

# RAM

● ROBOTICS  
AND  
MECHATRONICS

## DESIGN OF AN EYE-GAZE INTERFACE FOR CONTROLLING AN ASSISTIVE ROBOT ARM

M.H.W. (Marinus) Bos

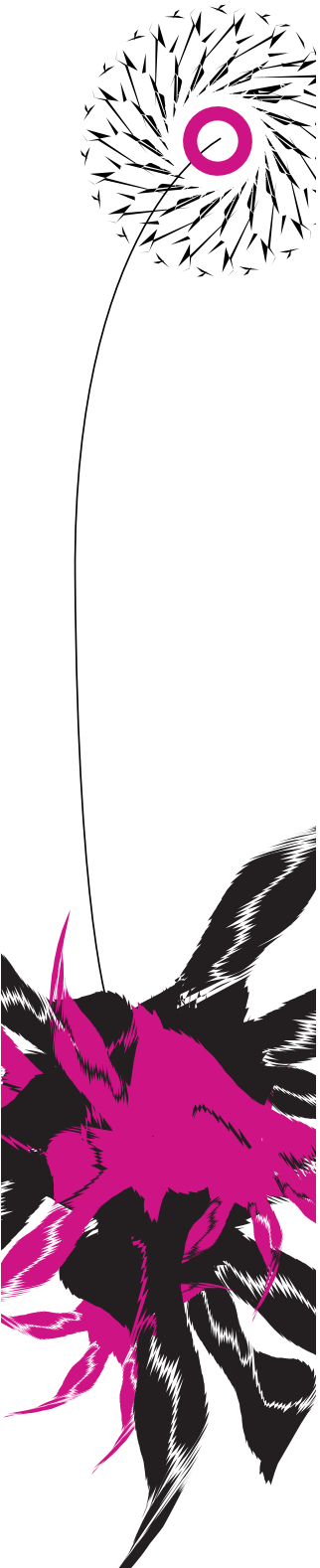
BSC ASSIGNMENT

**Committee:**

dr. ir. E. Dertien  
dr. ir. D. Dresscher

July, 2024

041RaM2024  
Robotics and Mechatronics  
EEMCS  
University of Twente  
P.O. Box 217  
7500 AE Enschede  
The Netherlands



## **ABSTRACT**

A system and interface were made for a client to control an assistive robot arm using gaze on a 2D web interface with the goal of allowing them to drink freely. The system was built upon prior work, extending it with improved gaze-controlled interfaces based on background research into gaze-controlled interfaces and control of assistive robot arms. Two interfaces were produced and tested with different balances between speed and avoidance of accidental inputs. Each interface provides full 3DOF control with the arm facing forward and control commands moving the arm continuously in an axis until the user stops it. The system was tested with 3 participants who gave generally positive feedback and a mixed preference between the two interfaces. A test with the client revealed that the system was difficult and straining for them to use, with the researcher believing that the client may lack the spatial awareness to translate a 2D interface into 3D space. Future research is suggested to create a system which can automatically move the arm to perform tasks.

## **ACKNOWLEDGMENTS**

Thank you to everyone at Ability Tech and the RAM group at the University of Twente as well as all participants.

# CONTENTS

<b>1</b>	<b>Introduction</b>	<b>5</b>
1.1	Research questions	5
<b>2</b>	<b>Background Research</b>	<b>6</b>
2.1	Context and stakeholders The client • Ability Tech • Interview with client	6
2.2	Prior work	8
2.3	Literature review The MASE guideline • Analysing existing interfaces	8
2.4	State of the Art Kinova Robotic Arm control • Tobii Eye Tracker	10
<b>3</b>	<b>Methods and Techniques</b>	<b>12</b>
3.1	The Creative Technology design process	12
3.2	Applying the design process	12
3.3	Understanding stakeholders	12
3.4	Testing	12
<b>4</b>	<b>Ideation</b>	<b>13</b>
4.1	Concept generation	13
4.2	Final concept	14
<b>5</b>	<b>Specification</b>	<b>17</b>
5.1	Requirements Quantitative Requirements • Qualitative Requirements	17
5.2	System components	18
5.3	Assistive robotic arm	18
5.4	Connecting to gaze-controlled computer	18
5.5	Camera	19
5.6	Mapping buttons to arm movements	19
5.7	Control axis	19
5.8	Hand control	20
5.9	Further features	20
5.10	Interface design	20
<b>6</b>	<b>Realisation</b>	<b>21</b>
6.1	System diagram	21
6.2	Raspberry Pi	21
6.3	Controlling the arm Continuous movement in cardinal directions	21
6.4	Connecting to the user's computer	23
6.5	Camera	23
6.6	Web interface Technical implementation • Interface description	25
6.7	Running the system	26
6.8	Allowing future work on the system	26
<b>7</b>	<b>Evaluation</b>	<b>28</b>
7.1	General user tests Testing method • Testing setup and procedure • Results	28
7.2	Testing with the client Testing • Evaluation	34
<b>8</b>	<b>Discussion &amp; Future Work</b>	<b>37</b>
8.1	Analysing requirements	37
8.2	Challenges	37
8.3	Comparison to prior work	37
8.4	Future work	38

<b>9 Conclusion</b>	<b>40</b>
9.1 Answering the research questions . . . . .	40
9.2 Conclusion . . . . .	40
<b>References</b>	<b>41</b>
<b>A Final concept</b>	<b>42</b>
<b>B User Interface Screenshots</b>	<b>46</b>
<b>C Explanation for testers</b>	<b>51</b>
<b>D Questionnaires</b>	<b>52</b>

# 1 INTRODUCTION

People with severe muscle impairments are restricted in their ability to interact with the outside world. One method which is commonly used to interact with a computer is gaze-control using an eye tracker, allowing the user to perform actions by looking at them. This form of input does not require the use of any muscles below the neck, as such it is often available even when most motor functions of the rest of the body are not usable, like for people with locked-in syndrome which are only able to communicate via vertical movement of the eye [1]. By connecting this input to a robotic arm, this form of control could be extended to the physical realm, granting more freedom and autonomy to the user.

This work is part of Ability Tech, which is an initiative where mostly students work on creating technology to increase the quality of life for people with impairments. One of its customers, Andrei, is unable to control most of the muscles below the neck and cannot speak. As such, he relies on a gaze-controlled computer mounted to his wheelchair to interact with the outside world. His wish is for more autonomy, including being able to move around, open doors and drink by himself. This research aims to help him accomplish this wishes by allowing him to control a robotic arm, building upon prior work by D. Gaethofs where it was proven that such a system was possible [2]. Furthermore, the output of this research could be distributed to other clients via Ability Tech, and parts of this research could be used for further development of other products.

This research implements a gaze control interface for a multi-axis robotic arm. The goal is to work towards creating a system which will enable a user merely using gaze to, among other tasks, drink from a container independently. The interface should be accessible via an existing gaze-controlled computer and the system mountable to a wheelchair. A multi-axis robotic arm is a complicated machine with many variables and the potential for unsafe behaviour, as such controlling it via the limited and unreliable input of gaze-control poses a challenge.

## 1.1 Research questions

This research attempts to answer the following research questions:

**Research question:** How can a system be designed to enable an assistive robot arm to be controlled using gaze?

**Sub-question 1:** What is the current state-of-the-art of gaze-control and controlling assistive robot arms?

**Sub-question 2:** Can users accurately stop the robot arm at a desired position by performing a gaze interaction at a precise time during its movement?

**Sub-question 3:** Is a separate confirm button preferred for preventing accidental movements?

**Sub-question 4:** Should control of the robot arm be from the reference frame of the hand instead of the base of the arm or the physical position of the user interface?

**Sub-question 5:** Is full manual control suitable for a person with disabilities to control a robot arm?

In chapter 2, background research is performed. In chapter 3, the methods and techniques for performing the research are defined. In chapters 4, 5 and 6 the product which will be tested is developed during ideation, specification, and realisation phases. In chapter 7 the system is tested where the remaining sub-questions will be answered and in chapter 8 the results are evaluated and discussed, and future research is suggested. In chapter 9 the research questions are answered, and conclusions are made.

## 2 BACKGROUND RESEARCH

The research started by collecting relevant information from various sources, including existing research papers as well as stakeholders. First a literature review was performed to analyse design aspects and existing gaze-controlled systems to learn more about different aspects of gaze-control. From this, a guideline was created to highlight aspects which are especially important for interfaces designed for gaze. Then, state-of-the-art technology as well as the prior work were analysed. Finally, an interview was performed with a client of Ability Tech. The results of this research serve as a basis for designing the system.

### 2.1 Context and stakeholders

The first part of the background research was to understand the goals and context of the project. This information was gathered through meetings with the supervisor where the initial assignment was explained, as well as various informal meetings with the stakeholders.

#### 2.1.1 The client

The goal is to build an interface for the client to control a robot arm using gaze. The client, Andrei, has disabilities which mostly prevent him from controlling most of his muscles except for his eyes and to a limited extent his neck and mouth. He also cannot talk, although he can produce a limited set of sounds through his mouth. When with the researcher, Andrei is usually accompanied with his dad who acts as a caretaker. Andrei's caretaker is, among other things, responsible for feeding him and moving his wheelchair, as well as acting as a sort-of interpreter between other people and Andrei. Andrei can communicate through moving his eyes, mouth, neck, making sounds through his mouth as well as the intensity of movements in the rest of his body, although it is unclear as to the amount of direct control he has over most of these actions. Communication with Andrei is usually done via his caretaker who interprets Andrei's movements and sounds as well as using techniques to get clearer communication via Andrei's eyes. Andrei does understand spoken and written language, although he appears to understand communication by his dad better than the researcher.

Andrei uses a Tobii Gaze-Controlled computer to do tasks such as writing and drawing. This computer is mounted to his wheelchair and can be removed if Andrei wants to better look at his surroundings or rest his eyes. Extended use of the gaze-computer is tiring for Andrei, and he usually prefers to be socially involved in those around him. Although Andrei can use this computer to talk via text-to-speech, the researcher has not seen him do this. This research should extend the use of his gaze-controlled computer to be able to physically interact with the outside world to allow him to grab objects or drink autonomously.

#### 2.1.2 Ability Tech

This project is made for Ability Tech, an organisation for which Andrei is currently the primary client. Ability Tech operates out of the University of Twente. They manage projects which they work on themselves as well as turning them into assignments for students. Their projects include a gaze-controlled wheelchair, gaze-controlled sjoelrobot, and testing the viability of a button mounted to the headpiece of Andrei's wheelchair which could be used as an emergency stop. They also make projects for other clients, such as a system to help deaf kids sleep better by informing them of what's happening where in the house. More information about Ability Tech can be found on their website.<sup>1</sup> This project, the gaze-controlled robot arm, is part of the projects to grant more autonomy to Andrei.

Throughout the project, various informal meetings with stakeholders from Ability Tech were held. These meetings were not specifically about this project, but helped to keep contact with the stakeholders to learn more about their priorities and wishes. From this, it was clear that the priority of this project should not just be on the research about the user interface and user-friendliness of the system, but also on creating a system which can be expanded upon by future developers and become usable not only by the main client, but that is also reproducible to be used by future clients. There is also a focus on using readily available parts, so that those with the right technical knowledge can reproduce, repair and modify the system without the involvement of an organisation like Ability Tech.

The researcher also attended a symposium organised by Ability Tech where the organisation presented their current projects and discussed their goals and philosophy alongside guest presenters who talked about a variety of accessibility tech related projects. From this it became clear that a goal of Ability Tech is to provide happiness through increased autonomy. This is reflected in the currently active projects and should be reflected in choices made about the design and implementation of the project.

#### 2.1.3 Interview with client

An informal interview was conducted with Andrei and his dad about their experience with gaze-controlled systems. It took place during a visit from Andrei and his father to a meetup of Ability Tech. The interview was prepared with a few questions, but the interview was not intended to be structured. During the interview

---

<sup>1</sup><https://abilitytech.nl/>



**Figure 1.** The client, Andrei, using his gaze-controlled computer to control a sjoel-robot created by Ability Tech

Andrei demonstrated various uses of his gaze-controlled computer while several questions were asked to him and his dad. Due to Andrei's reliance on his computer to communicate it was difficult for him to answer more complex questions and he seemed hesitant to type out an answer. Instead, most communication went via his dad who either answered via his experience being with Andrei or by asking the question to Andrei and interpreting his body language and non-verbal audio responses. During the interview a few notes were taken, and afterwards a summary of the observations was made.

In the demonstration Andrei demonstrated multiple programs he uses and the way he interacts with them. One program he showed was Tobii Dynavox Communicator which he used as an alternative for talking. It features a place to type out a message and the ability to read it out loud by the computer. The user interface shown used large buttons on a grid, making selection easy but limiting the number of buttons available at once and requiring the keyboard to be split into multiple pages [3]. He also showed himself creating art using a software intended to be controlled using a mouse. This software had small buttons and relied on holding a mouse button while moving the cursor. To enable the use of this program, software was used to translate the gaze-input into mouse commands while providing tools to improve the usability of programs not optimised for gaze-input. Gazing at a part of the screen caused that part to be magnified, then gazing at any point on the magnified display caused a mouse action at that point and removed the magnification. The mouse action to be performed was determined beforehand via a menu opened by gazing at a bar on the edge of the screen. Although Andrei was able to successfully use these interfaces, it was clear a number of concessions were made to make them work with gaze-control.

During the demonstration various observations were made and topics were discussed based on prepared questions, the given demonstrations and topics brought up by the parent. In the beginning of the demonstration Andrei appeared to be struggling with the interface. This was likely due to Andrei not being perfectly aligned with the display, causing the calibration to be incorrect. This may cause if the head moves at a time when correct input is time-sensitive. An observation was made that selecting the bar to change mouse function appeared to be difficult, but it is unclear whether this was due to its size, placement along the edge of the display or due to the improper calibration. A major topic brought up by the parent was the need for resting the eyes and ability to observe the surroundings. He stated that using the computer was tiring, and that Andrei often had to take breaks. As such, it is important for them to be able to look away from the display to rest his eyes at any moment. Another reason for looking away is to be able to interact with people around them. The parent stated that Andrei values looking at other people and considers it an important part of being involved with people around them. An observation was made that he has his dwell-delay set to 800 ms, which is the upper value found in the literature review. Even then, the parent explained that accidental inputs were a factor when he wanted to look at possible options to select. A particularly frustrating situation mentioned was when he looked at the words selected for auto-completion and accidentally select the wrong one. This then requires multiple inputs to remove each erroneously placed letter one at a time. An undo button was suggested as a solution to this issue. A final point discussed was familiarity, where unfamiliar interfaces may take time to learn and cause more fatigue. An example of this was that although a new version of the mouse control software was available, Andrei chose to stay at an older version out of preference. In conclusion, this interview revealed a number of requirements and preferences for the final interface.

## 2.2 Prior work

This project has been worked on before by Damian Gaethofs under Ability Tech in his report titled Adaptive Control of a Kinova Assistive Robotic Arm [2]. This work will form the basis of the current project in both research and implementation. Damian's work is a proof-of-concept, showcasing the possibility for a robot arm to be controlled using gaze. It uses a Kinova Jaco robot arm controlled via the ROS software system. A webpage is hosted by a local webserver (figure 3 through which the robot arm is controlled via simple commands, most of which each move the arm for 1 second in a given axis. There is also the functionality of opening and closing the hand, resetting the arm via an automatic calibration procedure and enabling/disabling all commands to the arm. A further analysis of this interface was performed during the literature review in section 2.3.2. This project intends to build upon this existing system, implementing recommendations from it and expanding it while using mostly the same hardware.

The prior work runs four command-line applications simultaneously. These include the Kinova-ROS stack to send commands to the arm, a custom ROS node which forwards external messages to the Kinova-ROS stack, a HTTP server to host a web interface, create a camera stream, and forward button presses to ROS, and finally a rosbridge server which allows the HTTP server to send messages to ROS applications. Figure 2 contains a diagram created by the author of this work which displays the components of the system and the communication between them. The web interface is JavaScript-based, with buttons intended to be pressed by Tobii software which interprets gaze movements as mouse movements and mouse button presses. When a button is clicked, a message is sent to the web server via a websocket. The webserver is written in Python using Flask as well as CV2 to host the camera stream. It receives the button inputs from the web interface and translates them into messages for the ROS node, including assigning axis to each directional button and repeating directional commands a hundred times which causes the arm to move for one second. The webserver sends messages to rosbridge via a websocket, where these messages are received as JSON and forwarded as ROS messages. These ROS messages are received by the ROS node written in C++ which translates them into Kinova-compatible messages and service calls.

Some decisions made during the prior work will carry over during this project. For example, the prior work decided to control the robot arm in coordinate space and not in joint space based on an interview with the clients, this will carry over to this research. The research also decided not to use complicated methods to derive a 3D setpoint, this will remain out of scope for this research. The goal of Andrei to be able to drink autonomously will continue to be a target. A camera was used as the display with the interface may block the view of the robot arm, a concept which this research expands upon. The research had a goal to integrate neatly with other projects such as the gaze-controlled wheelchair and to use Andrei's existing gaze-controlled computer, which this research will work towards.

The previous work left a number of recommendations which this research will attempt to improve upon. The main recommendation was to lower the completion time of a task, for instance by designing a different control interface which is a major goal of this research. It recommends moving the back-end of the system to a Raspberry Pi or similar computer and establishing a local network, which this project accomplishes. It states the need to consider the mounting point of the camera which this research works on. Lastly, the previous work did not do a test with Andrei, this work does perform a test with them.

Some recommendations of the prior work are not expanded upon in this research, such as mounting the robot arm to the wheelchair. A drinking mode will not be implemented, and only a beginning is made on the implementation of a system for saving and reusing points in space. The prior work also defines a testing procedure for the arm, but this cannot be replicated using the system in this research.

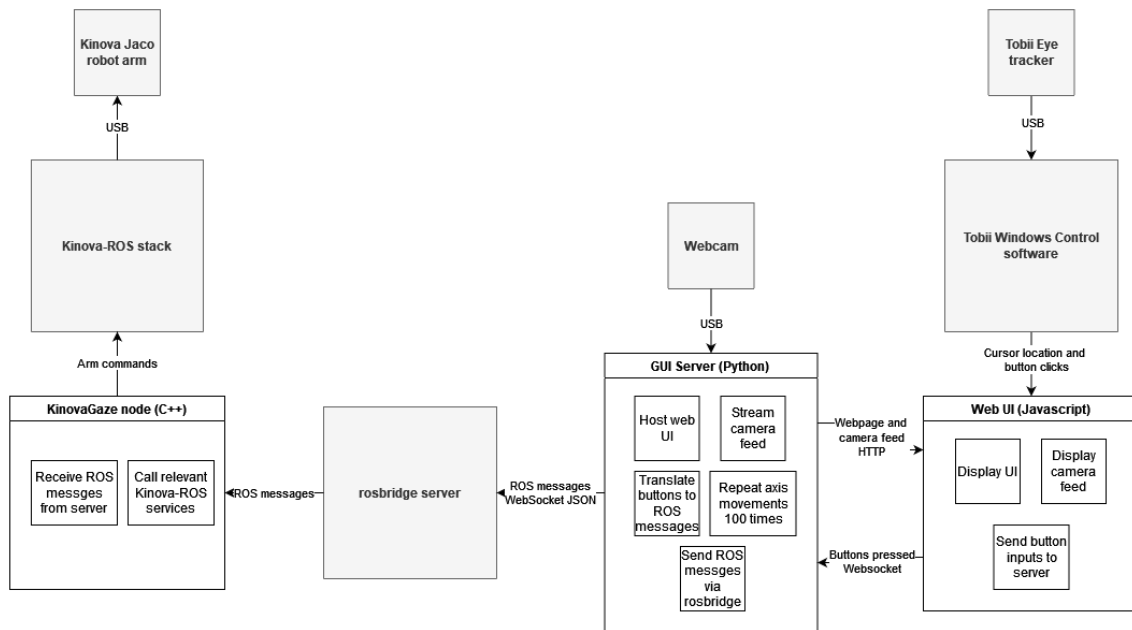
## 2.3 Literature review

A literature review was done to explore previous work to build upon. The goal was to find which attributes and limitations were important when designing for gaze-control, which solutions were found to be effective or ineffective and to analyse interfaces used in the literature for inspiration. This research was performed in a separate unreleased paper, a summary of the relevant parts is provided below.

