real or fictional) you randomly pick at the beginning by asking yes or no questions. the system of Kurnia et al. (2004) is able to not only understand yes or no, but the features of objects like color, shape, position, or size (Kurnia et al., 2004).

Eye-tracking

Tracking the movement of the pupil and inferring where the user is looking based on that, or recognizing patterns in the movement is the basic goal of eye-tracking. It can imitate a normal computer mouse on a basic 2D screen or can be used to point at something in 3D-space. Eye-tracking most often suffers from imprecision due to calibration errors and occlusion of the pupil through eyelids and the resulting false positive pupil detections. Because of the technical difficulties, many researchers in the field of robotics who want to use the eye-gaze rather use an approximation of it, the head-gaze. Palinko, Rea, Sandini, and Sciutti (2016) performed a study to examine if the head-gaze can replace eye-gaze in natural interactions with a robot. They concealed to the subject how the robot interaction is tracked and found that with this uninformed subject the eye-gaze performs much better for selecting specific objects (Palinko et al., 2016). The reason for this is that humans do not always move their heads when looking at an object, especially when the points of interest are spatially close. Additionally, Palinko et al. (2016) point out that joint attention between two persons is established by looking at the point of interest and before and/or after into the eyes of the counterpart. This can enable more robust natural behavior of the robot if the system is aware of this attention establishment.
Despite the trend in HRI to replace eye-gaze with head-gaze, because of the remaining technical difficulties, there is still active research going on. For example Barea, Boquete, Bergasa, López, and Mazo (2003) utilize eye-tracking to control a wheelchair with a basic 2D user interface. They use electro-oculography to detect eye movement and employ an inverse eye model to determine the direction the user is looking. Based on the generated information Barea et al. (2003) developed different control methods. A direct input method in which the action buttons on the screen need to be dwelled on for a certain amount of time to activate, a sweep method where the specific eye movement (called *tick*) selects a command and a continuous method where the eye position, independent of the screen, is used for command activation (Barea et al., 2003). A few years earlier Yanco (1998) developed a similar, more basic, wheelchair control based on the same technology (Yanco, 1998).
In recent research, concerning the physically impaired and eye-tracking, Tanaka, Mu, and Nakashima (2014) constructed a meal-assistance robot that utilizes a special **U**ltra**s**onic **M**otor (USM) and eye-tracking for user-control. The robot is not a

[1] https://en.akinator.com/game

multi-joint robotic arm, instead it uses the USM for simple orthogonal movements of the spoon. A food-holding tray with tracks is used, in which the food is placed in bite-sized chunks that fit onto the spoon. The eye interface is laid out like a number pad where each number is a command, for example, *Select tray, Pushing, Send food, Return food, etc.* For selection, the user looks at one option to select it and blinks for confirmation. Because of the robots simplicity, it needs to be placed at a specific height and distance to the user (Tanaka et al., 2014). This and the blink are glaring disadvantages in the design of Tanaka et al. (2014), especially because blinking is not something that can be totally controlled. Many factors determine the ability of a human to blink or not blink reliably, like dryness of the eye for example.

Head-tracking

Head-tracking is the most robustly working of the three input modalities discussed here. In its simplest form, the rotational movement of the head is tracked by an **I**nertial **M**easurement **U**nit (IMU) attached to the head, for example with a strap. An IMU is a small electronic chip that contains an accelerometer, gyroscope, and sometimes a magnetometer which enable it to determine its angular acceleration. Ahmad, Ghazilla, Khairi, and Kasi (2013) reviewed the different IMU technologies in their paper and illustrate many common day-to-day applications for them (Ahmad et al., 2013). The biggest disadvantage of IMUs are their sensor inaccuracies and the resulting drift.

Baldi, Spagnoletti, Dragusanu, and Prattichizzo (2017) developed a wearable control interface for a robotic arm which utilizes a **M**agnetic, **A**ngular **R**ate, and **G**ravity (MARG) sensor, which is similar to an IMU, and three **E**lectro**m**yo**g**raphy (EMG) electrodes as user input. These sensors are integrated into a wearable cap. The EMG electrodes replace a button press by detecting the contraction of the frontalis muscle and the head roll and pitch tilt translate to continuous motions of the robotic arm. They use four modes that can be switched between. Two of which are for translational movement, where one is controlling motion in the horizontal plane and the other for up and down. The other two modes are similar, but for the rotation of the end-effector. With this interface they ran an experiment in which users had to perform two scenarios, one task was to fill a glass with water and the other was to fit an object into a hole. The times for task completion were about twice as long in compared to a normal joystick at about 120 seconds for scenario one and 100 seconds for scenario two, which are pretty good results for hands-free interactions (Baldi et al., 2017).

Another wheelchair system was developed by Tomari, Kobayashi, and Kuno (2012) which uses head-gaze for user input in combination with a simple button to switch modes, similar to Baldi et al. (2017) who used EMG electrodes as button replacements. Tomari et al. (2012) point out that other input methods could be used as button replacement, like voice commands or blinking. For the head-gaze tracking, they use a simple webcam image and the software *FaceAPI* [1] which is able to calculate the necessary gaze data in real-time, of which they only need the yaw angle. Their approach is to have a semi-autonomous wheelchair that uses the yaw angle as a goal direction and performs the lower-level movement tasks itself, leaving the user only having to intervene

[1] https://www.seeingmachines.com/

when the directions need to be changed. Besides the semi-autonomous mode, there is a manual mode, which is used for precise turning towards a specific direction. For autonomous movement the wheelchair is equipped with a *Microsoft Kinect*, a laser-sensor in addition to an IMU. Tomari et al. (2012) experiment is also similar to Baldi et al. (2017) in that they compare their input scheme with the performance of a joystick. Results also show that the joystick is twice as fast as their semi-autonomous and manual modes, but the semi-autonomous mode navigated to all target locations in the shortest overall distance (Tomari et al., 2012).

As shown in this section there are interesting approaches for these input modalities and many more that are not outlined here. Besides these three major modalities there are plenty of other ideas that try to employ even more experimental approaches like a brain interface (Finke, Knoblauch, Koesling, & Ritter, 2011) or tongue input (Saponas, Kelly, Parviz, & Tan, 2009), that may lead the future.
What most if not all of these input modalities have in common is the heightened workload for users and a limited degree of freedom they provide without much mode-shifting or similar techniques. Consequently, more autonomous robotic devices are often used like in (Choi, Anderson, Glass, & Kemp, 2008; Tomari et al., 2012). These replace the lower level input with high-level commands and reduce the users' workload. Now the question arises how much autonomy is too much and where does the users' trust come into play?

2.2.2 Autonomy and trust

Specifically, in research for physically impaired, the term *autonomy* is used quite often in two different contexts. One has to be aware of the varying subjects this term can refer to. The first is the autonomy of a robotic system, which includes its capability to perceive the world around it and adapt to this changing environment by itself. The more autonomous a system is, the more it can do by itself and does not need a human to guide it. The other subject is the physically impaired user of the system and the goal to provide him with technology to increase his autonomy and independence. Important to note is that just maximizing the users' autonomy is not always desired. Most research is not only focused on replacing caretakers with new technology, rather to increase the impaired persons' well-being and overall satisfaction by enabling them to take care of themselves.
Unintuitively there is no direct correlation between raising the autonomy of a robotic system and resulting generated autonomy for users. Indeed, a system that can act on its own is potentially more useful for a tetraplegics who can not drink by themselves or get a glass from a cupboard. Now it may already be seen as just another external helper that is relied upon and not an extension of the user that enables him/her to do the task by themselves. D.-J. Kim et al. (2011) made an exploratory study in which they tried to find a correlation between task completion and satisfaction for spinal cord injured subjects. One group used an autonomous robotic arm to pick and place objects from a shelf with two different height levels. The other group was tasked to do the same with a manually controlled arm that included a few different input methods. The study took place over a period of three weeks. While the task completion was the same for both methods the

effort for autonomous control was significantly lower, nevertheless the satisfaction was not heightened in autonomous mode, in fact, the raw data in satisfaction was lower. An indication for a reason is given in the semi-structured interview following the study where it was indicated that even though the users could benefit from the autonomous system, they felt more in charge during the manual mode (D.-J. Kim et al., 2011).
Another study, performed by Gopinath, Jain, and Argall (2016), was interested in how differently impaired users would decide to tune the degree of autonomy the system has. Gopinath et al. (2016) specifically were interested to give options for customization because physical impairments differ widely between subjects and the input requirement may even change over time (in case of rehabilitation or worsening of the disease). The users had to tune system autonomy at the beginning before performing pick and place tasks with a robotic arm and after the first half of the experiment had the opportunity to adjust their settings. The test subjects where divided in two categories, user with spinal cord injuries and non-injured people. The results show that the customization not only improved performance but also reduced the differences in performance between injured and non-injured subjects. Most interesting is the fact that the custom-mode was not optimized by users to be time-optimal or minimum-effort, rather something more complex, which coincides with the results of D.-J. Kim et al. (2011) that more autonomy and therefore less effort, is not always a primary goal (Gopinath et al., 2016).

Another reason why system autonomy is not always desired, besides possible reduction of user autonomy, may be an issue of trust. J. D. Lee and Moray (1994) studied how and when an automated mode of a system is used, in contrast to a manual mode, by operators in a semi-automatic pasteurization plant. The two big influences they highlighted are the self-confidence of operators in themselves versus the trust they have in the machine. As they put it:

> In general, automation is used when trust exceeds self-confidence and manual control when the opposite is true. (J. D. Lee & Moray, 1994, p. 1)

Additionally, they point out that this interplay is especially elevated in physical contact situations, where actual harm to the operator is possible when the system faults. The formed goal for the design of such systems is that considerations need to be made for this dichotomy and the system should reflect its certainty and how much it can be trusted at any given moment (J. D. Lee & Moray, 1994).
Dragan and Srinivasa (2013) build a system that tries to grab an object to which it is directed by a user via hand motions. Their algorithm calculates a prediction based on user input and tries to make an arbitration on which object to grab. It can follow the user input until the decision is made, after which it will start grabbing autonomously. The interesting part is the possibility to adjust the arbitration function in order to make the robot seem more timid or aggressive. Aggressiveness, in this case, correlates with higher autonomy, because the robot starts to move earlier on its own and needs less user input. Dragan and Srinivasa (2013) made a small study to determine the reception of these two behaviors and were most interested in user preference in the four cases where the grasping task is difficult/easy and the robot's prediction is right/wrong for both the timid and aggressive version of the robot. Resulting from the study a clear time

difference between the two modes is visible, which has a decent correlation to user preference (Pearson's $r(30) = .66, p < .001$), but it does not capture the user's experience completely. The user preference for both cases where the robot is wrong is much higher for the timid version than the aggressive. Additionally, there was no preference difference in the easy and right task for either timid or aggressive. Only for the hard task which was predicted right is the aggressive mode the preferred one. Especially the two cases where the robot is wrong coincide with the findings of J. D. Lee and Moray (1994), that a system should reflect its confidence in some way to build trust. They point out that the robot should not just be quick, rather it should be intent-transparent (Dragan & Srinivasa, 2013).

3 The use of augmented reality

As stated in the introduction, augmented reality is a useful tool for the visualization of 3D-information. As seen in chapter 2 all kinds of information would be beneficial to have, while controlling a complex machine like a robotic arm. From showing safety warnings or intend of the arm to having virtual buttons following the user or robot for interaction purposes. The ability to place information in 3D space as *holograms* may be especially useful in these scenarios. This basic concept of AR, the age-old idea to have virtual elements integrated with the real world, was initially inspired by science fiction like *Star Wars* or *Star Trek*. Interestingly the idea of a hologram is much older than these films and started in the early 20th century with the work of Gabriel Lippmann (1908) on color photography based on light interference (Gabriel Lippmann, 1908). Modern AR technology is doing on a technical level today something very different. Even if the displayed 3D objects from these devices are often called holograms, the technology does not fit the scientific definition. The scientific field, called holography, defines the word hologram specifically as recording (an then displaying) the shape of light after it reflects off a physical object, enabled through its wave-like properties. This is a wildly simplified description of what Richardson and Wiltshire (2017) give in the introduction to their book *The Hologram* (Richardson & Wiltshire, 2017, pp. 1–5). In this book they specifically mention that AR today uses not the wave-properties of light but rather stereoscopy and environment registration to achieve their goal (Richardson & Wiltshire, 2017, pp. 17–25).
Besides misuse of the term hologram, AR in addition to VR and MR have another problem with the definition, which will be alluded to in the following section.

3.1 The Reality-Virtuality continuum

Defining a subject matter is always important to negate misunderstandings and confusion on what is actually said. Especially in situations where a field grew over a longer period of time and new aspects arise that were previously not considered. Augmented as well as virtual reality are such topics that are intertwined, have similarities but are still quite different in many aspects. To straighten these vague connections and clear things up Milgram and Kishino (1994) performed a taxonomy of this area of research and clearly defined what these different terms mean and how they relate to each other. They came up with an intuitive visualization of their so-called *Reality-Virtuality continuum*, which gives the aforementioned terms a quite natural place. Figure 3 shows the continuum which is a simple one-dimensional graph that spans from actual real environments to complete virtual ones. They focus on display technology, which come with different properties and combination levels of reality and virtuality. At the core they say everything on this continuum between the two extremes is a subcategory of **M**ixed **R**eality (MR) because they combine the two to some degree or another and therefore *mix* them together. In total Milgram and Kishino (1994) define six different classes of MR displays, going from monitor-based and non-immersive video displays that have graphics overlaid, to see-through HMD, and large-screen displays where real objects can be integrated (Milgram & Kishino, 1994).
A year after their first taxonomy Milgram, Takemura, Utsumi, and Kishino (1995) elaborated on their previous work and interestingly set out to more precisely define AR in terms

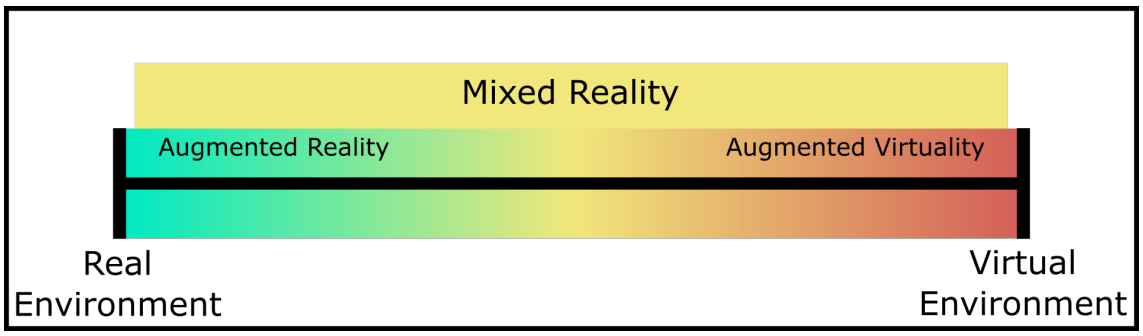


Figure 3: The Reality-Virtuality continuum. Based on (Milgram & Kishino, 1994)

of their continuum. As if they knew that *Microsoft* would years later come and try to define these terms themselves, Milgram et al. (1995) gave them a clear idea of what they mean by the term AR. Milgram et al. (1995) even mentioned most of the issues which are still problems the *HoloLens* has today, like the small **F**ield **O**f **V**iew (FOV), being uncomfortable to wear and inaccuracy with head-tracking (Milgram et al., 1995).
But it did not seem to help, after many years of accepted terminology in the research field Microsoft redefined the virtuality continuum with a subtle difference, which brought back the confusion on what is actually meant when using some of these terms. To specify, on Microsofts renamed Reality-Virtuality continuum (called Mixed reality spectrum) they define AR as the polar opposite of VR which both lie right next to the two extremes. They say:

> ...Thus the experiences [mobile phones] offer cannot mix between physical and digital realities. The experiences that overlay graphics on video streams of the physical world are *augmented reality*, and the experiences that occlude your view to present a digital experience are *virtual reality* (Bray Brandon, McCulloch Jesse, Schonning Nick, Zeller Matt, 2018).

This contrasts with the original AR description of Milgram et al. (1995), which defines it as a range starting at Bray Brandon et al.'s definition and ending somewhere in the center of the continuum, where the prominence of reality and virtuality converge. For this reason, Microsoft calls the HoloLens a MR device, which fits with both definitions, but specifically says that it is not an AR device (Bray Brandon, McCulloch Jesse, Schonning Nick, Zeller Matt, 2018). Which it is in the original sense.
Concluding, in this thesis the first definition of Milgram and Kishino (1994) is being used and both terms, AR and MR, are applicable for the HoloLens. Mostly used is the term AR to refer to its capabilities because it is a narrower term and describes them more precisely than MR would.

3.2 What does augmented reality bring to the table?

As stated, AR spans almost half of the virtuality continuum and begins close to actual reality. A common example of AR close to that side is the widely popular game *Pokémon GO* [1] which uses GPS and IMUs of the smartphone to place the 3D models in specific locations but is not able to let them interact with the real world in any way. Much farther

[1] https://www.pokemongo.com/

to the middle of the continuum is Microsofts previously mentioned HoloLens, which uses in total eight different cameras. Four environment tracking cameras for map building and head tracking, two depth sensors where one is for the spatial mapping and the other for hand-tracking and two **I**nfra**r**ed (IR)-reflective cameras which are unaffected by lighting conditions and help for depth calculation (Zeller, Gedye, Ong, Schonning, & McCulloch, 2018). Through spatial mapping and head tracking the registration of the HoloLens is enabled. Additionally, the 3D-mesh of the surroundings, which is generated during this process, allows integration of 3D-objects into the real world and lets them interact with it.

To have a convincing realness of the holograms a high precision of registration is necessary. Especially because humans are visual creatures and can spot visual mismatches up to a very fine difference, as Azuma (1997) elucidates in his survey of AR (Azuma, 1997, pp. 18–22). As the HoloLens was the only widely available modern AR device, it was used extensively in research and its performance got measured and tested on multiple occasions. Y. Liu, Dong, Zhang, and El Saddik (2018) for instance made a general survey of the technical capabilities of the HoloLens in which they evaluated head localization, real environment reconstruction, spatial mapping, hologram visualization, and speech recognition (Y. Liu et al., 2018). For hologram visualization, they measured an average visual distance of 1.25 cm with a standard deviation of 0.25 cm when overlaying a 3D model exactly on top of its real counterpart. Which points to a small imprecision of either the display technology or registration of the HoloLens. In contrast to Y. Liu et al. (2018), Vassallo, Rankin, Chen, and Peters (2017) performed a more directed study of HoloLens' hologram stability. Their goal was to examine if the device has the technical capabilities for intraoperative clinical use, as very high precision is required in surgeries. They devised four different actions that were performed during the study which might occur in clinical procedures. Walking, Sudden Acceleration, Occlusion, and Object Insertion where separately performed and resulted in slight drifting of the 3D models. Displacement errors were at 5.83 mm with a standard deviation of 0.52 mm (Vassallo et al., 2017). These results show that the hardware is by no means perfect yet, but already advanced enough to handle situations where precision is important, up to a point.

Another important part of modern AR is the FOV in which the 3D models are displayed. This is one of the biggest flaws of the HoloLens, as its FOV is only 30°x17.5° horizontally and vertically respectively. This limits the viewing area of holograms to something alike to looking through a small window in front of you onto the virtual parts. Not only limits this immersion but makes it harder to locate holograms in space. All virtual objects which normally would still be in your FOV are not displayed and head-movement is required to center them and bring them into the FOV of the device. Other current hardware has only slightly better FOVs (Magic Leap, HoloLens 2), which are still too small for many tasks and not even close to the near 200° horizontal FOV of humans (Ruch & Fulton, 1960).

Hand tracking is another feature modern AR devices often integrate. It is not part of the display technology and therefore does not fit into Milgram and Kishino (1994)s definition of the virtuality continuum. It nonetheless is an important step to get closer to the center of a perfect blend between reality and virtuality. Additionally, it just makes sense to manipulate a virtual object with your hands, because it is the most natural thing to do

with actual objects. At this point in time, the hand tracking is still mostly limited to simple gestures like an *air tap*, which just replaces a normal button click. Actual hand and finger tracking similar to what the *Leap Motion* [1] achieves is not yet possible with modern AR-devices.

3.3 The state-of-the-art augmented reality HMDs

Besides the HoloLens which was discussed in the previous section in the context of general AR capabilities, there are a few alternative AR devices that are generally technologically on par with the HoloLens. Most notably the *Magic Leap One* which has similar capabilities as the HoloLens, like inside-out tracking with infrared cameras, spatial registration, and gesture recognition. The device has two advantages. One is the FOV, which is bigger (40° x 30°) and the other is its weight, which is almost half what the HoloLens weighs (600g vs 320g). The caveat with its weight is that it has an additional small computing unit, which can be attached to the belt (Magic Leap Inc., 2019; Sauter, 2018).
As a successor to the HoloLens, Microsoft announced the HoloLens 2 in early 2019. It promises a better weight distribution for more comfort while wearing the device, included eye-tracking and a bigger FOV which rivals the Magic Leaps with 43°x29° (White, 2019).

3.4 Literature: How AR can enhance the use of robotic systems

Since the release of the HoloLens, research on virtual augmentation of robotic systems had a big surge. Before AR with head-mounted displays became viable research focused mostly on projecting information into the real world via projectors. An example is the system of Leutert, Herrmann, and Schilling (2013), where a projector shows the information on what the robot is going to do, like a path it is going to follow (Leutert et al., 2013). Research focused on object detection/recognition is deliberately left out in this section and will be discussed in sections 4.4.2 and 4.4.3.
An interesting direction was to use AR to help the researcher debug their autonomous systems when they fail. This was achieved by spatially displaying the sensor data of the robotic systems at their real-world locations, which gives the developer an easier way of interpreting it and finding reasons for failure. Collett and Macdonald (2010) provide a systematic analysis of the challenges and approaches to visualize the robotic data for debugging. Giving architectural ideas on a general software system that is applicable for a wide range of different robotic setups (Collett & Macdonald, 2010). Based on Collett and Macdonald (2010)s research, Renner, Lier, Friese, Pfeiffer, and Wachsmuth (2018) used the HoloLens years later to not only show the sensor data of the robot via the new hardware possibilities but also used it as a back-channel to provide the robot with additional data gathered by the HoloLens itself (Renner et al., 2018). At that point, the data visualization was not only intended for researchers but as a general help for users to understand what the robot is seeing and going to do.
This line of reasoning was also the basis for the research by zu Borgsen, Renner, Lier, Pfeiffer, and Wachsmuth (2018) who wanted to improve Human-Robot handover using different augmented reality approaches. They visualized the robot laser scans, the path it

[1] https://www.leapmotion.com/

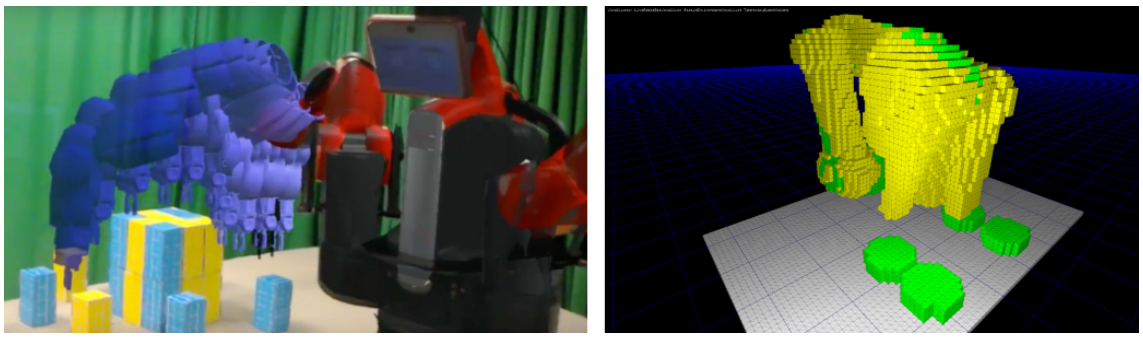


Figure 4: Side by side comparison of the robotic arm motion visualization by (Rosen et al., 2017) (left) and (Bolano, Roennau, & Dillmann, 2018) (right).

will take, its battery life, and pose via the HoloLens for an AR scenario. Additionally, they added a virtual body part to the robot, namely the head, to give it facial expressions during the handover and help the human to better understand what is going to happen. These enhancements were based on a previous study for handover without AR which showed that non-experienced users had problems understanding the robot. A future study with a fully or partially virtual robot will be conducted based on the design of their paper (zu Borgsen et al., 2018). Communication of motion intend is especially important when a human and robotic arm work collaboratively in close proximity to each other. While humans communicate in subtle ways what they are going to do when working close together, a robot lacks such natural behavior. For that reason, quite a bit of research is conducted to help this kind of information exchange. For instance (Bolano, Roennau, & Dillmann, 2018) and (Rosen et al., 2017) use similar approaches for this problem. They utilize AR to show the motion volume (or sweeping area) of the robotic arms next motion to the user to inform him about possible collisions. Rosen et al. (2017) conducted a study that indicates an increase in prediction accuracy of 16% and a decreased task completion time of 61% with the AR interface. Figure 4 shows the differences in their visualizations, where Bolano et al. (2018) (right) used a voxel-based volume to show the motion area and Rosen et al. (2017) (left) a more realistic representation but only snapshots of the motion in short intervals.
For non-autonomous robotic arms Gadre et al. (2019) build a system followed by a study, where the user can plan the robot's motion via waypoints using the HoloLens. The difference to the previous research is that the users themselves are responsible for the motion and therefore want to know if the planned path works as intended (no collisions). For this reason, the path is shown in AR similar to the approach of (Rosen et al., 2017) in figure 4. When the user notices unintended motions they can adjust the waypoints until it conforms to what they intended. In the performed study this AR interface was compared to a basic 2D one, where the users needed to plan a path for picking and placing objects at specific positions. Their results strongly support the use of the AR interface (Gadre et al., 2019).

Krupke et al. (2018) compared different input techniques, enabled by AR and the HoloLens. They lay a virtual version of the robotic arm over the real one and when the user picks a target, an animation is played that shows the upcoming movement of the

real robotic arm. When the user is certain that the robotic motion is what they expected they can confirm it and the real arm moves. Krupke et al. (2018) focus was on the two kinds of input modalities to select a specific target via pointing, which they compared in their study. The first method was the head gaze of the user, like a laser pointer, coming from between their eyes and shooting in the direction they look. For confirmation, speech input was utilized in both techniques. For the second method, the user needs to point with a finger that is tracked by the HoloLens, and a ray shoots from the head position through the tip of the finger for selection. The results of their small study indicate that the head ray laser pointer method is faster, more precise, and has a lower taskload than the finger-pointing method (Krupke et al., 2018).

4 Advancements in object and pose recognition

Object recognition is the next logical step for many systems that interact with the real world in one way or another. Autonomous cars should stay on the road and try to avoid any object, but when unavoidable it should be able to choose a trash bin on the side of the road, rather than a child trying to catch a ball. Or a furniture AR application could not only show the new couch in the living room before you brought it but could easily place it in the correct location in front of the TV. Especially when working with robotic arms it opens the possibilities for autonomy in object manipulation, or as in this thesis, gives users additional information about the objects and helps them with their tasks.
In this section, the focus lies on the specific sub-task of object recognition, pose recognition, which is necessary for precise interactions with objects that are not in a perfectly controlled environment. First, a proper definition will be given, to make clear what the differences between the terms object classification, object localization, object detection, and pose detection are. Following statements on what makes these tasks difficult to solve.
In the second half, we will look at the different approaches on how to solve the object-pose recognition problem. Lastly will be an examination of the research regarding not solely object-pose recognition but the intersections of it and Robotics/AR.

4.1 Defining object and pose recognition

Object detection/recognition has been a constant research field in the area of computer vision and went through several phases in the last decades. Beginning in 1963 when Lawrence Gilman Roberts (1963) published his thesis "Machine perception of three-dimensional solids". Where he developed a program that could detect the outlines of simple 3D-objects from images and match them to predefined internal representations. If successful the object can be drawn virtually on the screen from any angle (Lawrence Gilman Roberts, 1963). Since then the field evolved extensively and went through a few different technological phases. As a basis for the comparison of new approaches, a few different terms emerged along the way. These start with the easiest task and get progressively harder to solve subsuming the previous one, as stated by Russakovsky et al. (2015). The first one is *Image Classification* which denotes the prediction of an object type (class) in a cropped image. The next is *(Single-)Object Localization* which finds the type and defines a bounding box around the objects in a given cropped image. The third is *Object Detection/Recognition* which expands the first two, giving an object classification with a corresponding bounding box of all objects in a given image, not just one. Apart from these three main terms for the stages of object recognition exist other more specific ones for different sub-tasks. A specification of general object detection is *Object Segmentation* which gives not only the objects bounding box, but rather all the pixels on the image it occupies (Russakovsky et al., 2015).
Lastly, the term *Pose Detection/Recognition* brings the 2D information of the image into the third dimension, trying to give the detected objects rotational and translational information relative to the camera, which took the given image. The terms detection and recognition are interchangeable and are used by different researchers but mean the same thing. In this thesis, the preferred term will be object recognition.