### 2.3.1 The MASE guideline

Gaze-control has various attributes that an interface should be designed around to provide a good experience. To analyse whether an interface is designed appropriately, the MASE guideline was introduced. This guideline proposes the following four qualities that a gaze-controlled interface should be designed for: Midas Touch, Accuracy, Speed and User Experience. Midas Touch was defined by R. Jacob [4] to be the issue of unintentionally interacting with objects when the user is trying to read information from the display or resting their eyes. This is often avoided by introducing a dwell-delay ranging from 80 ms [5] to 800 ms [6], with a lower delay being preferred when the results of incorrect inputs are easy to undo [4]. However, this does not provide a solution for continuous input [7]. The other main limitation of gaze-control is accuracy, which is a limitation of our current technology as well as our eyes [5]. Small and densely packed buttons are





**Figure 2.** System diagram of the prior work by D. Gaethofs [2], as interpreted by the author of this work.

difficult to accurately select [8]. Smoothing inputs can improve accuracy [9], and magnification can help make existing interfaces usable [5]. Midas Touch and Accuracy are the main limitations of gaze-tracking, and accounting for them is required for a gaze-controlled interface.

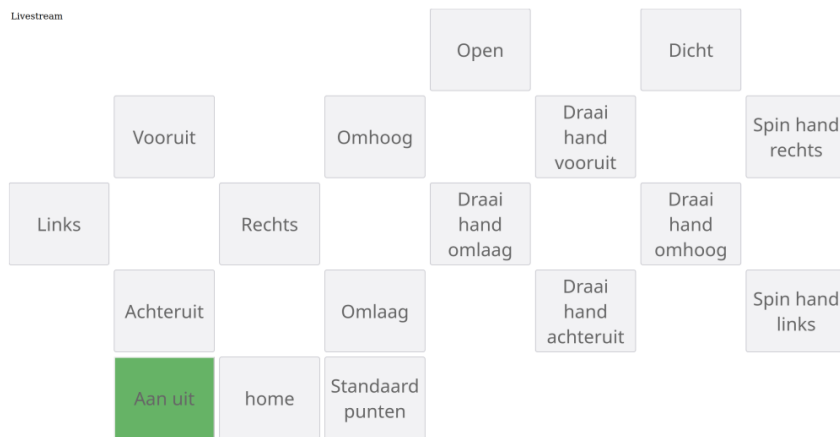
However, accounting for Midas Touch and Accuracy can come at the expense of the Speed and User Experience of the interface. The common dwell-delay method adds a minimum time to every input [5], and the replacement of continuous inputs with multiple individual inputs slows interfaces even further [10]. Furthermore, one should consider the real-world consequences of design decisions. Long dwell times can cause eye-fatigue [8], and freedom to look away from the screen is important for resting the eyes, social interactions and observing the surroundings [10]. Looking down too much or being locked in a virtual reality headset can be socially excluding and uncomfortable [11]. Autonomy is also considered very important for those relying on gaze-controlled interfaces [11], and a virtual reality headset sacrifices autonomy for those who cannot adjust or remove it themselves [10]. Overly automated systems should also be avoided for the same reasons. Real-world experiments about the speed and user experience of an interface should be performed to ensure the interface is nice to use.

In conclusion, the MASE guideline tells us to focus on eliminating accidental inputs (Midas Touch), to consider the limited accuracy of gaze-input, to look at the speed of an interface when comparing options and to keep in mind the user experience. Furthermore, autonomy is important for people with limited mobility, as such they should not have to rely on other people to use the interface.

### 2.3.2 Analysing existing interfaces

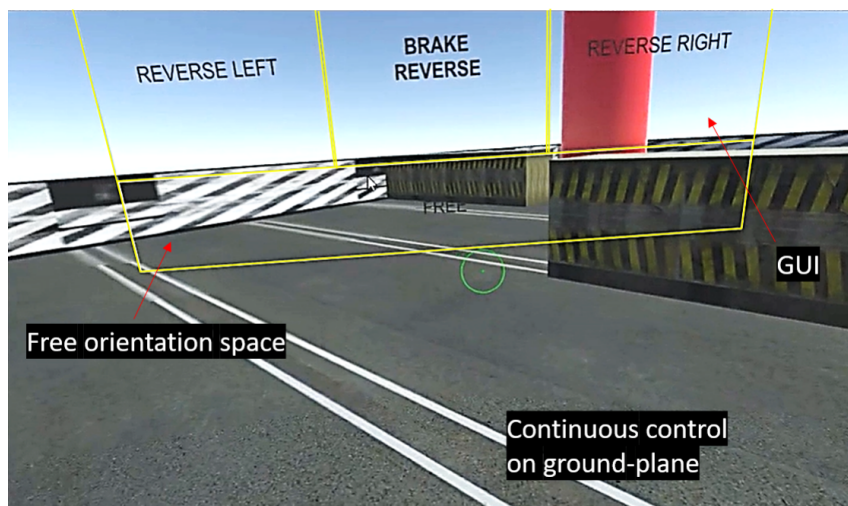
The interface used by D. Gaethofs [2] shown in figure 3 was analysed as a starting point for this project. The interface contains buttons activated by dwell-delay which move the arm by a set amount in a certain direction or axis. The buttons are overlaid on a camera feed of the robotic arm, allowing for feedback on the movement of the robotic arm without looking away from the interface. The buttons are positioned in a checker-board pattern, allowing the user to gaze between them to observe the robotic arm without performing an action. There are also buttons to move the robotic arm to preset states, as well as an on/off button to disable the other buttons when no action is intended. Analysing this interface reveals a number of qualities and shortcomings. The buttons are semi-translucent with text labels on top and are positioned along the entirety of the camera feed, allowing for accidental inputs when observing the state of the robotic arm or when reading the labels. Accuracy was mostly accounted for by making the buttons large, but the paper does mention an issue where it was easy to accidentally trigger a button from its corner. The speed of the interface is low as a result of the dwell-delay buttons having no way to move a large distance in a single movement, which the paper notes as a primary point of improvement. Overall, this interface is easy to use but there is room for improvement in a number of categories.

The interfaces used by J. Araujo et al. [10] to control a wheelchair were also analysed with a focus on the continuous-input interface shown in figure 4. This interface allows the user to move a motorized wheelchair by looking in a certain direction on the ground, where looking farther away makes the wheelchair



**Figure 3.** Robotic arm interface from D. Gaethofs [2]. The white background was replaced with a camera feed of the robotic arm during operation and the buttons are translucent.

move faster. Three buttons at the top allow for braking and reversing in three directions without looking backwards. A space is provided where the user can analyse their surroundings without affecting their movement. UI elements float in front of the user and have no background, only an outline. This interface was used via a Virtual Reality headset. This interface uses a significantly different design from the previously discussed solution, and as such provides different qualities. As the actions are activated immediately when gazed at, there is a lot of opportunity for Midas Touch-related issues. There is no way to read the label of buttons without triggering their actions, and no way to ensure the user is looking at the right spot below the free space without moving. There is also no way to disable movement altogether, and as the VR headset covers most of the field of view there is no way to look away from the free space to rest the eyes. The speed of this interface is high as the elements respond instantly to gaze. Movement is continuous and the speed of moving forwards and turning can be controlled. This interface provides a contrast to D. Gaethofs [2] by sacrificing Midas Touch-reduction but enhancing speed.



**Figure 4.** Continuous control interface from J. Araujo et al. [10] for controlling a wheelchair. The original transparent grey borders are made yellow for this illustration.

## 2.4 State of the Art

### 2.4.1 Kinova Robotic Arm control

To gain an understanding of currently available solutions relevant to this project two existing products were analysed, starting with the control interface Kinova® Jaco® assistive robotic arm and the associated joystick controller [12]. This is the robot arm which was used during the development of the product and is specialized for use by people with disabilities. The default control scheme is designed around a 3-axis joystick which can tilt on two axis and also rotate as a third axis, although an alternate control scheme without the rotation axis of the joystick is also available.

The control schemes are split up into different modes, depending on which axis of the joystick are enabled. The 3-axis scheme separates control into four separate modes, translation, wrist, drinking and fingers. Translation moves the hand in space while maintaining its orientation, wrist controls the orientation of the hand, drinking allows the robot to make a pouring motion, and fingers allows the user to open and close either 2 or 3 fingers of the robot. Two-axis control creates an additional mode and moves the vertical translation and wrist rotation into this mode. The hand moves relative to its orientation, meaning that pushing the joystick forward means that the hand will move towards the direction it is facing, not necessarily the direction that the joystick is pushed. Finally, there are buttons to move the arm to preset states including home and parked, as well as a control for the speed of the arm. In all modes the user only ever has direct control of the hand, the movement of the joints is controlled automatically to achieve the desired state. This control scheme can be used as the basis for the available features and organisation of a gaze-controlled interface.

#### **2.4.2 Tobii Eye Tracker**

Secondly an analysis was performed of the Tobii Eye Tracker 5, specifically for gaming applications. Gaming applications for gaze-tracking are relevant because gaming often has a 3D world projected onto a 2D screen similarly to a camera view of a robotic arm. In their marketing material it shows that in certain games played from a first-person perspective, the player can aim their weapon at an enemy merely by gazing at them with their eyes [13]. A similar system may be used to mark a target for the robotic arm to grab.

### 3 METHODS AND TECHNIQUES

To create and evaluate a system, a system was followed which defines various steps to get from an initial topic to an evaluated system which is suitable for the needs and desires of the stakeholders. To aid with this, various points of contact were used to gain a better understanding of the stakeholders and to allow for a feedback loop. The general concepts are described in this section, and the execution and results are described in the rest of the paper.

#### 3.1 The Creative Technology design process

The design process described in "A Design Process for Creative Technology" by Angelika H. Mader and Wouter Eggink [14] will be used as the basis for creating the design of the user interface. In this process, an edition is performed on a design question, in this case the research question, to create initial designs for a product. It then defines four phases, ideation, specification, realisation and evaluation. The ideation involves collecting stakeholder requirements, collecting available existing technology and generating creative ideas towards a solution. After some ideas are generated the specification phase is entered where various prototypes are created, evaluated and often discarded. These prototypes are often of small parts of a total system to allow for clear results and rapid iteration. In the realisation phase the product is realized, often in individual parts which are then combined and validated. Finally, an evaluation part looks at the result and evaluates whether it fits the requirements set in the ideation phase and whether it satisfies the user's needs.

The Creative Technology design process is applicable to this project as there is a strong connection between the projects that the process is intended for and the specifics of this project. There is a design question, namely how to make a good interface for controlling a robot arm using gaze. The technology used in this project already exists and is in widespread use, however the full potential has not yet been explored and there is active research into the use and development of gaze-controlled interfaces [13]. Finally, the result is meant for direct human usage.

#### 3.2 Applying the design process

To define the methods and techniques which were be used during this project, the Creative Technology design process was applied. For the ideation phase stakeholder requirements were elicited through an interview with the client as well as relevant stakeholders. Background research has been performed to collect technology, existing implementations and results of prior work. Idea generation was done via brainstorming sessions as well as gathering ideas during other parts of ideation. Afterwards a list of requirements, ideas for features of a user interface and a mock-up of an interface were produced and discussed with stakeholders. For the specification phase a number of tests were produced and performed to test individual parts of the proposed interface and the concept was adjusted based on the results. To perform tests using the robotic arm the setup used by D. Gaethofs [2] was replicated and adjusted. In the realisation phase the setup used during the previous phase was expanded upon to implement all features of the design. The system was then installed onto the wheelchair of the end-user and various tasks were performed using it. For the evaluation phase feedback from the user was collected as well as various data on the product such as task performance time and prevalence of incorrect inputs. Finally, the requirements were checked, and conclusions and recommendations were drawn.

#### 3.3 Understanding stakeholders

To gain a better understanding of the stakeholder needs and desires, close contact will be held with the stakeholders throughout the duration of the project. Part of this contact will be in the form of (informal) meetups at Ability Tech, usually over dinner and during times they work on a system with a close relation to this project. The researcher also joined a symposium held by Ability Tech where various speakers held talks related to living with disabilities and designing products to help those in need. Finally, weekly meetups with a supervisor who also supervised the prior work by D. Gaethofs [2] and understands the longer-term goals of the project. These meetings help shape the focus of the project and are an opportunity to get feedback from stakeholders with more experience using gaze-interfaces and working with people with disabilities.

#### 3.4 Testing

Testing will be done during the creation of the project as well as with the client and multiple users. A short feedback loop of implementing a feature and then using it is used to discover immediate changes to improve the use of the product to a state in which results from further testing should provide more useful results. Testing with external users will be done not just be done in a strict, controlled environment with predefined actions and outputs to minimise external influences, but also in a more lenient way to allow the testers to give more opinionated feedback which may otherwise be lost in assumptions of the researcher during the design of tests and variables.

## 4 IDEATION

To begin with development of the system, an ideation phase was performed to generate initial concepts. Applying the Creative Technology design process [14], the ideation phase began with the research question "How can a system be designed to enable an assistive robot arm to be controlled using gaze?" First background research was performed as described in section 2. Then, concept generation was performed to eventually arrive at a final concept to bring to the specification phase.

### 4.1 Concept generation

Based on the output of the background research, concept generation was performed. As the hardware and software layer were already created by the prior work of D. Gaethofs [2], the concept generation focused on the user interface. First, a brainstorm was performed to come up with a list of elements that could be included in the interface, the result of which can be shown in figure 5. Then in another brainstorming session sketches were produced for potential interfaces, the output of which can be found in figure 6. Finally, these concepts were developed into a mock-up of a full interface and presented to stakeholders.

- Label next to button
- Dwell, then move slider for direction and speed
- Safe box to look at feedback
- Slider stays engaged but doesn't move while gazing in safe box
- Split slider into sections each with a speed
- Current slider section becomes larger
- Instant gaze in area to end movement
- Gaze off-screen to end movement
- Cancel button replaces options to switch interface options
- Buttons with stored locations
- Undo-button
- Separate buttons for nudging and for engaging
- Dwell multiple times to increase speed
- Circle interface with direction buttons
- Separate buttons for direction and speed/nudge
- Look off-screen in direction to change menus

Figure 5. List of potential interface elements resulting from first brainstorm.

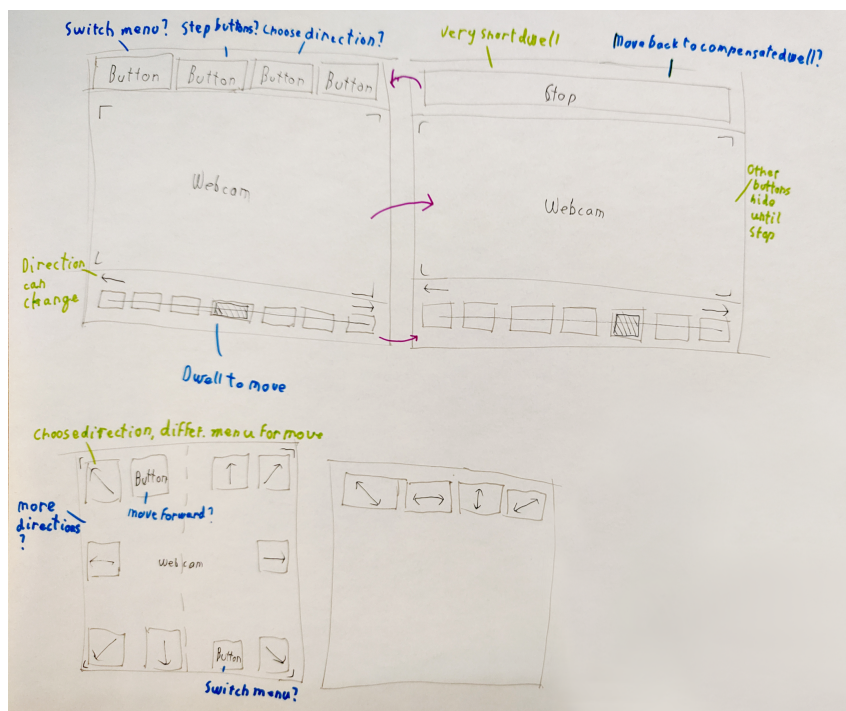


Figure 6. Sketches of interfaces resulting from second brainstorm

## 4.2 Final concept

The final concept was presented as a mock-up where an object is picked up. Figures 7 show some highlights, and the full mock-up is included in appendix A

The first screen of the interface contains three major elements: Mode buttons at the top, a camera feed in the centre and an information box in the bottom. In these initial concepts, the webcam view is mounted on top of the hand. This design element was inspired by the first-person games used in the demonstration of the Tobii Eye Tracker 5 for gaming [13] as well as the camera view when controlling moving vehicles like in J. Aurojo et al. [10]. This view is intended to help with judging whether the arm is facing the object and when the object is in the hand of the arm. The location of the camera is further discussed in section 5.5. The four mode buttons are big and do not overlap the webcam view to prevent accidental inputs. Buttons to switch between interfaces are placed at the top to create similarity between tabs in common existing interfaces like tabs in a web browser, and to create a visual hierarchy where buttons on top affect the options below. The information box at the bottom is intended to aid in learning the interface, while also keeping a consistent layout between modes. This way it is easier to know which parts of the screen are part of the webcam without gazing at them.

Once a mode button is selected it becomes highlighted and the interface for the selected mode is revealed, each interface following a similar structure. In all modes the user can switch between continuous and nudge mode, the buttons for which are overlaid over the information box to not take up space on the webcam. In all modes except rotate, yellow buttons are overlaid in the corners of the webcam feed to control the direction/axis that the movement will have. In the case of rotate, a direction can instead be chosen by dwelling on the camera feed and is indicated by a yellow circle and line. Once a direction is selected, the information box and continuous/nudge buttons disappear and are replaced by a speed select or nudge activation button, depending on which mode is selected. The top mode selection buttons also disappear and are replaced by a stop button. Once movement is initiated once, the direction buttons/circle become locked to prevent accidental direction changes. The stop button immediately stops all movements of the arm and removes the direction selection, returning the mode select and continuous/nudge buttons as well as the information box and unlocking the direction selection.

The interface is split into four main modes, these being rotate, forward, strafe and hand. Rotate allows the hand to be rotated among two axes to make it and the camera look at a specific object. The direction is chosen by looking at a point on the 2D webcam view relative to the centre, and the robotic arm moves in the direction so that the selected point crosses the centre of the view. Forward allows the hand to move forwards and backwards in the direction that it is looking. Strafe allows the hand to move in 8 directions on a 2D plane perpendicular to the looking direction of the camera. Finally, hand mode allows the hand to be opened and closed as well as spun in the roll axis which is not covered by the rotate mode. There is also a fifth mode which is engaged by selecting the current mode again, this exits the mode and goes back to the empty initial mode.

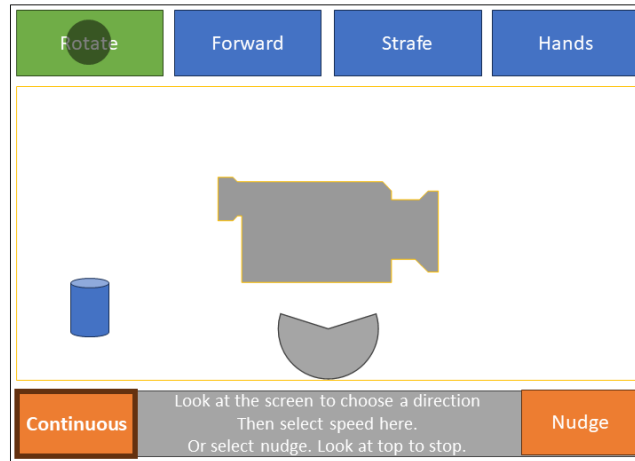
The modes of the interface are intended to be used in order for most tasks and are inspired by the 2D mode of the Kinova Assistive joystick [12]. To perform an action, the hand is first rotated to point towards the point of interest, it is then moved forward toward that point, strafe is then used to correct for minor errors and hand is then used to perform a task. The continuous mode is intended to perform big movements quickly, and the nudge mode is intended for small corrections. The continuous/nudge selection is retained between menu mode changes. Finally, the empty initial mode can be used to rest the eyes and look at the webcam view without buttons in the way.

This interface template was designed with a number of considerations. Each mode of the interface follows a standard layout to keep them familiar and to limit the amount of searching needed. The stop button is always in the same place and very large to make selecting it easy. It also has a negligible dwell-delay to ensure the arm stops immediately and precisely. The centre of the webcam view is designed to be the most important part of the interface, as such it is never covered by large menu items. However, the corners may contain the point of interest in the rotate mode, as such this mode does not overlay any buttons on the camera feed. Instead, this mode uses a selection mode inspired by the targeting system used by the Tobii Eye Tracker in first-person games as well as the continuous and waypoint control used in J. Aurojo et al. [10]. This selection mode is intended to make it easy to select a point of interest and rotate towards it with more precision than traditional directional buttons. In other modes yellow buttons are overlaid over the webcam view in the corners and may be accidentally pressed, but the consequences of this are minimal as a different yellow button can easily be changed again or the stop button can be selected to cancel the selection. The idea of not preventing accidental selection when the consequences are minimal is inspired by [4]'s approach to selecting dwell times, and this approach might also be used for selecting dwell times for the buttons depending on the results of testing.