Figure 5: What defines a chair? Comparison between different kinds of chairs[1] [2] [3].

4.2 What makes it difficult?

Even after almost sixty years of research the problem of object recognition is not wholly solved, even though many systems are already very good at correctly recognizing hundreds of different object classes, it is not even close to human-level object recognition. Only under the most simple conditions (few object, no occlusion, controlled illumination) is the task of object detection considered solved propose Andreopoulos and Tsotsos (2013) (Andreopoulos & Tsotsos, 2013). Mostly solved aspects are the environmental effects, which can change many properties of objects. The lighting conditions, even between dimly lit and bright sunlight, already change the colors of an image/object. Not to mention colored light, which alters them completely. Having occlusion or simply different orientations of objects changes their visible shape and can make them look very different in images. Simply relying on color detection or outlines is not a viable solution for these changes. But with modern approaches, many of these basic discrepancies in objects can already be overcome.

What makes classifying all objects even for only one class still extremely hard is their massive variety, called within-class variation. Even something trivial like a chair has great diversity, as figure 5 demonstrates. Color, shape, amount of legs, and material are all different between these three chairs. The main chair-defining characteristics are: you can sit on it (horizontal surface) and in this case, it has a backrest, but that may not even be required for a chair. To solve this problem and be able to extend the classification to yet unseen, newly designed, chairs would maybe require a complex reasoning structure that can extrapolate and cross-reference specific aspects. Ot it could have the ability to extract the commonalities of previously learned chairs and differences between these and, for instance, a couch. Whatever the solution is, the complexity of replicating what humans can easily identify as a chair is a highly difficult task yet to be perfectly solved. These remaining problems are discussed in more depth in the survey of Zou, Shi, Guo, and Ye (2019) (Zou et al., 2019).

Pose recognition is a special case of object recognition and therefore inherits its difficulties, which makes it an even harder task because it has some additional hurdles itself. First, it

[1] http://pngimg.com/download/6862
[2] https://commons.wikimedia.org/wiki/File:Ngv_design,_ron_arad,_tom_vac_chair,_1997.JPG
[3] https://de.wikipedia.org/wiki/Datei:Ngv_design,_frank_o._gehry,_wiggle_side_chair,_1972.JPG

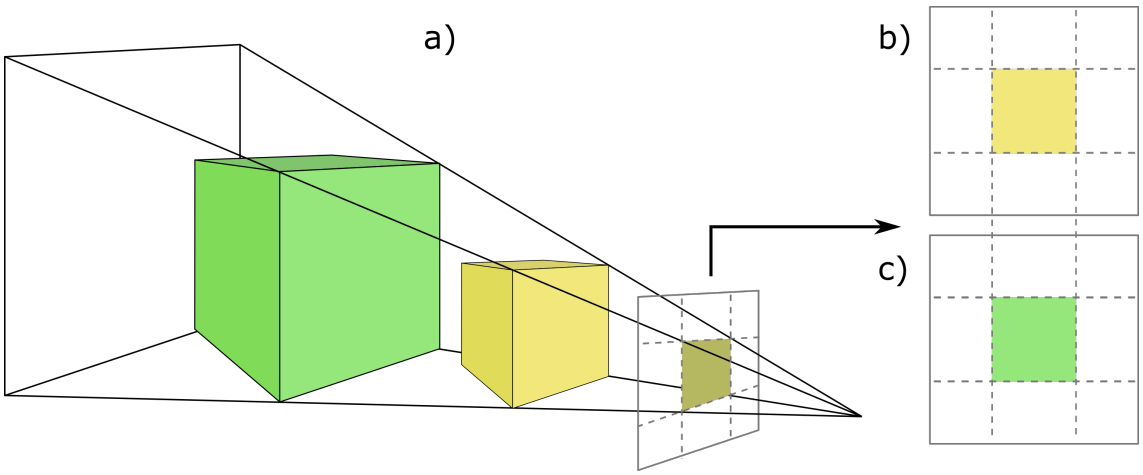


Figure 6: Visualization of depth-size ambiguity. a) Two objects with different sizes and distances that, when projected on the screen, have the same size. b) Screen view of the small object. c) Screen view of the big object.

has an even bigger problem with rotational or occlusion-based ambiguity, as it makes it impossible to know the exact pose, even for humans. As Manhardt et al. (2019) point out in their research, which addresses this problem specifically (Manhardt et al., 2019). The added dimension, the distance to the camera, is calculated based on the camera parameters (dimensions, focal length, etc.) and the object size. As one might expect, if the object to be positioned exists in multiple sizes, the distance calculation from the camera may be wrong. It may take the wrong object-size, and therefore place it closer or farther away from the camera as it is in reality. Figure 6 demonstrates how two differently sized object, which look the same in an image (different colors only for readability), may actually be differently sized, only placed at different positions in space. Lastly, the majority of modern solutions are based on neural networks, which need a huge amount of training data for learning. These are pretty easy to manually label for standard object recognition, where a square is drawn around the object in question on each training-image. For many pose recognition algorithms, this 2D square is transferred into the third dimension which makes labeling them a hard and arduous task. Seeing the rotation of the object and correctly placing its 3D bounding box accurately on the image is additionally not very precise.
All these problems lead to the fact, that careful considerations about the scope and exact problem statement need to be made when designing or building a pose recognition system. For example, leaving out obscure objects or having a partly controlled environment, in which all objects are known beforehand.

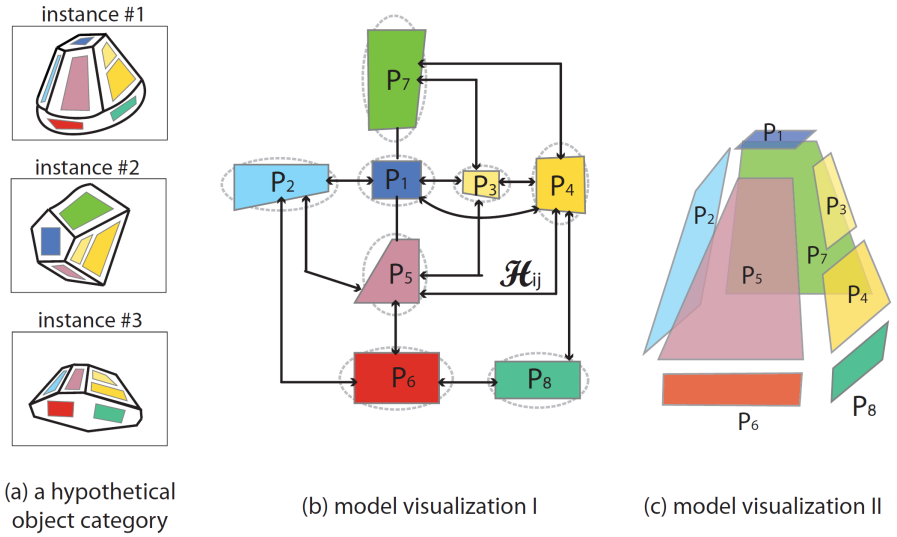
4.3 Approaches to object-pose recognition

Over the years a few different approaches were popular in the field which still have some advantages and disadvantages over each other. The three most important ones will be discussed here. They are feature-based, template-based, and convolutional neural network-based approaches.

4.3.1 Global and local feature extraction

The basic idea of this approach is to extract specific attributes from the image, called features, which drastically reduces the problem space from the high dimensionality of the image resolution. These features are categorized into local and global features. Global features contain information that encodes the whole image, e.g. a histogram of the color intensity. The fact that they encode the whole image makes global features susceptible to background clutter and occlusion. For this reason, they are often used for direct image matching and in combination with the local features to enhance object recognition. As the name implies, local features are deployed only on parts or a region of an image. This is especially handy for object detection, because it ignores the image background, and focuses only on the important parts. These features may be color, texture, or specific shapes in the image (Lisin, Mattar, Blaschko, Learned-Miller, & Benfield, 2005).
One of the first recognition systems using local features was developed by Lowe (1999), who expanded on the ideas of Schmid and Mohr (1997) and used the **S**cale **I**nvariant **F**eature **T**ransform (SIFT) algorithm to extract transform, scaling and rotational invariant features from a given image for detection purposes (Lowe, 1999; Schmid & Mohr, 1997). He points out that his approach is very robust against occlusion because it only needs three *keys* (which are key locations in the image/object) to identify an object. The more detailed an object is, the more keys it has making it easier to detect. The detection is robust up to 60° rotation away from the camera in any direction for planar objects. For a 3D object with rotation in depth, it has a rotational range of about 20°. Other similar approaches inspired by the work of Schmid and Mohr (1997), which were published around the same time where Allezard, Dhome, and Jurie (2000), Baumberg (2000), Mindru, Moons, and Van Gool (1999), Tuytelaars and Van Gool (2000). All of these approaches, including the one from Lowe (1999), where evaluated in a study by Mikolajczyk and Schmid (2005), where Lowe (1999)s outperformed most others (Mikolajczyk & Schmid, 2005).

Lepetit, Pilet, and Fua (2004) treat the point matching of extracted local features as a classification problem. During the training phase, they perform statistical classification of a large number of synthesized views of the keys. This produces a compact description of said key points, which can then be used during runtime. The goal of these prior offline calculations is to offload the work for higher runtime performance. To extract the object pose a standard RANSAC-based method is applied after the image correspondence is found. Their experiments showed, that the pose estimation for 3D boxes and faces is reliable and accurate from most view angles, but they do not give any exact numbers on the results (Lepetit et al., 2004).
A different idea came from Savarese and Fei-Fei (2007) who proposed a system that constructs a model of an object out of detected parts. These parts are distinct sections with many local invariant features and get connected through their projective relation. Figure 7 shows how such a model is internally connected. With this approach, they generate compact representations of the appearance and geometry of the corresponding object class. The object pose is obtained for each part in the model during the construction phase. The view of a part that has the most area facing the camera is its base-pose. After the recognition of an object, the pose can be found by looking at the



(a) a hypothetical object category

(b) model visualization I

(c) model visualization II

Figure 7: Visualization of object model part connections. a) three sample training images, b) the internal model graph and c) a more intuitive visualization of the graph (Savarese & Fei-Fei, 2007).

visibility of each part and on that basis, the pose is estimated (Savarese & Fei-Fei, 2007). Building on this work Savarese and Fei-Fei (2008) extended their system to recognize previously unseen object poses. This is done by synthesizing two existing views of connected parts into a new one during the recognition phase. Through linear interpolation, a new, synthetic view is created which is used to predict the unseen object pose (Savarese & Fei-Fei, 2008).

The *Vuforia Library* is an AR **S**oftware **D**evelopement **K**it (SDK) that is widely used for commercial applications. It is able to track multiple planar objects simultaneously as well as a couple of 3D objects via natural local feature extraction. Not much is known about the actual algorithms used, but it seems to be based on the research paper by K. Kim, Lepetit, and Woo (2012). The proposed system has a small database of 3D objects (around 50) of which up to 14 different objects can be tracked simultaneously and in real-time. Additionally, it enables the user to integrate new objects of primitive shapes very fast and easy. The key to its real-time capabilities is a clean separation of tasks into a foreground and background thread, of which the foreground thread tracks the feature points for every consecutive frame and the background thread recognizes the targets and estimates the 3D poses.
Figure 8 shows the process of adding new objects into the database. The user can add local coordinates on one of the captured frames to outline the base shape of its primitive and extrude that side into 3D space. Afterwards, the object is added to the database for matching, which holds a description of the corresponding local features inside the user made outline (K. Kim et al., 2012).

Especially after K. Kim et al. (2012) published their work, the feature-based object and pose recognition is pretty reliable and usable in real scenarios as shown by Vuforia. The biggest disadvantage is the need for local invariant features on objects in order to track them reliably. Many objects in the real world, however, do not have the appropriate

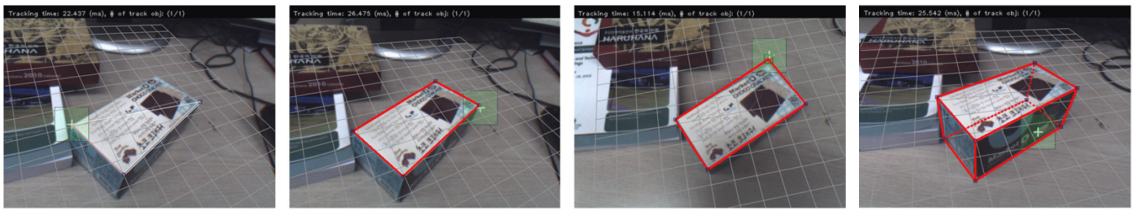


Figure 8: Procedure for adding a primitive object (K. Kim, Lepetit, & Woo, 2012).

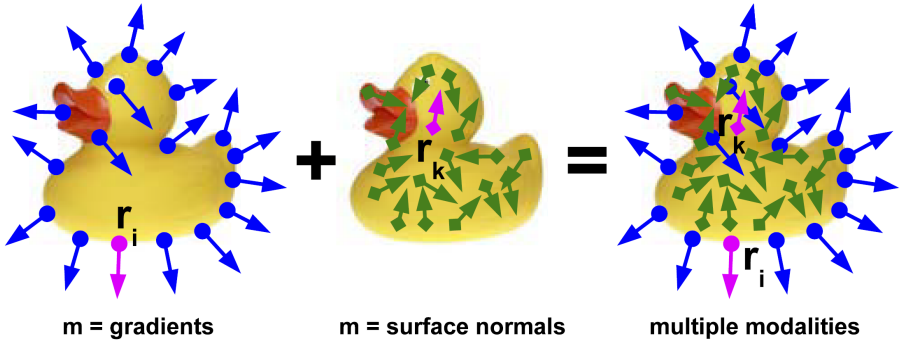


Figure 9: Two complementary modalities, gradient, and surface normals, for object template representation (Hinterstoisser, Cagniart, et al., 2011).

number of features for robust tracking and therefore exceed the limits of approaches based on feature extraction.

4.3.2 Template-based

The template-based approach for object pose recognition was developed in direct response to the fact that local feature extraction is often dependent on object texture and natural distinctive features. In addition, it has advantages to other statistical approaches that require a learning phase to build the classifier, which generally takes time and is required for every new object. It was mostly Stefan Hinterstoisser who lead the research on this template-based pose recognition, which is the only distinct different approach to neural networks that started to emerge at that time.
His work is inspired by Dalal and Triggs (2005) and their **H**istograms of **O**riented **G**radients (HOG) algorithm (Dalal & Triggs, 2005). In *Dominant orientation templates for real-time detection of texture-less objects* Hinterstoisser, Lepetit, Ilic, Fua, and Navab (2010) first presented the approach they named **D**ominant **O**rientation **T**emplates (DOT). The basic idea is to use the gradient orientation of an image, which is a global feature, and compare it with an input image gradient orientation at a specific location. For comparison of the orientations only the magnitude of the gradients, not the actual values, are used. This makes it more robust to illumination changes and noise. For more tolerance to small transformations and better computing performance the images are divided into regions on a regular grid and only the dominant orientation in that region is used, meaning only the gradient with the biggest magnitude has an impact on matching.

For performance evaluation, they performed a comparison experiment against a few different approaches (nine others in total). They used the Graffiti and Wall Oxford datasets in which they increased the viewpoint angle from 20° to 60°. They not only

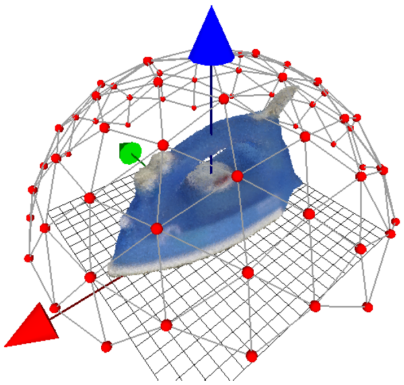


Figure 10: Regularized grid around an icosahedron for viewpoint taking of an object 3D model (Hinterstoisser et al., 2012).

outmatch all other tested approaches but achieve a remarkable 100% matching score for both the Graffiti and Wall datasets (Hinterstoisser et al., 2010).
One big issue with this approach is its susceptibility to background clutter, which quickly degrades the results up to a failure of recognition. To address this problem Hinterstoisser, Cagniart, et al. (2011) changed their use to only the dominant orientation of a regular grid space. A similarity measure for the gradient orientation is used, which searches the neighborhood of an associated gradient location and tries to find the most similar orientation in the input image. This results in finding almost only the gradients on the object contours, not the background objects, making it much more robust to background clutter. Further enhancement of their previous approach is done by the added possibility to also integrate depth data as a second classification property. They extract the surface normals and process them in the same way as the gradient orientation, which leads to mainly finding matches on the object itself. Figure 9 shows the complementary nature of these two modalities, making the object recognition quite a bit more robust (Hinterstoisser, Cagniart, et al., 2011; Hinterstoisser, Holzer, et al., 2011).

While the results of Hinterstoisser, Cagniart, et al. (2011) are also giving the pose of the recognized objects, it is only a rough estimate, because of the sparse view-angles of the created templates. In robotics and other computer vision tasks, a precise pose is most often necessary, which is why Hinterstoisser et al. (2012) revised their previous work again to enable the possibility of acquiring precise orientation. The approach to produce a precise pose is the simple idea to regularize the encoded viewpoints to cover the whole object. Firstly, online template learning was moved to be offline and not user-operated anymore. A 3D model of the object is utilized to create all required views. In this new offline template learning stage, the images for encoding the template are taken in a spherical regularized icosahedral grid, shown in figure 10. On every red dot in the figure, a new training image is taken. Between each new viewpoint the camera is rotated by an angle of approximately 15° to cover the whole object. To note here is that they only cover the upper hemisphere of the object, as visible in figure 10 which only enables a rotation detection of 0-360° around the object and ±45° tilt (not an arbitrary rotation). Additionally to the new offline training stage, they created a new dataset for future

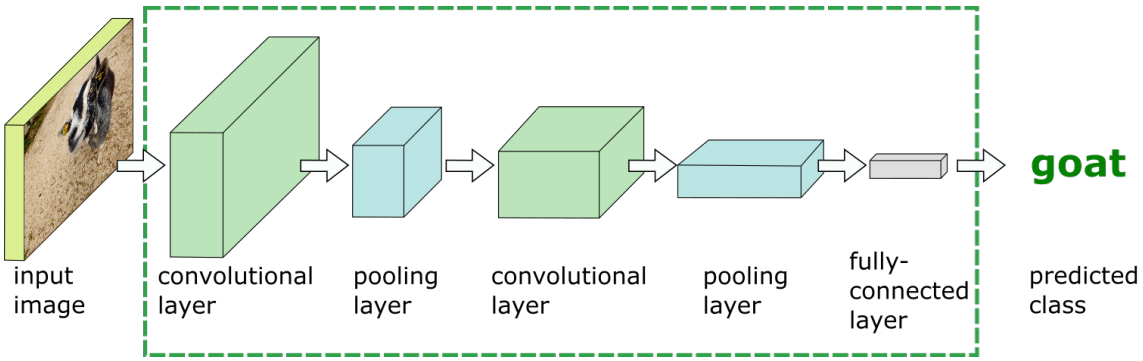


Figure 11: Example of the basic structure of a CNN.

comparisons with other pose detection approaches. It contains 15 mostly texture-less objects with over 1100 reference images each and the ground truth pose to compare against. In the performed experiment they had an average detection rate of 96.6% for all objects (Hinterstoisser et al., 2012).

4.3.3 The era of convolutional neural networks

As the computing power of modern machines increased significantly, research on artificial neural networks for machine learning had a resurgence. Especially deep neural networks brought new progress in the field and based on that the new sub-class of **C**onvolutional **N**eural **N**etworks (CNN) was created. These are especially well suited for high dimensional input data because they convolve the input across their multiple layers. Similar to other deep neural networks, CNNs have one input layer, an output layer, and multiple hidden layers in between (hence the name *deep* neural network). Figure 11 shows a basic CNN structure which reduces the high dimensionality of the input down to a certainty value for the best matching trained objects. Goodfellow, Bengio, and Courville (2016) explain in their book *Deep Learning* the workings of deep neural networks and specifically, starting from chapter 9, CNNs in detail (Goodfellow et al., 2016).

The modern CNN structure was developed for recognizing handwritten digits by LeCun, Bottou, Bengio, Haffner, et al. (1998) in 1998 (LeCun et al., 1998). Utilizing CNNs was re-popularized by Krizhevsky, Sutskever, and Hinton (2012), which won the *ImageNet Challenge*[1] in 2012, which focused only on 2D object recognition and not on full 3D pose. From that point onward CNNs became standard in object recognition and in the last couple of years it also became the basis for most pose detection systems.
In terms of performance, PoseCNN is a very successful network architecture for 6D pose estimation created by Xiang, Schmidt, Narayanan, and Fox (2017). It decouples the different tasks of pose estimation into three distinct components. First, it labels each pixel in an input image and assigns them to possible objects. Secondly, it predicts the objects 2D center as well as the distance to the camera center with the voting of each pixel from a corresponding object label. With the help of the intrinsic camera parameters, this can predict the 3D location of the objects relative to the camera.

[1] http://www.image-net.org/challenges/LSVRC/

Finally, through a regression of the convolutional features extracted inside the bounding box, a quaternion representation of the object rotation is generated. This is achieved by running the previously predicted visual features through 3 fully connected layers, which have two different optimization functions, one specifically to handle object symmetry. Their method uses just RGB image data for the 6D pose prediction, but can be coupled with the **I**terative **C**losest **P**oint (ICP) algorithm when depth data is available to boost the overall precision of pose estimation. The experimental results show slightly higher scores than comparable state-of-the-art approaches of that time (Xiang et al., 2017).
Around the same time as PoseCNN was developed, Tekin, Sinha, and Fua (2018) approached the problem from a different perspective. Inspired by the YOLO-network from Redmon and Farhadi (2017), they based their architecture around the idea to only have a single processing stage, which grants them real-time capabilities (Redmon & Farhadi, 2017). The standard are methods that use at least two stages, or as PoseCNN three, to predict the pose which takes additional computing time. The CNN they use predicts the 2D projections of all eight corners of the 3D bounding box and a center point for one object. Then uses the **P**erspective-**n**-**P**oint (PnP) algorithm to extract a 6D pose. For training, they point out that only the 3D-bounding-box of training objects needs to be known, not a complete 3D model. The experiments show that the network outperforms all other compared state-of-the-art approaches not only in prediction accuracy but also in computational runtime. The runtime speed is an impressive 50 **F**rames **P**er **S**econd (FPS), which is ten times higher then (Kehl, Manhardt, Tombari, Ilic, & Navab, 2017)s approach, which is also single staged (Tekin et al., 2018).

One disadvantage of many CNN-based systems is the fact that labeled training data is required for every object class or type. In the case of 6D pose, this data has a high complexity in contrast to the generation of data for 2D recognition and is, therefore, time-consuming to generate by hand. One approach to tackle this problem is to generate this data with synthetically rendered images from 3D models, which brings its own problems. Even photo-realistic rendering is not exactly similar to real photos and degrades the performance of the trained network. The minute differences produce different statistics on which the networks train, this is often called the *Domain Gap*. Hinterstoisser, Lepetit, Wohlhart, and Konolige (2018) elaborate on this and developed an approach that uses a pre-trained deep network, freezing the first layers which are responsible for lower-level image feature detection like edges, then training the rest with only synthetic images. This approach is simple and mostly overcomes the aforementioned problems (Hinterstoisser et al., 2018).
Sundermeyer, Marton, Durner, Brucker, and Triebel (2018) focus on the problem of expensive training data and object symmetry/partial occlusion, which creates ambiguous orientation, having also computational performance in mind. Their approach is two-staged and uses a **S**ingle **S**hot Multibox **D**etector (SSD) (W. Liu et al., 2016) to identify objects in the image and generate an image crop for each, then feeding this output into their **A**ugmented **A**uto**e**ncoder (AAE) for pose determination. The idea behind the AAE is to only implicitly learn the pose representation, as opposed to explicitly on labeled data, from rendered 3D model views. The augmentation controls

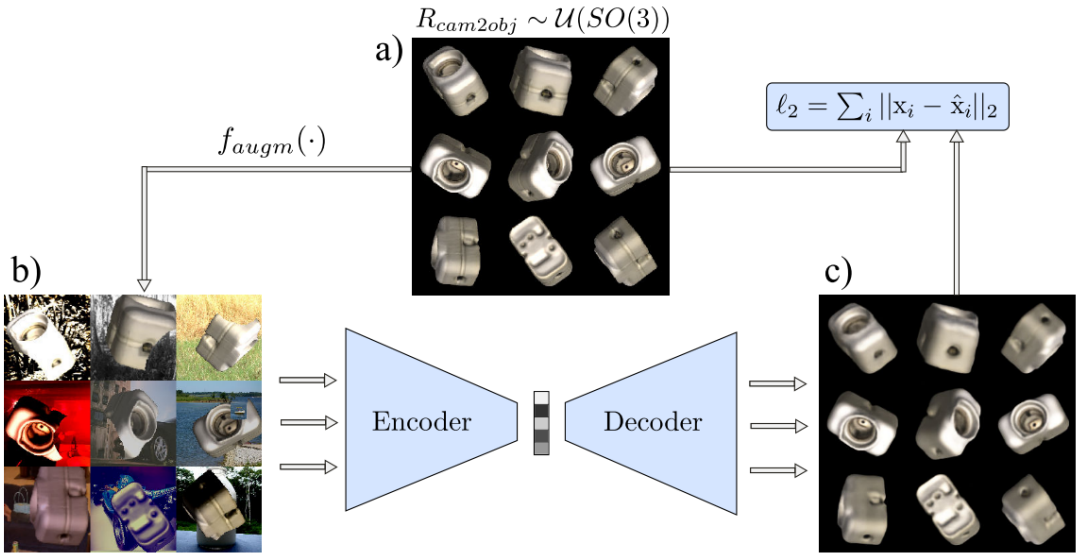


Figure 12: Training of the AAE. a) is a rendered comparison batch, b) is the same batch with domain randomization and c) is the reconstructed pose after 30000 iterations (Sundermeyer, Marton, Durner, Brucker, & Triebel, 2018).

what properties the latent representation learns, and to which it will become invariant. To achieve this they use *Domain Randomization*, which alters the rendered images in regards to lighting, occlusion, background, saturation, etc. as described by (Tobin et al., 2017). The implicitness of the representation learning combined with the domain randomization generates robustness against many of the previously mentioned problems like occlusion and symmetry. Figure 12 shows the training procedure of the AAE. The important part is the encoder/decoder stage, where the encoder produces a reduced representation of the high-dimensional input, which the decoder can later convert back to the higher dimensionality. For identifying pose on unseen images the image crop will then be encoded to this representation, which is invariant to all the domain randomized features, and compare it to the learned representation. The best fit is then chosen which inherits the originally encoded object pose (Sundermeyer et al., 2018).

DeepIM is an approach by Yi Li, Wang, Ji, Xiang, and Fox (2018) that focuses only on pose accuracy and refinement, not as much on computation time or data generation. Their idea is an iterative CNN, that uses an initial pose estimation. It refines this estimation iteratively by rendering the previous pose with the 3D model of the corresponding object and matching this synthetic image against the original input. A mask helps to segment the object and zooming into the image at the correct location to make the object larger and ignore the background or other objects. The reason to enlarge the object is to gain more detail/features across the whole image that gets fed into the network. Their approach has a very good performance in comparison to other systems like PoseCNN, to which they compared it. They do not talk about the computational time it takes the network to generate a result, but the iterative nature and rendering stages hint to poor speed (Yi Li et al., 2018).

Besides the here described approaches there are many others, which focus on similar problems using only RGB image data. For further readings, here are a few interesting

ones: (Rad & Lepetit, 2017), (Pavlakos, Zhou, Chan, Derpanis, & Daniilidis, 2017), (Peng, Liu, Huang, Zhou, & Bao, 2019).

Different from all aforementioned methods, which use only RGB data, the following approaches utilize additional depth data from a RGB-D camera. This boosts the performance of pose estimation, because of the added dimension of depth. In the last years, RGB-D cameras became much cheaper, making them more viable to use. But like it is with everything, this approach brings its own additional problems. For one, normal depth cameras have a sweet spot in which they generate usable depth data, but everything further away or too close degrades in data quality. Additionally integrating this additional depth channel into the pipeline and using the information properly is not as easy as one might expect, because the depth channel has a different space compared to the RGB data.
DenseFusion from C. Wang et al. (2019) finds a way to tightly couple the RGB-D input data and use it to get much more robust results in cluttered scenes and cases of occlusion. They generate features from the two channels (color and depth) separately and therefore retain the intrinsic structure of the source data. After this feature extraction, they perform a per-pixel dense fusion, which is local instead of global, making it more robust to clutter and background noise. *Fusion* is the procedure of bringing the two feature types, namely geometric and color, together, by projecting each 3D point onto the corresponding 2D-image-feature plane with the use of the intrinsic camera parameters. The resulting per-pixel features are then fed into a network that predicts the 6D pose. In contrast to PoseCNN which could use depth data to improve the pose estimation in a concatenated ICP refinement step, this approach is much faster and has fewer problems as aforementioned clutter and background does not affect it. Their experimental results use PoseCNN+ICP and others as a comparison. DenseFusion has a slightly better performance than PoseCNN in cluttered scenes and is 200 times faster (C. Wang et al., 2019).