This concept was shared with stakeholders to gather feedback. The dad of the client from the interview

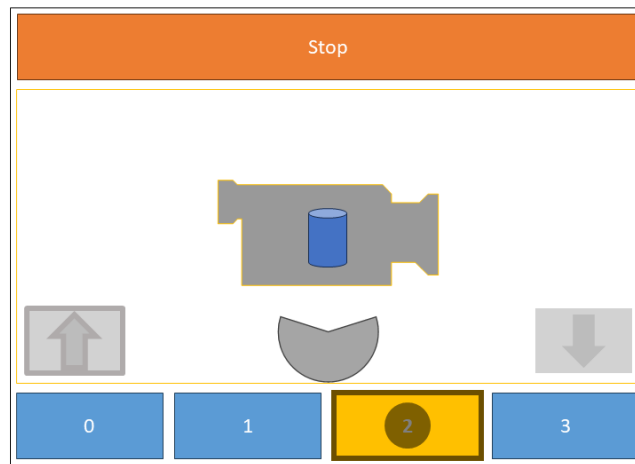
**Figure 7.** Final concept mock-up highlights

First we select the rotate option to make the webcam face the object we want to grab. Since we want to make a large adjustment, we'll keep the mode on continuous.



**(a)** Final concept in rotate continuous mode without a direction selected.

We'll choose a moderate speed of 2.



**(b)** Final concept in forward continuous mode currently moving

in chapter 2.1.3 was shown the interface and was positive about it with no suggestions for improvement. Another stakeholder from Ability Tech was worried that the stop button would be too difficult for the user to hit, especially in the case of panic or calibration issues. To remedy this issue a suggestion was made to stop the robot arm moving when the user looks away from the screen or a button, further testing should reveal whether this is necessary as it does sacrifice the ability to look at the robot without looking via the robot arm.



## 5 SPECIFICATION

A specification was made for the system, derived from the background research and the ideation. First a list of requirements was made, then a description of specifications was made to fulfill these requirements.

### 5.1 Requirements

A list of requirements was created for the final prototype. As this project builds on the prior work of D. Gaethofs [2], a number of requirements are continued from it.

#### 5.1.1 Quantitative Requirements

**Requirement 1:** The system must be able to control a robotic arm using an eye tracker

This is the base requirement for the system, required to test upon the main research question.

**Requirement 2:** The interface must be controllable using an existing gaze-controlled computer.

This is a requirement for the main customer, as they already have a computer which is mounted to their wheelchair which they are used to.

**Requirement 3:** The user should be able to pick up a container of liquid and move it with the robotic arm.

Meeting this requirement is needed to reach the eventual goal of the client being able to drink with the robotic arm. Meeting this goal shows potential to reach the stakeholder wish of using the robotic arm to drink.

**Requirement 4:** The interface should be able to be disabled

As specified by both the MASE guideline and the interview with the stakeholder in the background research, the interface should be able to be disabled to allow the user to rest their eyes and read the interface without accidentally triggering an option.

**Requirement 5:** The system must be completely mounted on and powered by a powered wheelchair.

The system needs to travel with the client, and the client should not be dependant on an external power outlet or device for the system to function.

**Requirement 6:** The user should be able to use the arm without looking away from the screen

As stated by D. Gaethofs, [2], the screen may block the view of the robotic arm. As such, the system should be usable without looking away from the arm. This could be accomplished by incorporating a camera feed.

**Requirement 7 :** All controls needed to operate the system should be easily accessible via gaze

This ensures full autonomy by the user of the system, as was found to be important in the background research. If an option is not easily accessible via gaze, this might require help from a caretaker and as such reduces autonomy.

#### 5.1.2 Qualitative Requirements

**Requirement 8:** The system should be simple to setup for customers

This is important for allowing further customers to use the system as described in section 3.3.

**Requirement 9:** The system should be easy to learn

The interface should be intuitive and overly complex. As found in the stakeholder interview, the interface should not be too dissimilar to what the client is already used to.

**Requirement 10:** The system should be fast to use

The main point of improvement of the previous work by D. Gaethofs [2] was that the system was slow. As such, the new system should be faster to perform a task than the previous work.

**Requirement 11:** The system should prevent accidental inputs

The MASE guideline defined in the literature review states that Midas Touch and low accuracy means that buttons should be big and well separated to prevent accidental inputs.

**Requirement 12:** The system should cause little fatigue or discomfort to the user

As found in the literature review and stakeholder interview, using gaze-tracked interfaces can be fatiguing and cause discomfort if used for a long time, especially if there are long dwell times [8] or uncomfortable angles [11].

**Requirement 13:** There should be minimal risk of unintentionally hitting objects

A robotic arm has the potential to cause risk to the surrounding environment, this should be minimised. This means good control over the movement of the hand and being able to stop it quickly. The arm between the hand and the attachment point should also not hit surrounding objects.

**Requirement 14:** The system should be well-documented

As described in section 3.3, it is important for future developers to be able to continue to work on the system, as well as users to be able to make modifications to the system. As such, good documentation should be provided to aid with this.

**Requirement 15:** The system should be easy to reproduce.

As described in section 3.3, it is important for the system to be reproducible so copies can be built for other customers and so that existing systems can be repaired.

**Requirement 16:** The system needs to be reliable.

The client needs to be able to rely on this system in his daily life, as such the system should be reliable and not have frequent failures.

## 5.2 System components

To meet the requirements the system should consist of at least 4 primary components, this being a display with eye-tracker, a robot arm, a camera and a controller. Controller in this case refers to a device responsible for managing the different functions of the device, hosting the interface, relaying camera input, responding to user input and commanding the robot arm. In the case of the initial customer Andrei, the display with eye-tracker should be his Tobii computer. This reduces the difficulty of learning the system as he is already used to this computer, and it is already mounted to their wheelchair thus making installation easier. This computer has the capability of acting as the controller, however this likely requires installing a lot of proprietary software which may not run on their computer without complication, making the installation significantly more difficult. As such a separate controller is used. For testing without the Tobii computer, a different computer with external gaze-tracker and appropriate software may be used.

Using a separate controller also helps with allowing the system to keep working in the future as external support for the hardware and software ends. The software needed to control the arm is nearing its end of life at the time of writing, with the Kinova-ROS control stack not having been updated in a year as the company developing it moved on to new software which is not compatible with the older arm used in this project. The software on which the stack runs, including the operating system, will be end-of-life and thus stop receiving security updates in less than a year as of the writing of this report. The control stack is based on ROS 1, which will be deprecated once support for the current version ends and control stacks for future hardware will likely all be based on a newer system. By running this outdated control software on a separate device, security issues are reduced as this system is not directly exposed to the internet, and compatibility issues are resolved as the software for other hardware can be ran on a completely separate device.

## 5.3 Assistive robotic arm

The robot arm to be used in this project was a Kinova Jaco v2 6DOF arm with 3 fingers. This arm was used as it was already available and used in the prior work by D. Gaethofs [2]. The Kinova Jaco arm is advertised as an assistive arm, and it is intended to be mounted to a wheelchair and powered by the wheelchair, thus satisfying requirement 5. This arm is most commonly controlled by a joystick, but for this project a software interface solution should be used.

## 5.4 Connecting to gaze-controlled computer

The connection between the display with eye-tracker and the controller needs to be reliable, universally supported, easy to set up and fast enough to carry a live camera feed. Although the last requirement is not necessary if the camera is connected directly to the display instead, this adds extra complexity to the user experience as their computer now needs to be correctly set up to use the camera. The connection needs to be universally supported as this increases the reproducibility of the system. Reliability is key as to improve the performance of the system, and a sudden failure of this connection may cause the robot arm to hit something due to the loss of control. Using external communication such as an existing Wi-Fi router should be avoided as this decreases reliability and may not always be available, and wireless connections in general should be avoided as they are less reliable depending on interference nearby.

## 5.5 Camera

The placement of the camera is a complex issue and will need further research, as such a solution with major compromises will need to be used in the system. The camera needs to aid the user in many tasks, including aligning the hand with objects to grab, placing objects in the right location, not colliding the hand with surroundings and determining the current position, orientation, movement speed and movement direction of the hand. Furthermore, the camera needs to be mounted onto a moving wheelchair and not get in the way of moving around. Finally, since the display is 2D, the camera also cannot easily convey depth information. Attempting to solve all these requirements leads to many difficulties. The need for both an up-close view of the hand as well as all surroundings, needing to be able to see the hand relative to surroundings but not be occluded by the arm, needing a good perspective of the position of arm in the axis parallel to the direction of the arm relative to the chair without having the camera stick out far from the chair. A solution may involve the use of multiple camera's, alternate sensors and/or complex algorithms, but this is beyond the scope of this project.

To stay within a reasonable scope, the system in the current project will have one camera placed on top of the hand with both the forward surroundings and the fingers within view, with a fluid and responsive video feed. This helps with the tasks of aligning the hand with object to grab but does not provide a wider view. Instead, users are expected to glance over the screen to look at the arm. While not a perfect solution, it does provide an opportunity to get feedback into which perspectives people would want the system to have. The technical requirement for the video feed is for it to be responsive and fluid, as this is assumed to be important for intuitive understanding of the motion of the hand as well as preventing moving the hand too far due to latency in the feed.

## 5.6 Mapping buttons to arm movements

The simplest control scheme makes it so that looking at a button immediately causes the arm to move in the given direction, however the background research tells that care has to be taken to ensure a good user experience. Most importantly this option would likely suffer from Midas' Touch, as such a dwell-delay should be included to prevent accidental activation. To mitigate Midas' Touch even further, a separate confirm button could be used requiring two interactions before movement occurs. Design considerations in the placement and size of the buttons should also be taken to limit accidental inputs, and the user should have the ability to hide these buttons to rest their eyes.

Because this form of input requires the user to continuously look at a button, their ability to observe the movement of the arm is diminished. To allow the user to observe the movement and position of the hand while it is moving, the button could stay engaged for a certain time or even indefinitely after it is no longer hovered. As this increases the severity of an accidental input, an increasingly effective Midas' Touch mitigation should be used even if this sacrifices input speed. Furthermore, for inputs where the user cannot stop movement immediately by looking away, a stop button should be implemented for stopping the movement prematurely. This is important for both decreasing the severity of mistakes, as well as allowing for shorter and more precise movements than the disengagement delay would normally allow. Even with a stop button, this form of continuous input is not well-suited to small movements as input delay, user reaction time and dwell-delay on the stop button or disengagement delay on the movement button could prevent movements below a certain length.

Instead of holding a button continuously, a discrete or stepped movement option could be used to allow the user to glance at a button and be ensured that the hand moves the desired distance. This was the approach used by D. Gaethofs [2], where a single click of a button causes the arm to move for one second. A downside of this system became clear during testing, where performing a simple task of navigating towards and picking up an object took over 5 minutes on average. An upside of this method is that the amount of movement for each input is guaranteed, instead of relying on the user to perform an interaction at the right time while the arm is moving. Options could be added for the length of a single movement to allow the user to choose between step precision and input speed. If stepped movement is used as an option alongside continuous movement, large step sizes may not be necessary.

## 5.7 Control axis

The robot arm moves in 3D space, but our input is only in 2D space, as such a translation has to be performed. One approach is to define three axis (X, Y, Z) which can be independently controlled. To define these axes, a reference vector is used. This can be a static point such as the base of the robot arm or a moving point such as the hand of the arm, or a combination of both. The joystick for the Kinova Jaco as analysed in the background research appears to use a combination, where left/right and front/back movements are based on the direction of the arm, while up/down is always the axis of the base.

Most interfaces use their 2D nature to align two of the axes in a natural way, usually a plus shape with one axis vertical and another horizontal. The third axis can be included in a number of different ways. D.

Gaethofs [2] attempts to keep the third axis as equal to the other two as possible, arranging it vertically next to the plus arrangement of the other two axis. S. Sunny et al. [15] applies a separation between XY and Z, where Z is placed in a separate box from the XY and Z grid. More complex interfaces can introduce diagonal buttons, allowing users to combine multiple axis into one movement. With the 2D nature of the interface and the complexity in adding the Z dimensions, only diagonal buttons for combining X and Y movements are trivial to implement, although as stated by D. Gaethofs, consideration should be taken not to crowd the interface with buttons.

An alternative method of controlling the arm is to control its individual joints instead. This has not been pursued in this research as D. Gaethofs [2] states that the client specified a preference for the former type of control, but it could be worth further research. One of its advantages is that it is a more direct form of control, bypassing the inverse kinematics of the robot arm which can sometimes slow down or refuse to move to an impossible position against the user's intuition.

Given the complexity of implementing translation in 3D movement, it was decided not to include rotation for this research and focus on implementing translation. This means that the control axis should clearly be defined by the predetermined orientation of the arm. However, which direction becomes which axis is not an easy decision to make. Given the position of the camera, it makes intuitive sense to align the left/right movement to the X axis and the up/down movement to the Y axis, as this aligns an X Y movement cross with the camera view. However, an argument can be made that having the Y axis be forward/backward instead makes more intuitive sense as this leaves the only axis affected by gravity to the separated Z axis. The current system will have the X Y axis aligned with the camera, and user feedback will be gathered during testing about the alternative mapping.

## 5.8 Hand control

Aside from controlling the arm, the system will also provide the ability to open and close the hands of the arm. The robot arm used has three fingers which can each be set to an arbitrary percentage of curl and automatically stop moving when a grabbed object prevents it from closing further. For this research, the scope of the system is limited to sending commands to fully open or close all fingers of the tool at the same time. This is the same limitation as the prior work by D. Gaethofs [2] where it was proven to allow for picking up a liquid container, satisfying requirement 3. More delicate control can be made available but is not a target and left to future research.

## 5.9 Further features

The interface should also contain a button to lock all inputs and a button to reset the arm. The lock button is intended to satisfy requirement 4, disabling all inputs except for a button to unlock the interface. The reset button moves the arm to a set starting position. The position which the arm moves to when reset should ensure that it is facing in the predetermined orientation and is in a position which affords it good freedom to move without reaching the limits of the kinematics.

## 5.10 Interface design

The design of the interface requirements should be based heavily on that created during ideation, but with all controls for rotation removed. This means putting 8 buttons for X Y movements including diagonals on one screen and then putting buttons for the Z axis in a separate screen. The controls for opening and closing the hand can be placed on the same screen as those for movement in the Z axis to consolidate the number of screens. The controls for locking the interface and resetting the arm should be placed away from the webcam or any place where they can be easily activated, as such they could replace unused buttons in the top bar.

While prior work by D. Gaethofs [2] uses the gaze-tracker software to determine when a button is actuated, this system will only use the gaze-position output and apply a dwell-delay. This is for a number of reasons. Firstly, click actions in the gaze interface are mainly designed for general Windows applications which are not optimised for gaze-input, thus a zoom before clicking is standard which should not be needed for the large buttons of this interface. Furthermore, the software does not know if the user is hovering over a button, only beginning a dwell-delay after the user's gaze is stationary for a while which causes a slower response. Finally applying a dwell-delay inside the system allows for different dwell-delays between buttons as specified in the ideation design.

Two variants of the interface should be made for comparison during testing. One should implement a start-stop system where a direction is selected and confirmed using a start button to begin movement, and another should begin movement immediately when an axis is selected and stop when the button is unhovered for more than a second. This delay in stopping should allow the user to glance at the position of the arm during a movement, and to allow for more precise movements a stop button with low dwell-delay should also be present. These two interfaces should provide a point of comparison between a slower method which prevents accidental inputs, and a faster system which does less to prevent accidental inputs.

## 6 REALISATION

The system was realised by building upon the work by D. Gaethofs [2]. Although their code was used as a basis, eventually most of it was replaced as every major part of the system needed to be reworked to include the features from the specification as well as to improve the flexibility of the code for future expansion.

### 6.1 System diagram

Figure 8a contains an overview of all major hardware components of the system and how they are connected to each other. The robot arm and camera are connected to the Raspberry Pi via USB, and the eye-tracker is connected to the user's computer via USB or built-in to the computer. The user's computer and Raspberry Pi are connected over Ethernet.

Figure 8b shows all hardware and software components of the system, how they communicate with each other and a summary of the functions of different components. A ROS system running the Kinova-ROS stack, a custom node and rosbridge server receive WebSocket packets from a Web UI running JavaScript which is hosted by a Python HTTP server. A DHCP server is responsible for setting up the connection between the Raspberry Pi and the user's computer. Everything is executed via Systemd services.

### 6.2 Raspberry Pi

The device that the software would run on was chosen to be a Raspberry Pi 3. The Raspberry Pi was chosen for its form factor, price, support, ubiquity and ability to turn on immediately when connected to power. The form factor helps attach the system to a wheelchair where space may be limited. The price helps keep the system affordable. The ubiquity ensures that copies of the system can be produced easily by creating an image of the SD card of the original system, and in case of failure the PI can be easily replaced. The PI turn on immediately when connected to power, removing a step from the setup procedure for each session of the system. Finally, the Raspberry Pi 3 was chosen specifically as it is the only model that functions correctly with the software provided by Kinova to interface with the robotic arm. A downside is that the Raspberry Pi 3 has limited processing power and memory, limiting future expansion of the system which may require more resources.

### 6.3 Controlling the arm

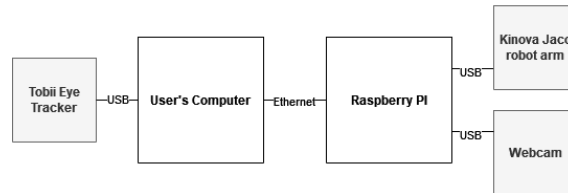
To send commands to the arm, a ROS node was implemented to send messages and execute services in the Kinova-ROS stack. The Kinova-ROS stack provides a ROS interface for Kinova robot arms, allowing a range of commands to be sent to it. This stack is ran as a separate task which also includes the main ROS system. The arm can be connected to the system though USB or ethernet, of which USB was chosen as it was easier to set up. Commands are sent to the Kinova-ROS stack through either calling services or via an action client. Services are used for simple actions which cannot be interrupted, such as homing the arm or enabling/disabling the arm. Calls sent via an action client return a success or failed state and have the ability to be cancelled in the middle of their movement. One type of service call can be cancelled, this being the `AddPoseToCartesianTrajectories` service which allows multiple Cartesian coordinates to be combined into a single movement and can be cancelled by calling `ClearTrajectories`.

The ROS node implements a variety of ROS message receivers which perform a number of tasks to provide a toolkit for implementing features into the front-end. For moving the arm, it provides continuous, relative and absolute movement options in a Cartesian coordinate space. Continuous movement moves the arm in a direction until a stop command is sent, relative movement moves the arm in a direction by a given distance, and absolute movement moves the arm to a set position and rotation relative to the robot arm base. The fingers can be controlled by a percentage value for how far open each finger should be. A stop method is provided which stops all movement of the arm. Finally, it can pass through a command to home the arm.

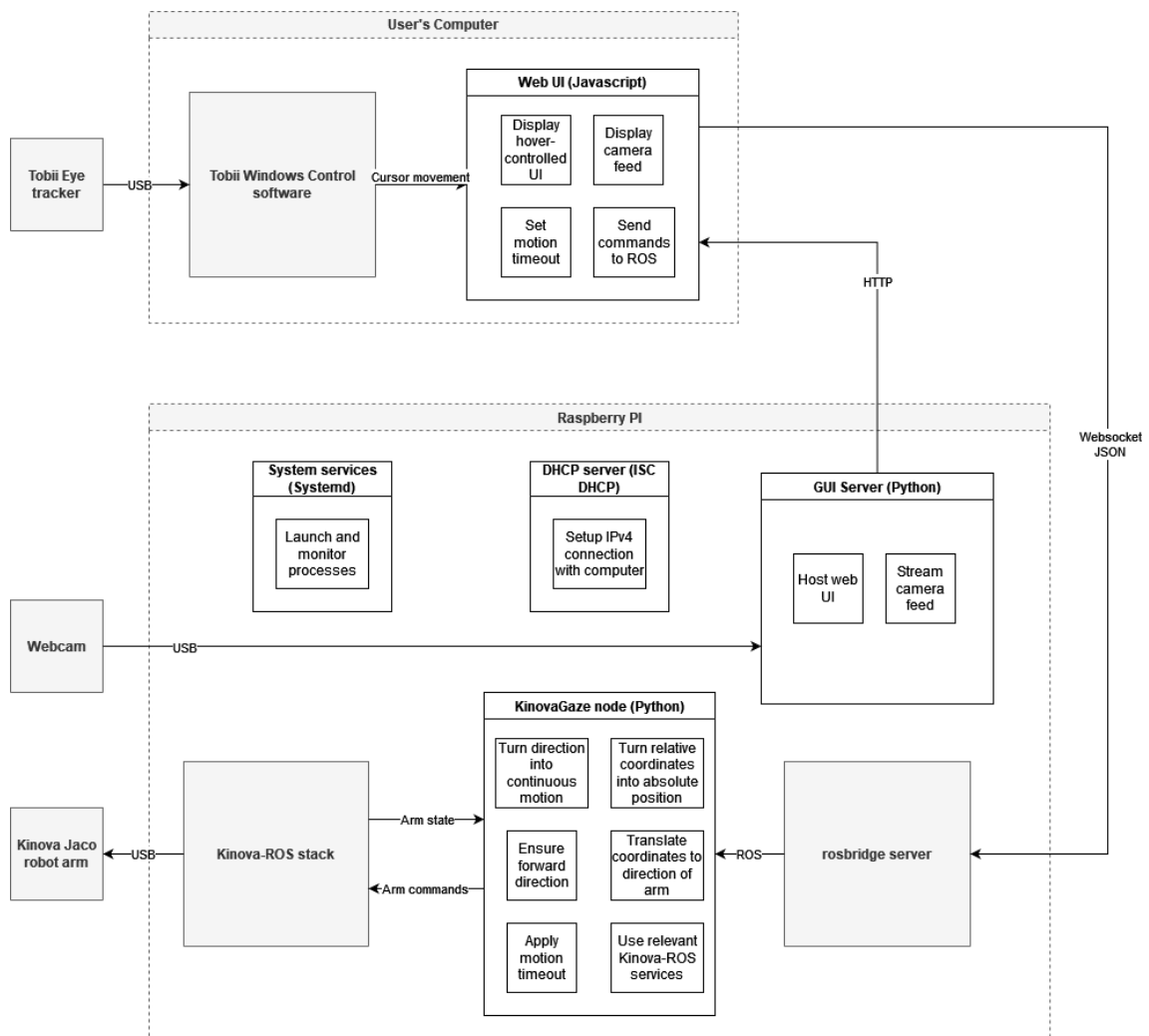
Another feature of the ROS node is the motion locking system, a safety feature which prevents the arm from moving as a result of a system oversight. When the stop movement method is called, it sends a command to the arm which stops the effects of all movement commands from the ROS node until a subsequent command is sent to start accepting movement commands again. The start command is sent when a new movement is requested as part of the request handler. The stop movement method can be called by sending a message to the ROS node and should be mapped to a button in the front-end which is always available when the arm is moving.