4.4 Bridging the research areas

After looking at research that focuses almost solely on object-pose recognition, the following sections discuss research that integrates pose recognition into different scenarios. It will cover the topics relevant to this thesis, robotics, and augmented reality. Ending with research on systems that combine all three main points of this literature review.

4.4.1 Robotics and object-pose recognition

Having a robotic arm grab objects autonomously is crucial in many scenarios, especially when the environment is unknown. The actual difficulty to accomplish this task depends on the specific scenario. Collet, Berenson, Srinivasa, and Ferguson (2009) were able to build a robotic system that is able to grab objects on its own pretty reliably by restricting the objects to be only highly textured and known beforehand. This enables them to adapt the core algorithm introduced by Gordon and Lowe (2006), which itself is directly based on Lowe (1999)s previously discussed pioneering work, for local feature extraction

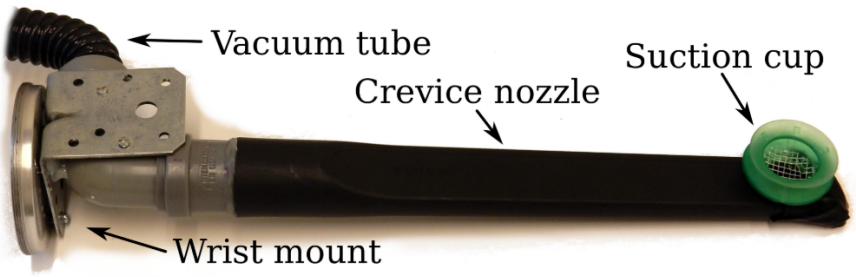


Figure 13: End-effector consisting of a crevice nozzle and suction cup. It lifts objects with the suction strength of an attached vacuum cleaner (Eppner et al., 2016).

(Gordon & Lowe, 2006; Lowe, 1999). With these restrictions in place, their system can detect multiple objects poses simultaneously and is robust to outliers, occlusion, and illumination changes. Using an existing path planning algorithm for their robotic arm, they conducted an experiment to evaluate the performance of their system. They build a cluttered scenario out of four different objects (a can of cola, a juice bottle, a cereal box, and a notebook). Each object was grasped 25 times amounting to 100 tries, with 91 succeeding. Additionally, they measured the rotation error of the pose prediction, which degraded slightly for rotations higher or lower than 45/-45 degrees for some objects. The stated reason for this is the sparseness of local features at those rotations. As future work they pursuit to integrate multiple camera views or even active sensing to move the arm into a better viewing position, which would alleviate occlusion problems and rotational degradation (Collet et al., 2009).

In 2015 the online store *Amazon* posed the *Amazon Picking Challenge*, to accelerate research on solving their specific tasks in the warehouse. Autonomously finding and grasping objects on shelves. They posed this challenge three years consecutively but stopped in 2018 to switch their focus to direct funding of specific research. In the paper of Eppner et al. (2016) they describe their winning system from the 2015 challenge. Their end-effector for picking objects was a clever design, which alleviated some problems a normal gripper might have. The usage of a vacuum cleaner and its crevice nozzle equipped with a small suction cup at its tip helped picking objects of up to 1.5 kg. The setup is shown in figure 13. This slim design has the big advantage that it can grasp objects even in small spaces, e.g. standing in between other objects. The object detection is done based on RGB-D data from a camera on the robot's forearm. The arms agility helps to scan the shelf where the objects are located. First, they track the shelf itself, with all its object bins, via the depth map, and extract local features from the bin regions. With this information, they estimate the probability that a pixel belongs to an object and then assign labels to each of the estimated object pixels. With these labels, the objects get segmented in the image, and finally, a bounding box is fitted with an ICP-algorithm. Jonschkowski, Eppner, Höfer, Martin-Martin, and Brock (2016) describe this detection process in a second paper (Jonschkowski et al., 2016). Their evaluation was the Amazon Picking Challenge where they scored 148 of 190 points, taking 87 seconds per pick. This means they were able to pick ten of the twelve objects and could maximally try to pick 14 times in the 20 minute time frame of the challenge. In comparison to second- and third place, they won by a vast margin, as those only scored 88 and 35 points respectively. They attribute the reason for

this large margin to the second place by clever design decisions, not only for the gripper, but different things like the camera placement and leveraging their mobility for the generation of better RGB-D data results (Eppner et al., 2016).
The winner of the Picking Challenge two years later in 2017 had a very interesting approach for picking objects out of a bin. In that year's challenge, you had to stow specific objects from a very cluttered bin into a different one. Zeng et al. (2018) split the problem into two parts. Firstly, they use a category-agnostic affordance prediction algorithm to calculate a grasping possibility. This selects one of four different primitive grasping behaviors to execute on one of the objects in the bin because it is category-agnostic this object selection is random. Only in the second stage, they use a cross-domain image classification framework on the picked object to identify it. This network can handle previously unseen objects by matching them with product images from the web (as it makes sense for Amazon products). Through this two-stage procedure, the object is not occluded by anything and much easier to identify. When it is identified to be one of the correct objects, it is stowed into the output bin. If it is not one of the correct objects it is placed into an intermediate bin. As a side note, because of this approach, the system does not actually use pose detection, because it only makes an affordance prediction. The challenge results were a 100% success rate for recognition of the picked object and 66.65% for picking attempts. They where the only entry to succeed in stowing all objects within the allotted time-frame (Zeng et al., 2018).
These examples show how individualized these systems still need to be in order to perform well enough for real use-cases. But in actuality, it is remarkable that even at the intersection of two only partially solved research topics such performance can be achieved.

4.4.2 AR and object-pose recognition

Similarly to robotics, AR and pose recognition have a natural connection as well. The fact that AR and robotics have the goal to interact with the physical world in some way, makes obtaining useful 3D information very important. The previously mentioned AR glasses, like HoloLens, use their sensors to make a 3D mesh of its surroundings to position (register) itself in space. This problem is called **S**imultaneous **L**ocalization **a**nd **M**apping (SLAM) and is similarly important for autonomous robotics as in AR, as Durrant-Whyte and Bailey (2006) describe in their article (Durrant-Whyte & Bailey, 2006).
What SLAM does not offer, at least in current AR technology, is actual recognition of objects in the constructed 3D scene. Because of this, different systems were proposed to solve this problem. Garon, Boulet, Doironz, Beaulieu, and Lalonde (2016) developed a system that provides additional high-resolution data on the HoloLens. They attached a *RealSense* depth-camera to it, which communicates across a desktop PC with the HoloLens. The depth data that is sent to the PC can be processed and then send back to the HoloLens, which is preferable to just sending the raw data because of bandwidth limitations. They present the usefulness of their system by implementing pose detection for small objects using the template-based approach mentioned in section 4.3.2 by (Hinterstoisser, Holzer, et al., 2011), (Garon et al., 2016).
Sun, Kantareddy, Bhattacharyya, and Sarma (2018) used the proposed system from

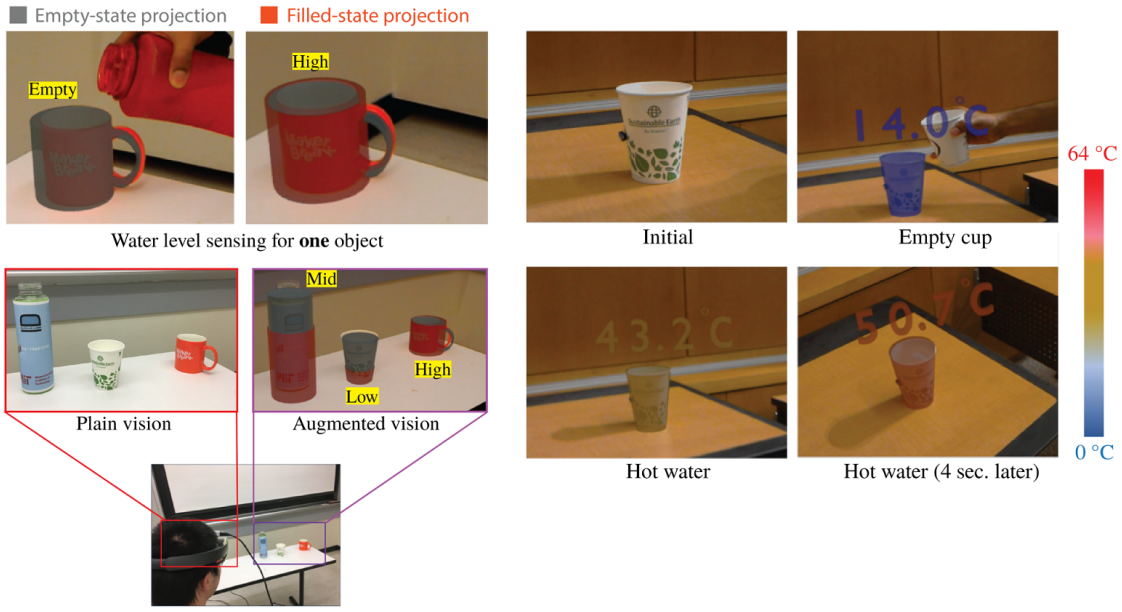


Figure 14: Augmentation of objects with sensor data from RFID chips. Left: Fill state of a cup or bottle. Right: Temperature of liquid in cup (Sun, Kantareddy, Bhattacharyya, & Sarma, 2018).

Garon et al. (2016) as a basis for their AR tool *X-Vision*, which can detect objects and their corresponding poses via the RealSense depth camera and additionally extract different sensor information via **R**adio-**f**requency **Id**entification (RFID) chips attached to these objects. Their object-pose detection pipeline uses a basic local feature extraction technique, refined by an adapted version of the ICP algorithm using the point-cloud data of the depth camera. A parallel pipeline is collecting the RFID sensor data with a simple antenna plus reader setup. Both the pose and RFID data is send to a cloud server, which processes them and sends the resulting data to the HoloLens. Figure 14 exemplifies some possibilities of information that can be displayed to the wearer of the AR glasses. An experiment showed that a stable range for accurate recognition between the camera and target is from 40 cm to 75 cm and for the readability of the visualization around 100 cm to 150 cm (Sun et al., 2018).

One of the most commonly advertised applications of AR is assisting with assembly tasks. Y. Wang, Zhang, Yang, He, and Bai (2018) developed an assembly assistance application, which is marker-less and has stable tracking, which is normally not the case and therefore a big negative of most similar applications. Markers are often not practical or possible in industrial settings, which reduces the usefulness of such systems. Y. Wang et al. (2018) based their tracking, similar to (Garon et al., 2016), on the work of (Hinterstoisser, Cagniart, et al., 2011) combining it with the **O**riented FAST and **R**otated **B**RIEF (ORB) algorithm developed by Rublee, Rabaud, Konolige, and Bradski (2011). ORB is a fast alternative to the aforementioned SIFT (4.3.1), which is two orders of magnitude faster with similar matching results (Rublee et al., 2011). Y. Wang et al. (2018) use a common two-stage approach (offline and online phase), where the offline part generates templates for each assembly step, from which the ORB features are extracted. These results are saved in a E**x**tensible **M**arkup **L**anguage (XML) file, for

ease of access to other systems. In the online part, these features are extracted from the live camera image and matched with the previously saved data to calculate the camera pose. Concluding with overlaying a 3D model of the next assembly pieces to be attached. Through the use of the template matching in combination with the ORB feature extractor, which are two very time-efficient algorithms, they achieve stable 30 FPS (Y. Wang et al., 2018).
In contrast to Y. Wang et al. (2018), Su et al. (2019) used a CNN as the recognition backbone. Their focus is the detection of multi-stage objects, which Y. Wang et al. (2018) did not consider and simply have the user pick the current assembly stage. Su et al. (2019) combine two CNN architectures to generate a new network that has a state estimation and pose estimation branch. The used networks are TridentNet (Yanghao Li, Chen, Wang, & Zhang, 2019) and R-CNN (Ren, He, Girshick, & Sun, 2015). The difficulty for stage detection is the high ambiguity because of the similarity of the objects and the fact that the different stages have their own geometric center. This means the pose cannot easily be calculated based on the same local coordinate system. Through their combination of the network structures, they solve this problem by not relying on the 2D bounding box. They achieve 30 FPS as well because they split the detection of pose and stage. The higher performant and more important pose detection runs every frame (at 30 FPS) and the stage estimation with 2 FPS. After pose and stage of the object are found, the next assembly step is overlaid in AR similar to Y. Wang et al. (2018) (Su et al., 2019).

Hettiarachchi and Wigdor (2016) have a much more uncommon and novel use for pose recognition in AR. They use a *Microsoft Kinect* to obtain depth data for their AR system, which they call *Annexing Reality*. The idea is to integrate haptic feedback into AR by dynamically searching the environment for objects that have a similar shape as the virtual models they want to display. Then overlaying the 3D model over the real object, which acts as a proxy for an actual prop with the exact shape of the virtual one. In figure 15 an example of the working system is shown, which tries to find the best fit for all available 3D models and real objects. This system works by finding primitive shapes in the point cloud generated by the Kinect and scaling the 3D models to fit the proxy objects better. To find these matching objects they detect horizontal surfaces in the environment, extract objects on top of them, and extract their features (like size, primitive shape, orientation). After this is done, all real objects are compared with all virtual ones and a voting scheme decides if they fit or not. When an object is identified for tracking, their object pose tracking pipeline takes over and positions it at the correct pose in space, realigning it every frame. The tracking pipeline cuts a cubic area around the rough pose of the object in the point cloud and performs an ICP algorithm on it. The rough pose is generated with a RANSAC based pose estimation. A small study was performed to test the **U**ser **I**nterface (UI) and the satisfaction with the system's performance. The results showed that the participants were generally satisfied, but the predictability of the voting scheme needs to be improved in the future (Hettiarachchi & Wigdor, 2016).

Figure 15: Annexing Reality finds the best fit for 3D models in the real world. (Hettiarachchi & Wigdor, 2016).

One important fact becomes clear when looking at AR applications with pose recognition. Performance is very important for the usability of different applications. Not only for the FPS of rendering but also for pose estimation. If pose recognition is done too sparsely it cannot follow the users' movement. One problem here is the inherent mobility of AR devices like smartphones or the HoloLens, which limits their computing power. Zhang, Han, and Hui (2018) for instance offload the pose estimation to an *Edge Cloud* to reduce computational load (Zhang et al., 2018). Another approach by C.-H. Lin, Chung, Chou, Chen, and Tsai (2018) is to involve GPS data to only consider objects in close proximity to the user in their navigation app (C.-H. Lin et al., 2018). To only name a few.

4.4.3 Combining the three: Robotics, AR and object-pose recognition

This final section deals with research focused on the same intersection of technologies which were discussed in the previous three chapters of this thesis, robotics, AR, and object-pose recognition. One direction of research is the visualization of sensor data specifically for autonomous robots and their object detection systems. These are primarily intended for the development and debugging of such systems with help of AR visualization (Chen, Wulf, & Wagner, 2006; Stilman et al., 2005). This idea makes sense because it makes debugging and refining the recognition systems much easier when you just see what the robot *thinks* it sees/recognizes.

More closely related to this work is the research done by Gao and Huang (2019) (Gao & Huang, 2019), Gong, Ong, and Nee (2019) (Gong et al., 2019) and Rudorfer, Guhl, Hoffmann, and Krüger (2018) (Rudorfer et al., 2018). All three research teams use AR and object detection as a means to simplify the work of programming a robot for pick and place tasks. Gao and Huang (2019) and Gong et al. (2019) both utilize a projection-based augmented tabletop and acquire RGB-D data from the *Kinect 2* camera. Despite their many similarities they do not refer to each others' work, this may be due to the fact that two of the papers came out the same year, 2019. In the following the differences between the approaches are described.

Gao and Huang (2019) mount the Kinect 2 to the ceiling and use it for tracking the user's

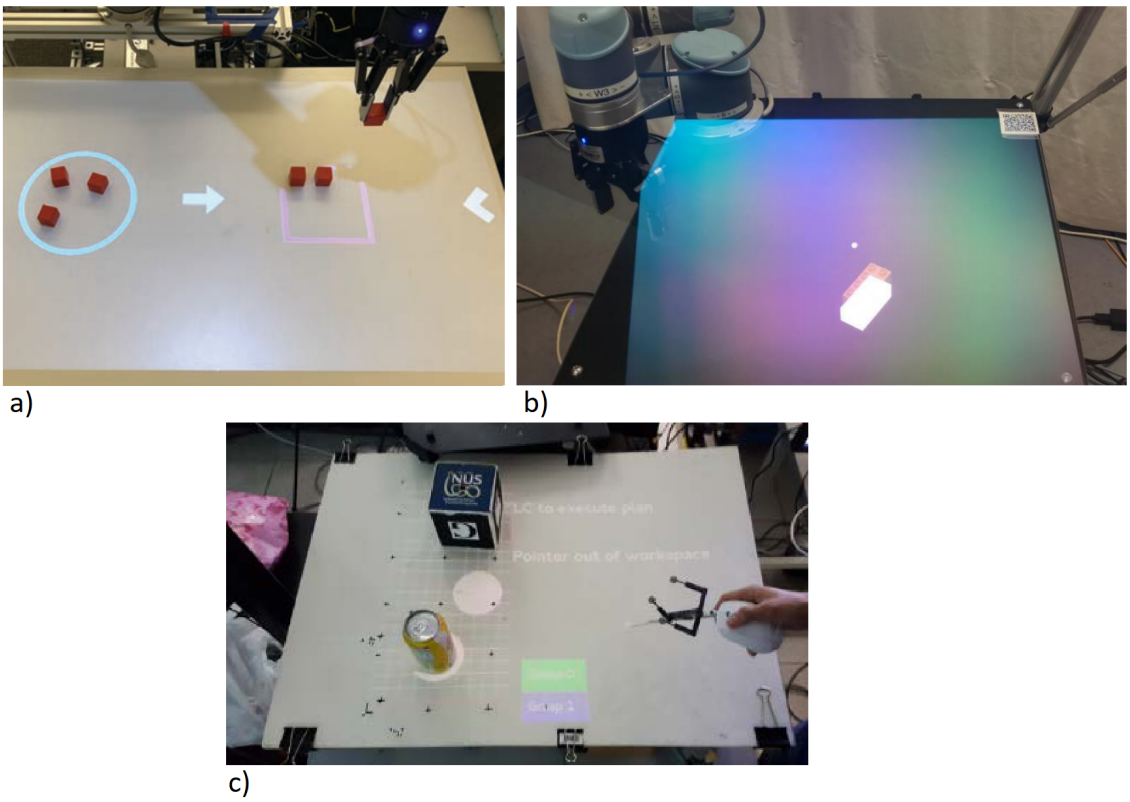


Figure 16: Visualization contrast of three similar robot programming systems through AR. a) is the projector-based tabletop by (Gao & Huang, 2019). b) is the view through a HoloLens, by (Rudorfer, Guhl, Hoffmann, & Krüger, 2018). c) is a second projection-based AR system by (Gong, Ong, & Nee, 2019).

hand gestures as well as the objects lying on the table. A point-cloud library uses the generated data to detect the objects poses on the table, which are then refined by the use of the RANSAC algorithm. Their focus lies on the interface projected on the tabletop, which can be used to program the robot with simple touch gestures. Figure 16 a) shows how the user can define areas in which i.e. objects should be picked and a destination area where picked objects are then placed by the robot. As an evaluation of their system, they performed a user study with 17 participants in which they tried to determine whether their system is more efficient than the built-in system of the robotic arm, in regards to learning time, confusion, and programming speed of the user. Their results indicate that the proposed interface was better on all measured attributes (Gao & Huang, 2019).

Gong et al. (2019) in addition to the Kinect 2 use *OptiTrack* an infrared motion tracking system. As a programming platform, they use the **R**obot **O**perating **S**ystem (ROS), which is widely used in the world of robotics. Most of their subsystems use existing modules from ROS, like the **O**bject **R**ecognition **K**itchen (ORK) or *RVIZ* (ROS Visualization). The role of the augmentation on the tabletop is to give the user guidance information through the process of task definition (robot programming). The help consists of information or error text prompts, buttons to select desired configurations, and a shadow of the selected object at the pointer location. This shadow, therefore, shows the objects placing location in 2D. The visualization is visible in the comparison figure 16 c). For placing they assume the tabletop as standard height, except when an object is detected at that location then they add the additional object height to it in order to not crash into anything. With

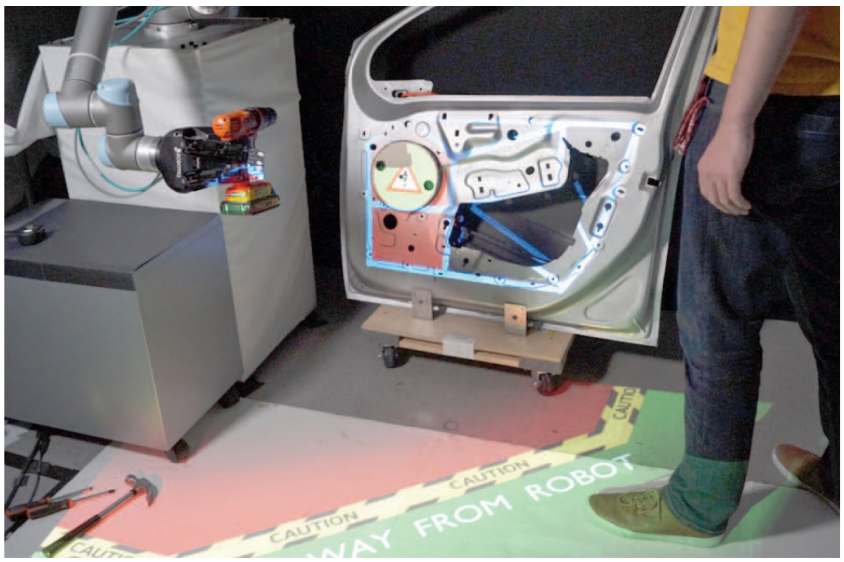


Figure 17: Example of visual cues for HRC. Robot working area on the ground and assembly helper on the car door (Ganesan, Rathore, Ross, & Amor, 2018).

their interface, the user can define several pick and place tasks, that are then executed in order. Gong et al. (2019) call this assembly planning, as each task can be part of a bigger assembly task (Gong et al., 2019).

In contrast to Gao and Huang (2019) and Gong et al. (2019), Rudorfer et al. (2018) focus on the adaptability of their system and separate each module they use to make switching them out very easy. In this proof of concept, they employ notably simple components which later on may be switched out, corroborating their modular design. In this system the user sees which objects are detected through the HoloLens' display, via visualization of the objects 3D model overlay, visible in figure 16 b). Now the user only needs to place the object at a different position, through the HoloLens gestures he can select an object and new position via air-tap. The object detection is using a simple rectangle recognition of the library *OpenCV* [1], limiting the possible detection to very simple blocks (Rudorfer et al., 2018).

The final paper is called *Better Teaming Through Visual Cues* by Ganesan, Rathore, Ross, and Amor (2018). Their system is designed for HRC and displays visual cues. Firstly, it provides instructions for procedural tasks by recognizing the pose of the assembly piece and projecting the next steps onto the work environment. Secondly, they give robot attention cues, which gives the human information on what the robotic arm will be doing and where it may not be save to stand. Figure 17 shows both types of cues. The working area of the robotic arm on the ground and some blue assembly helper lines on the car door. Unlike the previously mentioned approaches, they spend a considerable amount of research into the pose recognition focusing especially on occlusion. Which makes sense, because the human worker is often in front of the assembly piece, occluding it. They dedicated a second paper to their algorithm Brahmbhatt, Amor, and Christensen (2015), in which they describe their approach. It is based on the HOG templates mentioned in section 4.3.2. Brahmbhatt et al. learn not only object templates, but also occlusion templates which are used when the matching is weak. The object template in that area is then

[1] https://opencv.org/

replaced with the occlusion template, which is then used for matching the object.
Finally, a user study with 15 participants was performed, in which they tested the efficiency and effectiveness of their visual cues against a version with printed instructions and a mobile screen. Additionally, they tested accuracy and understanding time for the three types. A statistically significant improvement of their visual cues over the other two in all tested categories was reported. Especially the task understanding time is only a fraction for two of the three sub-tasks the subjects had to perform. Subjectively the participants preferred the visual cues. Major positive themes were the perception of their own abilities, the systems performance and the HRC experience in general (Ganesan et al., 2018).
The discussed research shows that often in these complex systems the actual pose recognition part is very basic, with the exception of Ganesan et al. (2018). This makes sense, due to the fact that the focus is often more on other parts of the system. In the approaches for robot programming the use of depth-cameras seems reliable enough to use, especially because not a very high precision is needed. Gong et al. (2019) are interestingly close to visual cues that will be used in this system, namely their shadow. Only their projector-based AR limits them to 2D.

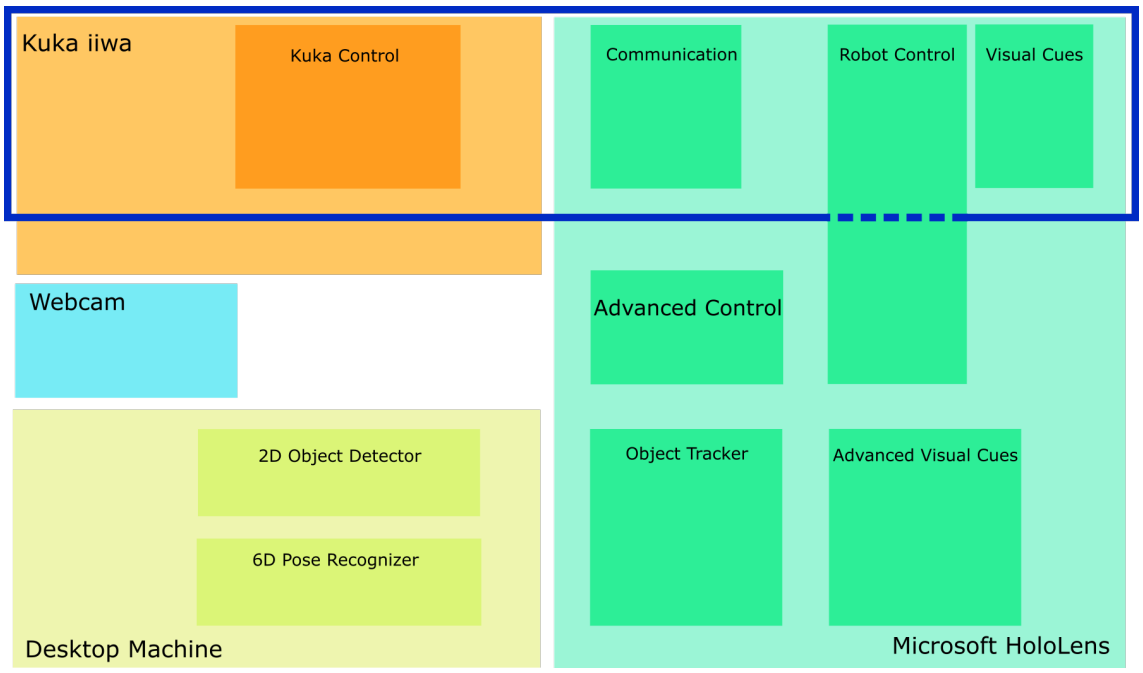


Figure 18: Simplified overview of the developed application. The blue outlined area is external from this thesis, by Franziska Rücker. The lighter shaded rectangles describe the hardware devices, while the darker ones are logical components.

5 Method

With knowledge from the research, we can take a closer look at how the underlying system is constructed and discuss why certain decisions were made. Figure 18 gives a rough overview of the most important components of the system. Including the ones from Franziska Rücker's master thesis, which is marked with a blue border. A more thorough description will be given in section 6.1. Here the important aspect is the separation of what is part of this thesis and what is not. Additionally, it is important to clearly state the constraints that had to be considered. These constraints are categorized into different aspects, like the used technology, the working environment and the users of the system.

Constraints

Users:

The target group are physically impaired people who are still mentally fit. Mainly tetraplegics or otherwise paralyzed people who cannot move their lower body and are bound to a wheelchair. The resulting system should help them integrate into work life, assembling objects, and providing them with more autonomy.

Technology:

The Microsoft HoloLens is a see-through HMD-AR device that has built-in SLAM to orient itself in space. For input, it provides simple gestures and additionally speech input and head tracking through SLAM. This device not only fills all initial requirements mentioned in 1.2 but is also available for our research.

The robotic arm is a KUKA iiwa, which is designed for HRC. It can lift up to 7 kg and has a maximum reach of 80 cm. The seven joints give it the same DoF as human arms, having almost infinite possibilities to reach a specific point in space within its reach. It is

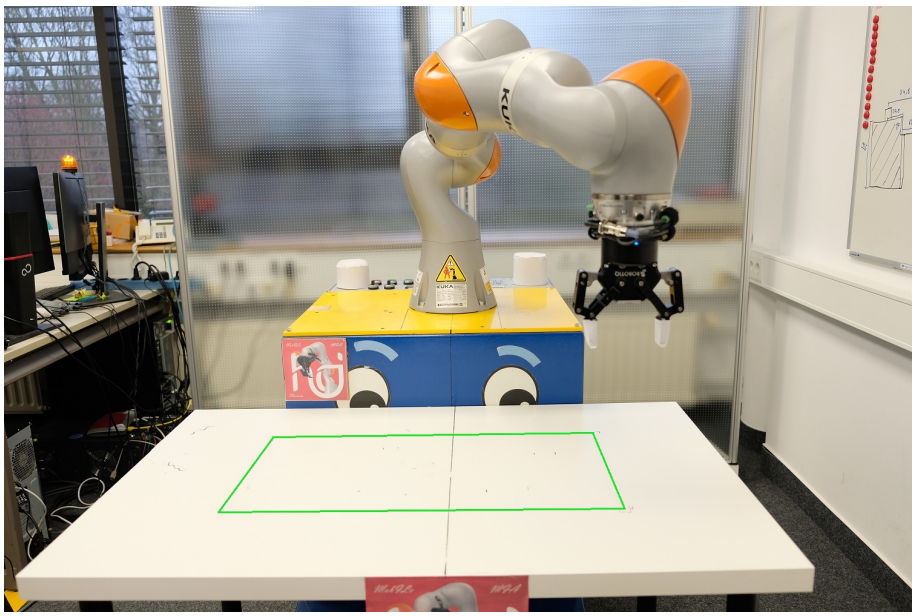


Figure 19: Workstation with the Kuka iiwa in its standard pose, from the view of the user. The working area is outlined with a green rectangle.

equipped with torque sensors in each joint to prevent high force collisions for close-range interactions. This arm was put in operation during the time of this thesis with the help of Franziska Rücker and other members of the HCI-Group of the Westphalian University of Applied Sciences and is the only available robotic arm for this project.

Environment:
We have a clearly defined region in which the robotic arm is working. This area is a rectangular shape on top of a table with the dimensions 28,5 cm x 60 cm. The operator sits in front of this table with a distance of about 120 cm. The working area was defined in the early prototype this project is based on, because of the nature of assembly tasks at hand (small part assembly) and the fact that our target group is not agile. The setup is depicted in figure 19.

Robot Control
One of the big decisions for the operation of the robotic arm was to limit its main movement to be in a specific horizontal plane, 40 cm above the tabletop, in order to prevent collision while keeping the broad movement simple. The exact nature of the movement will be presented in section 6.4. Additionally to the movement plane, the decision was made to keep the gripper facing downward at all times. This reduces the possible rotation to be only along the up-axis, keeping the control as lightweight as possible. Figure 19 shows the standard pose of the robotic arm. This is an ambivalent trade-off that was made because, on one hand, it limits the usefulness for assembly tasks and to rotate a wrongly placed object quite a bit. On the other hand, it does simplify many things for the control scheme, which is important for the user base. Making this scheme as simple as possible, while maintaining all possible movements of the robotic arm, was not the main focus of this project and therefore the decision was made in favor of reduction and simplicity.

Input Modalities

For input modalities, we settled on speech and head movement. As highlighted in section 2.2 another option would have been eye-tracking or more technologically novel approaches like brain-interfacing. The latter one were not considered, because they have many different disadvantages like the need of additional sensors on the user (brain-interface, EOG), additional technology which was not present, or they are in early research stages and therefore not very stable (brain-interface) and many need calibration for each usage. The last two disadvantages are also reasons why we disregarded eye-tracking.

In previous research, we used the *PupilLabs*[1] eye-tracker, which had a tedious calibration phase and was still very unstable to the point of uselessness. This PupilLabs eye-tracker is an attachment for the HoloLens, which was the reason for the poor performance. The HoloLens fixation to the head is build to leave wiggle-room and oftentimes slides a tiny bit on the user's head during movement. For the eye-tracker, each movement of the HoloLens ruins the calibration, because it assumes a fixed view on the eye. While an external eye-tracker may not have these drastic disadvantages, they were not an option because using HMD-based AR is the basis of the project and therefore the user will wear some sort of device on his head occluding the eyes. Furthermore, it does seem that eye-tracking is in general not at a point yet where it is a viable replacement for head-tracking/head-pointing. Qian and Teather (2017) made a Fitts' law based study to compare selection times and accuracy of head movement, eye movement, and a combination of both. The results show that the head movement performs the best and is also strongly preferred by users (Qian & Teather, 2017).

The choice to use head movement was supported by a few strong arguments. Firstly, head-tracking is technically mature and therefore robust enough to use reliably. Secondly, it is a versatile input modality. It can be used for discreet input, by pointing at a dwell-button, or for more continuous input with head tilt or point distances. Finally, it was clear that we use the HoloLens based on the kind of AR we want to utilize in this project, which already has head-tracking integrated into the device.

Even with the disadvantages of false positives, speech input was the most appropriate addition to head-tracking. It is similarly already provided by the HoloLens, thus no additional hardware is required. Furthermore, these two modalities work well together, because they can easily be used simultaneously as humans do every day. The last positive is that for input no visual clutter is generated, which keeps the users view free.

Pose Recognition

Arriving at the final setup of the pose recognition, based on initial technological constraints, was a long-winded process. The first constraint is the 3D-mesh the HoloLens builds of its environment. The idea was to use this mesh to forgo the need for object recognition, just using the bumps in the mesh generated by objects that extrude from the working area for the visual cues. Experiments showed that the created 3D-mesh is very coarse and smooths over anything that is just a small object extruding from a larger

[1] https://pupil-labs.com/

plane. Even objects of up to 15 cm were reduced to a minuscule bump in the mesh, making it not usable for this purpose.
With actual object pose recognition needed, it became clear that the HoloLens' infrastructure for object recognition was not sufficient for our purposes. The only large recognition systems that are available on the HoloLens are the, in chapter 4.3.1 mentioned, Vuforia SDK which is only able to detect highly textured objects and an incomplete port of the *OpenCV* [1] library. Vofuria is used in this project for basic marker detection for the initial alignment of the coordinate systems of the HoloLens and the Kuka iiwa robotic arm.
Because of the relatively low computational power the HoloLens has, in combination with missing infrastructure, the decision was made to outsource the actual pose recognition to a separate desktop machine with which the HoloLens communicates. This leaves the HoloLens' resources for the actual application, which is important because a stable and high frame rate is necessary for appropriate application quality. Additionally, the possibility to use multiple cameras is opened up, as long as their position is known in the HoloLens coordinate system. This enables different viewpoints of the working area than the user has, a higher resolution image than the HoloLens Webcam for better detection and the camera can also be much closer than the user itself.

The considerations for the pose recognition network that was used as a basis of this recognition system, are the following: firstly, one of the most important aspects is the fact that normal assembly task objects often have almost no texture, which makes these kinds of feature-based approaches (like Vuforia) not suitable. Secondly, in this scenario where the working area is limited, partial occlusion among objects and from the robotic gripper itself is common. This gets further amplified when objects are rearranged and grasped by the gripper. As described in 4.3.2 template-based approaches handle textureless objects well but have bigger problems with partial occlusion, leaving them not very robust in these situations. Another aspect, which was lessened in importance by the possibility of multiple cameras, is the detection efficiency for distances of 1.5 to 2.5 meters from the working area. As the user should be placed outside the maximum envelope of the robotic arm for safety reasons, even with the Kuka iiwa which has the ISO 10218-1 certificate for HRC.
With these constraints in place, we decided that using a state-of-the-art CNN-based network would be the best approach. Based on this decision another aspect/constraint comes into play. A neural network needs a large amount of labeled training data to learn from, which is difficult to generate as elaborated in chapter 4.2. Because of this, the system by Sundermeyer et al. (2018) was chosen as the basis because it fits the constraints very well (Sundermeyer et al., 2018). It focuses on the synthetic generation of training data, partial occlusion robustness, symmetry robustness, and can handle textured as well as textureless objects. The basic idea of how this network works is described in chapter 4.3.3 and in more detail in 6.6.2.

Additional considerations

[1] https://opencv.org/

With knowledge of the working area, it is possible to adjust the found pose of objects, if it is not perfectly aligned to the table surface. If an object is virtually placed inside or slightly above the tabletop it can be adjusted and therefore alleviate small detection errors of this kind. The same can be done for small rotational errors, where a flat surface of the object is tilted and therefore not flat on the table, which can also be adjusted.

The system by Su et al. (2019) discussed in section 4.4.2, utilizes different update frequencies for their separate sub-systems. The recognition runs at approximate 30 FPS and the state estimation with only 2 FPS. This approach makes sense for this application as well, because the HoloLens application needs to run at 60 FPS, but the pose of the object can be tracked much less frequent for two major reasons. The first is the fact that a once recognized object can simply be placed in the virtual environment of the HoloLens, which itself tracks its pose in space. Once the object is placed and not moved in the physical world, there is no need for another pose recognition as it is simply held in place by the HoloLens' SLAM algorithm. The second reason being that the manipulation of the objects is well known from the applications point of view, as long as no user error occurs. When an object is grasped it will move exactly as the robotic arm does, and its movement is known/controlled through the application, making the object manipulation also well defined and needing no further pose recognition in between picking and placing the object. Only when the object is pushed by the gripper without grasping it, falls over or something similar happens, a new pose recognition is required. This, in theory, allows a much less frequent update rate of the pose recognition, possibly even only on certain events.

One approach that is used by a few systems presented in the literature review, like the one from Hettiarachchi and Wigdor (2016), is to utilize an additional depth camera to get the missing data the HoloLens does not provide (Hettiarachchi & Wigdor, 2016). With the integration of additional cameras to add a different viewpoint of the working area, it makes sense to consider employing depth-cameras and use the additional information in this application. Depth-cameras were disregarded as they introduce more processing overhead, system complexity, and additional hardware which was not feasible in the time-frame of this thesis. Therefore it will be further discussed in the future work section 8.1.

The developed advanced visual cues which will be described in section 6.5 utilize the pose recognition in different ways. The most complex one is the *Object Ghost*, which is similar to the *Object Shadow* of Gong et al. (2019). The key difference being the three-dimensionality of this ghost in contrast to the 2D version Gong et al. (2019) use. The advantages of this method will be highlighted in the aforementioned section 6.5. Gong et al. (2019) additionally use an object aware place which is implemented in this application as well. Details of which are illustrated in section 6.4 (Gong et al., 2019).

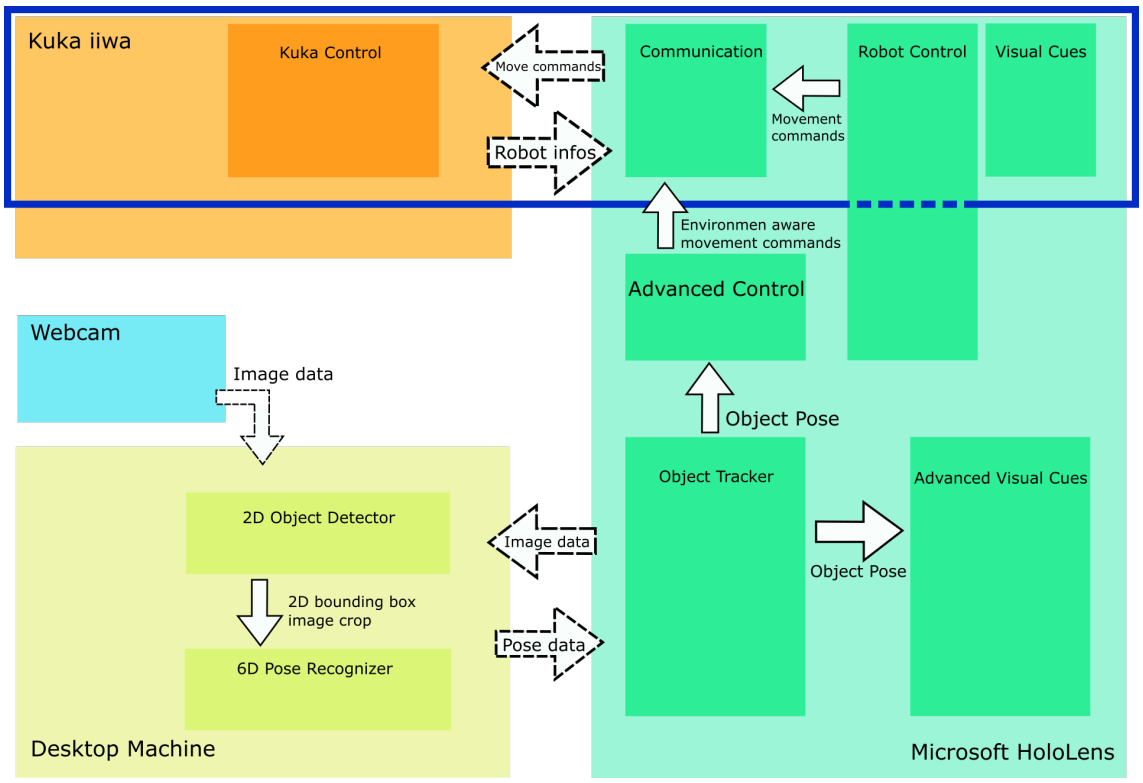


Figure 20: Overview of the developed application. The blue outlined area is external from this thesis, by Franziska Rücker. The lighter shaded rectangles describe the hardware devices, while the darker ones are logical components. The arrows indicate communication between components (continuous line) or devices (dashed line).

6 The application prototype

The application developed for this theses will be described in the following sections. Starting with a general system overview with a precise description of what the different system components are, how they interplay, communicate, and what data is passed between them. For clarification of what the user is doing, while using the application, the workflow will be run through. Subsequently, the general interaction types and control concepts for the robotic arm with the added benefit of object pose information will be described. Following a depiction of the advanced visual cues and lastly all components of the object pose recognition system.

The implementation for the HoloLens application was done in *Unity* [1] in combination with the *HoloToolkit* [2], a plugin for Unity. C# is the used programming language for this part. Python is the programming language used for the pose recognition system, which runs on *Ubuntu* [3] 16.

6.1 System Overview

The complete system consists of three separate devices, visualized in figure 20 with the lighter shades of each color yellow, green, and orange. The orange is the control cabinet of the Kuka iiwa, which is responsible for the robotic arm. This includes its safety

[1] https://unity.com/
[2] https://github.com/Microsoft/MixedRealityToolkit-Unity/
[3] https://ubuntu.com/

settings and general controls. For our prototype, a small application is running on this cabinet to communicate with the HoloLens via an **U**ser **D**atagram **P**rotocol (UDP) interface. It can process incoming movement, rotation, and gripper state changes through a simple protocol and sends back information on gripper positions, state, and completed movements. As visible in the figure, this communication is part of Franziska Rücker's thesis and will be described in detail there.
The HoloLens application consists of six logical main components, of which three are part of Franziska Rücker's thesis as well. The Communication component is the interface to the Kuka iiwa but is additionally responsible for the conversion of spatial data between the two coordinate systems of the hardware devices. The Kuka iiwa's and HoloLens' coordinate systems have different handedness and are not aligned in space. Robot control is for the basic environment-unaware control of the robot, like gripper rotation and precise movements. As the figure suggests, this component was created in cooperation with Franziska Rücker, with no clear boundaries. As the general movement is also important for the understanding of this thesis, the functionality of the individual control commands will be described in section 6.4. The last component, that is external to this project are the visual cues. These are virtual helpers that work completely without object detection, only through use of the spatiality of the holograms.

The Object Tracker component is handling everything needed for the pose recognition on the HoloLens application side. It captures a webcam image, sends the raw image data across the local network to the desktop machine via a **T**ransmission **C**ontrol **P**rotocol (TCP) connection and processes via UDP incoming pose data from the desktop machine. Two different protocols are used to reduce the communication overhead as much as possible. TCP is able to stream a large amount of data, which is needed for the raw webcam image. Whereas it is much simpler to send small data packets via UDP which can only send a maximum of 65.535 bytes - 20 bytes (Size of the IP header) = 65.515 bytes (including 8 bytes UDP header), but is very lightweight. Additionally, the Object Tracker processes the pose data and places the corresponding object inside the virtual environment of the HoloLens. With redundancy checks and small pose corrections. This is further explained in section 6.6.
Advanced Control is the component responsible for all movement of the robotic arm, which is environment-aware. This mainly includes picking and placing and is also discussed in section 6.4 together with the other movement scheme.
The final component on the HoloLens side are the Advanced Visual Cues. They use the object pose data to create virtual helpers that are environment-aware and are therefore able to support the user drastically with manipulation of an object. These cues are *Occlusion Cue*, *Gripper Region Indicator*, *Laser Pointer*, *Virtual Extension* and *Ghost Object*. They are described in detail in section 6.5.
On the desktop machine are two components and a directly attached webcam. The 2D Object Detector waits for incoming image data, regardless of which webcam they originate from (HoloLens or directly attached), and processes the data by detecting all known objects and making rectangular crops of them with corresponding labels. These crops are then given to the 6D Pose Recognizer, which uses these crops to infer an object pose in

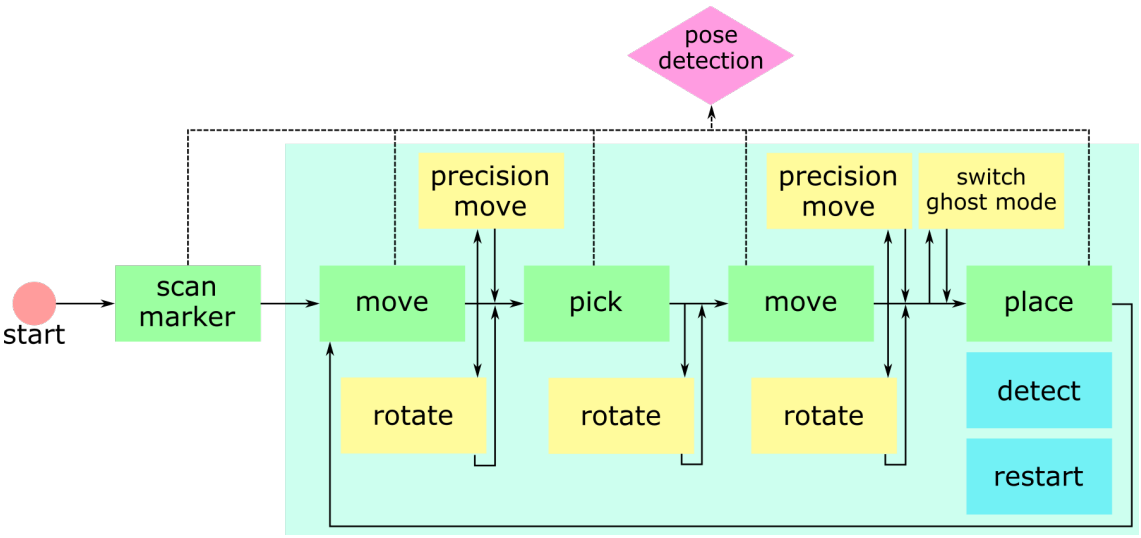


Figure 21: Intended workflow for the prototype application. The rectangles represent user interaction and can be repeated as often as necessary, while the diamond shape express independent system actions. The green rectangles define the *main path* of the application which cannot be skipped. The yellow ones are optional. The blue rectangles can be activated at any time in the big light blue rectangle.

camera space. This data is then sent to the HoloLens as a bundle, where all recognized object data is concatenated. The data consists of a 4x4 transformation matrix containing the object rotation and position. The details for this are explained in section 6.6.

6.2 Workflow

Multiple hardware devices and different subsystems like the robot control, visual cues, and the pose recognition system give this system a certain complexity. To show that the user's actual interaction-loop is straight forward, figure 21 gives an example of the application interaction workflow. Right after the application is started the user needs to scan the marker to set up the virtual environment, after which they are in the main loop of the application. The path of the green rectangles represents the minimum interactions for moving objects around the workspace. Each time one of them is triggered, the pose recognition system rescans the area, indicated by the dotted lines and the violet diamond shape. During the green main loop, the user can intersperse the rotation and precise movement for alignment at any time. The *switch ghost mode* action toggles the helper for placing a picked object and therefore only makes sense after the user picked something, and then moved the arm again, before activating it. In any situation after the marker was scanned the restart or the detection command can be issued if something unexpected happens.

6.3 Interaction types

The two input modalities that are used in this prototype are speech and head gaze, as mentioned before. The specific interactions for each one will be described in this section. The speech input is very basic, in that it uses only single-word commands that the user

can issue.
Following is a list of all commands:

- *Pick:* Lowers gripper, closes it and goes back up
- *Place:* Lowers gripper, opens it and goes back up
- *Open:* Opens the gripper
- *Close:* Closes the gripper
- *Turn:* Rotates gripper to cardinal direction
- *Move:* Moves the gripper to the gaze location
- *Cancel:* Cancels precision movement mode
- *Occlusion:* Virtually overlays all currently recognized objects with their 3D model
- *Detect:* Updates object poses
- *Restart:* Resets the gripper to its default starting position

These commands will be described in more detail in the following sections.
The head gaze has two different interaction types. The first one is its use as a normal cursor for pointing, like a computer mouse. This interaction happens for dwell buttons, which are displayed in figure 22 a). When the gaze cursor is above one of these buttons, the small white pin at the top of the button spins around the circle once. When it reaches the end, the button is activated. This *dwell time* guards against accidental activation. The second interaction type uses the cursor head gaze at its foundation as well. But here only the distance and direction from a certain position is important, not its exact position. This interaction is used for the *Precision Mode*, which will be described in detail in the following section as well.

6.4 Robot control

As previously mentioned, the input methods to control the robot consist of simple voice commands and head movement. The control can be split into three different groups, the movement, the rotation, and lastly the object interaction (namely pick and place).

Movement
Broad movement is done by pointing at a specific location with the head gaze pointer of the HoloLens and then issuing the command *Move*. This triggers the robotic arm to move to the pointed location, remaining in its horizontal movement plane. If it was previously not at the correct height it first moves upwards to reach it. This enables the user to easily move the robot across the working area without needing to consider its prior height and with a single uncomplicated input. The disadvantage of this movement is its imprecision caused by the use of pointing. When the user is not holding the head steady or is not looking at the exact spot, the gripper may not be aligned perfectly.
For exactly these fine-grained movements is the second control scheme. The *Precision*

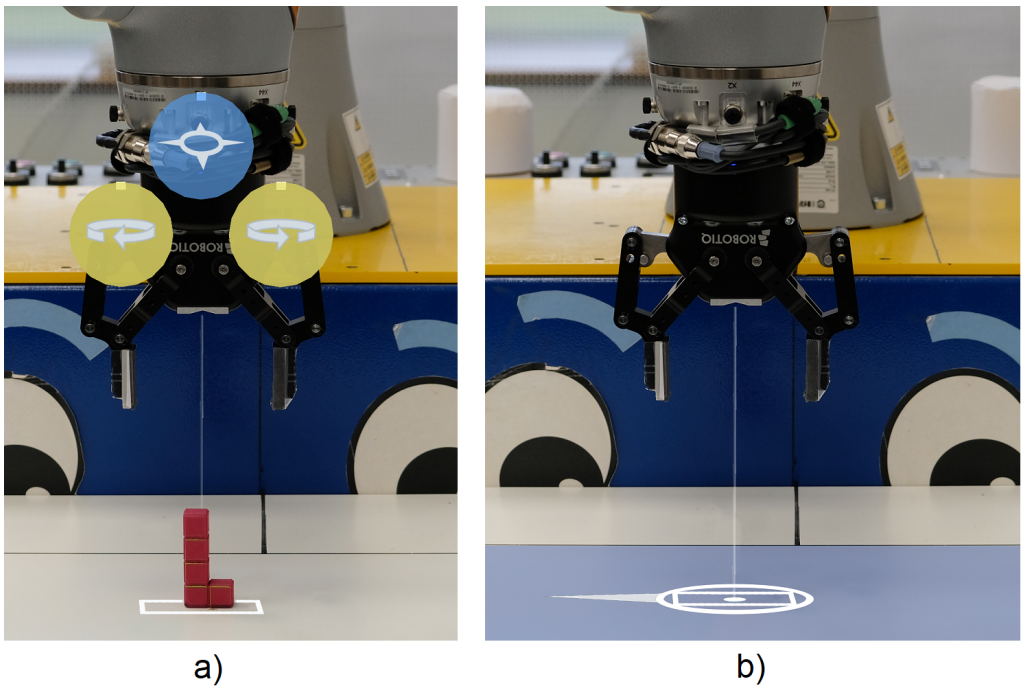


Figure 22: Robot control. a) shows the rotation and precision dwell buttons. b) is the precision mode visualization (blue underground only for visibility).

Mode can be activated by gazing at a dwell button that follows the robot gripper or with the speech command *Slow*. Once it is activated the robot gripper starts continuously moving towards the pointing location of the user in a straight line. This movement is limited to only one axis direction in the z-plane of the grippers' local coordinate system. It accelerates when the distance between the gripper and gaze pointer is larger and decelerates the closer they are together, stopping when the pointer reaches the circle in the center. To indicate this, an arrow is visualized reaching from the circle to the gaze pointer and stretching accordingly. Figure 22 b) shows these visualizations. This allows for very precise alignment of the gripper. The deactivation is done by either dwelling into the circle in the center for a short time, or if it is occluded by an object by issuing the voice command *Cancel*.

Rotation

As discussed in section 5 the rotation in this prototype is limited to one axis. It has a broad and precise manipulation type similar to the movement. The broad rotation is issued via a speech command, namely, *Turn*, that rotates the gripper from its current rotation to the nearest cardinal direction. For example, if it is at 78° it would rotate to 90° and if it was at 24° it would rotate to 0°. If it already was at one of the cardinal direction points, it automatically rotates to the other. Because the gripper is symmetric, it is not too important which direction it rotates towards.

The precise rotation is done through two additional dwell buttons, one for each direction, which follow the gripper. Different from the precise movement this is not a mode to activate and deactivate. Instead, when the dwell button is activated the gripper rotates in the corresponding direction in a 10° step. 10° seemed fine-grained enough to fit our purpose, but could easily be tuned.

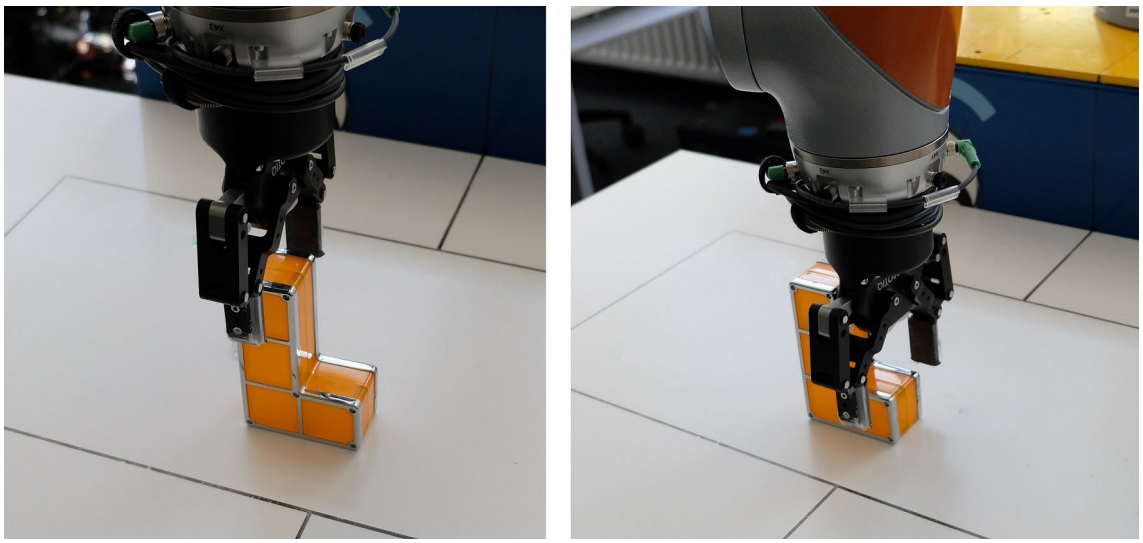


Figure 23: Example of different automatic grab heights based on the grippers relation to the object.