The motion locking system is mainly used by the timeout system, which requires the front-end to send frequent commands to keep the arm unlocked. The ROS node contains a value which is always counting down so long as the arm is unlocked and which will stop and lock the arm when it reaches zero. To prevent this from happening, the front-end has to periodically send a message to set the timeout value. The front-ends can take multiple approaches to this. For example, one interface continually sets the value to 1 second, preventing the arm from automatically locking so long as the interface is connected. Another interface only

**Figure 8.** System diagrams



**(a)** Diagram of all hardware components and their data connections.



**(b)** Diagram of all relevant hardware and software components of the system.

sets the value so long as the user is looking at a movement button, allowing it to reach zero and stop the arm after the user stops looking at the button for a set time. The arm reset button sets this value to the expected duration of the motion. The back-end also sets this value to 1 second every time the movement is unlocked to ensure it does not immediately get locked again.

Although the ROS node supports rotation, this is intentionally disabled with the tool orientation goal of every movement forced to a set value. This is done to simplify the use of the arm during testing, limiting the buttons needed to learn during the test and removing complexity from the test objectives. As the arm has a tendency to rotate when it reaches the limits of its movement, this implementation allows the arm to rotate back to the forward orientation when another movement is made.

A bunch of functions are in the code for aiding with processing arm orientation. Some of these are used by the tool orientation enforcing code, a part of it was intended for an unused and unfinished feature to rotate any movement commands to the perspective of the hand instead of the base. This would for example mean that a command to move left would move the arm left relative to the orientation of the hand rather than the base. This was intended to make control with the camera mounted to the hand more intuitive, but ultimately was left unused as changing the orientation of the tool was disabled.

### **6.3.1 Continuous movement in cardinal directions**

Two methods were tested for moving the robotic arm in a cardinal direction and rotation for an undetermined amount of time. The first method used a feature of the Kinova API to directly control the cartesian space velocity of the arm, where each command makes the arm move in a direction and rotation for 1/100th of a second. This method of control give responsive control over the motion of the arm, however it caused unintended movements as shown in figure 4. This is likely the result of inaccuracies in the inverse-kinematic system of the arm which are likely compensated for over longer movements via some sort of feedback loop or movement planning, but the lack of persistence between commands to the arm in this mode does not allow for this.

The second method used was sending a command for the arm to move to a point far away in the chosen direction and rotation, and then cancelling the movement when a stop signal was sent. This system worked for lateral movements, but for rotation the movement it has to stop at less than half a rotation to ensure it goes in the correct direction. To solve this issue, a feature was used to chain multiple points together which the arm would turn into a single movement. This control method has multiple downsides. Firstly, it does not provide any velocity control as the arm always moves to its destination at the fastest speed it is able to. During testing it also did not respond to stop commands reliably, although this may have been an error in the implementation of this project. It also introduced a slight stutter into rotational movements every time it switched to the next point in the movement. Finally, it stops more quickly when the inverse kinematics prevent it from moving in a straight line without colliding with itself. While this behaviour is preferable over the drifting behaviour of the direct velocity control method, it made movements difficult especially where the arm is near its base.

## **6.4 Connecting to the user's computer**

As the system should be used with the existing gaze-controlled computer of the user, a connection between the Raspberry Pi and this computer was needed. For this, Ethernet was chosen as it is the only common wired connection supporting a direct connection between the two devices without extra hardware or software on the user's computer. A fixed IP address and a DHCP server were installed on the Raspberry Pi, allowing it to create a local network when connected directly to a computer. The Raspberry Pi uses the IP address 192.168.137.1 and assigns computers in a range of 192.168.137.100-192.168.137.200. The IP 192.168.137.0/24 was used because it is a local IP which isn't assigned to any organisation and thus unlikely to collide with another IP. The interface is accessed via a web browser by connecting to 192.168.137.1:5000. If the user's computer does not have an ethernet port available, an ethernet to USB (C) adapter may be used.

To host the interface to the computer, a http server was created in Python. This server hosts both the files for the web interface as well as the camera stream. The server framework used was Flask as this was already used by the previous work by . Although Flask warns not to use it in a production environment due to its inability to handle more than one request simultaneously per default, this was not considered an issue as the current system should only have one user at a time. It is also not built with the security of a more proper server, but this was also not considered an issue as the service is not designed to be exposed to the world wide web.

## **6.5 Camera**

The camera used in the system is a simple webcam mounted on the top side of the hand of the arm using double-sided tape. It is mounted behind the fingers, pointing towards the facing direction of the hand and looking slightly down so it can see the fingers even when they're spread apart. The shape of the hand and the

**Figure 9.** Start and end position of the robot arm after attempting to move it in a singular linear axis.



**(a)** Starting position with angle highlighted



**(b)** Ending position with angle highlighted

```
twist_linear_x: 1.0  
twist_linear_y: 0.0  
twist_linear_z: 0.0  
twist_angular_x: 0.0  
twist_angular_y: 0.0  
twist_angular_z: 0.0
```

**(c)** Movement axis used during the movement



simple mounting method did result in the camera being crooked which was not corrected for during testing. The camera wire was left loosely hanging from the hand. This means that certain movements may allow it to get tangled in the arm which may constrain or damage the system, however this was not considered a large issue as the system did not support tool rotation thus limiting the potential for tangling.

The camera feed is streamed using OpenCV to capture images from the camera which are then embedded in a http package as text. It was noticed that the latency of the feed was high and it suffered from frequent stutters. To resolve these issues, the resolution of the feed was lowered from 1920x1080 to 640x360. This was considered to still provide a clear enough image while having low latency and less frequent stutters. The implementation of the camera in code does have room for improvement, as different cameras may require changes to the code before they work with the system which make installation of the system and replacing parts more difficult.

## **6.6 Web interface**

The web interface represents a major part of the system, implementing the design as imagined during ideation and evolved during the specification.

### **6.6.1 Technical implementation**

The web interface was built using p5.js by creating a custom gaze-button interface system. p5.js is a tool intended for learning code and creating art using JavaScript, providing simple graphical commands to draw onto a canvas. It was chosen due to its similarity to Processing which is a similar tool but for offline Java applications. Using p5.js the interface provides custom hover-activated buttons with a more advanced logic system to allow for a dynamic interface. This is in contrast to the prior work by D. Gaethofs [2], which uses a mostly static interface with standard HTML buttons and a minimal amount of JavaScript to relay button presses to the web server. The new interface consists of multiple parts, including a framework for gaze-controlled interfaces, a class for communicating with the ROS node, a class for recording system interactions during testing, and the logic combining these two systems into a functioning interface.

The first part of the system is a framework for gaze-controlled interfaces featuring an abstract class for hover-activated buttons and helper classes to create a functioning interface. The main part of the framework is the button class, which provides a basic functionality of a hover-controlled button and is intended to be extended into the specific functionality that the designer needs. The basic button is a simple rectangle overlaid with either text or a (vector or raster) image as a label, which can be made any size and placed anywhere within the canvas. A dwell-delay can be defined, which determines how long the user has to look at a button before it activates. The button uses colour to communicate its state, gradually changing colour when hovered until it snaps to a different colour to indicate its activation. All colours can be customized per button. The button can also be deactivated which will cause its brightness, saturation and opacity to decrease, or the button can be hidden completely.

Helping the button class is a cursor class, which stores the last position of the mouse cursor and activates any hover-controlled buttons it is located above. The cursor objects normally hides the cursor, but it has an optional feature to replace the cursor with a simple target indicator. There is also a grid class, which helps with creating a button layout for an interface. Implementing the button class is done by overwriting the class with a custom implementation. Per default the button does nothing when activated and cannot be deactivated, this behaviour is intended to be defined by overwriting the onHover, onUnhover, onActivate and onReset methods. This way the code can define custom button times such as latching buttons, button groups where only one can be active, and buttons which gradually disable over time. Implementations of the button class also commonly use callbacks, where each button can define a method to call when a certain event occurs. Together, this framework provides a flexible and extendable toolkit for implementing gaze-controlled interface and is intended to be expanded upon with more types of input.

Alongside the gaze control interface system are helper classes for communicating with the ROS node and recording user actions during testing. The ROS node helper class sets up the communication with the rosbridge server and contains methods for the various functions in the custom ROS node. It creates publishers for all the actions implemented in the backend, instantiates a connection with rosbridge and sets up callbacks for changes in the connection state. The recorder class contains a video recorder for the interface canvas as well as a logging system which outputs a csv file. A recording can be started and ended with methods, and when the recording is stopped it downloads the video recording and log file to the user's computer.

Finally is the logic which implements these two systems into a functioning interface. This implements the gaze interface system, the ROS communication system and the recorder if enabled. All buttons are created initially with their label assets downloaded from the server, and are then enabled, disabled and hidden as the user navigates the interface. A loading screen is implemented, where the buttons are replaced by a message until a connection with the rosbridge server is established. The interfaces are also responsible for managing the timeout timer in the ROS node, which different variants of the interface handle differently.

The start-stop interface accomplishes by regularly setting the timeout to 1 second at all times, while the hold interface repeatedly sets it to 1 second only when a movement button is active and dwelled, allowing it to reach 0 when the button is released. If the reset button is pressed, the timeout is set to a value determined to be the maximum time needed to reach the starting position.

### 6.6.2 Interface description

Figure 10 shows both variations of the interface. More images of the interface are included in appendix B. The interface consists of three main interactable elements, this being the top bar, the axis buttons and the start/stop buttons. The top bar contains a button to lock the interface, a button to reset the arm to its starting position and buttons to switch between groups of axis buttons. In the centre of the screen behind all buttons is the video feed from the camera, stretching the full width of the webpage. Above the video feed is the top bar with a green lock button, yellow axis group buttons and a blue reset button. Axis buttons are yellow and surround the camera view with gaps between the buttons. Below the video feed is a bottom bar, the contents of which depend on the interface variation. For the start-stop variation the bottom bar is filled by a green start button, whereas in the hold interface it is filled with a red stop button. In the start-stop interface, the stop button replaces all buttons in the top bar so long as an axis is selected, and the start button is only enabled if the stop button is visible.

The current interface has two groups, strafe (up, down, left, right, diagonal) and other (forward, backward, grab, release). When selecting an axis group, the corresponding buttons appear around the video feed and the axis group button is highlighted green. If the stop button is hit, the current axis is deselected but the axis group remains selected. If the lock button or the reset button are hit, the axis group is deselected and all axis buttons are hidden. This also deselects any selected axis. If the lock-button is activated, all other buttons are disabled, its colour changes to red and its icon changes to a closed lock. When the lock-button is deactivated, all buttons are enabled again.

The user is expected to follow a certain order of operations when operating the hand. If this is the first use of the system after a power cycle, the hand must first be reset so it is calibrated. Then the user first selects an axis group. Within the axis group, the user selects a direction or hand command to perform. The activation and deactivation method depend on the variation of the system, but a stop button can always be used.

If the user is in the start-stop interface, selecting an axis causes the start button to enable and the stop button to appear. The user can still change their axis selection. Otherwise, they can now press the start button to begin the movement. After an axis is selected, regardless of whether it is started or not, the stop button can be used to stop movement and deselect the axis. The stop button remains until they look away from it, at which point it is hidden and the top bar returns.

If the user is in the hold interface, selecting an axis causes the movement to begin immediately. The movement continues until the user looks away from the button for longer than a second, or until the stop button is activated. The stop button intentionally has a very low dwell delay to allow for stopping the arm more precisely.

## 6.7 Running the system

To enable all the functions of the system, five Systemd services are set to start at boot and restart if they exit. Each service manages either the Kinova-ROS stack, the custom ROS node, the rosbridge server, the GUI server or the DHCP server. Starting the system simply involves supplying power to the Raspberry Pi, where the system will be fully started after a few minutes assuming all components are connected. There is currently no mechanism to safely shut-down the system and prevent data corruption aside from connecting via an SSH session and manually running the shutdown command. There also is no way for the user to manually restart (parts of) the system in case they stop working correctly.

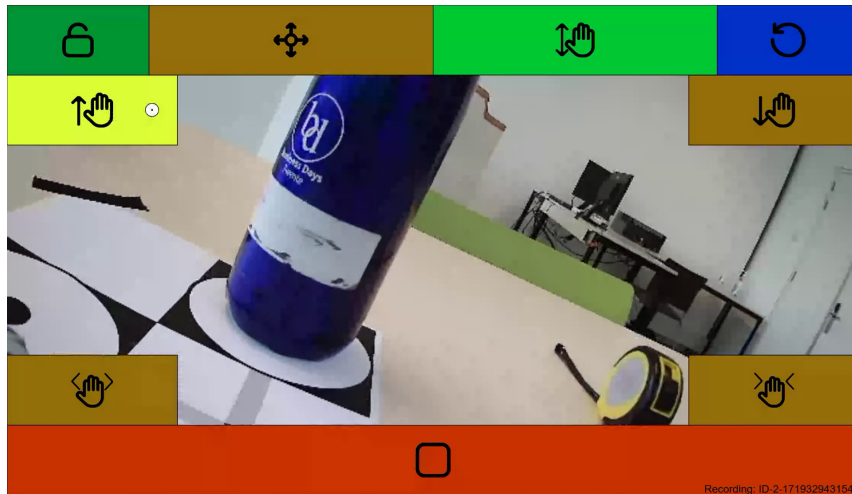
## 6.8 Allowing future work on the system

To allow for future work on the system, the code was documented and instructions for replicating the system were made. A readme file was created containing an overview of system functionality and detailed instructions on how to replicate the system. The function of the separate parts is explained, and code has been documented with class and method-level documentation. All files and documentation for this project can be found on GitHub under the Ability Tech organisation.<sup>2</sup>

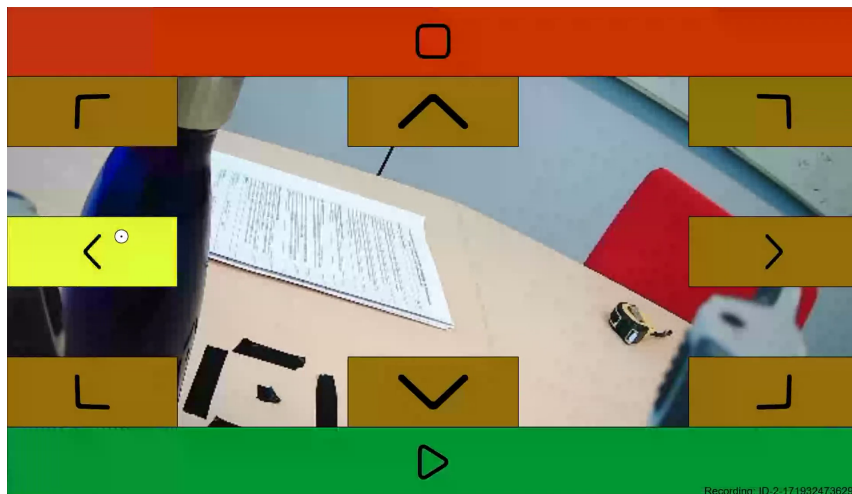
---

<sup>2</sup><https://github.com/abilitytechlab/KinovaGaze>

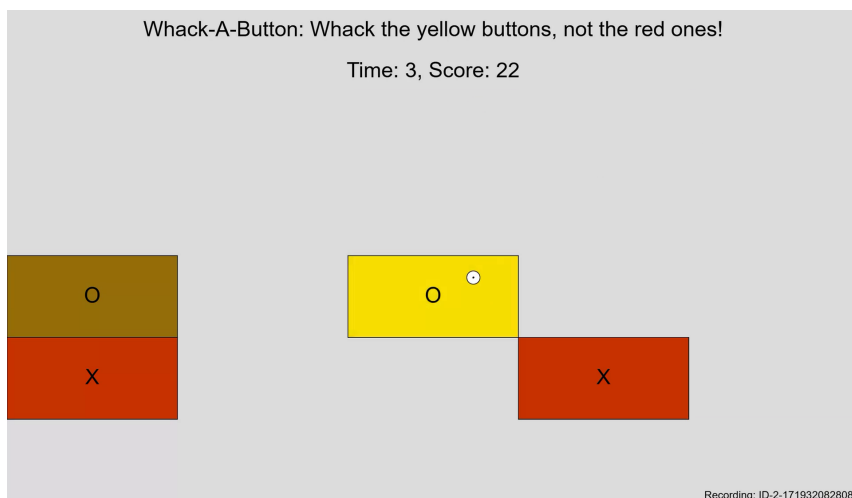
**Figure 10.** Screenshots of the interfaces.



**(a)** A screenshot of the hold interface being used to move the arm forward to grab a bottle.



**(b)** A screenshot of the start-stop interface being used to move the arm sideways to grab a bottle.



**(c)** Whack-A-Button game with 3 seconds left and a score of 22. Two good buttons and two bad buttons are on-screen with a good button being hovered.

## 7 EVALUATION

After realisation, the system is evaluated to test its effectiveness. Testing is done with both the main client and other participants to gather a larger sample size for general performance as well as specific feedback from the client. The testing method between general participants and the client are different, with the first subsections of this section describing the method used with general participants.

### 7.1 General user tests

#### 7.1.1 Testing method

The goal of this testing is to gather information about the effectiveness of different parts of the system as well as the system as a whole. This is measured in the amount of time it takes to accomplish a simple task, their opinions after experiencing the system and observations made by the researcher during the tests and by analysing the test recordings afterwards. Test recordings include both a log of all commands given and a video recording of the interface including camera feed, and the opinions of the system are gathered via questionnaires.

Initial participants are gathered from the researcher's network or from anyone else interested, mostly those at the university where this study is performed. The goal is for this test to be completed by at least 4 participants, to allow for two variants of the test to be performed each with two points of data. Although involving multiple participants should increase the accuracy of the results if compared to the general public, the selection method risks a bias in the results stemming from homogeneity of respondents. Despite the effort of the university, respondents are likely to be WEIRD (Western, Educated, Industrial, Rich, Democratic) as well as have a background in technology. However, it can be argued that most of these labels also apply to the main client and are thus a lesser issue to the main goal of this research, although they should not be forgotten to the detriment of others who may be able to benefit from this research.

To provide a point of comparison, two versions of the interface are tested. The first is the continuous movement interface, where the user selects a direction, then selects start to begin movement, and finally presses stop to end movement. The second is a discrete movement interface based on the system used by Damian, where the user selects a direction which causes the arm to immediately move in that direction for one second. Neither interface has the ability to control rotation of the robotic arm, only providing movement in 3 Cartesian axes as well as being able to close or open the arm. The arm begins in a state where it already points towards the bottle and away from the user. The user will be asked to complete the same task 4 times, alternating between the two control styles. Furthermore, the testing alters between starting with the continuous or the discrete interface between testers.