Object Interaction

The very basic interaction is opening and closing the gripper to grasp or release an object. The speech commands *Open* and *Close* enable these. The higher-level commands *Pick* and *Place* are leveraging the object pose recognition to simplify the picking and placing process. First, the gripper needs to be positioned over the object that the user wants to pick. Once the Pick command is issued the robotic arm lowers itself until it is in grasping reach of the object, closes the gripper, and moves back up to its initial position. The stopping height is calculated based on the table height and the scale of the detected object below the gripper. The object complexity, as long as it is correctly detected, is irrelevant because the picking height is dependent on where the gripper is positioned above the object. This is programmatically precise enough to differentiate between the different heights of the object below the gripper and lower it appropriately. An example is shown in figure 23, where based on the gripper position above the object, it lowers to the lower protrusion of the L-Shape or stays at the higher part. The robustness is entirely dependent on the precision of the pose recognition.

The advanced Placing also takes the recognized objects into account. With knowledge of the object held by the gripper and on the table, it calculates and moves the distance the robotic arm needs to lower the gripper, then opens it, and, similar to picking, moves back to its starting position. This enables the stacking of objects on top of each other without too much exhausting precise manual lowering of the gripper. If no object is detected below the gripper, the height for picking and placing is the table height.

This height aware pick tries to ride a fine line of supporting the user but not replacing their tasks by too much automation. One aspect of this is, that the pick only makes vertical movements itself and leaving the horizontal movement and gripper rotation to the user. This is additionally done because oftentimes it is necessary for the user to grasp an object at a specific point in a certain rotation, which a fully automatic pick could not predict. This higher-level strategizing is part of why a human as an operator is useful in

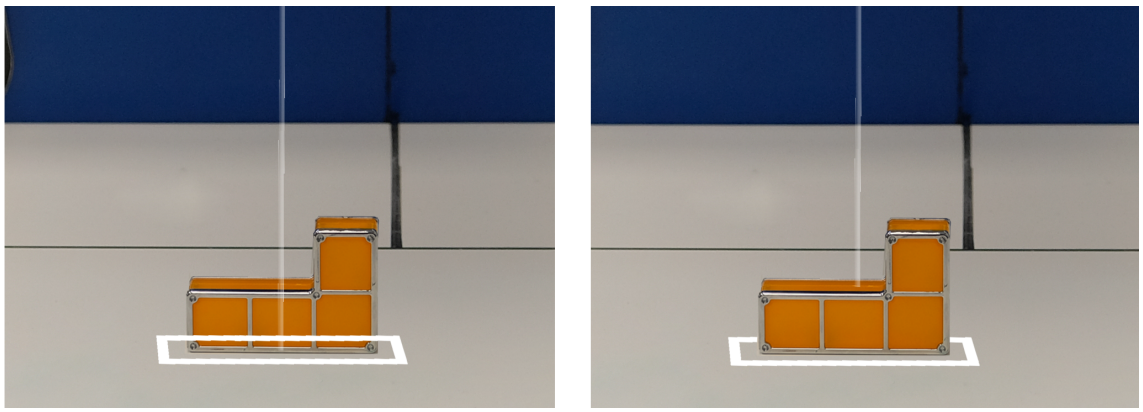


Figure 24: Example of how the occlusion cue enables correctly looking occlusion of virtual element and the real world.

the first place. Height aware placing mirrors these decisions. A more in-depth discussion on why certain design decisions for the part of this control scheme that is not utilizing object recognition where made, can be found in Franziska Rückers thesis.

6.5 Advanced Visual Cues

These visual cues are called *advanced* because they utilize the object recognition system for more environmental information. With this information, they can create better depth information and react to possible collisions to support the user, in contrast to the ones not using object pose recognition.

Occlusion Cue

The fundamental and probably most important visual cue is the *Occlusion Cue*, which is enabled by pose detection. When an object is recognized its virtual model will be placed on top of the real one. This virtual model is shaded to be invisible and **not** occlude the real world, but simultaneously occlude everything virtual that is behind it. Figure 24 demonstrates the difference between having this occlusion and not having it. Only through the occlusion cue, most other cues are actually usable, because they can be properly perceived by the user. Especially for depth perception, these are crucial information to have as humans use occlusion to recognize relations of objects and distances between them.

Gripper Region Indicator

The *Gripper Region Indicator* has the purpose of giving the user a clear notion wherein the horizontal plane the gripper is positioned at any moment even when it is high above the table. To be effective, the Gripper Region Indicator is displayed on top of the table and makes use of the Occlusion Cue to clearly show when the gripper is hovering over an object, as demonstrated in figure 24. Secondly, the Gripper Region Indicator resizes according to the current gripper span (distance between the fingers). When the gripper is half-closed, the Indicator matches its size. Giving the user a clear understanding of its span and if an object is pickable at the current moment or not. Figure 25 a) and c) show how it works in action.

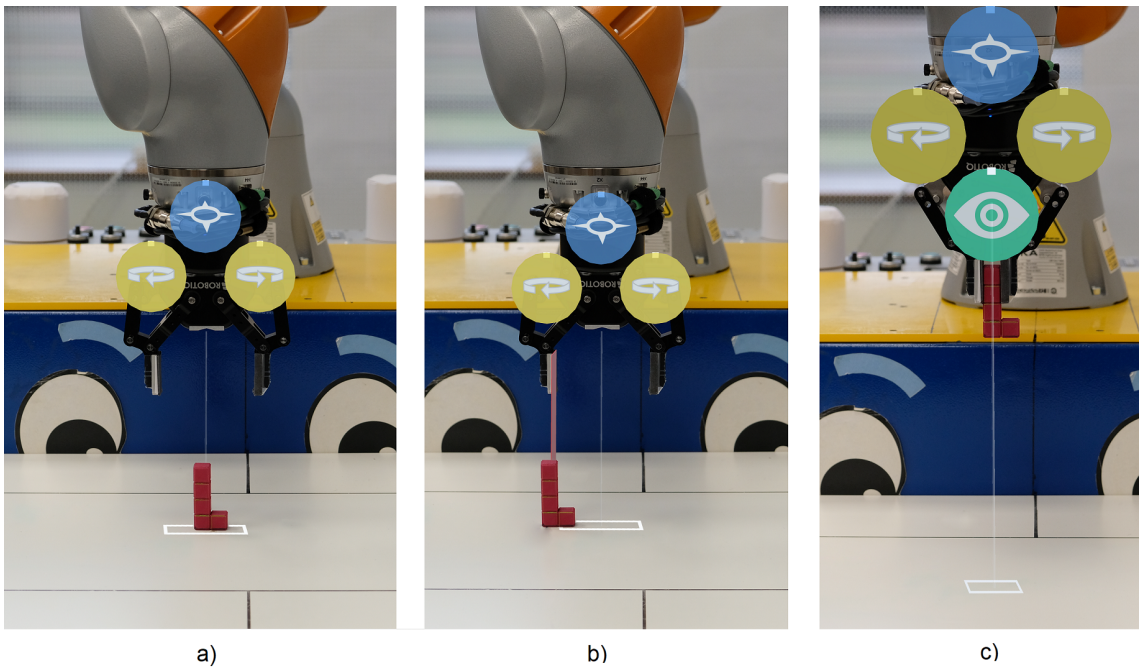


Figure 25: Advanced Visual Cues for picking. a) shows the working occlusion of the gripper region and laser pointer coming down the gripper center. b) gives a demonstration of a collision, which is highlighted by the red Virtual Extension of the gripper finger and c) shows the picked object with the change of the Gripper Region Indicator.

Laser Pointer

Another cue for aligning the gripper is the *Laser Pointer* which marks its center. It travels vertically down to meet the Gripper Region Indicator on the table and is visible in figure 25. Its purpose is to quickly see the gripper alignment to the left and right side, even when the Gripper Region Indicator is blocked from vision. Additionally, it adds a third dimension, giving the user information for higher or non-uniform objects. Similar to the Gripper Region Indicator, the Laser Pointer is taking advantage of the Occlusion Cue and stops on the objects surface.

Virtual Extensions

The last cue which is primarily for picking objects are the *Virtual Extensions*. These are an additional help to minimize wrong alignment of the gripper and reduce accidental object-collision. They are extensions of the gripper fingers and, similarly to the Laser Pointer, go down vertically to the tabletop. They normally are invisible to keep the visual clutter low and only show themselves when they collide with an object. This collision happens as soon as the gripper is above an object, reducing the potential of unnoticed misalignment. This is exemplified again in figure 25 b), where the Virtual Extension collides with the object and is displayed as a slightly transparent red rectangle. Red was used as a signal color that the users attention is required, in contrast to the white of the other cues. The difference to the other two alignment helper, this one gives active visual feedback instead of being passive like the Gripper Region Indicator and the Laser Pointer. The reason the other two are still important is, that they work on bigger/complex objects. These may not fit entirely into the gripper region, therefore needing to be grasped at small outcroppings when the Virtual Extensions may still be

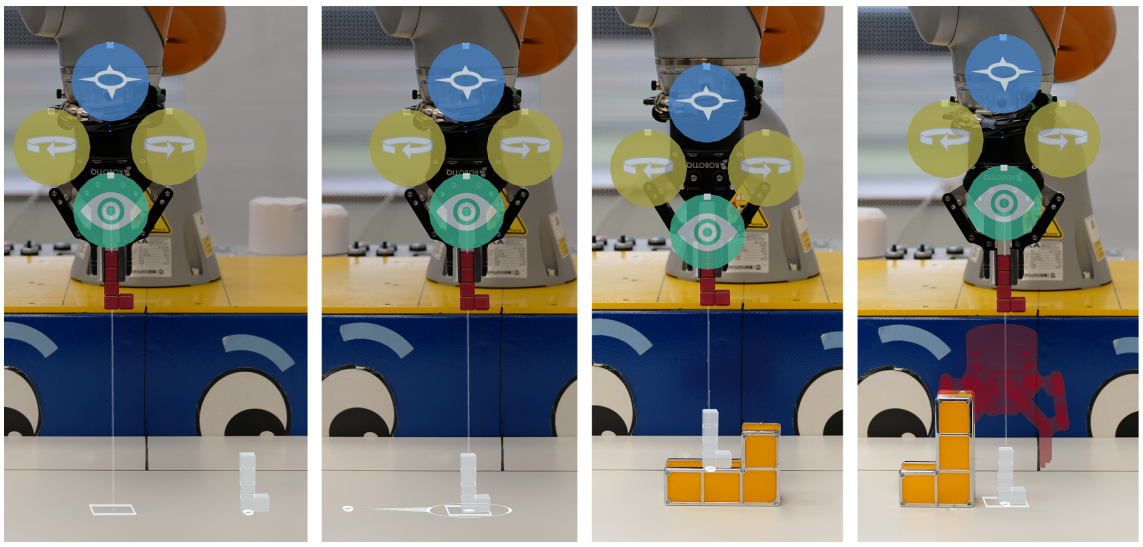


Figure 26: Advanced visual cues: the ghost object. a) shows the object ghost in the Gaze-Follow pattern and b) in the Gripper-Follow pattern. c) demonstrates, that the ghost adjusts its height. c) shows the gripper ghost caused by proximity to a different object.

triggered. Secondly, when the pose recognition system fails, they still work with some limitations. They would then occlude the object, but still give some information about object-gripper relations.

Object Ghost

For placing, there is only one advanced visual cue, the *Object Ghost*, which is the most complex one. As can be seen in figure 26 a), it is a copy of the currently grasped object. This copy slides along the table adjusting its height when above another object, following the users' gaze, as visible in figure 26 c). It only moves inside the working area of the robot, stopping at the exact location where the border is, which the robot itself will not pass. This allows the user to know exactly where the object will end up when they move the robotic arm. The user can align the Object Ghost and say the move command with knowledge about the placing position of the object.

In addition to this *Gaze-Follow* pattern, which was just described, the Object Ghost has a second behavior pattern. The *Gripper-Follow* pattern can be manually activated and deactivated through the *Seeing Eye* dwell button that only appears when an object is grasped. It also switches into this pattern when the robotic arm is in precision mode. While in this pattern, the ghost is projected exactly under the real object, following the grippers' slow movement and giving the user an opportunity for very precise alignment. This is shown in figure 26 b).

The last important feature of the Object Ghost is the normally invisible Gripper Ghost, shown in figure 26 d) in red. This Gripper Ghost is attached to the Object Ghost and moves around with it. Just like the Virtual Extensions it only shows itself when it collides with another object. Allowing the user to know exactly if they can place the object without hitting a nearby one with the gripper itself during placement.

All these advanced visual cues were designed with the intention of supporting the users

without overloading them with information. This was achieved by reducing visual clutter as much as possible, which is especially important on the HoloLens as the FOV is very small. Only displaying elements when they are important and sticking to transparent visualizations.

6.6 Object and pose detection

At the core of this prototype lies the object pose recognition. Without it, the described robotic control scheme and the advanced visual cues will not work. With the considerations in section 5 the recognition system has two cameras with different viewpoints of the working area. The webcam integrated into the HoloLens and an additional webcam placed on the right side of the table, facing the working area from an elevation of 20 cm above the tabletop. Only on key events an image from each webcam is send to the pose recognition system to update the objects in the scene.
The events in which a new pose update is done are listed here:
On...

- ...system startup
- ...restart finished
- ...robot move finished
- ...gripper opened
- ...gripper closed
- ...*Detection* command issued

These few events, excluding the *Detection* command, cover most of the situations in which an object might be moved without the application having information about it. The speech command is there as a fail-safe for the user if something went wrong in other situations.
As outlined in the system overview, the pose recognition is compartmentalized into two steps, the 2D object detection, and the 6D pose recognition. These will be described in the following two subsections.

For a user-study which was performed and will be discussed further in section 7.2 a fake object pose recognition was utilized. As the workspace is fixed during the study the object pose was well-known and could be hardcoded into the system. When an object is normally moved with the gripper, these changes are known as well, as the gripper position itself is specified by the application. If no errors occur on the operators' side, like pushing or dropping objects, the system functions as if it had working pose recognition for the purpose of the study. This was done for two reasons. First, because the study was interested in other criteria than the accuracy of the object recognition and therefore it was not important to use it. It was more important to be consistent for the user. Secondly, the pose recognition system had trouble with the used objects and was therefore not at an usable point. This will also be further discussed in section 7.2.

The following, the three components of the pose recognition system will be described in the order of operation in the application.

6.6.1 Object detection network

The first stage of the full pose recognition system is the 2D object bounding box detection network (object detection network). The *Keras-Retina* network was chosen based on a recommendation by Sundermeyer et al. (2018), because it outperforms most other established object detection networks and its single stage nature gives it considerable speed as well (T.-Y. Lin, Goyal, Girshick, He, & Dollár, 2017). The implementation that is used in this thesis is by Gaiser et al. (2019) and is available on *Github* [1] (Gaiser et al., 2019). The novelty of T.-Y. Lin et al. (2017)'s object detection is the loss function called focal loss. It focuses on the imbalance in training data in regards to many easy, negative examples in contrast to few hard, positive examples. A big problem with one-shot (single-stage) object detectors is this imbalance which leads to an overemphasis on easy cases and the rare hard cases are not appropriately trained. Focal loss solves this problem by dynamically scaling the impact of correct detections during training based on the given confidence of the detector. The higher the confidence of class detections the less impact it has on training. This leads to an emphasis on harder examples and a reduction of the impact of easy examples.
Data for training the network is generated synthetically to drastically reduce the turnaround time for testing different objects. As the basis for data generation serves the script by Sundermeyer et al. (2018). This basic generation script takes a random background from a list of open access images and renders a random amount of training objects onto this background in a random transformation (position, rotation). For each new image an XML file with the information of all object locations, their classes, and bounding boxes is generated.
Augmentation of this synthetic data generation was done to align it more with our use-case. As we used the same object multiple times with different colors it was necessary to add an object color randomization to the data generation. Additionally, because of the different cameras and corresponding different distances to the working area the variation in distance offset to the camera was heightened. For each newly generated image, a different random base distance is chosen which gets added to all rendered objects after their normal random transformations are applied. This gives the effect of the virtual camera standing at different distances. The final augmentation is the enabled possibility of object overlap. In the basic synthetic data generation, no objects could overlap and occlude each other, which drastically reduces the performance of the object detection even for slight overlaps. Because the generated XML file is not the correct format for training the *Keras-Retina* network, a small conversion software that parses all generated XML files and has a **C**omma-**S**eparated **V**alues (CSV) file with all needed information as the output was developed.

Pre-trained weights were used for training the network, which helps to overcome the domain gap of synthetically generated data as elucidated by Hinterstoisser et al. (2018) (Hinterstoisser et al., 2018). It additionally speeds up the training itself. The used

[1] https://github.com

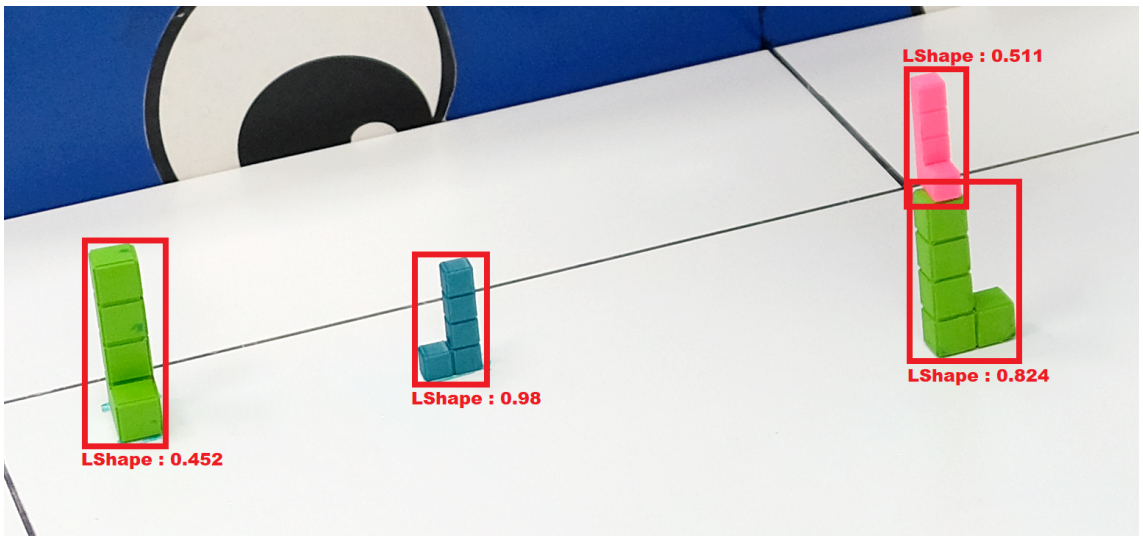


Figure 27: Result of object detection with the bounding boxes, classes, and confidences drawn on the input image.

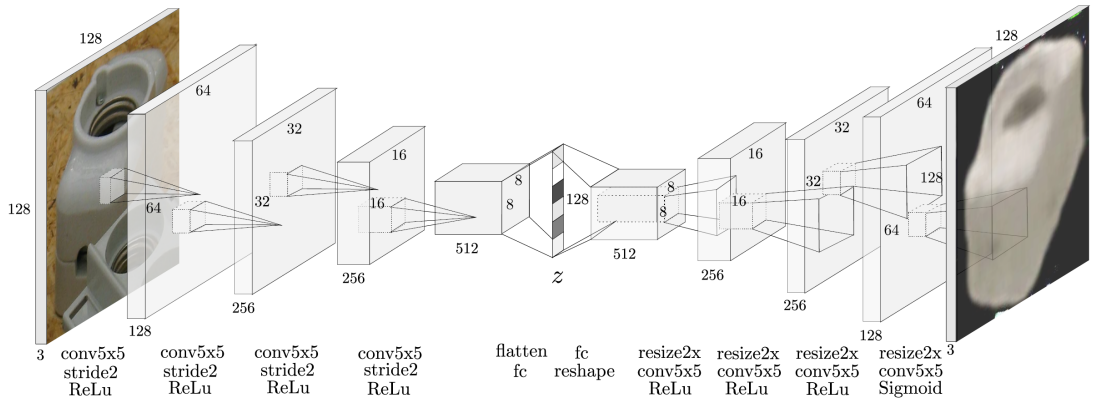


Figure 28: AAE CNN architecture (Sundermeyer, Marton, Durner, Brucker, & Triebel, 2018).

pre-trained weights came from the *COCO* [1] dataset and the feature extraction is done with the *ResNet-50* [2] architecture. The training time amounts to approximately 42 hours on an Nvidia Tesla K40 graphics card.

An exemplary result of the network testing on a webcam image is given in figure 27. The red boxes are the individual detected bounding boxes of each object, the word under it is the class of the detected object and the value is the confidence of the detection in percent [0,1]. These values in the form of three separate arrays get fed into the pose detection network subsequently for each detected object.

6.6.2 Pose detection network

As previously mentioned the AAE network is adopted from Sundermeyer et al. (2018). Following, the network will be described in more detail.

The architecture of the AAEs CNN is shown in figure 28. For training the network, a pixel-wise **L**east square (L2) error loss function is utilized. It is only computed on the

[1] http://cocodataset.org/

[2] https://www.kaggle.com/keras/resnet50/home

pixels with the largest errors in order to reconstruct finer details and not converge in local minima during training. For the training 20000 synthetic images are generated that are placed in a random 3D orientation facing towards the camera with a constant distance. Except for the scene lighting, all other domain randomizations are added during training. The training batch size is 64 and it performs 30000 iterations, with the *Adam* optimizer, amounting to a training time of approximately 2.5 hours on an Nvidia Tesla K40 for a single object.

After the training stage is finished a so-called codebook is created. Images of the object are generated from each point of a full view-sphere, which is a highly refined icosahedron with the object at its center. It is shown in the offline stage of figure 29. For each point of this view-sphere, the camera is rotated in-plane to cover the whole rotation space of the object. In total there are 2562 equidistant viewpoints times 36 in-plane rotations equaling 92232 created images. All of them are encoded with the trained network resulting in a code-point for each, that is subsequently inserted into the codebook together with its rotation matrix. This codebook serves as a lookup reference in the online test phase.
Figure 29 additionally shows the online phase (testing). The given image is first run through the Object detector, which crops the objects to their bounding box. This crop is then encoded with the trained AAE and then compared with each codebook entry, using the Cosine Similarity. This similarity can be computed efficiently on a graphics card. The entry with the highest similarity to the input image is then selected and the corresponding rotation matrix is the resulting orientation that the network predicts.
Finally, the translation is predicted with the following two equations:

$$t_{pred,z} = t_{syn,z} * \frac{l_{syn,max\_cos}}{l_{test}} * \frac{f_{test}}{f_{syn}} \tag{1}$$

$$\begin{pmatrix} t_{pred,x} \\ t_{pred,y} \end{pmatrix} = \frac{t_{pred,z}}{f_{test}} \begin{pmatrix} (bb_{cent,test,x} - p_{test,x}) - (bb_{cent,syn,x} - p_{syn,x}) \\ (bb_{cent,test,y} - p_{test,y}) - (bb_{cent,syn,y} - p_{syn,y}) \end{pmatrix} \tag{2}$$

Equation (1) predicts the objects distance $t_{pred,z}$ to the given pinhole camera model. All variables containing the subscript $_{syn}$ are the known synthetic parameters either from the synthetic camera or the codebook entry which was matched in the orientation prediction step. Therefore, $t_{syn,z}$ is the distance to the synthetic camera. l_{test} and l_{syn,max_cos} are the diagonals of the bounding boxes of the objects. The first being of the detected object and the second being from the matched codebook entry. Finally, f_{test} and f_{syn} are the focal lengths of the real and synthetic camera sensors. The second equation (2) calculates the two missing dimensions, x and y, of the translation vector t_{pred}. Here p_{test} and p_{syn} are the principal points of both camera parameters and $bb_{cent,test}/bb_{cent,syn}$ are the centers of their corresponding bounding boxes.

For additional clarification, the system described in this subsection was created by

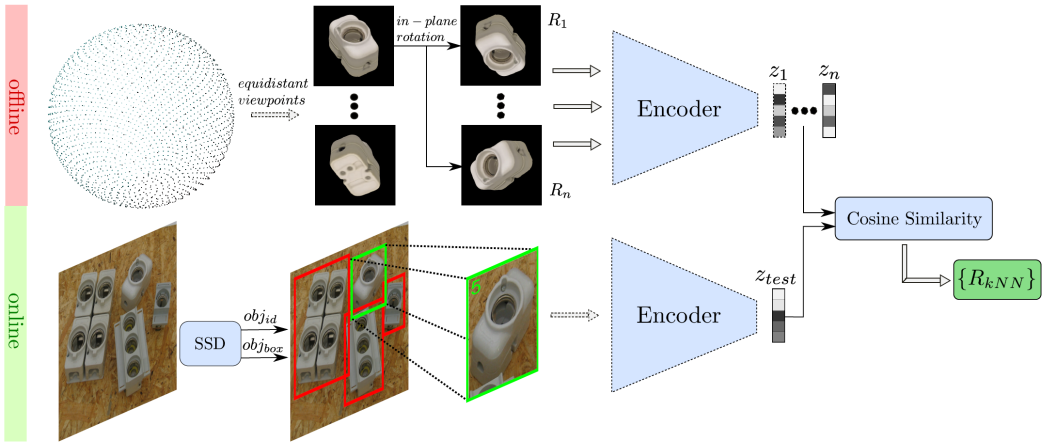


Figure 29: Upper half is the codebook creation, which is done offline after training the AAE network. Lower half is the online testing of an image (Sundermeyer, Marton, Durner, Brucker, & Triebel, 2018).

Sundermeyer et al. (2018) and only slightly adjusted for the purpose of this thesis, namely the synthetic image generation. For additional robustness against similar objects overlapping each other's bounding boxes in a given test scene, the domain randomization was augmented with additional objects rendered onto the background. It is often the case in assembly tasks that multiple instances of the same object are present and lying close to each other in the working area. This addition is used to explicitly guard against errors that occurred during the Cosine Similarity selection step.

6.6.3 Pose result processing

After the 6D pose data for all objects in the current webcam images are generated, they are sent to the Object Tracker component in the HoloLens application, where they are converted into virtual objects in the scene. This occurs in five main steps, the *message parsing*, *coordinate system transform*, the *object selection*, the *pose adjustment* and finally the *de-duplication*.

The *message parsing* uses the message to generate a translation vector and rotational quaternion for each object, still in the original camera space of the webcam. Additionally, it saves the given object class and which webcam it came from.

Coordinate system transform is then converting the generated transformation data into the corresponding camera space. This includes a scale adjustment of the translation because of different coordinate system units in the pose recognition system (centimeter vs meter).

Thirdly, based on the distance of the objects to the table surface the *object selection* selects which size the object has. Two different sizes were used for all objects, of which the pose recognition system only knows the smaller dimensions. Meaning that it always assumes it to be the smaller object and the given distance to the camera is therefore not correct if the found object is the bigger of the two, as it *thinks* that it is closer to the camera than it actually is. This depth-size ambiguity was described in section 4.2. To find out if the object is the correct one the distance to the table surface along the camera

forward ray is taken and if it is larger than a small threshold (4 cm) then the bigger size is used.
For *pose alignment*, the same distance information as for the *object selection* is used and the object is translated along the camera forward ray to place it perfectly on the table. For rotation, the object axis with the highest Cosine Similarity to the tables up-axis is found and then the object is rotated to perfectly align the found axis with the table axis. Finally, the objects get *de-duplicated*. As there are two cameras, all objects from each camera are still present, even if both cameras detected the same object. To eliminate this duplication, first every object from one camera is intersection checked with all objects from the second one. The closest objects that overlap and are the same class and size, will then be merged into one. This merging takes the most robust values from the translation of each object. The distance to the camera has often small errors, as it is inferred in the conversion process of the 2D image into 3D space. Therefore this value is not used in the merging process, making the pose recognition slightly more robust. For the other two axes, the mean of each between both cameras is calculated.

After these five steps, all detected objects are placed in the virtual scene. They can be utilized for collision detection and further transformed based on user interaction until new pose data arrives at the Object Tracker component.

7 Evaluation

This thesis consists of two main parts, the pose recognition system and the design of the advanced visual cues. In order to properly verify if both parts function as intended they were tested in two separate small studies. The reasons for splitting them into two studies instead of having one bigger study are manifold. First is the dependence of the advanced visual cues on the pose recognition. If the pose prediction is too imprecise the study participants cannot effectively use the cues and it would be impossible to get any conclusive information if they function as intended. Additionally, the design verification study for the cues was part of a bigger study conducted for the MIA project (described in section 1.1), which needed to take place at a specific time, and the pose system seemed not reliable enough at that point. The in 6.6 mentioned fake pose recognition was therefore implemented to have a predictable and precise outcome for each participant. Lastly, the approach for testing both parts are vastly different. For the advanced visual cues, it is important to test the design choices with different potential users of the system to get an impression if they work as intended. Pose recognition works independently of the user and can be tested best in a technical evaluation. Here the prediction accuracy and stability are important to know, which cannot properly be measured in a user study.
The decoupling of these two parts has the disadvantage of failing to state how well the overall system works. This was not seen as a huge drawback, because it is straightforward to predict the possible performance of the whole system based on the results of the two smaller studies, as the two evaluated parts are cleanly separated.