The objective results of this test will not be able to be compared to prior work, as the testing setup is sufficiently different. The experiment done in the prior work by Damian [2] uses a setup where the arm is required to rotate to accomplish the goal. As the system used in this test does not allow for rotation, the testing procedure cannot be replicated and as such the results cannot be compared. They also only performed the experiment once themselves with no external users, seriously limiting the usefulness of the results. Another similar system was tested by Sunny et al. [15], however their tests also involved the participants controlling the movement of a wheelchair. Furthermore, they placed the object in various places without documenting the details of the starting and ending positions for the task, thus the tests cannot be sufficiently replicated to compare results. Subjective results may be comparable; however, this is a very imprecise result.

#### 7.1.2 Testing setup and procedure

The testing setup involves a Kinova robotic arm mounted to a table, and behind the arm is a laptop with a Tobii Eye Tracker as the user interface. The interface on this laptop has been programmed to start recording once the first input is given, and to stop recording when the robot arm is reset. A second laptop is connected for the researcher to use, both to make notes and to intervene in case of problems with the system. Two spots are marked on a printed paper taped to the same table as the robot arm, each spot a square with a diameter of 10 cm. One spot, the ending point, has its centre about 45cm in front of the starting position of the robotic arm's hand. A second spot, the starting point, is located 10 cm closer to the arm and 10 cm to the left of the first spot. The starting spot will have a bottle placed on top of it in the beginning, where the ending spot is the goal where the bottle needs to be placed onto.

The testing procedure involves four main phases. The procedure is preceded by the participant reading and signing the consent form and information sheet as well as asking any questions they might have. Meanwhile the researcher loads the correct interface, tests the system to ensure it is working, placed the bottle on the starting point and assigns a unique ID to the tested which is connected to all questionnaires and recordings to ensure they can be connected to each other. Then they move to the first phase, where they will be asked to complete a questionnaire asking some questions about topics which may affect the results of the test. This includes the user's prior knowledge and experience with gaze-controlled systems and robotic arms, as well as any eye or motor disabilities which may affect their ability to control the system.



**Figure 11.** An image of the testing setup prepared to begin a test session.

The second phase is designed to get the user started with the eye-tracker. First, the user's gaze is calibrated to the screen using the utility included in the Tobii eye-tracking software. Then, the user is asked to play a game intended to familiarise them with moving a cursor using their eyes. This game is inspired by the classic game Whac-A-Mole. In this game titled Whack-A-Button, the player has to trigger randomly appearing buttons before they disappear. These buttons function similarly to the buttons in the robot arm interface, gradually changing color when hovered and triggering after a dwell delay. The chosen dwell delay was 0.5 seconds, and the buttons disappear upon being triggered. There are also bad buttons which the player should avoid, with the final score of the game being the number of good buttons hit with each bad button hit subtracting 3 points. The game gradually increased in difficulty, with buttons appearing and disappearing faster and more bad buttons appearing. The game ends after 30 seconds, with the user being informed of their final score as well as average time before triggering a button. A recording of the game as well as a file logging all actions and statistics were saved for later analysis.

The third phase contains the main assignment. Here, users will be tasked to use first the start-stop interface and then the hold interface to pick up, move and place down a bottle from a set starting position to a set goal position and then reset the arm. The user first receives an explanation of how to use the interface and the task they should attempt to accomplish. The arm and bottle are then reset, and the participant is allowed to use the interface to accomplish the task. When the task is accomplished, the interface is switched to the hold interface and the task is repeated. If the tester attains a significant delay in achieving the task, such as toppling the bottle, the task is reset and the tester tries again.

In the final phase the participant is asked to complete a questionnaire regarding their experience with the system. The participant may also communicate with the researcher to more directly give their opinion or feedback, which the researcher then writes down.

### **7.1.3 Results**

In the general testing, four testing sessions were performed each with a different participant. The results of one participant were mostly discarded as they were unable to properly use the gaze-tracker, likely due to their strong corrective eyewear. One other participant also wore eyewear, but the lower strength of the lenses allowed the gaze-tracker to perform to an acceptable quality. Still, this participant removed their eyewear several times during the session. All other participants did not report to use or need corrective eyewear. Testers also responded that they had little to no experience controlling robotic arm or using gaze-controlled interfaces.

**Completion time** The time of completion for each task was measured, starting at the first movement command sent to the arm and ending at the last hand open command sent. Figure 12 contains the best time per user for both interfaces. The hold interface has on average a lower task accomplishment time, although this is not universal across all participants. It must be noted that there is likely a bias in this data, as all participants used the start-stop interface first and were thus more familiar with controlling the robot when the hold interface was tested.

**Post questionnaire** The post questionnaire is intended to gain insight into people's opinion of the system and preference regarding the two interfaces. The questions were on a scale from 1 to 5, with 1 usually meaning bad and 5 meaning good. Some questions asked for preferences where choosing 3 meant that the current state was preferred and 1 or 5 meant the tester wished for a change in one direction or the opposite direction. Some questions regarded the general system or a particular interface, whereas other questions compared the two interfaces. The questions comparing the two interfaces were usually split into two separate scales of 1-5, so that the results could be compared between the two interfaces or combined to show the satisfaction with the general system. The questionnaire concluded with an open text field to elaborate on answers or to add recommendations or further thoughts which are discussed after the observations. The full questionnaire can be seen in figure 25 and the results can be seen in figure 23.

Figure 13 contains the results of the post questionnaire. Some questions were asked separately for both interfaces with the results averaged out while others were only asked once for the whole system. The results are generally positive, with only one result below the centre line. The question "How would you rate your experiences with the system" stands out as being answered with the highest score possible even though testers did not rate every part of the system as perfect. While this result does likely show that most parts of the system were not considered bad to use, it does not show that the system is absolutely good as testers were not provided a point of comparison with other interfaces or other modes of interacting with a robot arm. They also only used the system for a limited amount of time in a controlled environment which does not test the long-term satisfaction with the system. Furthermore, testers consisted of people within the social circle of the researcher, thus the results may be positively skewed as testers may be hesitant to critique the researcher's work.

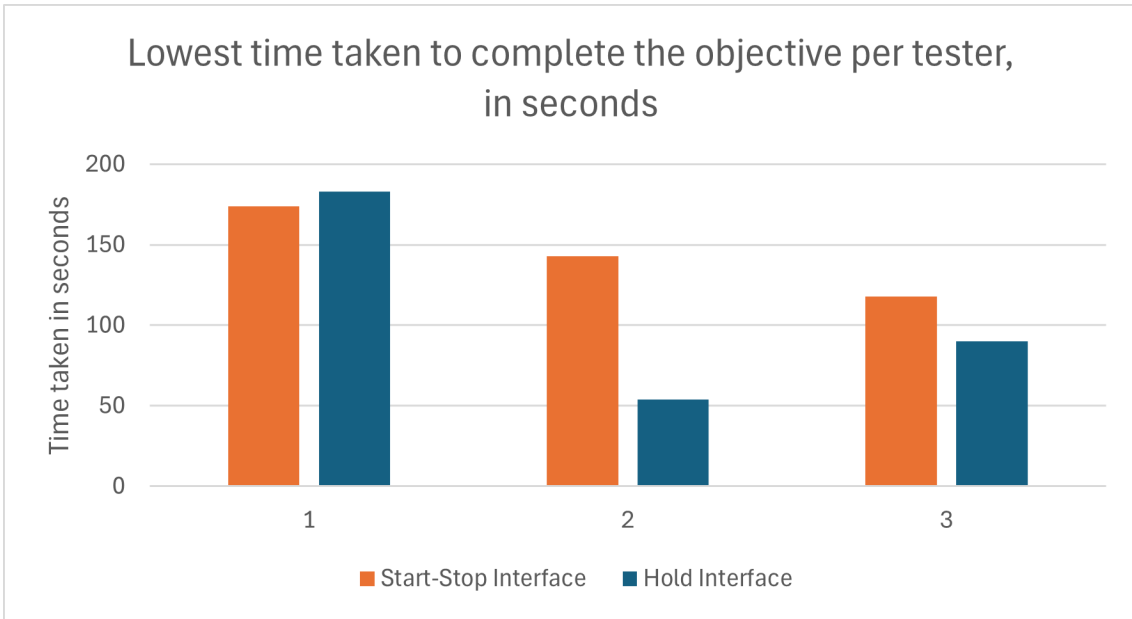
The other question to get the highest score possible was whether users considered the camera responsive, with the result indicating that users had no issues with the responsiveness of the camera. This shows that the system hardware is capable of providing a camera feed which is of high enough quality to be used in this context. However, this result is incomplete as there was no question regarding the resolution or clarity of the camera feed and only one camera position was tested. Other questions regarding the camera like the one about the usefulness of the position of the camera show that the implementation still has room for improvement.

People responded that they felt well in control of the arm but that they struggled to stop in the right place, which may be because they did not feel well aware of the position of the robot arm while moving it. This is likely because the position of the camera has room for improvement, as well as the hold interface not letting users look away from the button while moving. Small movements were observed to be difficult, which do not leave any room for looking at the arm as the user has to quickly switch from starting the movement to looking at the stop button. This struggle to stop in the right place does not seem to be caused by the system not responding well to eye movements as the results for this were quite positive. Learning the controls was rated as moderately difficult, thus showing room for improvement.

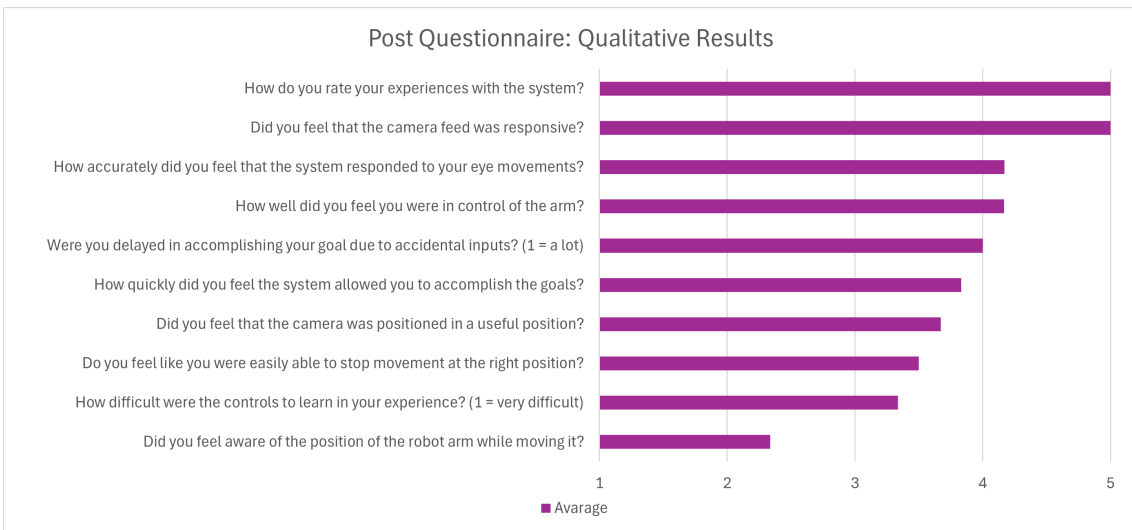
Figure 14 shows a comparison between questionnaire results for the start-stop interface and the hold interface. These results show a marginal difference between results, although neither interface scores significantly better in any one category. These results are not concrete due to the marginal difference and low sample size.

The results indicate that the start-stop interface gave a better feeling of control and awareness over the arm. It scores higher in the ability to stop the arm at the right position and having less trouble with accidental inputs. The most significant difference between the two interfaces is the feeling of awareness over the position of the robot arm while moving it, which is in favour of the start-stop interface. This is likely because the interface allows the user to look away from the interface and directly at the arm while it is moving without the arm stopping. While the hold interface has a button retention delay to allow people to look away, during testing it became apparent that this was not enough for testers to be comfortable in looking away. This can also be seen in figure 15 which shows that testers preferred looking at the object directly in the start-stop interface, while in the hold interface they looked more through the camera likely due to it being closer to the button and thus easier to glance at.

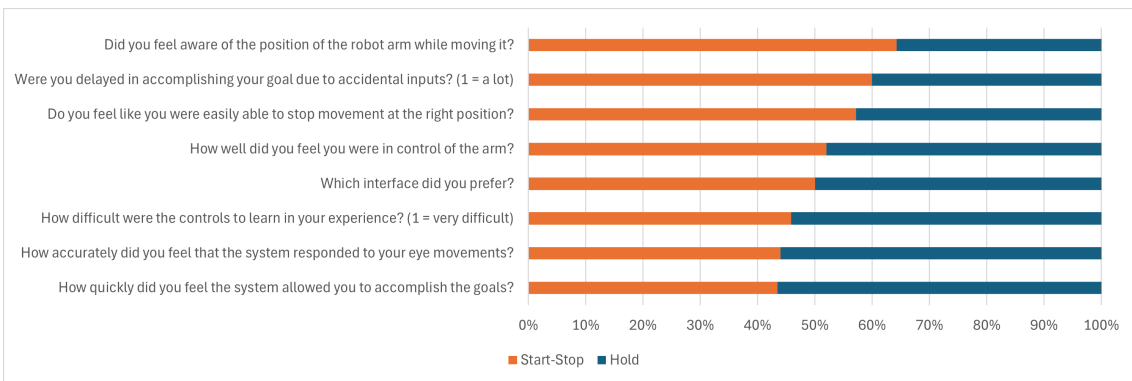
In contrast, the results indicate that the hold interface was considered slightly quicker which aligns with the completion time results. This shows that the accidental inputs were not so bad as to negate the speed advantage of the arm immediately moving after hovering a button. It also shows that the hold interface



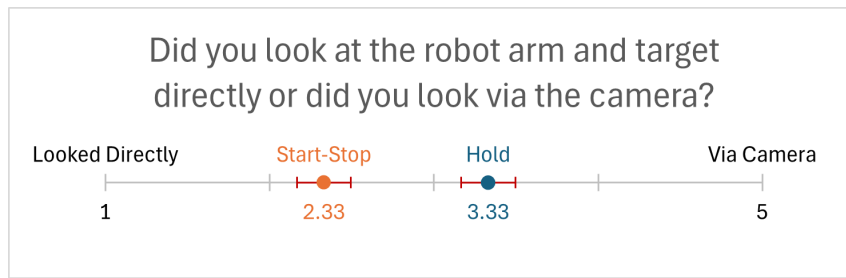
**Figure 12.** Task time in seconds per client



**Figure 13.** Post questionnaire system scores in different categories, longer means better.



**Figure 14.** Post questionnaire qualitative comparison between start-stop and hold interface, normalized to sum of 100%, longer means better.



**Figure 15.** Post Questionnaire question result: looking through camera or at object directly. Orange and blue dots indicate average results, red lines indicate sample variance.

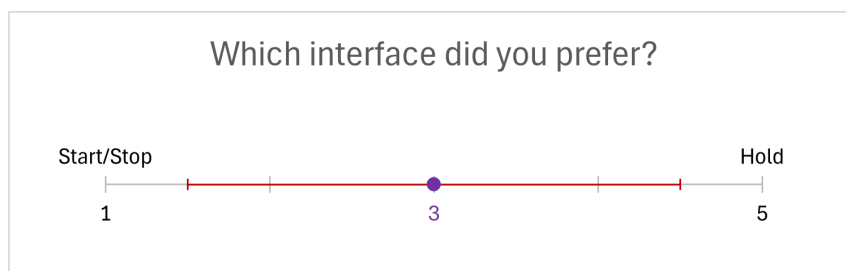
appeared to respond better to eye movements, which is likely caused by an issue observed during testing where testers struggled to hit the start button in the start-stop interface. The hold interface was also considered slightly easier to learn, though this result could be because testers used the start-stop interface first and were thus already accustomed to the layout of the interface.

The last question of figure 14 is expanded with the sample variance in 16. It shows that while the average is perfectly equal between the two interfaces, there is a large variance in preference between testers. Given the results from the other questions, this indicates that testers likely had different preferences or experiences between the feeling of control over the arm and preference for a faster interface. It indicates that a final interface may benefit from providing options to use either variation, although further research should be done to discover whether different combinations of interface behaviour may result in a singular interface which is significantly preferred.

A similar result can be found in figure 17 which shows the responses to the question of whether users preferred to use the stop button in the hold interface or just to let the button disengage automatically after 1 second. This result shows that there is a large variance between testers showing that including both may be the best option. However, these results may differ after changing options such as the location of the stop button or the release delay of the directional buttons. Figure 18 shows that the robot speed was fine according to testers despite the inability for it to be changed from its factory speed, and figure 19 shows that the buttons in the interface were also of a good size.

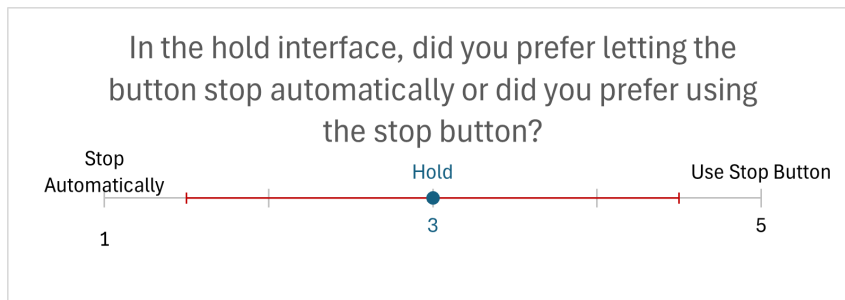
**Observations** During testing, the researcher made multiple observations. One of these observations was that almost all testers confused the up/down and forward/backward buttons with the other axis. This indicates that it may be more intuitive for the movement plane to be placed parallel to the floor rather than the tool and webcam. The diagonal axis buttons were also barely used, with only the third tester discovering that one of the diagonal buttons had a mistake in its implementation causing the arm to move in an incorrect direction. As such it may be better to remove them and/or replace them with other functions to reduce the number of interface pages. Another issue many testers encountered was the implementation of the lock button. When the interface is locked, the lock button is replaced with an unlock button which is immediately dwelled by the lingering gaze of the user and replaced with the lock button once activated, forming a cycle which is only interrupted once the user looks away. The dwell-delay of unlocking the interface was longer than that of locking the interface, as such testers struggled with unlocking the interface without accidentally locking it again. A fix would be to move the unlock button away from the lock button.

Users were also likely to target buttons in their central icon even when this meant a larger travel distance for their eyes, which also caused accidental inputs with nearby buttons. The start and stop buttons are wide, stretching the entire width of the interface. However, most of this area went unused as the centre was usually

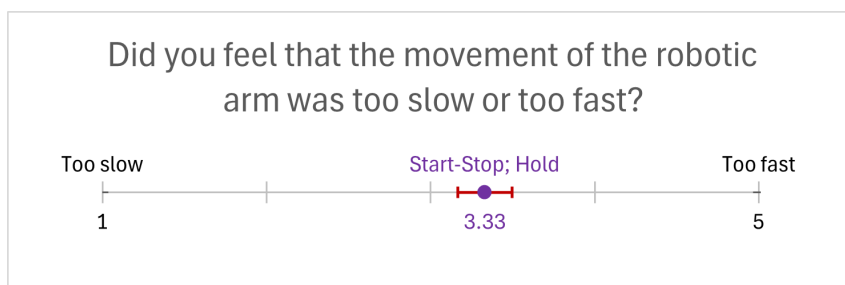


**Figure 16.** Post Questionnaire question result: interface preference. Purple dot indicates average result, red lines indicate sample variance.

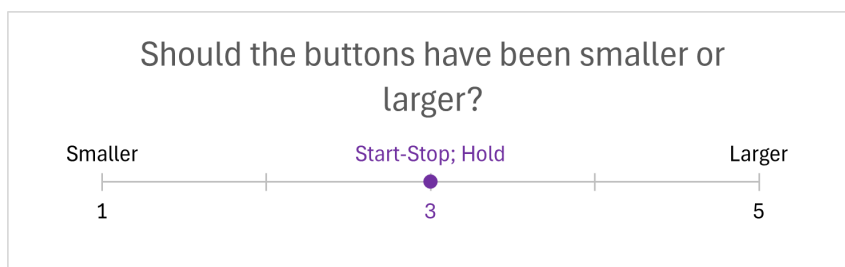




**Figure 17.** Post Questionnaire question result: use of stop button in hold interface. Blue dot indicates average result, red lines indicate sample variance.



**Figure 18.** Post Questionnaire question result: movement speed of robot arm. Purple dot indicates average result, red lines indicate sample variance. Results were equal between hold and start-stop interface.



**Figure 19.** Post Questionnaire question result: size of buttons. Purple dot indicates average result, sample variance was zero. Results were equal between hold and start-stop interface.

the target point for users, as such these buttons could be shrunk to create room for other interface elements. It was observed that testers often accidentally selected the down button instead of the start button, with the down button being immediately above the centre of the start button. This may be caused by a calibration error as some testers also struggled to reach the bottom buttons in the Whack-A-Button game. Another potential cause of this is the instinct to look at the camera feed while moving the interface, which is also located above the start button. To limit this issue, the down button could be moved away from the centre of the screen, perhaps rearranging all axis buttons away from the centre column.

**Feedback** Feedback from the testers was taken both via a textbox in the post questionnaire as well as by the researcher discussing with the testers directly. The researcher asked for general feedback, and also asked some preference questions directly. The researcher asked about the camera position and whether another camera would have helped, they asked about swapping the forward/backward and up/down axes, about the release-delay of axis buttons in the hold interface, and about whether they would have liked a stepped movement option.

The testers generally agreed that a second camera feed was needed, as the current one did not provide enough information. No tester suggested moving the current camera outside of correction its crooked angle, as such the current angle was considered useful to the testers. The main suggestion for the physical position for the second camera was to mount it above the fingers of the arm pointing downwards to help with aligning objects with targets. Another suggestion was a view to help with seeing the height of the hand relative to the ground. The location of the second camera feed on the display was more difficult for testers to imagine, one suggestion was adding a secondary display alongside the main one. Alternatively, the existing camera feed could also be toggled between multiple cameras with a button.

Opinions on switching the forward/backward and up/down axes were mixed. There was some feedback given outside of these testing sessions to include an arrow indicating the direction the arm is about to move in, however it is unclear how this could be implemented on the 2D display. This topic should likely be left for further research which also implements arm rotation. There was some feedback that the grab/release buttons should have clearer icons with the fingers visually spread or closed instead of this being indicated by arrows.

All testers agreed that the release delay in the hold interface was not nice. They did not feel that it allowed them to look away, while it did hinder their ability to stop the arm in the right place or make small movements. One tester stated that the latency in the starting of an arm movement was too much in the system and also recommended adding an acceleration curve to the robot to make smaller movements easier. Adding a stepped movement option was welcomed by all testers, although they stated that completely replacing the continuous movement would have made the interface worse.

Some final suggestions included changing the stop button to always be adjacent to the axis button in the hold interface, to make it easier to stop in a specific position and to make small movements. One also stated that they used the sound of the bottle to indicate when the bottle was being pushed by the arm and thus surrounded by the hand, leading to the idea of implementing audio feedback in the system. One tester wished for the dwell-delay in buttons to be lower, although others sometimes accidentally activated incorrect buttons. As such it may be beneficial to allow each user to set their own dwell-delays.

## **7.2 Testing with the client**

After the tests with general participants, a testing session with the client was performed where the client would interact with the system and perform a task while the researcher observed. Afterwards, questions were asked to the client and a discussion about the system was had with the dad. The goal of this session was to evaluate what the user could accomplish with the current system and where future work should be performed to improve the system. The session took approximately two hours.

### **7.2.1 Testing**

While this testing session was initially intended to be similar to the ones performed with general testers, it was quickly apparent that this would not give useful results. Instead, testing was performed in a loose sense, where the client used whichever interface they wished to use and was free to experiment with and use the arm as they wanted, only encouraged to pick up a bottle.

The reason for not performing the regular test was the client's performance when using the system made these tests too complex for them to perform. The client struggled to understand the interface, repeatedly attempting to use buttons which were not available in the current system state (such as pressing the start button without selecting a direction) or seemingly intentionally moved the arm in a direction away from the goal. The client's inability to talk made it difficult to understand the cause of these issues, making it difficult to correct potential issues during the test.

There were also other issues, such as the lack of an option to directly move the mouse based on gaze of their new gaze-controlled computer, necessitating the switch to an older computer. Their gaze tracking did



**Figure 20.** The client using the system via their gaze-controlled computer, attempting to pick up a bottle.

not appear to be as stable as other's, often freezing on a particular spot. As the client moved the arm away from the bottle in directions not expected by the tester, it often got caught in a state where it could no longer move in the direction that the client input into the interface. Because of the issues experienced during the testing session, no task accomplishment time was captured.

Throughout the evaluation attempts were made to help the client use the system. The research tried explaining the interface again and the task was simplified to only picking up a bottle right in front of the arm. The researcher switched from the start-stop interface to the hold interface, which the client seemed to work with better. The researcher thought that they may accidentally be triggering buttons when looking besides the display at the arm, so the interface was resized to move buttons away from the edge of the display. The client and their wheelchair were moved to provide a more natural perspective of the arm, so that selecting forward on the arm moved the arm directly away from the client's point of view. The researcher attempted to provide step-by-step instructions to pick up the bottle. Eventually the client was able to pick up the bottle with a lot of help by his dad, whose instructions the client seemed to better understand.

### **7.2.2 Evaluation**

After testing, the client was asked questions about the system. This was done via their dad, who asked a question with two options and held up fists representing each option, and the client then looked at a fist to select an option or in-between the fists to indicate their answer was somewhere in-between. The client was very tired from using the interface which limited the number of questions which could be asked and made interpreting the results more difficult. The questions were also loosely asked and changed, therefore the answer may be a bit different from the question as written. The following is an interpretation of the questions and answers.

**How difficult was the arm to control?** Intense.

**Were the gaze-controlled buttons nice to use?** Not having to click to interact with the buttons is nice.

**Did you prefer using the webcam or looking at the arm directly?** Both webcam and looking directly.

**Was the speed of the arm too slow or too fast?** The speed was good.

**Was the system easy or difficult to learn?** In-between.

**Did you have issues because of accidentally pressing buttons?** Accidental inputs were not an issue.

**Did you understand where the arm needed to go?** It was difficult to know where the arm should go.

**Did you enjoy using the system?** It was nice, but too intense.

**Did you prefer having a separate start button, or prefer the buttons activating movement directly?**  
Unclear.

A potential reason for the performance of the client during testing may be their inexperience controlling any object in 3D space. Their disability prevents them from moving their own appendages, and to our knowledge the client has very little experience interacting with 3D space via their computer.

A discussion was had with the dad about the progress on the system and where future work could continue. The system in its current state was too intense to be put into practice. He expressed a believe that the system would be better with more automation, for example if it could detect and automatically grab objects. Perhaps a marker-based system could be used, where an object is attached to commonly grabbed objects so the arm can locate it. To accomplish the goal of drinking, it would also need to detect the location of the client's mouth and move to the right location.

## 8 DISCUSSION & FUTURE WORK

After the evaluation, a discussion is held where the results are analysed and suggestions for future work are made.

### 8.1 Analysing requirements

In the specification phase a number of quantitative requirements for the system were set which have been achieved to different degrees. Requirements 1, 2 and 3 have all been reached as the client was able to use his existing gaze-controlled computer to use a robot arm to pick up and move a bottle with water. Requirement 4 has been achieved by the implementation of a lock feature. Requirement 5 has not been reached, as the system has not been installed onto a powered wheelchair and certain questions such as how the system will be powered have not been answered. Requirement 6 has only partially been achieved, as the position of the webcam view is obscured when a bottle is grabbed. As for the last requirement, requirement 7, this has only partially been reached as important functions such as homing the arm and shutting down or restarting the system have not been implemented with gaze-accessible buttons.

A number of qualitative requirements were also defined. Requirement 8 specifies that the system should be simple to setup for customers and has not been tested, however the researcher currently has no concerns that a system with correctly installed software will pose a challenge for customers to install. Requirement 9 has not been achieved completely. Testing participants did not rate the ease of learning the system particularly high, and the client struggled to use the interface in the beginning. However there has also not been any comparison with other systems for neither testing participants or the client, as such it is unclear as to how much better these results could have been. Requirement 10 has been progressed on compared to prior work, as testing participants rated the speed of the system quite high. Requirement 11 has been achieved also as during testing few accidental inputs were observed.

Requirement 12 has not been achieved. The results depend greatly between the client and other testers. While other testers did not appear to feel much fatigue as they rated their experience as good, the main client clearly struggled with fatigue as a result of the system to the point that failing this requirement might mean the entire system needs to be changed majorly. Requirement 13 has not been tested, although testers generally appeared to have good control over the arm. Requirement 14 is WIP. Requirement 15 has been achieved, as a system image has been produced and instructions for installing the system are available alongside the system files. Finally, requirement 16 is partially achieved, as the system did not have any major technical faults during testing, but it sometimes did not turn on correctly requiring the researcher to manually restart parts of the system before it could be used.

### 8.2 Challenges

The main challenge of this project was the technical work. It was overestimated how much of the technical work was completed by prior work, and it was underestimated how many challenges would be encountered in creating the system. This limited the amount of testing which could be done, thus limiting the chance of iterating and improving the system and the chances for the client to use the system. A compromise resulting from this was to exclude tool rotation from the system.

### 8.3 Comparison to prior work

This work builds upon prior work by D. Gaethofs [2]. In the prior work a proof-of-concept interface is built.

This work progresses on the prior work in a number of ways. The literature is expanded by looking further into the design of gaze-controlled interfaces, looking at multiple existing implementations which may not be directly related to moving a robot arm. For moving the robot arm, this research also analyses the existing joystick control for Kinova Jaco robot arms to inform its own control layout. It also performs a deeper analysis of the goals and wishes of the client to transform the proof-of-concept into an implementation of the system which aligns well with the wishes of Andrei and Ability Tech.

The design and specification have been extended with requirements about both the performance of the system as well as its practical implementation with a focus on the ideals set by Ability Tech. Further specifications have been set about the design of the user interface and robot arm control, the camera placement and implementation and the integration of functions into a separate device which communicates with the user's computer. The design has become more intentional, moving away from the limited control provided by the built-in click function of the gaze-tracker software and the limited functionality of simple JavaScript buttons. Buttons are no longer all placed onto a single screen with little organisation while overlapping the camera feed, and the robot arm can be moved a considerable distance in a small number of inputs. The system is now in a state where it could be reasonably used by a customer without a researcher being involved to turn it on.

In the realisation, the software has become more dynamic and flexible allowing for more interface features and freedom in experimenting with different designs and functions. The back-end has moved

from a simple implementation only able to blindly move in a direction to a system with multiple features for manipulating the arm in a way which complements the robot arm while implementing safety features to prevent the loss of control. Communication between the front- and backend can now happen directly instead of going through a separate server, with the front-end no longer being stuck with a limited set of single-purpose commands which each have to be implemented in the server. The camera implementation has been improved to remove the delay and stutter issues encountered in the prior research.

The evaluation has been expanded beyond a simple test performed only by the researcher, now the main client as well as multiple participants have used and evaluated the system. The tests with the main client have shown that the current implementation of this system as well as one envisioned by the prior work may not be suitable for the client. Because the prior work did not involve the main client, it mistakenly concluded that exhaustion was not an issue. This has been proven incorrect in this work, which is an important fact which should majorly influence future work. The extended testing also revealed that positional awareness of the user of the robot arm requires extra attention, especially with the client's seeming struggle to understand moving the arm in 3D space.

This research does regress on the prior work in a few areas. The main one is the lack of rotation in the system, which prevents certain tasks from being accomplished. It also does not have a stepped movement mode, which arguably means that the prior system made it easier to move in small steps. The prior work also has a button for homing/calibrating the arm, which this system does not. Finally, the prior system works in the Mozilla Firefox browser, whereas the system is incompatible with the latest version of Firefox as of the writing of this report. This incompatibility can be attributed to the increased complexity in the new system, which can arguably be seen as a step back.

In the prior work D. Gaethofs describes recommendations for future work, the main one being lowering task accomplishment time. While the testing results between these reports cannot be compared directly as the tasks are different, user results show a decent satisfaction in the speed of the system. Another recommendation was moving the system to a Raspberry Pi, which this system accomplishes. This research also advances in the question of where a camera should be mounted.

Some recommendations for future work have not been advanced upon. This research has not implemented a system for controlling the arm by selecting a point on a 2D plane. It also does not implement a function for saving and returning to arbitrary points, although it does lay the groundwork for this in the back-end. It also does not implement a drinking function. Aside from the camera it does not advance on the question of where and how the arm should be mounted onto a wheelchair, and aside from implementing a basic safety feature it does not advance on the question of safety around other people.

#### **8.4 Future work**

Future work should be performed to iterate upon the results of this research and produce a better system. This should include future research into different aspects of the system, further implementation of features left unfinished in the current system, and creating systems which use different approaches to reach the same goals. Below are my recommendations for future work on the system.

Future research should start with analysing what Andrei or other clients want to do with the system and which approaches would be best for accomplishing these goals. It should also work towards understanding the client's ability to understand 3D space and the suitability of current systems. More testing sessions for the client to get used to the system may provide further insight into what they are capable of and how a system should be designed around them. Perhaps a training course could be created to familiarise the client with 3D control and perspective.

To aid with the usability of the system, research should be done into how to best represent the 3D world onto the 2D display. This may involve different graphical elements to visualise the depth and/or give more intuitive feedback. As part of this work should be performed to find the most effective camera angles to best convey the current state of the system and its surroundings. Perhaps sensors could be used to help convey data such as distance to an object. Other senses could also be used improve spatial awareness, similarly to the beeping of a car reverse parking sensor.

Research should be done into practical aspects of the system. Where should the arm be mounted and how should the system be powered? How should a camera cable be routed from the hand? What precautions should be made to ensure the system is safe for the user and bystanders? How can the system be used for eating and drinking, especially when the user cannot control the location of their mouth? Can the user control the arm with the robot arm in front of their face? Can the user choke on food or liquid if the system is not aware of how fast they are consuming it? Even with safety precautions, is it wise to leave the client alone when they use the system in case a safety feature fails to be effective?

A number of features should be implemented before the current system is ready to be used by clients. Simple features, such as buttons for homing the arm as well as shutting down or restarting the system should be provided to remove the need for using text commands to access basic features. These buttons could be

implemented under a sub-menu, perhaps alongside a settings button. This should allow for the possibility of changing the behaviour of the interface without the need for accessing different webpages. This would include settings for changing whether the start button is used or directions activate immediately, changing the dwell-delay of different button types and changing the disengagement delay of direction buttons. The lock button should also be improved to make it less easy to repeatedly change states. Finally, error messages could be improved such as indicating communication issues with the arm and potential fixes.

Some other features should be added to improve the usability of the system. This includes the ability to control the arm in steps, the ability to save points to be reactivated later, the ability to set boundaries on the arm's movement to prevent collisions and the ability for multiple cameras to be used. The stepped control could be implemented in the form of a toggle, potentially with the ability to change the size of the steps. Saving and loading states could be done from a sub-menu. Different cameras could be displayed simultaneously or switched between using buttons. Boundaries could be programmed at the installation stage, helping the user avoid collisions with their own wheelchair.

The system could be expanded with more movement options. Arm rotation could be reintroduced into the system to give more freedom in moving the arm. This could also be accompanied by the ability to move individual fingers of the hand. Perhaps control of individual joints could be added, or a system for moving in common arcs. Better feedback such as indicating when the arm has reached the end of its inverse kinematics range for a given command could be added to improve the ease of use for these more complex commands.

The results of the client testing point towards considering radically different approaches of accomplishing the client's wishes. This could mean diverging attention away from an interface for direct manipulation of the arm and towards a system where the arm can automatically grab objects or perform tasks. One of these features would be implementing a drinking mode where the arm uses sensors to automatically move towards the user's mouth and then tips a cup just enough to give the user a sip of water. Another extension would be the ability for the arm to automatically detect and grab objects, either through a marker-based system or computer vision recognition. Depth sensors could be combined with user gaze data to move the arm towards any point in space by them just looking at it or a representation of it on their computer.

Perhaps a different system entirely could be created for aiding with specific tasks such as drinking. One idea could have a liquid container with a straw and pump mounted which automatically moves the straw towards the user's mouth where the user activates the pump using gaze-control.

## 9 CONCLUSION

### 9.1 Answering the research questions

The research questions are answered based on the results of this paper.

**Research question:** How can a system be designed to enable an assistive robot arm to be controlled using gaze?

Designing a system for controlling a robot arm means researching gaze-controlled interface design and assistive robot arm control design. It involves learning about the goals and desires of the customer and implementing these into a list of requirements. The system needs an interface which is first imagined during a brainstorm, then refined by applying requirements and finally implemented during realisation. The system needs a back-end which should be designed for flexibility in the frontend and needs to run on a device where it is easy to install and use. The system should be tested with multiple people including the client to ensure it achieved their requirements.

**Sub-question 1:** What is the current state-of-the-art of gaze-control and controlling assistive robot arms?

The current state-of-the-art uses gaze-control with dwell buttons or as assistance with traditional controls by looking at objects in a space. For controlling assistive robot arms a joystick is used with robot axis mapped to the joystick axis in Cartesian space. It also has predefined home and resting positions as well as modes specifically for drinking.

**Sub-question 2:** Can users accurately stop the robot arm at a desired position by performing a gaze interaction at a precise time during its movement?

Users are able to stop the robot arm at a desired position by selecting a button at a specific time, although the final position may not be accurate enough to pick up an object and small movements are difficult to make.

**Sub-question 3:** Is a separate confirm button preferred for preventing accidental movements?

It depends. If this button makes it so people can look away from the interface afterwards it appears to be effective, although not necessarily preferred over directly activating a direction.

**Sub-question 4:** Should control of the robot arm be from the reference frame of the hand instead of the base of the arm or the physical position of the user interface?

This was not tested during this research.

**Sub-question 5:** Is full manual control suitable for a person with disabilities to control a robot arm?

Likely not, as the client struggled to understand how to move the arm and suffered from exhaustion after using the system for a limited time.

### 9.2 Conclusion

A system was created for controlling an assistive robot arm with gaze. User testing showed that the system could be used to pick up, move and place down a bottle in 2-3 minutes. Testing with a client revealed that the system was complex and straining for them to use, as such future work should focus on creating a system which is simpler to use.



## REFERENCES

- [1] M. Cross, L. Qiu, M. Zhong, Y. Wang, and Y. Shi, “One-Dimensional Eye-Gaze Typing Interface for People with Locked-in Syndrome,” in *UIST '22 Adjunct: Adjunct Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*. New York, NY, USA: Association for Computing Machinery, Oct. 2022, pp. 1–3.
- [2] D. Gaethofs, “Adaptive control of a kinova assistive robotic arm,” Ph.D. dissertation, University of Twente, 02 2024, unpublished.
- [3] Tobii, “Tobii Dynavox Communicator 5 Getting Started,” 2016. [Online]. Available: [https://download.mytobiidynavox.com/Communicator/documents/TobiiDynavox\\_Communicator5\\_GettingStartedGuide\\_v1-6-1\\_en-US\\_WEB.pdf](https://download.mytobiidynavox.com/Communicator/documents/TobiiDynavox_Communicator5_GettingStartedGuide_v1-6-1_en-US_WEB.pdf)
- [4] R. J. K. Jacob, “The use of eye movements in human-computer interaction techniques: what you look at is what you get,” *ACM Trans. Inf. Syst.*, vol. 9, no. 2, pp. 152–169, Apr. 1991.
- [5] C. Lutteroth, M. Penkar, and G. Weber, “Gaze vs. Mouse: A Fast and Accurate Gaze-Only Click Alternative,” in *UIST '15: Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. New York, NY, USA: Association for Computing Machinery, Nov. 2015, pp. 385–394.
- [6] X. Zhang, X. Ren, and H. Zha, “Modeling dwell-based eye pointing target acquisition,” in *CHI '10: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery, Apr. 2010, pp. 2083–2092.
- [7] A. R. Ramirez Gomez, C. Clarke, L. Sidenmark, and H. Gellersen, “Gaze+Hold: Eyes-only Direct Manipulation with Continuous Gaze Modulated by Closure of One Eye,” in *ETRA '21 Full Papers: ACM Symposium on Eye Tracking Research and Applications*. New York, NY, USA: Association for Computing Machinery, May 2021, pp. 1–12.
- [8] H. Istance, R. Bates, A. Hyrskykari, and S. Vickers, “Snap clutch, a moded approach to solving the Midas touch problem,” in *ETRA '08: Proceedings of the 2008 symposium on Eye tracking research & applications*. New York, NY, USA: Association for Computing Machinery, Mar. 2008, pp. 221–228.
- [9] M. Choi, D. Sakamoto, and T. Ono, “Bubble Gaze Cursor + Bubble Gaze Lens: Applying Area Cursor Technique to Eye-Gaze Interface,” in *ETRA '20 Full Papers: ACM Symposium on Eye Tracking Research and Applications*. New York, NY, USA: Association for Computing Machinery, Jun. 2020, pp. 1–10.
- [10] J. M. Araujo, G. Zhang, J. P. P. Hansen, and S. Puthusserypady, “Exploring Eye-Gaze Wheelchair Control,” in *ETRA '20 Adjunct: ACM Symposium on Eye Tracking Research and Applications*. New York, NY, USA: Association for Computing Machinery, Jun. 2020, pp. 1–8.
- [11] G. Zhang and J. P. Hansen, “People with Motor Disabilities Using Gaze to Control Telerobots,” in *CHI EA '20: Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery, Apr. 2020, pp. 1–9.
- [12] Kinova, “Jaco® assistive robot User guide,” 2023. [Online]. Available: <https://assistive.kinovarobotics.com/uploads/EN-UG-007-Jaco-user-guide-R05.pdf>
- [13] —, “Kinova Assistive Robot Arm,” Apr. 2024, [Online; accessed 25. Apr. 2024]. [Online]. Available: <https://assistive.kinovarobotics.com/product/jaco-robotic-arm>
- [14] A. H. Mader and W. Eggink, “A Design Process for Creative Technology,” in *Proceedings of the 16th International conference on Engineering and Product Design, E&PDE 2014*. The Design Society, Sep. 2014, pp. 568–573. [Online]. Available: <https://research.utwente.nl/en/publications/a-design-process-for-creative-technology>
- [15] M. S. H. Sunny, M. I. I. Zarif, I. Rulik, J. Sanjuan, M. H. Rahman, S. I. Ahamed, I. Wang, K. Schultz, and B. Brahmi, “Eye-gaze control of a wheelchair mounted 6DOF assistive robot for activities of daily living,” *J. NeuroEng. Rehabil.*, vol. 18, no. 1, pp. 1–12, Dec. 2021.