7.1 Design verification

The goal of the small user study was to discover if the designed advanced visual cues, together with the movement scheme, are used as intended and are understandable after a short introduction. Additionally, a task was chosen which consists of multiple sub-tasks that escalate in difficulty in order to identify if and how the system supports the user in especially hard situations. For verification of these questions, information on precision, time, and the success rate for each sub-task was recorded. On top of that, a small interview was conducted at the end of the task to get subjective information from the participants. In total seven participants between the age of 19 to 55 took part in the study. Five of them already worked with or programmed a robotic arm before, six experienced AR/VR at least once before and two had already programmed something for AR/VR. Initially, the goal was to have twelve participants in the study, which was not achievable because the COVID-19 pandemic cut the study short after the first week, as it would have been irresponsible to proceed with the study and risk infecting the participants.

7.1.1 Setup and procedure

In total, the duration of the study is one hour for each participant, consisting of four distinct parts. The first part was the introduction to the robotic arm, the complete system itself and all required commands to control the robot. This takes around 10 to 15 minutes. The second part is the training phase, in which the participant puts on the HoloLens and performs a small training task to get familiar with the system. The training task consists

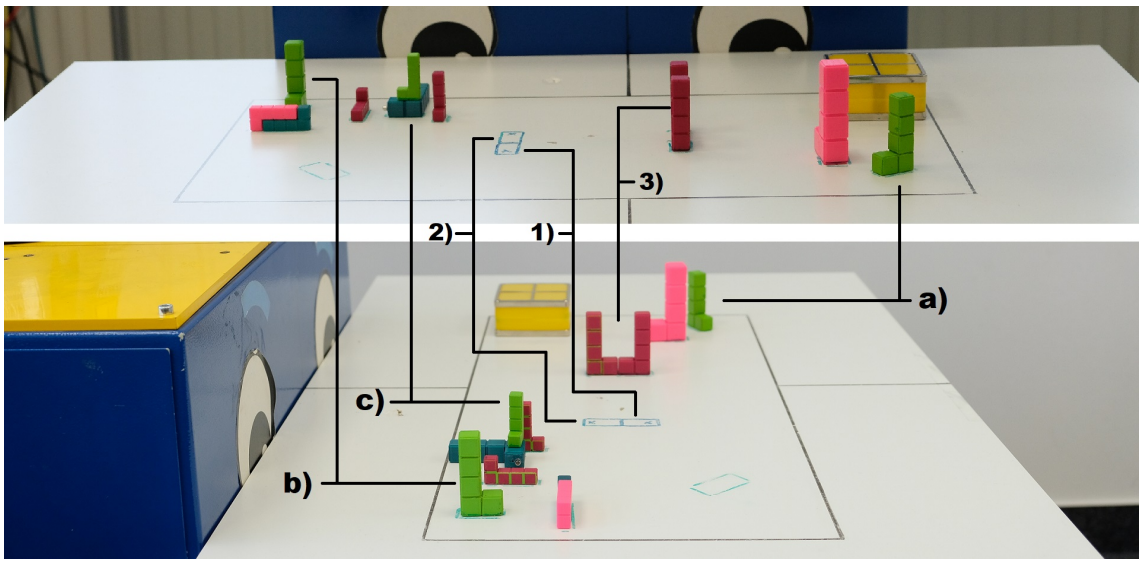


Figure 30: Study task setup of objects starting and marked end locations. The sequence the participant moves the objects in is a) to 1), then b) to 2) and finally c) to 3).

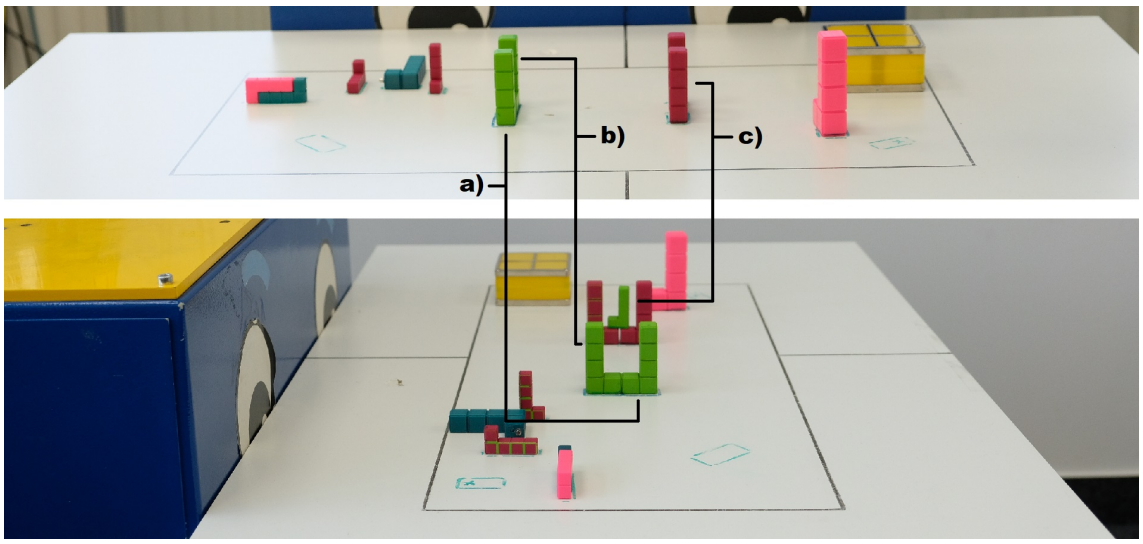


Figure 31: Study setup after completion of the task.

of picking and placing an object at different locations. This part takes approximately 5 to 15 minutes.

The third and main part is the main task, which was repeated two additional times and takes up to 30 minutes of the study. The setup of the chosen task is shown in figure 30 and the state at the end of the task is presented in figure 31. For each figure, the upper image is taken from the perspective of the participant, and the lower one from the side to give a proper overview. Picking and placing the lime green objects, a), b) and c), are the three sub-tasks. The first step is picking up object a) and placing it at location 1), then object b) to location 2) and lastly object c) to location 3). Object a) is straightforward and the only difficulty is not hitting the big pink object next to it while picking. Picking is slightly more complicated for object b), as it is close to multiple other objects, which occlude it partially. Placing object b) is harder as well, because the participant has to place it behind a), which is already at location 1) and therefore blocking the view. Hardest is the third object, c), which is smaller than the first two, elevated and very close to the

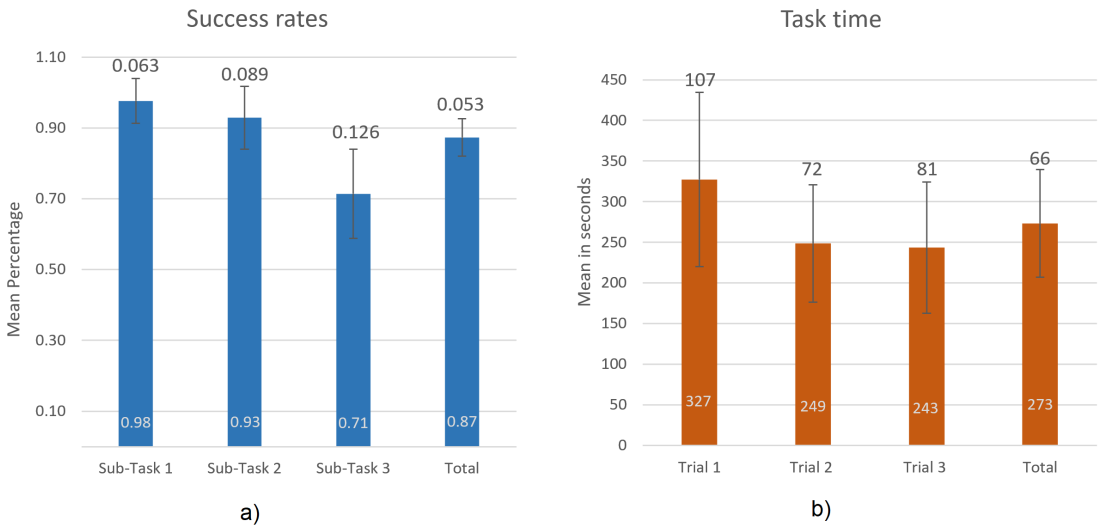


Figure 32: a) are the mean success rates and standard deviation for each sub-task and the total. b) shows the mean task time and standard deviation for each trial of the full task and the total across all trials.

red objects. Placing this object is very difficult, as the view is not only blocked by the closer red object at location 3), it additionally cannot be placed too far back as it would hit the further red object. During the whole task, the participants are not allowed to tilt their head or body to the side to get a better view of the scene. This was done because the original target group of this system are physically impaired people who would not be able to tilt their head or body for a better view.
The final part of the study was the data collection, namely some questionnaires and the short interview. The questionnaires were mostly for the aforementioned overarching study. This part takes up to 10 minutes in total.

7.1.2 Results and analysis

In general, the study shows that the designed system works as intended. The overall success rate of every participant across all three trials is at **M**ean (M) = 87.3% (**S**tandard **D**eviation (SD) = 5.3%). A sub-task was considered successfully completed if the object was picked and placed in close vicinity of the goal position, without it falling over. The participants had the chance to recover small, non-fatal errors. Splitting the success rates into the three sub-tasks gives a clear indication of the final sub-tasks' difficulty. Figure 32 a) shows that first and second sub-task have a success rate of M = 97.62% (SD = 6.3%) and M = 92.86% (SD = 8.9%) respectively and the third one M = 71.43% (SD = 12.6%). The reason for the comparatively high failure rate of this sub-task may be the fact that a recovery is often not possible. When the object is misplaced it will fall over which is unrecoverable in this scenario. For the first two tasks, a misplaced object would most likely just be standing on the table and could be picked up again to rectify the placement, which is impeded in the third one due to the slight elevation.

The time to complete the full task averaged at M = 273 seconds (SD = 81 s) across all trials, and for the individual trials it took M = 327 s (SD = 66 s), M = 249 s (SD = 107 s) and M = 243 s (SD = 72 s) respectively, as visible in figure 32 b). Of interest here is the

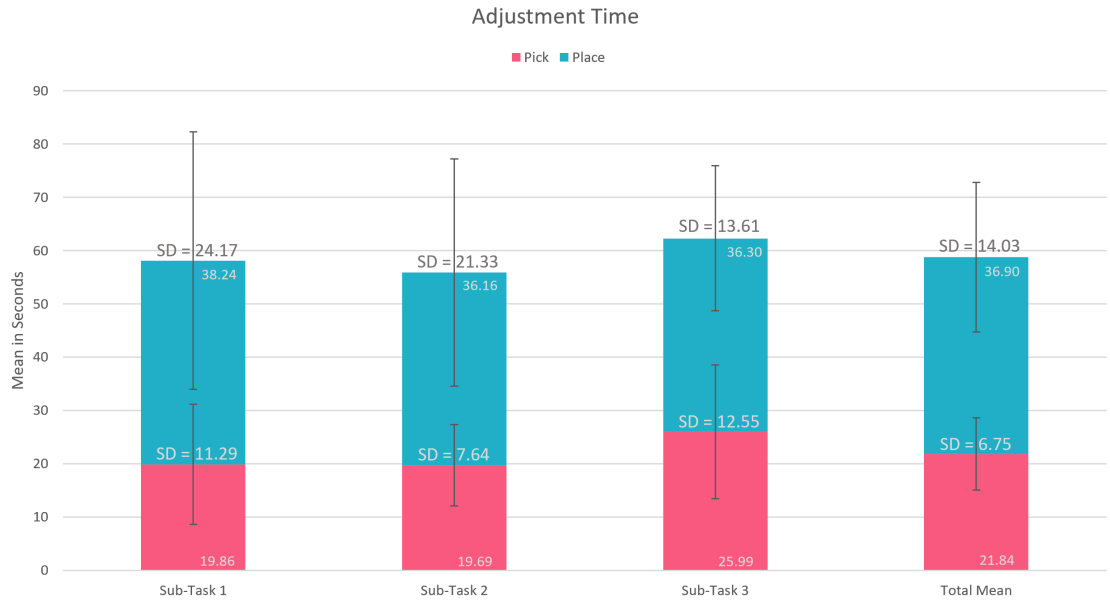


Figure 33: Mean adjustment time for picking and placing an object for each sub-task and the complete task. This is the time from the move command to the pick/place command.

difference of the first trial in respect to the latter two, as it is much higher. This may be a result of a learning process that was not alleviated by the training task. It makes sense that it takes longer to do such a specific task the first time because an approach needs to be developed which can be repeated in the following trials.

Time for specific sub-tasks was taken from the first move command to the pick/place command. This represents the fine-adjustment time when the robotic gripper is close to the goal location. Figure 33 depicts the mean times for pick and place for each sub-task and the mean across all sub-tasks. It is visible that there are minimal differences in these times, which is interesting because of the fact that the sub-tasks get progressively harder. This could be caused by the earlier mentioned success rates specifically of the third sub-task. The fact that readjustment was mostly not possible after making a placing error, the first two sub-tasks accumulated more time when users adjusted the positions when misplacing the object. This is also visible in the huge SD, that it was either done quickly (no readjustment) or took a long time. In the third sub-task, this time was spend on the more difficult alignment, equaling out the times. Additionally, the fact that this adjustment time does not include the whole sub-task, but only the fine-adjustment time after moving, may introduce more noise. A different hypothesis is that the system works in a way that the fine-adjustment time is independent of the task difficulty. The Object Ghost makes it very clear where the gripper is positioned regardless of the task. You can clearly see when the position is correct and the fine-adjustment time is therefore only dependent on how close the initial move command brought the gripper to the goal position. Finally, it is also interesting to look at the precision of the participants. Figure 34 depicts the pick and place precision in millimeters. All sub-tasks are in a range of less than four mm for their mean precision and are in total at about 1 cm offset to the perfect position. Considering the task difficulty these results are very reasonable, but may not be enough for the assembly of a workpiece or something similar. Especially the first sub-task is pretty

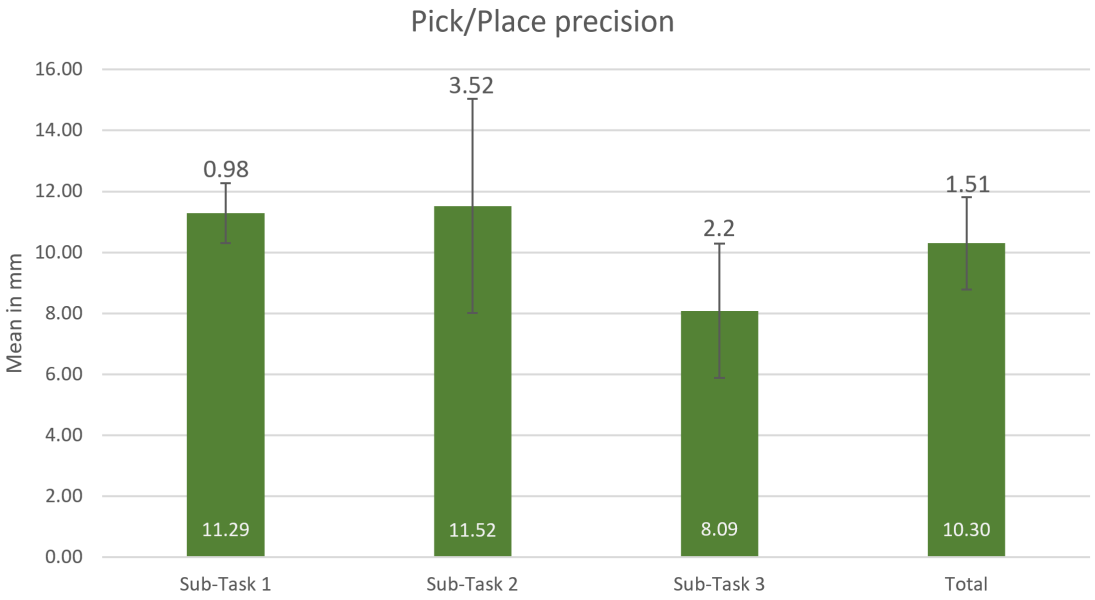


Figure 34: Mean offset to perfect picking and placing position in millimeter.

easy but regardless of that has the same precision, which was expected as the view of the placing area was unobstructed. Reasons for this may be the collected data, as error of the picking accumulates on the placing. Therefore, when the object is picked with an offset, barely being in the gripper, the user accommodates this offset when placing, but the system sees this accommodation as an additional offset.

In the short interview, the participants were asked what the most important helper for pick and place was. Five out of seven participants said that the Gripper Region Indicator was the most helpful. This fits the expectations as this cue gives the most information on which the Laser Pointer and Virtual Extension only expand. For placing the results are similar, here the Object Ghost was named five out of seven times as well. The Object Ghost is definitely the most versatile cue and the main helper for placing. On the question, if something about the prototype confused them, the limited rotation came up twice. This was a deliberate decision to limit the scope of the project, but the study further underlines that this is an important requirement for usage in the real world and actual assembly tasks.
For the Object Ghost, it was implied by two users that the opacity of the ghost is too high, therefore making it difficult to use in some situations. Other confusions were only based on insufficient explanation at the beginning. The last question asked what strategy the participants applied to fulfill the tasks. This was already visible during the study itself, nonetheless interesting to see if the subjective view of the strategy differs. The answers emphasized the fact, that all participants employed the same strategy, which consisted of first rotating the gripper, then moving to the object, and after that using the fine-adjustment and pick or place. The asked questions and the shortened answers of the participants are in the appendix tables 4 and 5 respectively.

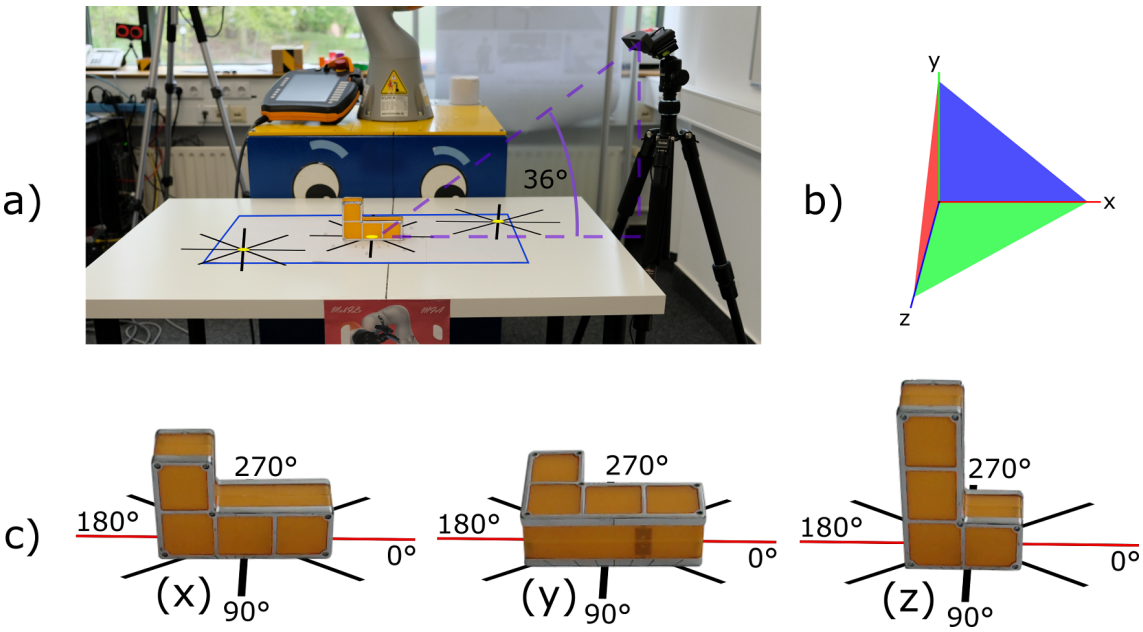


Figure 35: Overview of the test setup. a) is the working area with the three used positions (yellow) and the in-plane rotations. The purple triangle indicates the camera angle. b) is a visualization of the local coordinate planes, which are defined by the corresponding axis. c) is the object lying on all three planes, with marked in-plane rotation.

7.2 Recognition system accuracy and stability test

High accuracy and stability of the pose recognition system are extremely important for this prototype. If the information given to the user is not correct, they may quickly stop trusting the system, and with good cause. Therefore, two different variables were tested: occlusion and pose (position, orientation). Namely, the accuracy and stability of position and orientation predictions and the stability against occlusion. These tests were performed with one example object shown in figure 35 c). Especially the rotational prediction quality of the recognition system can vary between objects and would need to be considered for every new object added to the prototypes repertoire. For a broader coverage of the system's performance, many different objects would need to be tested and the results generalized across all of them. This was not feasible in the time-frame of this thesis, as training and testing of every object takes a long time. Instead, it was opted to settle for a sample of the already trained testing object, which has an average prediction difficulty. The object has only a small amount of textural features, is symmetrical around multiple axes, and therefore prone to self-occlusion which adds to the difficulty. The 3D bounding box of the object is 12 cm $*$ 8 cm $*$ 4 cm.
Different procedures were required to evaluate occlusion and pose, because of this they will be split in two sections.

7.2.1 Pose test setup

The procedure for testing the accuracy and stability of position and orientation was done as follows. The working area of the robotic arm defines the maximum extends of where the recognition still needs to perform consistently. Three different positions were tested in this area, the center of the working area, the lower right corner and the upper left

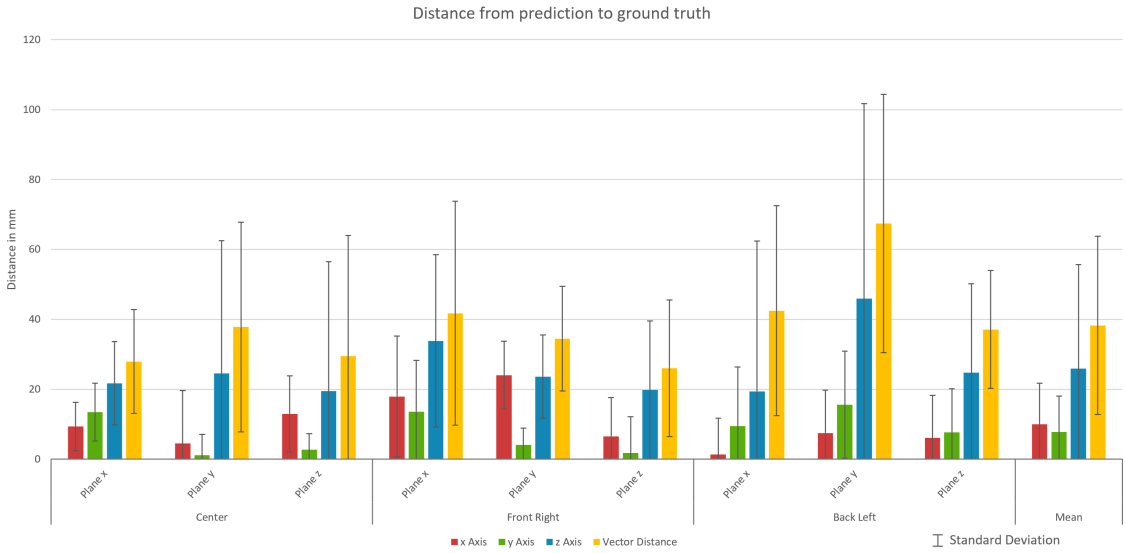


Figure 36: Mean prediction distance to ground truth in mm.

corner (from the camera's point of view), which gives an appropriate coverage. Figure 35 a) depicts the working area as the blue rectangle outline on the table and the points are visualized by the three yellow dots. For each point, the object was placed in all three of its local coordinate planes, which are defined by the local axes that stand perpendicularly on them as depicted in figure 35 b). The planes are colored the same as their defining normal axis, for instance, the green y-axis defines the green y-plane, etc. The circularly extruding lines in 35 a) and c) define the in-plane rotations that were performed for each of the three object axes planes. Each line increases the angle by 45°, to get adequate coverage of the whole rotational space. All eight in-plane rotations were done for all object axis. For each of the planes, no out-of-plane rotation was done, as the object can only lie on its perpendiculars and is symmetric along most axes. Figure 35 c) shows the three rotations for which the in-plane rotations were done. In theory, for (x) and (z) the object could be rotated 180° along the axis that lies on the red line. This would produce two novel perspectives on the object, but both were omitted for simplicity. Both are not viable in practice as the objects would tip over without scaffolding.
This amounts to 8 in-plane rotations * 3 object axis/planes * 3 object positions = 72 unique object poses. In order to measure the stability of each individual pose, 20 frames were evaluated respectively. The position and orientation predicted for each frame was saved in a text file for further analysis. As ground truth, or *perfect pose*, the setup was recreated in Unity and with a small script, these ground truth values were generated for each of the 72 poses and written into a text file as well. The position was saved as a 3D vector and the rotation as a quaternion.

7.2.2 Pose result analysis

To analyze the pose accuracy, a similarity comparison of the predicted and perfect values is required. The stability of each pose is implied by the **S**tandard **D**eviation (SD) across the 20 frames. For this analysis a small C++ application was developed which reads both

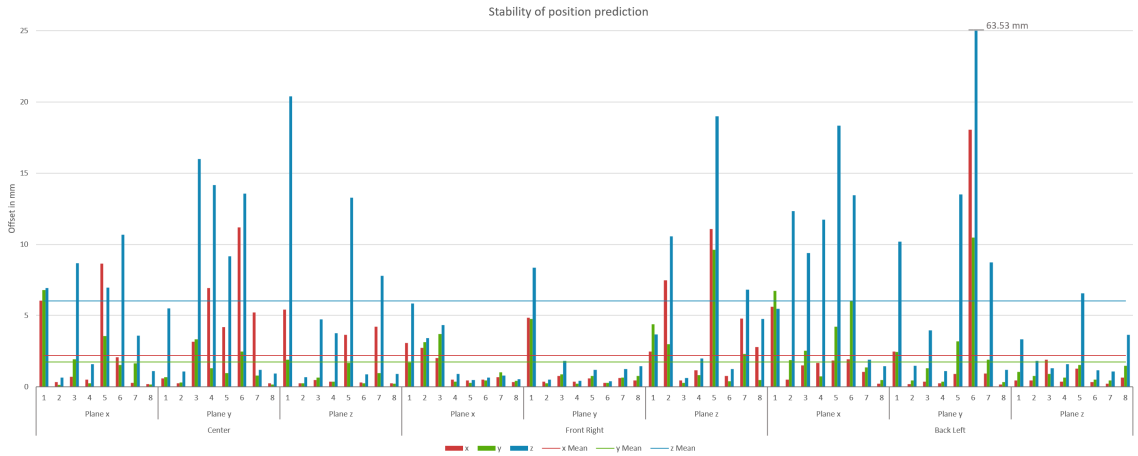


Figure 37: Stability measure of the position prediction. Measured as the SD of each pose over 20 predicted positions.

text files with prediction and ground truth values and outputs the similarity and SD for position and orientation. These results could then be further analyzed.
For position, the similarity measure was done in two different ways. Firstly the vector distance of the predicted value to ground truth and secondly the distance along each axis. Looking at each axis separately makes sense because especially the z-axis may differ drastically from the other two, as it is the depth value predicted from a 2D image. After initial analysis, the absolute value was used for the axial differences, as there was no visible pattern in the offset direction and it made the data much more inscrutable.
Orientation similarity is more complicated, as it has a few possible representations like Euler angles, rotation matrices, or quaternions which all have advantages and disadvantages. Quaternion representation was chosen, based on the recommendation of Huynh (2009) (Huynh, 2009). They compared metrics of 3D rotation similarity proposed by other literature and concluded, that quaternion is spatially and computationally more efficient and has no problem with iso-errors.
The similarity was calculated using the angle θ which represents the rotation required to get from one quaternion to the other around a specific axis. This avoids the problem that a quaternion q and -q represent the same orientation. The formula to calculate the angle θ is shown in (3). In order to bring it in the range of [0,1], θ is afterward divided by Pi.

$$\theta = cos^{-1}(2 * Dot(q_1, q_2)^2 - 1) \tag{3}$$

q_1 and q_2 are the prediction and ground truth quaternions. The inner product (Dot) of their four components is calculated exactly like an inner vector product.