## NOTE ON USE OF GENERATIVE AI

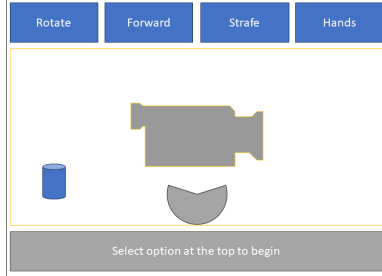
During the preparation of this work the author(s) used ChatGPT GPT-4o in order to help with technical issues, perform small coding and documenting tasks, brainstorm a report title and find correct terms to use in the report. No contiguous part of the report longer than a word has been generated by AI, and at least 95% of the contents of the files created for the technical system have been manually created. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the work.

## A FINAL CONCEPT

The final concept created during ideation. The original concept contained some animation, this has not been preserved in this publication.

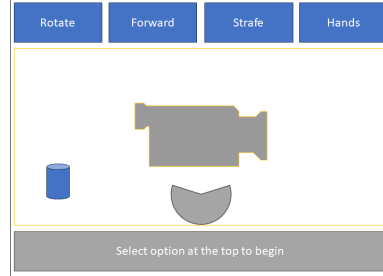
**Figure 21.** Storyboard of interface mock-up

The goal is to grab the cylinder with the pac-man shaped claw. Everything in the yellow box is webcam footage. The webcam is mounted on top of the hand.



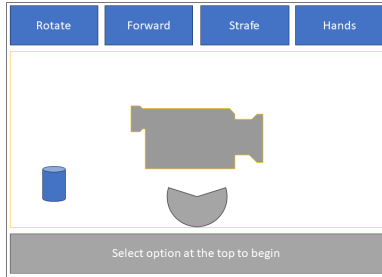
(a)

The arm moves relative to its orientation. Meaning, the forward option always moves in the same direction that the hand and webcam are looking, and the strafe directions are at a 90 degree angle to the forward option.



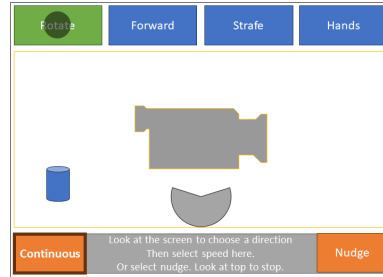
(b)

The place that was gazed at between slides to cause the given effect is indicated by a grey dot. View the slides in order to see a demonstration, or click on the blue buttons to skip to a part.



(c)

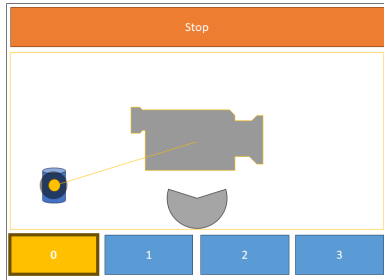
First we select the rotate option to make the webcam face the object we want to grab. Since we want to make a large adjustment, we'll keep the mode on continuous.



(d)

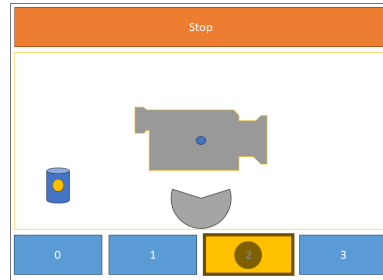
(Figure continues on next page.)

Now we dwell on the cylinder and follow it with our eyes. The hand will keep moving in the direction of the cylinder, and will slow down as we look closer to the center.



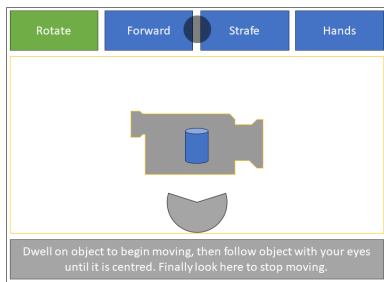
(e)

Now we dwell on the cylinder and follow it with our eyes. The hand will keep moving in the direction of the cylinder, and will slow down as we look closer to the center.



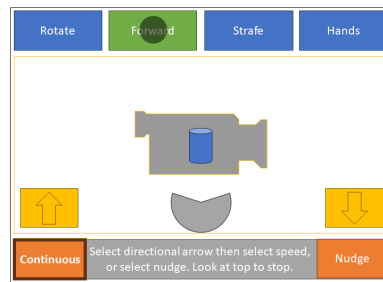
(f)

When the cylinder reached the centre, we look at the stop button to immediately stop moving.



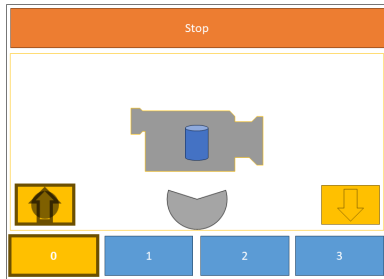
(g)

Now that the hand is facing the cylinder, we'll move forward towards it. We'll use the continuous mode.



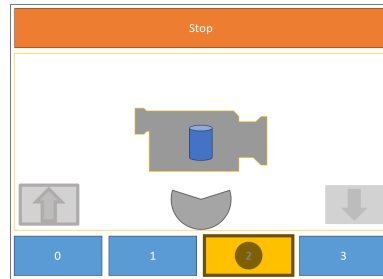
(h)

After selecting the forward direction, we can now select a speed at the bottom. The default speed is 0 so we are not moving yet.



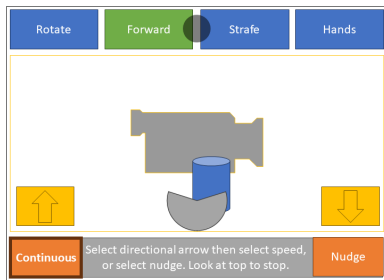
(i)

We'll choose a moderate speed of 2.



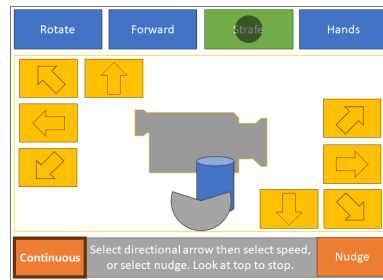
(j)

Once we are close to the cylinder, we'll look at the stop button and stop moving. Because we didn't face the cylinder perfectly, it isn't perfectly aligned with the hand.



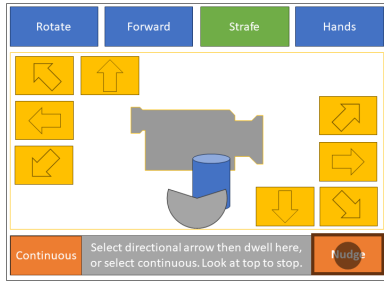
(k)

We'll use the strafe function to better align the hands. This mode allows us to move in 8 directions relative to the direction of the hand.



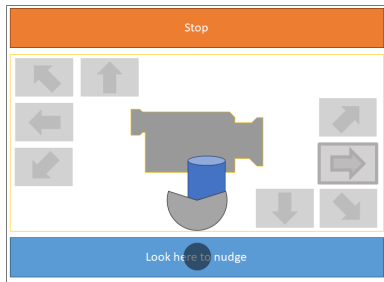
(l)

We'll select the nudge mode, as we want to make a small precise adjustment.



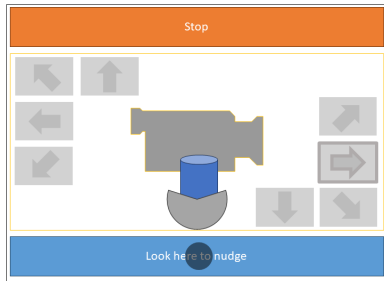
(m)

Now we'll dwell on the nudge button. The directional arrows become deactivated out so we can more freely look at the camera feed.



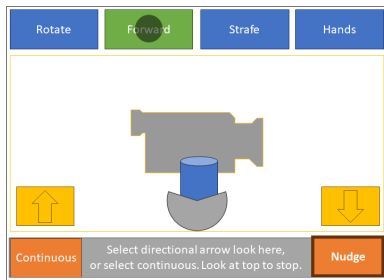
(o)

Last one, now we're lined up properly.



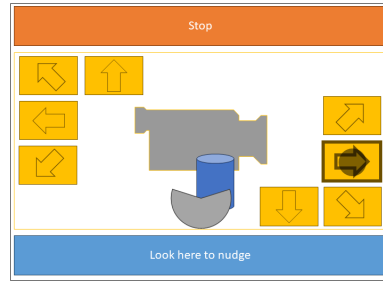
(q)

Now we'll switch back to the forward mode, as the cylinder isn't yet between the fingers. The interface remembered that we were in nudge mode while strafing.



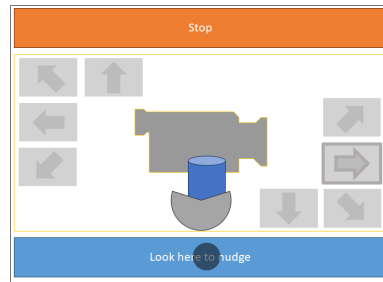
(s)

Next we'll select right, and the button at the bottom changes to a confirmation button.



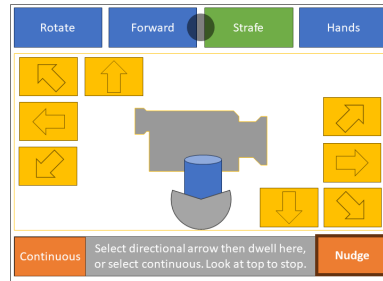
(n)

We'll have to press the nudge button a few times to get it lined up.



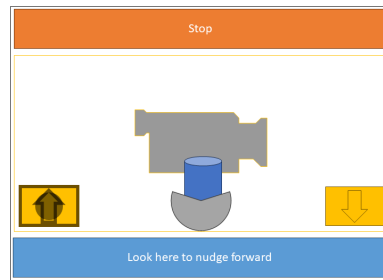
(p)

We'll look at the stop button to exit the movement.



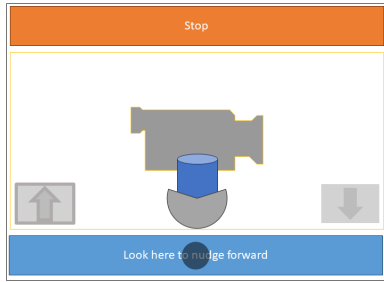
(r)

We'll select the forward direction.



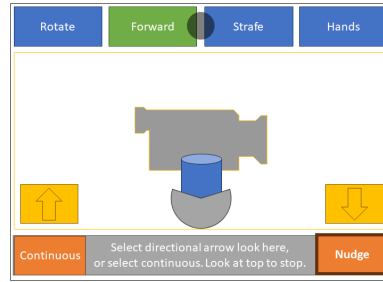
(t)

Now we'll nudge forward. Just 1 nudge seems to be enough.



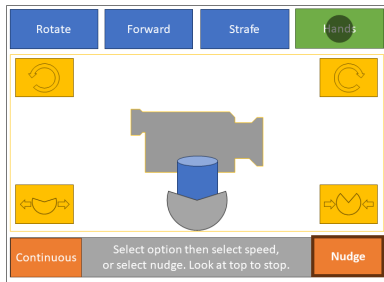
**(u)**

We'll glance at the stop button to exit.



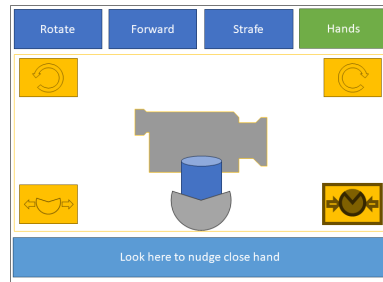
**(v)**

Now we want to close the hands. This menu also allows you to twist the hand, but we just need to close it. Once again it remembered us being in nudge mode.



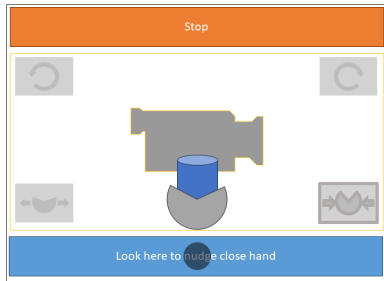
**(w)**

We'll select the hand close option.



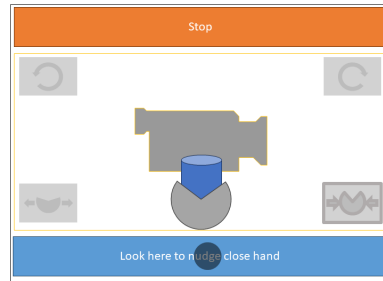
**(x)**

Now we start nudging until the hand has a good grip on the cylinder.



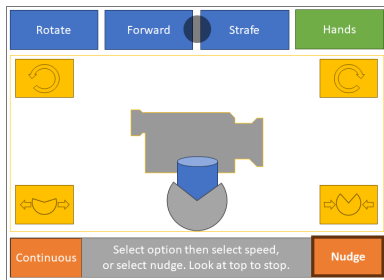
**(y)**

Nudge.



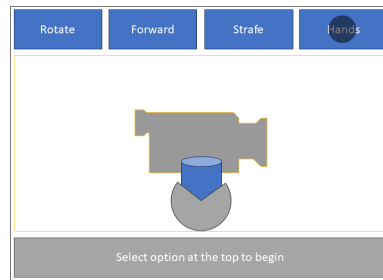
**(z)**

The cylinder is properly grabbed so we exit by glancing at the stop button.



**(aa)**

Finally we dwell on the Hands button to exit the movement mode and essentially lock the UI. We can now use all the options to freely move the cylinder around.

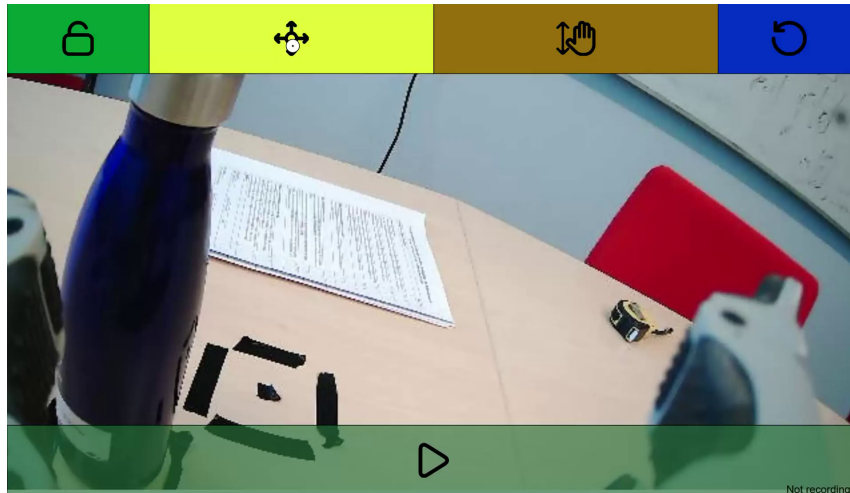


**(ab)**

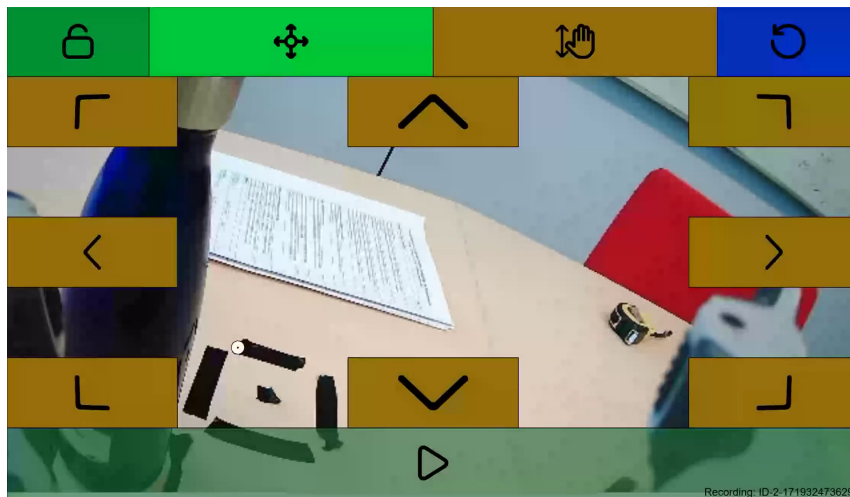
## B USER INTERFACE SCREENSHOTS

The final user interface shown as screenshots from recordings. Quality deficits such as low clarity are a result of the recording method and not an indication of the actual image quality during use.

**Figure 22.** Final user interface screenshots

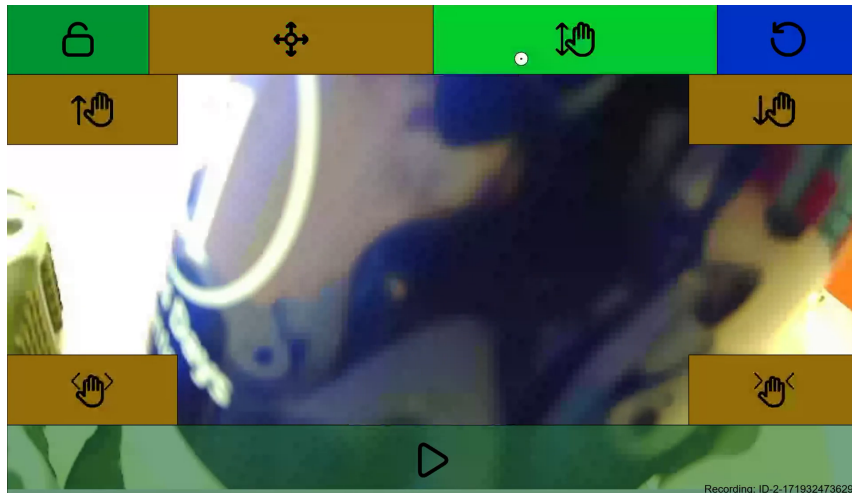


**(a)** Start-Stop interface with no axis group active and strafe axis group button hovered but not activated.

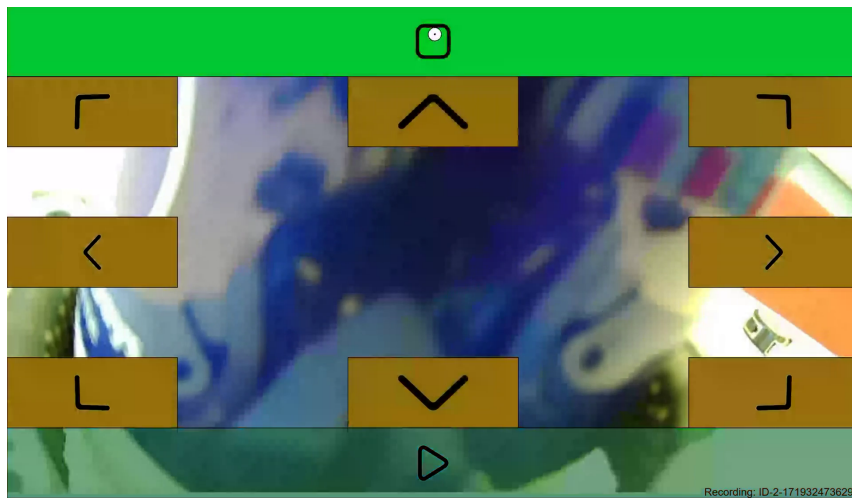


**(b)** Start-Stop interface with strafe axis group active but no axis selected.

**Figure 22.** Final user interface screenshots (cont.)

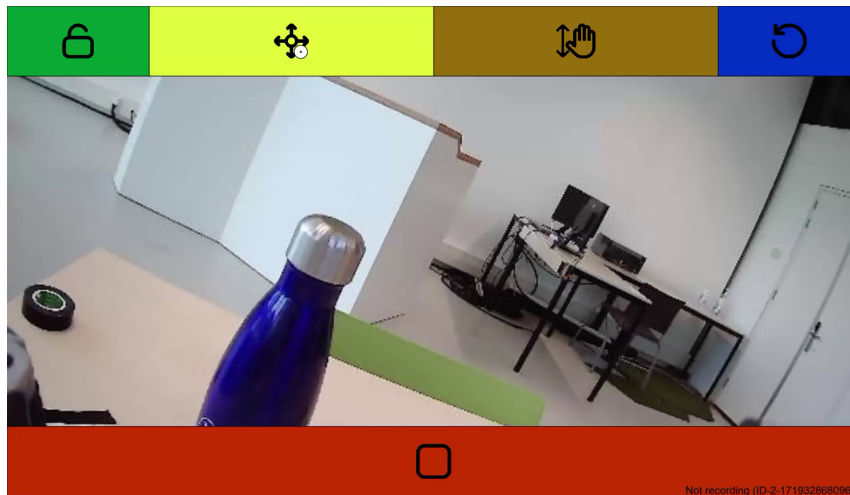


**(c)** Start-Stop interface with hand open and around bottle, forward/backward/grab axis group active but no axis selected.