The results of the mean distance to ground truth for each object plane in each of the three positions is depicted in figure 36. The overall mean of all axis is in the rightmost column of that diagram with a distance of M = 10.04 mm (SD = 11.75 mm) for the x-axis, M = 7.75 mm (SD = 10.35 mm) for the y-axis, M = 25.92 mm (SD = 29.71 mm) for the z-axis and M = 38.28 mm (SD = 25.53 mm) for the vector distance. All results are included in the appendix in table 1 for further reading. In general, there is no remarkable difference between the three positions, as well as the three different object

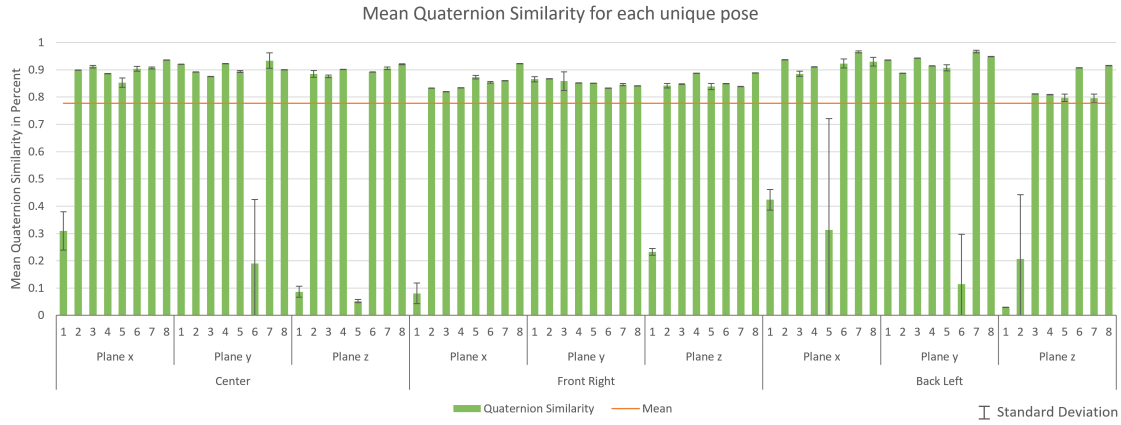


Figure 38: Quaternion similarity for all 72 poses, with the mean across all as an orange line.

planes. The z-axis has generally the highest offset, which was expected as mentioned before. The high SD on the z-axis gives further indication that the depth prediction is quite unstable and this affects the vector distance as it depends on all three axes.
Figure 37 depicts the stability of the position prediction for each individual pose. It represents the SD across all 20 frames taken for all poses. The horizontal lines are the mean stability for each axis. Both x- and y-axes have a similar mean of M = 2.19 mm and M = 1.72 mm offset respectively. With only a few poses where the deviation is higher than a 5 mm offset. This contrasts with the z-axis which has a mean of M = 6.01 mm offset and is almost always higher than the two other axes. On the Back Left position, Plane y, 6 is a remarkable outlier for the z-axis with a SD of 63.53 mm, which is three times as high as the second-highest value. All results are included in the appendix, table 2.

Quaternion similarity, in contrast to the positional offset, is bounded between 0° and 180° and can, therefore, be represented in percentage. 180° meaning that the predicted and ground truth orientations are in opposite directions (0% similarity), and 0° is perfectly aligned (100% similarity). The graph in figure 38 depicts all 72 orientation similarities. The mean across all orientation is M = 0.78 (SD = 0.021), the exact values for each similarity can be found in the appendix in table 3.
In total, 11 orientation similarities are lower than the mean, with all of them being drastically lower. This is caused by the fact that the orientation is either correctly predicted, but the last *fitting* to the end orientation is not perfect (similarity > 0.8), or the initial orientation already predicts the wrong base rotation and is therefore totally off (similarity < 0.5). This oddity is elevated by the fact that the test object is very angular and symmetric, meaning if the system *mistakes* one of the object sides for another the prediction is minimum 90° off. For these 11 low predictions the SD varies drastically from SD = 0.0001 for <Back Left, Plane z, 1> and SD = 0.407 for <Back Left, Plane x, 5>. These variations are caused by inconsistent base predictions. For the similarities with high SD, the prediction stability is very low and across the 20 taken frames, it switches the base prediction between two or more. For low SD the base prediction is always the same, but wrong. A pattern that emerges is the prediction failure for the first pose for Plane x and

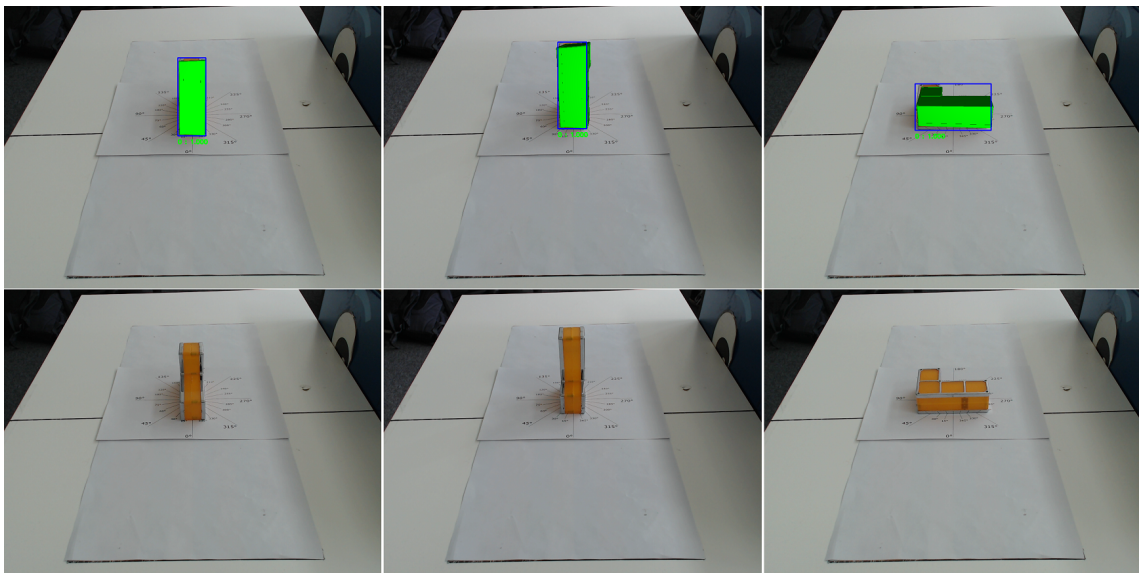


Figure 39: Example of first pose prediction for all three rotation planes. The first two are the first the x-plane, then the z-plane. The third one is the y-plane.

Plane z. All first poses for these planes are consistently wrong, which has to do with the view the camera has on the test object. Figure 39 a) shows that the camera angle makes it hard to discern if the small outcrop is in the front or the other side. This effect is not present for Plane y, as depicted in 39 b).

7.2.3 Occlusion test setup

Compared to testing the pose, occlusion testing is straightforward. Two objects are placed in the working area of the robot in specific relations to each other, first not overlapping in the camera frame. Then they are slowly moved closer together until the pose prediction starts to degrade. The overlap area of the bounding boxes at the point of recognition failure reveals how robust to occlusion the system is. This was done for two different spatial relations, each in 16 different orientations to cover different scenarios. These are shown in figure 41. In order to capture the moment of failure, a small addition to the recognition system was written. The predictions during the time of moving the objects closer together are done continuously, during this time it keeps the previous frame and prediction values. As soon as the prediction results meet one of three different cases, it saves the previous and current image and the prediction values and stops. Figure 40 depicts these three cases, the first case is when the size of the bounding box suddenly changes drastically, the second one is when a different amount of objects get detected and the third is when one of the predicted orientations suddenly changes drastically. This reliably captures the frame of failure and the frame right before its occurrence.

7.2.4 Occlusion result analysis

Evaluating the overlap percentage was done with *intersection over union*, which is an established statistical method for measuring the similarity of bounding boxes and other sample sets. It is calculated by dividing the intersection of two bounding boxes with the

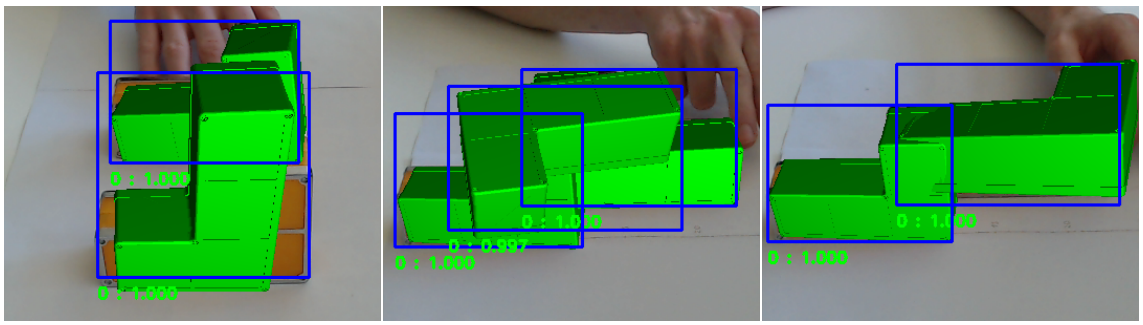


Figure 40: Examples of the three different failure cases through occlusion. Left image is a wrong bounding box size prediction, center image are too many objects detected and right image is wrong orientation prediction.

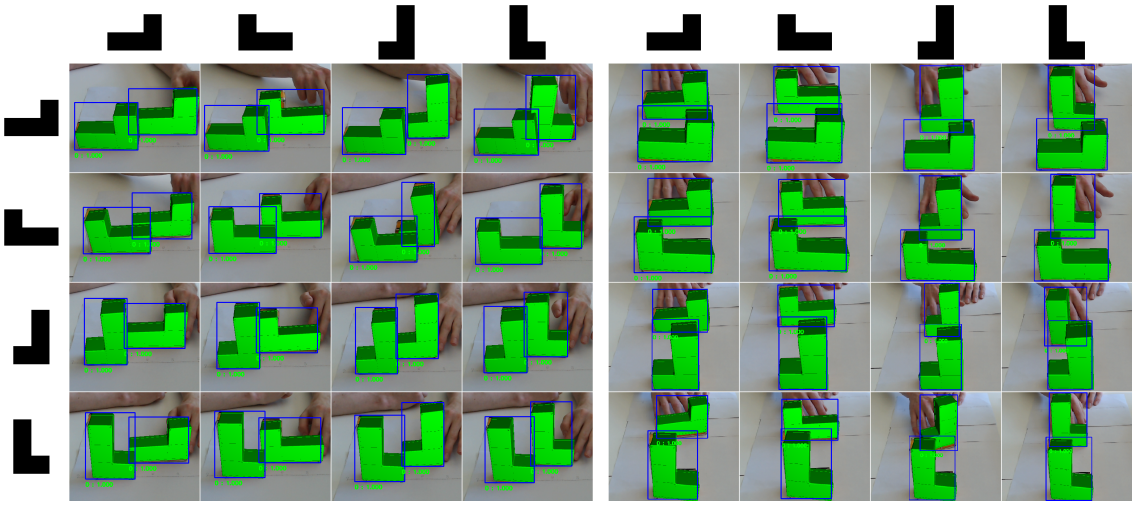


Figure 41: All 32 poses right before prediction failure through occlusion. The left 16 images are for relation one, the right for relation two.

union they form. Figure 41 shows all setup variations at the point of failure to give a general impression of the occlusion robustness. Each row represents a specific orientation for the stationary object and the columns for the moved object. In the left 16 images, the moving object is right of the stationary one and being pushed closer, and for the right 16, it is behind the stationary object. An overlap of 0 means no overlap and 1 is completely occluded. The mean overlap percentage from the camera frames' view for both object relations are depicted in figure 43. The total overlap percentage is the mean over both relations with M = 0.075 (SD = 0.044). The main failure case which leads to the low percentage is the prediction of additional non-existent objects, as shown in the center image in figure 40. 29 of 32 failures were caused by this, which is about 90%. Figure 43 depicts all 32 percentages. Here it is visible that the highest overlap is 21.9%, which is still not close to 50%. Interesting is number 3 of relation 1, which has no overlap. The reason for this may be a slightly too high movement speed of the object, that the previous frame was still not overlapping and the next already more than a few percents. As the mean percentage of relation 1 is M = 0.047 (SD = 0.031), only a centimeter or two might already be too much.

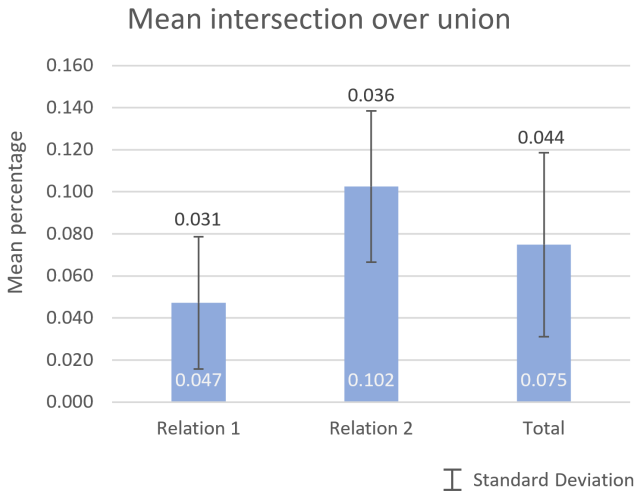


Figure 42: Graph of the mean overlap percentage, as intersection over union, of each relation and the total.

7.3 Discussion

The design verification and recognition system test gives a clear picture of the shortcomings and strengths of the system. The general result of the user based verification is, that the design of the robot control and the advanced visual cues are working as intended. The participants had no problem using the system and only minor details where stated in the interview when asked if something was confusing. Except for the limited gripper rotation, which was expected and considered. The success rates of the task additionally show that it is possible to perform precise object manipulation even after only a short introduction. Considering that this system is intended for utilization in an assembly workstation scenario, the users are expected to use it over longer periods on a daily basis. Which is likely to increase their performance and decrease the need for the system to be usable at an expert level from the beginning. The precision with a mean of approximately 1 cm offset may need improvement, but this study can only give a broad impression on this specific topic. The accumulation error for placing and the fact that the scenario is somewhat contrived makes it hard to transfer the results to different scenarios.

On the other hand, the recognition system still needs improvement. The main shortcoming is the frailty to occlusion. With the failure of recognition at a mean of approx. 7.5% overlap between the two test objects it is unsuitable for a working area with many objects, which might occlude each other. Especially in an assembly task, this may often be the case which makes it a crucial point. The point of failure was mainly the 2D object detection, which very quickly predicted wrong bounding boxes that propagate down to the pose system. Which assumes that the object will fill the given bounding box and therefore makes a wrong prediction as well. The position prediction has the worst performance for the z-axis (depth). In the complete system, this flaw is alleviated through the *pose adjustment* during the pose result processing, as it places the object on the table if the prediction lets it float in the air. The results of the recognition system test additionally give a target value for an *object selection* threshold, which is used to give a cutoff value for selecting the correct object. More details were previously described in 6.6.3. Rotational prediction quality was generally high enough for actual use, with a few exceptions. Es-

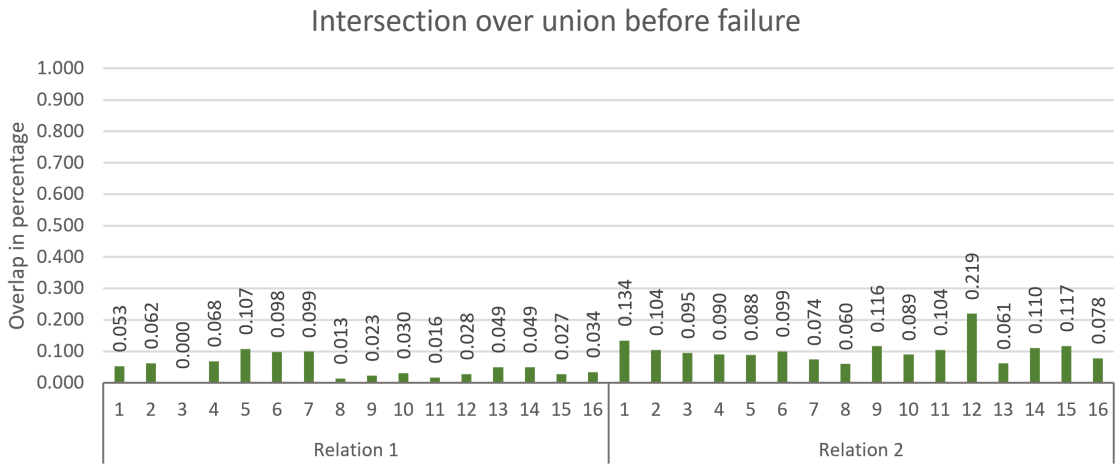


Figure 43: Graph for all 32 overlap percentages.

pecially the fact that for both the x- and z-plane the system consistently failed when the object was aligned face-on to the camera is undesirable. This is most likely caused by the high symmetry of the object in combination with the specific angle of the camera.
With both studies in mind, it is clear that the system cannot function at the level of precision it needs to, in order to support the user in tricky assembly tasks. The main reason is the recognition systems inconsistency. In many situations, it might be precise enough to help the user, but the performance is too dependent on arbitrary translational and rotational factors of objects. Making it unreliable and therefore untrustworthy for the user. High consistency would be required to allow the user to perform the tasks of, for example, the small user study.

8 Conclusion

In this thesis, a novel prototype was explored that uses AR in combination with object recognition to support physically impaired people to work with a robotic arm. Specifically trying to give missing perceptual information about the scene when the user is physically bound to a fixed viewpoint. The information is delivered through the use of spatially placed *holograms*, called advanced visual cues, that add missing relational context of objects and the robotic gripper. This is especially important for depth information, like the alignment of the gripper with objects, or objects with one another. To achieve this goal an AR control system for the robotic arm was developed in cooperation with Franziska Rücker, multiple advanced visual cues for picking and placing objects were designed that use a object pose recognition system, which was adapted from state-of-the-art research in computer vision and machine learning.

To evaluate the performance of the developed prototype, two studies were performed. The first was a small user study that evaluated whether the design of the control scheme and the advanced visual cues are usable and work as intended, with the usage of a fake recognition system to decouple the performance dependency. The second was a technical evaluation of the recognition system on precision, stability, and robustness against occlusion. The results of the user study show that the designed system works as intended and the study participants were able to perform difficult pick and place tasks with an accuracy of approx. 1 cm mean offset. It additionally showed that the system is intuitive enough that it can be grasped in a short amount of time. The technical evaluation revealed that the recognition system has, in principle, high precision for predicting the object poses, but is too unreliable to be used in a real-world scenario. While the precision is generally less than 1 cm positional offset and has a 90% accuracy for orientation, specific objects poses yield considerably worse results. Additionally, the robustness against occlusion is too low to allow real-world application.

In conclusion, even though the developed prototype was not entirely successful it achieves most goals that were initially set. The flawed pose recognition system provides insights on current problems in real-world applications of recognition systems. It additionally provides a sound basis for further research as it is decoupled from the main system and can easily be switched out or adapted further.

8.1 Future Work

The main topic for further work is the pose recognition system. There are multiple possible directions to proceed from the current state of the recognition system and initially it is important to decide which makes the most sense. The first direction would be, to try and improve the existing system further with the usage of hand-labeled real-world data for the 2D detection network or similar ideas. Secondly would be, to exchange parts of the current recognition system, like the 2D detection network, with other, more appropriate ones. For example, a 2D object segmentation network would be able to return only the pixels that contain the detected objects, separating it from the image background and making the pose detection less error-prone. Another possible direction is to not just switch the underlying networks, but overhaul the approach. With

the usage of a depth camera attached to the robot gripper, a fine-grained 3D mesh could be build that has a high fidelity in contrast to the one constructed by the HoloLens. Instead of taking images with a webcam triggered on certain events, the robotic arm could instead make a sweep above the working area and let the depth camera create new depth data and adjust the constructed mesh. With the use of such a mesh, pose recognition might be much more robust and accurate.

Another topic for further research is the rotational constraint of the robotic gripper. It made sense for this prototype to limit the rotation to one axis, but for a real-world system, it is necessary to have full rotational freedom. As it is currently not possible to get an object into a specific orientation, making it impossible to recover overturned objects. In order to achieve this, the control scheme needs to be drastically adjusted to accommodate for the added complexity. This is not an easy task as the system is already at an appropriate difficulty to learn and use, which might be hard to keep.
Technical limitations of the HoloLens are also another starting point for future work. The uneven weight distribution is painful after wearing the device for a while for people with no physical limitations, but make it almost impossible to use for many people with physical constraints, like tetraplegics. Additionally, the small FOV limits the user's spatial awareness for the holograms and leads to clutter in the center of the user's view. Therefore, switching to another AR device might be required at some point. The HoloLens 2 would be a logical choice, as it possesses a superset of the features the first HoloLens offers. It would not be too difficult to convert the prototype to it, as the technological basis is the same and it addresses the mentioned problems.
Lastly, as the conducted user study was done with participants who were not physically impaired, performing another study with the actual target group would be important. Not only might this give more insight into what the technical limitations of the prototype are but additionally provide a completely different perspective. As mentioned in section 2.2.2, people who are dependent on others due to physical impairment have a different focus on the kind of support given and on the degree of system autonomy. This would help shape further refinement of the prototype to bring it closer to a system that is usable in the real world.

Abbreviations

AAE **A**ugmented **A**uto**e**ncoder

AR **A**ugmented **R**eality

CNN **C**onvolutional **N**eural **N**etworks

CSV **C**omma-**S**eparated **V**alues

DoF **D**egree **O**f **F**reedom

DOT **D**ominant **O**rientation **T**emplates

EMG **E**lectro**m**yo**g**raphy

EOG **E**lectro**o**culo**g**raphy

FOV **F**ield **O**f **V**iew

FPS **F**rames **P**er **S**econd

HCI **H**uman-**C**omputer **I**nteraction

HMD **H**ead-**M**ounted **D**isplay

HOG **H**istograms of **O**riented **G**radients

HRC **H**uman-**R**obot **C**ollaboration

HRI **H**uman-**R**obot **I**nteraction

ICP **I**terative **C**losest **P**oint

IMU **I**nertial **M**easurement **U**nit

IR **I**nfra**r**ed

L2 **L**east square

M **M**ean

MARG **M**agnetic, **A**ngular **R**ate, and **G**ravity

MIA Human-Robot Interaction at the Workplace (**M**ensch-Roboter **I**nteraktion im **A**rbeitsleben bewegungseingeschränkter Personen)

MR **M**ixed **R**eality

ORB **O**riented FAST and **R**otated **B**RIEF

ORK **O**bject **R**ecognition **K**itchen

PnP **P**erspective-**n**-**P**oint

RFID **R**adio-**f**requency **Id**entification

ROS **R**obot **O**perating **S**ystem

SD **S**tandard **D**eviation

SDK **S**oftware **D**evelopement **K**it

SIFT **S**cale **I**nvariant **F**eature **T**ransform

SLAM **S**imultaneous **L**ocalization **a**nd **M**apping

SSD **S**ingle **S**hot Multibox **D**etector

TCP **T**ransmission **C**ontrol **P**rotocol

TCP **T**ool **C**enter **P**oint

UDP **U**ser **D**atagram **P**rotocol

UI **U**ser **I**nterface

USM **U**ltra**s**onic **M**otor

VR **V**irtual **R**eality

XML E**x**tensible **M**arkup **L**anguage

Glossary

Iterative Closest Point

Iterative Closest Point is an alogrithm that aligns point clouds. It needs an initial rough alignment and moves iteratively through each point to minimize the distance between them.

Perspective-n-Point

Perspective-n-Points calculates the pose of a calibrated camera given n-3D points and their corresponding 2D projections (Lepetit, Moreno-Noguer, & Fua, 2009).

affordance

According to psychologist Gibson (Gibson, 1977), affordance was defined to be the possibility of an action on an object or environment. The definition has been very widely used in many research fields such as perceptual psychology, cognitive psychology, environmental psychology, industrial design, human-computer interaction (HCI), interaction design, artificial intelligence, and robotics. In robotics, the concept of affordance has been mainly introduced in the research on traversability of environments (Yamanobe et al., 2017).

Cosine Similarity

The Cosine Similarity measures the similarity of vector orientations using inner product space. The range of this similarity is between [-1, 1] where two vectors facing the same direction have a similarity of 1, two vectors with a 90° angle have a similarity of 0 and when facing opposite direction the similarity is -1.

electro-oculography

Electro-oculography is a measuring technique for eye movement. It uses electrodes on either side of the eye to measure deviations of the standing potential that occur during eye movement in contrast to staying still (Gips, 1996).

end-effector

The end in a kinematic chain of joints and links. On a robotic arm this could be some tool or gripper.

Fitts' law

Is an established predictive model for human movement. It is used to model the act of pointing of any kind (phyical with finger, mith a mouse cursor, etc.) and evaluate its performance.

inside-out tracking

Inside-out tracking is the technique for tracking the position and/or orientation of a device without external sensor technology, everything is integrated in the specific device. This has the advantage of not confining the tracking to a specific location and enables moving around a larger spaces.

intrinsic camera parameters

Intrinsic camera parameters are components of the camera matrix, which define a pinhole camera model. They define three different camera properties, the focal length, the axis skew coefficient and the cameras' principal point.

RANSAC

Random sample consensus (RANSAC) is an iterative method for parameter estimation of a given mathematical model from a set of observed data that contains outliers.

registration

Registration (or 3D-registration) tries to reconstruct an environment in 3D from depth-sensor data and determines the relative motion of the camera therein (Bellekens, Spruyt, Berkvens, & Weyn, 2014).

RGB-D

RBG-D data is a color image with three channels (Red, Blue, Green) and an additional Depth channel.

stereoscopy

Stereoscopy uses 2D images to create the illusion that the image is 3D. This is done by producing two images of the same scene from slightly different angles, to imitate the position of the two eyes. This effect is only possible because of the binocular vision of humans (Amy Tikkanen, 2013).

List of Figures

References

Ahmad, N., Ghazilla, R. A. R., Khairi, N. M., & Kasi, V. (2013). Reviews on various inertial measurement unit (imu) sensor applications. *International Journal of Signal Processing Systems*, *1*(2), 256–262.

Allezard, N., Dhome, M., & Jurie, F. (2000). Recognition of 3d textured objects by mixing view-based and model-based representations. In *Proceedings 15th international conference on pattern recognition. icpr-2000* (Vol. 1, pp. 960–963). IEEE.

Amy Tikkanen. (2013). Stereoscopy. Encyclopædia Britannica, inc. Retrieved from https://www.britannica.com/technology/stereoscopy

Andreopoulos, A., & Tsotsos, J. K. (2013). 50 years of object recognition: Directions forward. *Computer vision and image understanding*, *117*(8), 827–891.

Aristidou, A., & Lasenby, J. (2011). Fabrik: A fast, iterative solver for the inverse kinematics problem. *Graphical Models*, *73*(5), 243–260.

Azuma, R. T. (1997). A survey of augmented reality. *Presence: Teleoperators & Virtual Environments*, *6*(4), 355–385.

Baldi, T. L., Spagnoletti, G., Dragusanu, M., & Prattichizzo, D. (2017). Design of a wearable interface for lightweight robotic arm for people with mobility impairments. In *2017 International Conference on Rehabilitation Robotics (ICORR)* (pp. 1567–1573). IEEE.

Barea, R., Boquete, L., Bergasa, L. M., López, E., & Mazo, M. (2003). Electro-oculographic guidance of a wheelchair using eye movements codification. *The International Journal of Robotics Research*, *22*(7-8), 641–652.

Baumberg, A. (2000). Reliable feature matching across widely separated views. In *Proceedings ieee conference on computer vision and pattern recognition. cvpr 2000 (cat. no. pr00662)* (Vol. 1, pp. 774–781). IEEE.

Bellekens, B., Spruyt, V., Berkvens, R., & Weyn, M. (2014). A survey of rigid 3d pointcloud registration algorithms. In *Ambient 2014: The fourth international conference on ambient computing, applications, services and technologies, august 24-28, 2014, rome, italy* (pp. 8–13).

Bolano, G., Roennau, A., & Dillmann, R. (2018). Transparent robot behavior by adding intuitive visual and acoustic feedback to motion replanning. In *2018 27th ieee international symposium on robot and human interactive communication (ro-man)* (pp. 1075–1080). IEEE.

Brahmbhatt, S., Amor, H. B., & Christensen, H. (2015). Occlusion-aware object localization, segmentation and pose estimation. *arXiv preprint arXiv:1507.07882*.

Bray Brandon, McCulloch Jesse, Schonning Nick, Zeller Matt. (2018). What is mixed reality? Retrieved May 28, 2019, from https://docs.microsoft.com/en-us/windows/mixed-reality/mixed-reality

Buss, S. R. (2004). Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods. *IEEE Journal of Robotics and Automation*, *17*(1-19), 16.

Chen, H., Wulf, O., & Wagner, B. (2006). Object detection for a mobile robot using mixed reality. In *International conference on virtual systems and multimedia* (pp. 466–475). Springer.

Choi, Y. S., Anderson, C. D., Glass, J. D., & Kemp, C. C. (2008). Laser pointers and a touch screen: Intuitive interfaces for autonomous mobile manipulation for the motor impaired. In *Proceedings of the 10th international ACM SIGACCESS conference on Computers and accessibility* (pp. 225–232). ACM.