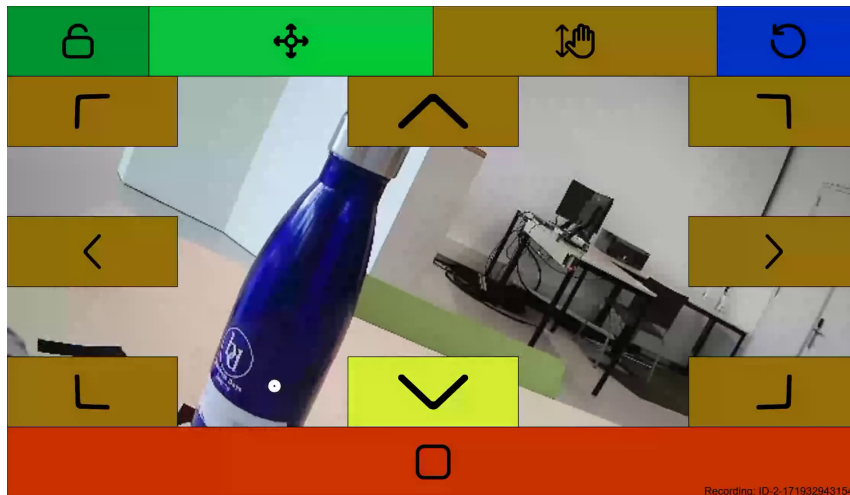


**(d)** Start-Stop interface with bottle grabbed and stop button hovered and activated, strafe axis group active but no axis selected as this selection was removed by activation of stop button.

**Figure 22.** Final user interface screenshots (cont.)



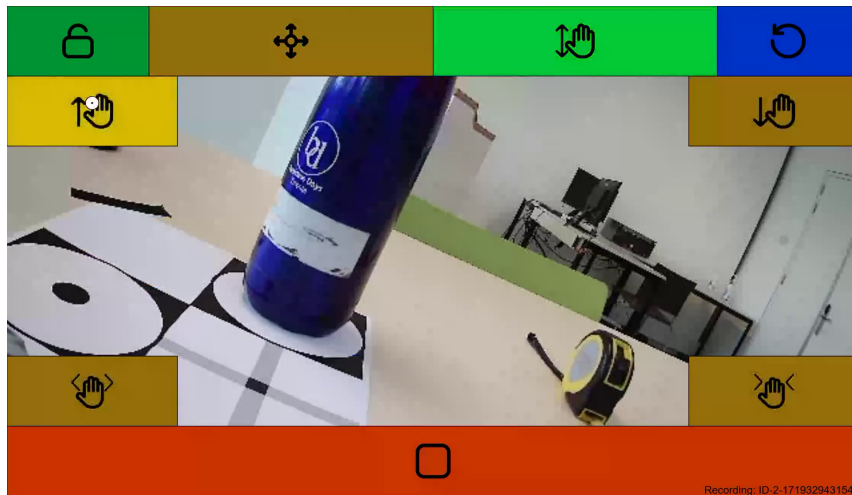
**(e)** Hold interface with no axis group active and strafe axis group button hovered but not activated.



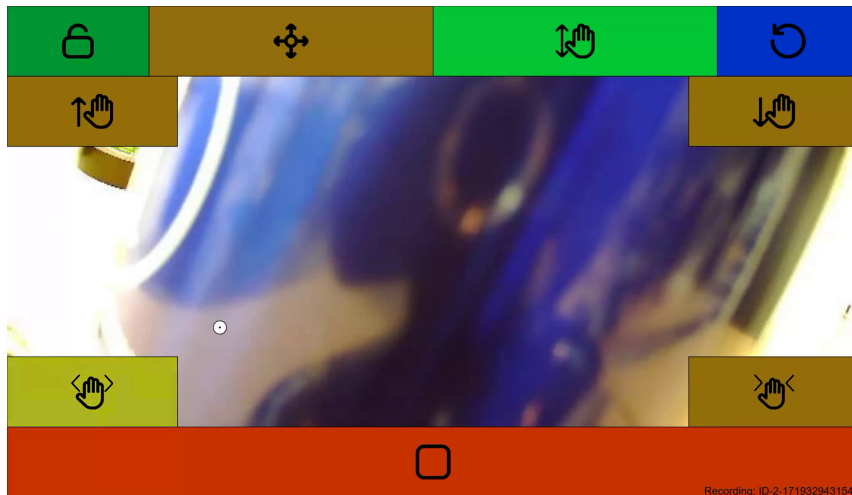
**(f)** Hold interface with down axis currently active by release-delay as it is no longer hovered.



**Figure 22.** Final user interface screenshots (cont.)

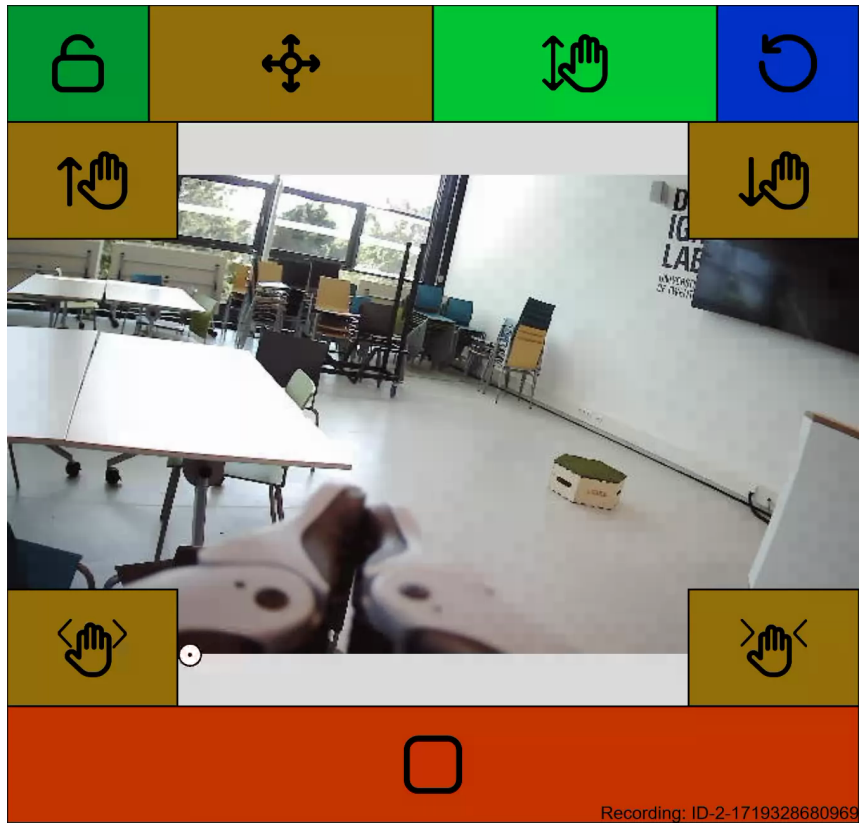


**(g)** Hold interface with forward/backward/grab axis group active but no axis selected and forward button currently being hovered which will soon activate.

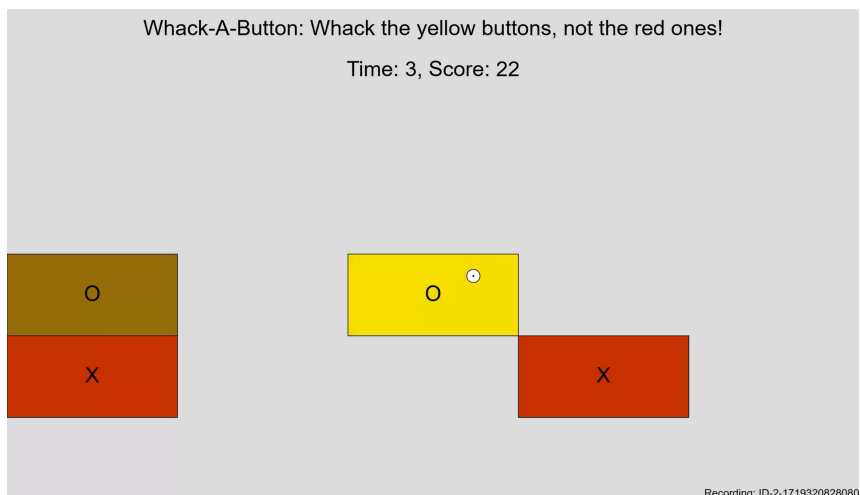


**(h)** Hold interface with hand around bottle and just let go as the hand open axis button is still active by release-delay.

Figure 22. Final user interface screenshots (cont.)



(i) Hold interface fit within a differently shaped window. Forward/backward/grab axis group active but no axis selected, hand closed.



(j) Whack-A-Button game with 3 seconds left and a score of 22. Two good buttons and two bad buttons are on-screen with a good button being hovered.

## C EXPLANATION FOR TESTERS

Before testers during user testing begin a task, an explanation is provided. The explanations are approximately as follows (though presented in a less formal tone), with actions performed by the tester in square brackets.

**Before the initial questionnaire:** "This questionnaire is about things which may influence the result of the test."

**Before starting Whack-A-Button:** "The game is like Whac-A-Mole, only with buttons instead. Buttons will randomly appear, and you need to activate them by looking at them for a small amount of time. Only hit the yellow buttons, not the red ones."

**After finishing Whack-A-Button:** "The files that were just downloaded contain a recording of the screen as well as a log of your actions. Do not mind them."

**Before using the start-stop interface:** "This first interface is as follows. [Point at lock button] this button locks the interface so you can look around safely. [Point at reset button] this button resets the arm to its original point; it should be hit before and after every task. [Point at strafe and forward/backward/grab buttons] these buttons allow you to switch to two sets of movement options. [Trigger strafe menu] these buttons move the arm relative to the webcam view, so up and down, left and right. The camera view is unfortunately crooked, imagine it being straight instead. [Select a direction] when you select a direction, the start button becomes available. Select it to start moving. [Select the start button] when you want to stop the movement, look at the stop button. [Select the stop button] the stop button doesn't disappear until you look away from it, at which point you can use the top buttons again. [Look away from the stop button, select the forward/backward/grab menu, point at forward and backward buttons] in this menu you can move the arm forward and backward, [Point at hand grab and release buttons] and open and close the hand. [Reset the arm, click the download pop-up away]. Your task is to pick up the bottle, move it to the other point, put it down, open the hand and then reset the arm. You may begin whenever you're ready."

**Before using the hold interface:** "This second interface works almost identically to the first, although now when you select a direction the arm will start moving immediately. It will stop moving after you look away for a second which you can use to look at the arm, or you can stop it faster using the stop button which is now located at the button. Otherwise, the interfaces are identical."

**Before the post questionnaire:** "Now I would like to ask you to give feedback on the system via this questionnaire. You may also speak out loud your considerations. Afterwards I will ask for your opinion on some potential changes to the system."

## D QUESTIONNAIRES

Results of post-questionnaire, and screenshots of all questionnaires. The exact results from the questionnaires have been omitted for privacy.

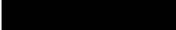

Question	Average	Sample variance	Values
In the hold interface, how well did you feel you were in control of the arm?	4	0	
In the start/stop interface, how well did you feel you were in control of the arm?	4.33	1.33	
In the hold interface, how accurately did you feel that the system responded to your eye movements?	4.67	0.33	
In the start/stop interface, how accurately did you feel that the system responded to your eye movements?	3.67	2.33	
In the hold interface, how quickly did you feel the system allowed you to accomplish the goals?	4.33	1.33	
In the start/stop interface, how quickly did you feel the system allowed you to accomplish the goals?	3.33	1.33	
In the hold interface, how difficult were the controls to learn in your experience?	4.33	0.33	
In the start/stop interface, how difficult were the controls to learn in your experience?	3.67	2.33	
In the hold interface, do you feel like you were easily able to stop movement at the right position?	3	1	
In the start/stop interface, do you feel like you were easily able to stop movement at the right position?	4	1	
In the hold interface, were you delayed in accomplishing your goal due to accidental inputs?	2.67	0.33	
In the start/stop interface, were you delayed in accomplishing your goal due to accidental inputs?	4	0	
In the hold interface, did you feel that the movement of the robotic arm was too slow or too fast?	3.33	0.33	1: Too slow, 5: Too fast
In the start/stop interface, did you feel that the movement of the robotic arm was too slow or too fast?	3.33	0.33	1: Too slow, 5: Too fast
In the hold interface, did you feel like the buttons should have been smaller or larger?	3	0	1: Smaller, 5: Larger
In the start/stop interface, did you feel like the buttons should have been smaller or larger?	3	0	1: Smaller, 5: Larger
In the hold interface, did you feel aware of the position of the robot arm while moving it?	1.67	0.33	
In the start/stop interface, did you feel aware of the position of the robot arm while moving it?	3	1	
In the hold interface, did you prefer letting the button stop automatically or did you prefer using the stop button?	3	3	1: Stop automatically, 5: Use the stop button
In the hold interface, did you look at the robot arm and target directly or did you look via the camera?	3.33	0.33	1: Looked directly, 5: Looked via camera
In the start/stop interface, did you look at the robot arm and target directly or did you look via the camera?	2.33	0.33	1: Looked directly, 5: Looked via camera
Did you feel that the camera feed was responsive?	5	0	
Did you feel that the camera was positioned in a useful position?	3.67	0.33	
Which interface did you prefer?	3	3	1:Start/stop, 5:Hold
How do you rate your experiences with the system?	5	0	


**Figure 23.** Post questionnaire numerical results, averaged with sample average

(Figures continue on next page.)

**Figure 24.** Initial questionnaire

## Kinova Gaze testing - Initial Questionnaire

 [Switch accounts](#) 

 Not shared

\* Indicates required question

Please enter your given identifier number \*

Your answer \_\_\_\_\_

Do you have any prior experience with gaze-controlled systems? \*

Yes - Often and recent (More than 3 times in the last year)

Yes - Infrequent and recent (1 to 3 times in the last year)

Yes - Often but not recent (More than 3 times ever)

Yes - Infrequent and not recent (1 to 3 times ever)

No

Could you elaborate on your experience with gaze tracked systems? (Optional)

Your answer \_\_\_\_\_

Do you have any prior experience controlling a robotic arm? \*

Yes - Often and recent (More than 3 times in the last year)

Yes - Infrequent and recent (1 to 3 times in the last year)

Yes - Often but not recent (More than 3 times ever)

Yes - Infrequent and not recent (1 to 3 times ever)

No - But I do have some prior knowledge and/or have observed people controlling robotic arms

No - I am not familiar with the control of robotic arms

Do you have (mild) visual impairment in one or both of your eyes? \*

Yes - Strong visual impairment which affects my ability to see even with assistive eyewear (e.g. glasses)

Yes - Visual impairment which seriously affect my ability to function without assistive eyewear but is mostly or fully corrected by assistive eyewear (e.g. glasses)

**(a)** Initial Questionnaire

- Yes - Mild visual impairment which does not seriously affect my ability to function without assistive eyewear and may be mostly or fully corrected by assistive eyewear (e.g. glasses)
- No
- Not sure
- Prefer not to answer

Are you currently wearing (corrective) eyewear? \*

- Yes - I am wearing glasses or other eyewear which covers my complete eye(s)
- Yes - I am wearing contact lenses or other eyewear which is directly applied to the eye
- No
- Other: \_\_\_\_\_

Do you currently suffer from dry eyes or something else which makes it difficult to hold your eyes open for long? \*

- Yes
- No
- Maybe
- Other: \_\_\_\_\_

Do you suffer from uncontrollable movements of your head or eyes? \*

- Yes
- No
- Prefer not to answer
- Other: \_\_\_\_\_

Is there anything else you believe may impact the result from this study? (Optional)

Your answer \_\_\_\_\_



Submit


Clear form

(b) Initial Questionnaire (cont.)

**Figure 25.** Post questionnaire

## Kinova Gaze testing - Post Questionnaire

 [Switch accounts](#) 

 Not shared

**\* Indicates required question**

Please enter your given identifier number \*

Your answer

In the hold interface, how well did you feel you were in control of the arm? \*

Barely in control   1   2   3   4   5   Fully in control

In the start/stop interface, how well did you feel you were in control of the arm? \*

Barely in control   1   2   3   4   5   Fully in control

In the hold interface, how accurately did you feel that the system responded to your eye movements? \*

Not accurate   1   2   3   4   5   Very accurate

**(a)** Post Questionnaire

In the start/stop interface, how accurately did you feel that the system responded \*  
to your eye movements?

	1	2	3	4	5	
Not accurate	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very accurate

In the hold interface, how quickly did you feel the system allowed you to \*  
accomplish the goals?

	1	2	3	4	5	
Slowly	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Quickly

In the start/stop interface, how quickly did you feel the system allowed you to \*  
accomplish the goals?

	1	2	3	4	5	
Slowly	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Quickly

In the hold interface, how difficult were the controls to learn in your experience? \*

	1	2	3	4	5	
Very difficult	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very easy

In the start/stop interface, how difficult were the controls to learn in your \*  
experience?

	1	2	3	4	5	
Very difficult	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very easy

(b) Post Questionnaire (cont.)



In the hold interface, do you feel like you were easily able to stop movement at the right position? \*

1 2 3 4 5

I was not able to stop at the right position      I was able to easily stop at the right position

In the start/stop interface, do you feel like you were easily able to stop movement at the right position? \*

1 2 3 4 5

I was not able to stop at the right position      I was able to easily stop at the right position

In the hold interface, were you delayed in accomplishing your goal due to accidental inputs? \*

1 2 3 4 5

Accidental inputs delayed me a lot      Accidental inputs did not delay me at all

In the start/stop interface, were you delayed in accomplishing your goal due to accidental inputs? \*

1 2 3 4 5

Accidental inputs delayed me a lot      Accidental inputs did not delay me at all

In the hold interface, did you feel that the movement of the robotic arm was too slow or too fast? \*

1 2 3 4 5

Too slow      Too fast

(c) Post Questionnaire (cont.)

In the start/stop interface, did you feel that the movement of the robotic arm was <sup>\*</sup> too slow or too fast?

	1	2	3	4	5	
Too slow	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Too fast

In the hold interface, did you feel like the buttons should have been smaller or <sup>\*</sup> larger?

	1	2	3	4	5	
Smaller	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Larger

In the start/stop interface, did you feel like the buttons should have been smaller <sup>\*</sup> or larger?

	1	2	3	4	5	
Smaller	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Larger

In the hold interface, did you feel aware of the position of the robot arm while <sup>\*</sup> moving it?

	1	2	3	4	5	
Not aware at all	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Aware at all times

In the start/stop interface, did you feel aware of the position of the robot arm <sup>\*</sup> while moving it?

	1	2	3	4	5	
Not aware at all	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Aware at all times

(d) Post Questionnaire (cont.)

In the hold interface, did you prefer letting the button stop automatically or did you prefer using the stop button? \*

1 2 3 4 5

Prefer automatic stop      Prefer using the stop button

In the hold interface, did you look at the robot arm and target directly or did you look via the camera? \*

1 2 3 4 5

Only looked directly      Only looked through the camera

In the start/stop interface, did you look at the robot arm and target directly or did you look via the camera? \*

1 2 3 4 5

Only looked directly      Only looked through the camera

Did you feel that the camera feed was responsive? \*

1 2 3 4 5

Not responsive      Responsive

Did you feel that the camera was positioned in a useful position? \*

1 2 3 4 5

Not useful      Useful

(e) Post Questionnaire (cont.)

Which interface did you prefer? \*

	1	2	3	4	5	
Start/stop interface	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Hold interface

How do you rate your experiences with the system? \*

	1	2	3	4	5	
Bad	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Good

If you have any elaboration on the questions above, suggestions for changes to the interface or further things you would like to share, please write them here  
(Optional)

Your answer

---

(f) Post Questionnaire (cont.)