Collet, A., Berenson, D., Srinivasa, S. S., & Ferguson, D. (2009). Object recognition and full pose registration from a single image for robotic manipulation. In *2009 ieee international conference on robotics and automation* (pp. 48–55). IEEE.

Collett, T. H. J., & Macdonald, B. A. (2010). An augmented reality debugging system for mobile robot software engineers.

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *2005 ieee computer society conference on computer vision and pattern recognition (cvpr'05)* (Vol. 1, pp. 886–893). IEEE.

Dragan, A. D., & Srinivasa, S. S. (2013). A policy-blending formalism for shared control. *The International Journal of Robotics Research*, *32*(7), 790–805.

Durrant-Whyte, H., & Bailey, T. (2006). Simultaneous localization and mapping: Part i. *IEEE robotics & automation magazine*, *13*(2), 99–110.

Eppner, C., Höfer, S., Jonschkowski, R., Martin-Martin, R., Sieverling, A., Wall, V., & Brock, O. (2016). Lessons from the amazon picking challenge: Four aspects of building robotic systems. In *Robotics: Science and systems*.

Finke, A., Knoblauch, A., Koesling, H., & Ritter, H. (2011). A hybrid brain interface for a humanoid robot assistant. In *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society* (pp. 7421–7424). IEEE.

Flacco, F., Kröger, T., De Luca, A., & Khatib, O. (2012). A depth space approach to human-robot collision avoidance. In *2012 ieee international conference on robotics and automation* (pp. 338–345). IEEE.

Gabriel Lippmann. (1908). The nobel prize "for his method of reproducing colours photographically based on the phenomenon of interference.". Nobel Media AB 2019. Retrieved from https://www.nobelprize.org/prizes/physics/1908/summary/

Gadre, S. Y., Rosen, E., Chien, G., Phillips, E., Tellex, S., & Konidaris, G. (2019). End-user robot programming using mixed reality. In *Proceedings of the ieee international conference on robotics and automation (in press). ieee.*

Gaiser, H., de Vries, M., Lacatusu, V., vcarpani, Williamson, A., Liscio, E., ... Dowling, D. (2019). Fizyr/keras-retinanet 0.5.1 (Version 0.5.1). doi:10.5281/zenodo.3250670

Ganesan, R. K., Rathore, Y. K., Ross, H. M., & Amor, H. B. (2018). Better teaming through visual cues: How projecting imagery in a workspace can improve human-robot collaboration. *IEEE Robotics & Automation Magazine*, *25*(2), 59–71.

Gao, Y., & Huang, C.-M. (2019). Pati: A projection-based augmented table-top interface for robot programming. In *Proceedings of the 24th international conference on intelligent user interfaces* (pp. 345–355).

Garcia, E., Jimenez, M. A., De Santos, P. G., & Armada, M. (2007). The evolution of robotics research. *IEEE Robotics & Automation Magazine*, *14*(1), 90–103.

Garon, M., Boulet, P.-O., Doironz, J.-P., Beaulieu, L., & Lalonde, J.-F. (2016). Real-time high resolution 3d data on the hololens. In *2016 ieee international symposium on mixed and augmented reality (ismar-adjunct)* (pp. 189–191). IEEE.

Gerken, Jens. (2017). Human-robot interaction at the workplace (mensch-roboter interaktion im arbeitsleben bewegungseingeschränkter personen). Retrieved from https://hci.w-hs.de/projects/mia/

Gibson, J. J. (1977). The theory of affordances. *Hilldale, USA*, *1*(2).

Gips, J. (1996). Towards an intelligent interface for eagleeyes. In *AAAI Symposium on Developing Assistive Technologies for People with Disabilities, MIT*.

Gong, L., Ong, S., & Nee, A. (2019). Projection-based augmented reality interface for robot grasping tasks. In *Proceedings of the 2019 4th international conference on robotics, control and automation* (pp. 100–104).

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. http://www.deeplearningbook.org. MIT Press.

Gopinath, D., Jain, S., & Argall, B. D. (2016). Human-in-the-loop optimization of shared autonomy in assistive robotics. *IEEE robotics and automation letters*, *2*(1), 247–254.

Gordon, I., & Lowe, D. G. (2006). What and where: 3d object recognition with accurate pose. In *Toward category-level object recognition* (pp. 67–82). Springer.

Hannappel, Philipp. (2015). Mechanisches glück. Retrieved from https://vimeo.com/122883656.

Harris Tom. (2002). How robots work. Retrieved from https://science.howstuffworks.com/robot.htm

Hettiarachchi, A., & Wigdor, D. (2016). Annexing reality: Enabling opportunistic use of everyday objects as tangible proxies in augmented reality. In *Proceedings of the 2016 chi conference on human factors in computing systems* (pp. 1957–1967).

Hinterstoisser, S., Cagniart, C., Ilic, S., Sturm, P., Navab, N., Fua, P., & Lepetit, V. (2011). Gradient response maps for real-time detection of textureless objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(5), 876–888.

Hinterstoisser, S., Holzer, S., Cagniart, C., Ilic, S., Konolige, K., Navab, N., & Lepetit, V. (2011). Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *2011 international conference on computer vision* (pp. 858–865). IEEE.

Hinterstoisser, S., Lepetit, V., Ilic, S., Fua, P., & Navab, N. (2010). Dominant orientation templates for real-time detection of texture-less objects. In *2010 ieee computer society conference on computer vision and pattern recognition* (pp. 2257–2264). doi:10.1109/CVPR.2010.5539908

Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., & Navab, N. (2012). Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian conference on computer vision* (pp. 548–562). Springer.

Hinterstoisser, S., Lepetit, V., Wohlhart, P., & Konolige, K. (2018). On pre-trained image features and synthetic images for deep learning. In *Proceedings of the european conference on computer vision (eccv)* (pp. 1–10).

Huynh, D. Q. (2009). Metrics for 3d rotations: Comparison and analysis. *Journal of Mathematical Imaging and Vision*, *35*(2), 155–164.

ISO/TC 299 Robotics. (2011). Iso 10218-1:2011 robots and robotic devices – safety requirements for industrial robots – part 1: Robots. Retrieved from https://www.iso.org/standard/51330.html

Jonschkowski, R., Eppner, C., Höfer, S., Martin-Martin, R., & Brock, O. (2016). Probabilistic multi-class segmentation for the amazon picking challenge. In *2016 ieee/rsj international conference on intelligent robots and systems (iros)* (pp. 1–7). IEEE.

Kehl, W., Manhardt, F., Tombari, F., Ilic, S., & Navab, N. (2017). Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *Proceedings of the ieee international conference on computer vision* (pp. 1521–1529).

Kim, D.-J., Hazlett-Knudsen, R., Culver-Godfrey, H., Rucks, G., Cunningham, T., Portee, D., ... Behal, A. (2011). How autonomy impacts performance and satisfaction: Results from a study with spinal cord injured subjects using an assistive robot. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, *42*(1), 2–14.

Kim, K., Lepetit, V., & Woo, W. (2012). Real-time interactive modeling and scalable multiple object tracking for ar. *Computers & Graphics*, *36*(8), 945–954.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097–1105).

Krupke, D., Steinicke, F., Lubos, P., Jonetzko, Y., Görner, M., & Zhang, J. (2018). Comparison of multimodal heading and pointing gestures for co-located mixed reality human-robot interaction. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 1–9). IEEE.

KUKA-AG. (2018). Lbr iiwa. Retrieved from https://www.kuka.com/en-de/products/robot-systems/industrial-robots/lbr-iiwa

Kurnia, R., Hossain, A. M., Nakamura, A., & Kuno, Y. (2004). Object recognition through human-robot interaction by speech. In *RO-MAN 2004. 13th IEEE International Workshop on Robot and Human Interactive Communication (IEEE Catalog No. 04TH8759)* (pp. 619–624). IEEE.

Lauzier, N., & Gosselin, C. (2011). Series clutch actuators for safe physical human-robot interaction. In *2011 IEEE International Conference on Robotics and Automation* (pp. 5401–5406). IEEE.

Lawrence Gilman Roberts. (1963). *Machine perception of three dimensional solids* (Doctoral dissertation, Massachusetts Institute of Technology).

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., et al. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

Lee, J. D., & Moray, N. (1994). Trust, self-confidence, and operators' adaptation to automation. *International journal of human-computer studies*, *40*(1), 153–184.

Lee, S.-D., Kim, Y.-L., & Song, J.-B. (2013). Novel collision detection index based on joint torque sensors for a redundant manipulator. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 4636–4641). IEEE.

Lepetit, V., Moreno-Noguer, F., & Fua, P. (2009). Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision*, *81*(2), 155.

Lepetit, V., Pilet, J., & Fua, P. (2004). Point matching as a classification problem for fast and robust object pose estimation. In *Proceedings of the 2004 ieee computer society conference on computer vision and pattern recognition, 2004. cvpr 2004.* (Vol. 2, pp. II–II). IEEE.

Leutert, F., Herrmann, C., & Schilling, K. (2013). A spatial augmented reality system for intuitive display of robotic data. In *Proceedings of the 8th acm/ieee international conference on human-robot interaction* (pp. 179–180). IEEE Press.

Li, Y. [Yanghao], Chen, Y., Wang, N., & Zhang, Z. (2019). Scale-aware trident networks for object detection. In *Proceedings of the ieee international conference on computer vision* (pp. 6054–6063).

Li, Y. [Yi], Wang, G., Ji, X., Xiang, Y., & Fox, D. (2018). Deepim: Deep iterative matching for 6d pose estimation. In *Proceedings of the european conference on computer vision (eccv)* (pp. 683–698).

Lin, C.-H., Chung, Y., Chou, B.-Y., Chen, H.-Y., & Tsai, C.-Y. (2018). A novel campus navigation app with augmented reality and deep learning. In *2018 ieee international conference on applied system invention (icasi)* (pp. 1075–1077). IEEE.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the ieee international conference on computer vision* (pp. 2980–2988).

Lisin, D. A., Mattar, M. A., Blaschko, M. B., Learned-Miller, E. G., & Benfield, M. C. (2005). Combining local and global image features for object class recognition. In *2005 ieee computer society conference on computer vision and pattern recognition (cvpr'05)-workshops* (pp. 47–47). IEEE.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). Ssd: Single shot multibox detector. In *European conference on computer vision* (pp. 21–37). Springer.

Liu, Y., Dong, H., Zhang, L., & El Saddik, A. (2018). Technical evaluation of hololens for multimedia: A first look. *IEEE MultiMedia*, *25*(4), 8–18.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Iccv* (Vol. 99, pp. 1150–1157).

Lumelsky, V. J., & Cheung, E. (1993). Real-time collision avoidance in teleoperated whole-sensitive robot arm manipulators. *IEEE Transactions on Systems, Man, and Cybernetics*, *23*(1), 194–203.

Lv, X., Zhang, M., & Li, H. (2008). Robot control based on voice command. In *2008 IEEE International Conference on Automation and Logistics* (pp. 2490–2494). IEEE.

Magic Leap Inc. (2019). Choose your bundle. Retrieved from https://shop.magicleap.com/#/

Manhardt, F., Arroyo, D. M., Rupprecht, C., Busam, B., Birdal, T., Navab, N., & Tombari, F. (2019). Explaining the ambiguity of object detection and 6d pose from visual data. In *Proceedings of the ieee international conference on computer vision* (pp. 6841–6850).

Mikolajczyk, K., & Schmid, C. (2005). A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(10), 1615–1630. doi:10.1109/TPAMI.2005.188

Milgram, P., & Kishino, F. (1994). A taxonomy of mixed reality visual displays. *IEICE TRANSACTIONS on Information and Systems*, *77*(12), 1321–1329.

Milgram, P., Takemura, H., Utsumi, A., & Kishino, F. (1995). Augmented reality: A class of displays on the reality-virtuality continuum. In *Telemanipulator and telepresence technologies* (Vol. 2351, pp. 282–292). International Society for Optics and Photonics.

Mindru, F., Moons, T., & Van Gool, L. (1999). Recognizing color patterns irrespective of viewpoint and illumination. In *Proceedings. 1999 ieee computer society conference on computer vision and pattern recognition (cat. no pr00149)* (Vol. 1, pp. 368–373). IEEE.

Occupational Safety and Health Administration. (2019). Industrial robots and robot system safety. Retrieved June 12, 2019, from https://www.osha.gov/dts/osta/otm/otm_iv/otm_iv_4.html

Palinko, O., Rea, F., Sandini, G., & Sciutti, A. (2016). Robot reading human gaze: Why eye tracking is better than head tracking for human-robot collaboration. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 5048–5054). IEEE.

Pavlakos, G., Zhou, X., Chan, A., Derpanis, K. G., & Daniilidis, K. (2017). 6-dof object pose from semantic keypoints. In *2017 ieee international conference on robotics and automation (icra)* (pp. 2011–2018). IEEE.

Peng, S., Liu, Y., Huang, Q., Zhou, X., & Bao, H. (2019). Pvnet: Pixel-wise voting network for 6dof pose estimation. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 4561–4570).

Qian, Y. Y., & Teather, R. J. (2017). The eyes don't have it: An empirical comparison of head-based and eye-based selection in virtual reality. In *Proceedings of the 5th symposium on spatial user interaction* (pp. 91–98).

Rad, M., & Lepetit, V. (2017). Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *Proceedings of the ieee international conference on computer vision* (pp. 3828–3836).

Redmon, J., & Farhadi, A. (2017). Yolo9000: Better, faster, stronger. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 7263–7271).

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems* (pp. 91–99).

Renner, P., Lier, F., Friese, F., Pfeiffer, T., & Wachsmuth, S. (2018). Facilitating hri by mixed reality techniques. In *Companion of the 2018 acm/ieee international conference on human-robot interaction* (pp. 215–216). ACM.

Richardson, M., & Wiltshire, J. (2017). *The hologram: Principles and techniques*. Wiley. Retrieved from https://books.google.de/books?id=WEkzDwAAQBAJ

Rosen, E., Whitney, D., Phillips, E., Chien, G., Tompkin, J., Konidaris, G., & Tellex, S. (2017). Communicating robot arm motion intent through mixed reality head-mounted displays. *arXiv preprint arXiv:1708.03655*.

Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). Orb: An efficient alternative to sift or surf. In *2011 international conference on computer vision* (pp. 2564–2571). Ieee.

Ruch, T. C., & Fulton, J. F. (1960). Medical physiology and biophysics. *Academic Medicine*, *35*(11), 1067.

Rudorfer, M., Guhl, J., Hoffmann, P., & Krüger, J. (2018). Holo pick'n'place. In *2018 ieee 23rd international conference on emerging technologies and factory automation (etfa)* (Vol. 1, pp. 1219–1222). IEEE.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, *115*(3), 211–252.

Saha, S. K. (2008). *Introduction to robotics*. Tata McGraw-Hill Education. Retrieved from https://books.google.de/books?id=Y-1AHBKJqCsC&lpg=PA27&dq=introduction%20robotic%20arm&hl=de&pg=PA16#v=onepage&q&f=true

Saponas, T. S., Kelly, D., Parviz, B. A., & Tan, D. S. (2009). Optically sensing tongue gestures for computer input. In *Proceedings of the 22nd annual acm symposium on user interface software and technology* (pp. 177–180). ACM.

Sauter, M. (2018). Für 2 milliarden us-dollar hätten wir mehr erwartet. Retrieved from https://www.golem.de/news/magic-leap-one-ausprobiert-fuer-2-milliarden-us-dollar-haetten-wir-mehr-erwartet-1812-137902.html

Savarese, S., & Fei-Fei, L. (2007). 3d generic object categorization, localization and pose estimation. In *2007 ieee 11th international conference on computer vision* (pp. 1–8). IEEE.

Savarese, S., & Fei-Fei, L. (2008). View synthesis for recognizing unseen poses of object classes. In *European conference on computer vision* (pp. 602–615). Springer.

Schmid, C., & Mohr, R. (1997). Local grayvalue invariants for image retrieval. *IEEE transactions on pattern analysis and machine intelligence*, *19*(5), 530–535.

Shamir, T. (1990). The singularities of redundant robot arms. *The International Journal of Robotics Research*, *9*(1), 113–121.

Stilman, M., Michel, P., Chestnutt, J., Nishiwaki, K., Kagami, S., & Kuffner, J. (2005). Augmented reality for robot development and experimentation. *Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-05-55*, *2*(3).

Su, Y., Rambach, J., Minaskan, N., Lesur, P., Pagani, A., & Stricker, D. (2019). Deep multi-state object pose estimation for augmented reality assembly. In *2019 ieee international symposium on mixed and augmented reality adjunct (ismar-adjunct)* (pp. 222–227). IEEE.

Sun, Y., Kantareddy, S. N. R., Bhattacharyya, R., & Sarma, S. E. (2018). X-vision: An augmented vision tool with real-time sensing ability in tagged environments. In *2018 ieee international conference on rfid technology & application (rfid-ta)* (pp. 1–6). IEEE.

Sundermeyer, M., Marton, Z.-C., Durner, M., Brucker, M., & Triebel, R. (2018). Implicit 3d orientation learning for 6d object detection from rgb images. In *Proceedings of the european conference on computer vision (eccv)* (pp. 699–715).

Tanaka, K., Mu, S., & Nakashima, S. (2014). Meal-assistance robot using ultrasonic motor with eye interface. *International Journal of Automation Technology*, *8*(2), 186–192.

Tekin, B., Sinha, S. N., & Fua, P. (2018). Real-time seamless single shot 6d object pose prediction. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 292–301).

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., & Abbeel, P. (2017). Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 ieee/rsj international conference on intelligent robots and systems (iros)* (pp. 23–30). IEEE.

Tolani, D., Goswami, A., & Badler, N. I. (2000). Real-time inverse kinematics techniques for anthropomorphic limbs. *Graphical models*, *62*(5), 353–388.

Tomari, M. R. M., Kobayashi, Y., & Kuno, Y. (2012). Development of smart wheelchair system for a user with severe motor impairment. *Procedia Engineering*, *41*, 538–546.

Tuytelaars, T., & Van Gool, L. J. (2000). Wide baseline stereo matching based on local, affinely invariant regions. In *Bmvc* (Vol. 412).

Vassallo, R., Rankin, A., Chen, E. C., & Peters, T. M. (2017). Hologram stability evaluation for microsoft hololens. In *Medical imaging 2017: Image perception, observer performance, and technology assessment* (Vol. 10136, p. 1013614). International Society for Optics and Photonics.

Wang, C., Xu, D., Zhu, Y., Martin-Martin, R., Lu, C., Fei-Fei, L., & Savarese, S. (2019). Densefusion: 6d object pose estimation by iterative dense fusion. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 3343–3352).

Wang, Y., Zhang, S., Yang, S., He, W., & Bai, X. (2018). Mechanical assembly assistance using marker-less augmented reality system. *Assembly Automation*.

White, J. (2019). Microsoft at mwc barcelona: Introducing microsoft hololens 2. Retrieved from https://blogs.microsoft.com/blog/2019/02/24/microsoft-at-mwc-barcelona-introducing-microsoft-hololens-2/

Xiang, Y., Schmidt, T., Narayanan, V., & Fox, D. (2017). Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. arXiv: 1711.00199 [cs.CV]

Yamanobe, N., Wan, W., Ramirez-Alpizar, I. G., Petit, D., Tsuji, T., Akizuki, S., ... Harada, K. (2017). A brief review of affordance in robotic manipulation research. *Advanced Robotics*, *31*(19-20), 1086–1101.

Yanco, H. A. (1998). Wheelesley: A robotic wheelchair system: Indoor navigation and user interface. In *Assistive technology and artificial intelligence* (pp. 256–268). Springer.

Zeller, M., Gedye, D., Ong, S., Schonning, N., & McCulloch, J. (2018). Hololens research mode. Retrieved from https://docs.microsoft.com/en-us/windows/mixed-reality/research-mode

Zeng, A., Song, S., Yu, K.-T., Donlon, E., Hogan, F. R., Bauza, M., ... Romo, E., et al. (2018). Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. In *2018 ieee international conference on robotics and automation (icra)* (pp. 1–8). IEEE.

Zhang, W., Han, B., & Hui, P. (2018). Jaguar: Low latency mobile augmented reality with flexible tracking. In *Proceedings of the 26th acm international conference on multimedia* (pp. 355–363).

Zou, Z., Shi, Z., Guo, Y., & Ye, J. (2019). Object detection in 20 years: A survey. arXiv: 1905.05055 [cs.CV]

zu Borgsen, S. M., Renner, P., Lier, F., Pfeiffer, T., & Wachsmuth, S. (2018). Improving human-robot handover research by mixed reality techniques. In *Proceedings of the 1st international workshop on virtual, augmented, and mixed reality for hri (vam-hri)*.

Appendix

Table 1: Data for figure 36.

		Mean Distance in mm				Standard Deviation in mm			
		x Axis	y Axis	z Axis	Distance	x Axis	y Axis	z Axis	Distance
Center	Plane x	9.390	13.492	21.755	27.952	6.917	8.240	11.921	14.832
	Plane y	4.499	1.102	24.567	37.808	15.168	5.973	37.939	30.011
	Plane z	12.983	2.692	19.529	29.508	10.896	4.657	36.948	34.526
Front Right	Plane x	17.938	13.611	33.813	41.749	17.264	14.641	24.661	32.062
	Plane y	24.032	4.148	23.648	34.466	9.717	4.746	11.868	14.963
	Plane z	6.542	1.837	19.849	26.018	11.091	10.282	19.744	19.539
Back Left	Plane x	1.314	9.485	19.432	42.497	10.419	16.848	42.992	30.068
	Plane y	7.518	15.586	45.907	67.408	12.171	15.341	55.824	36.916
	Plane z	6.136	7.713	24.747	37.106	12.093	12.420	25.497	16.831
Mean		10.039	7.741	25.916	38.279	11.748	10.350	29.710	25.528

Table 2: Data for figure 37.

		Standard Deviation in mm								
		Center			Front Right			Back Left		
		x Axis	y Axis	z Axis	x Axis	y Axis	z Axis	x Axis	y Axis	z Axis
Plane x	1	6.030	6.776	6.934	3.074	1.701	5.847	5.613	6.735	5.477
	2	0.317	0.108	0.650	2.725	3.125	3.410	0.500	1.858	12.345
	3	0.692	1.928	8.657	2.021	3.698	4.328	1.497	2.528	9.380
	4	0.501	0.234	1.569	0.480	0.351	0.897	1.657	0.729	11.738
	5	8.630	3.547	6.965	0.449	0.259	0.476	1.831	4.202	18.339
	6	2.076	1.521	10.672	0.493	0.431	0.625	1.920	6.021	13.450
	7	0.253	1.647	3.580	0.662	1.012	0.778	1.027	1.345	1.884
	8	0.169	0.152	1.085	0.330	0.402	0.518	0.216	0.478	1.445
Plane y	1	0.570	0.665	5.510	4.848	4.748	8.355	2.454	2.441	10.194
	2	0.239	0.291	1.073	0.363	0.236	0.506	0.168	0.434	1.473
	3	3.151	3.329	15.993	0.758	0.870	1.799	0.352	1.293	3.943
	4	6.924	1.293	14.153	0.350	0.196	0.402	0.241	0.357	1.083
	5	4.176	0.962	9.151	0.594	0.752	1.189	0.890	3.181	13.510
	6	11.184	2.469	13.566	0.263	0.266	0.392	18.054	10.461	63.529
	7	5.210	0.793	1.182	0.615	0.624	1.242	0.936	1.885	8.732
	8	0.234	0.148	0.913	0.448	0.757	1.433	0.152	0.319	1.176
Plane z	1	5.415	1.900	20.393	2.480	4.396	3.678	0.441	1.049	3.333
	2	0.234	0.248	0.678	7.464	2.967	10.556	0.423	0.740	1.809
	3	0.452	0.639	4.717	0.449	0.267	0.599	1.900	0.885	1.307
	4	0.358	0.337	3.753	1.144	0.812	1.978	0.343	0.641	1.578
	5	3.642	1.666	13.262	11.073	9.602	18.981	1.258	1.526	6.552
	6	0.294	0.224	0.856	0.755	0.367	1.233	0.332	0.481	1.156
	7	4.210	0.937	7.771	4.782	2.305	6.811	0.199	0.425	1.069
	8	0.224	0.220	0.903	2.792	0.479	4.745	0.639	1.475	3.630

Table 3: Data for figure 38.

		Center		Front Right		Back Left	
		Similarity in %	SD in %	Similarity in %	SD in %	Similarity in %	SD in %
Plane x	1	0.30914	0.07019	0.08031	0.03774	0.42356	0.03763
	2	0.89915	0.00068	0.83332	0.00023	0.93700	0.00004
	3	0.91141	0.00458	0.82008	0.00012	0.88524	0.00931
	4	0.88672	0.00007	0.83499	0.00005	0.91092	0.00008
	5	0.85270	0.01764	0.87270	0.00655	0.31347	0.40712
	6	0.90345	0.00882	0.85384	0.00260	0.92265	0.01667
	7	0.90667	0.00394	0.86085	0.00008	0.96518	0.00365
	8	0.93628	0.00007	0.92358	0.00006	0.92932	0.01578
Plane y	1	0.92018	0.00008	0.86529	0.00942	0.93560	0.00005
	2	0.89209	0.00007	0.86790	0.00023	0.88750	0.00005
	3	0.87551	0.00006	0.85826	0.03445	0.94314	0.00006
	4	0.92260	0.00003	0.85206	0.00005	0.91524	0.00003
	5	0.89370	0.00354	0.85122	0.00006	0.90735	0.01148
	6	0.18961	0.23493	0.83371	0.00003	0.11425	0.18255
	7	0.93349	0.02850	0.84581	0.00368	0.96727	0.00465
	8	0.90086	0.00004	0.84173	0.00005	0.94902	0.00004
Plane z	1	0.08631	0.02012	0.23269	0.01278	0.02938	0.00011
	2	0.88472	0.01240	0.84213	0.00826	0.20615	0.23583
	3	0.87627	0.00486	0.84720	0.00091	0.81251	0.00005
	4	0.90143	0.00007	0.88738	0.00006	0.80983	0.00007
	5	0.05213	0.00624	0.83814	0.01133	0.79710	0.01333
	6	0.89235	0.00003	0.85000	0.00026	0.90724	0.00012
	7	0.90509	0.00504	0.83858	0.00007	0.79567	0.01522
	8	0.92025	0.00212	0.88905	0.00025	0.91631	0.00004

Table 4: Questions asked for the short interview at the end of the study.

Q1	Tell us about the strategy you used to successfully execute the task.
Q2	What helped you the most to succeed in the task?
Q3	What help you the most to pick an object and what help you the most to place an object?
Q4	Did you feel the urge to move your body to left or right to change your visual angle?
Q5	Did you feel confused at any point? How? When? Explain your answer

Table 5: Shortened answers given by the participants in the interview (Questions from 4).

	Answers (Shortened)	Occurances	Percentage
Q1 Strategy	Rotation Gripper	6	85.714
	Incorporate moving virtual objects	1	14.286
Q2 Helped Most	Cursor	1	14.286
	Ghost	3	42.857
	Gripper Extensions	2	28.571
	Gripper Reach	1	14.286
Q3 Help Pick	Cursor	1	14.286
	Gripper Reach	5	71.429
	Laserpointer	1	14.286
Q3 Help Place	Ghost	5	71.429
	Gripper Reach	2	28.571
Q4 Urge To Move	No	1	14.286
	A Bit	1	14.286
	Yes	5	71.429
Q5 Confusion	Visualization of Ghost	2	28.571
	Limited Rotation	2	28.571
	Graspdifferences	1	14.286
	No	2	28.571
Various	More comfortable HoloLens	1	14.286
	Colorscheme	1	14.286
	No	1	14.286
	Perspective change	1	14.286
	Precision mode in all directions	1	14.286
	Improve moving virtual objects	1	14.286
	Less visual clutter	1	14.286
Total		7	100.00

Table 6: Content of the USB-Stick.

	Description	Location
Masterthesis	File of the thesis.	"/Dierks_Tim_MasterThesis.pdf"
Videos	Video of the prototype. Including the part of Franziska Rücker.	"/Videos/MasterThesis_Ruecker_Dierks.mp4"
	Demonstration of Pose Recognition system in small video clips.	"/Videos/6D_Pose_videos/*.ogv"
Sourcecode	Unity project of HoloLens application.	"/Application_Projects/UnityProject.zip"
	Python code of recognition system.	"/Application_Projects/PoseRecognitionProject.zip"
	XML to CSV conversion application.	"/Application_Projects/CSV_Converter_App.zip"
	Result analysis for stability/accuracy test application.	"/Application_Projects/AccuracyStabilityTest_App.zip"
Study Data	Results of user study.	"/Study Results/UserStudy_Data.xlsx"
	Results of stability/accuracy test.	"/Study Results/StabilityAccuracy_Data.xlsx"