Aalborg University
Biomedical Engineering and Informatics

Master's Thesis

Group 10410

Designing for Freedom: Shared Control in Assistive Robotics for Tetraplegia

Participants:

James Nicholas Gardiner
Kenneth Skovgaard Pedersen

Supervisors:

Ásgerður Arna Pálsdóttir
Lotte N. S. Andreasen Struijk

June 1, 2023

AALBORG UNIVERSITY
MASTER'S THESIS

AALBORG UNIVERSITY
MASTER'S THESIS

Project topic:

Master's Thesis

Project period:

February 2023 - June 2023

Project group:

gr10410

Participants:

James Nicholas Gardiner
Kenneth Skovgaard Pedersen

Supervisor:

Ásgerður Arna Pálsdóttir
Lotte N. S. Andreasen Struijk

Pages report: 65
Pages appendix: 17
Pages total: 82
Finished: 1st of June 2023

Department of Health Science and Technology
Biomedical Engineering and Informatics
Selma Lagerløfs Vej 249
9260 Gistrup
Denmark
https://www.hst.aau.dk

Abstract:

Introduction: Tetraplegia, a consequence of cervical spinal cord injury, has a profound impact on patients' quality of life and autonomy. The rising demand for advanced solutions due to healthcare personnel scarcity motivates the exploration of a proof-of-concept shared control system for robotic manipulator systems to aid those with tetraplegia.
Methods: A system combining user input with autonomous control was developed to handle the Kinova JACO^2 Assistive Robotic Manipulator (ARM). As the ARM approaches an object, a Graphical User Interface (GUI) displays a selection menu, triggering the ARM to autonomously secure the selected object and revert to its original position. The system, built upon the Robot Operating System (ROS), incorporates You Only Look Once (YOLO) V7 for object recognition, and Grasp Pose Detection (GPD) for generating grasp poses. Visual input was gathered via an Intel® RealSense™ D435 RGB-D camera.
Results: Evaluation of the system's performance was carried out focusing on object selection, autonomous grasp control, object detection and segmentation, user interface design, grasp pose generation, and grasp execution. Spatial accuracy tests revealed slight variations within the accepted tolerance. The autonomous robotic system demonstrated a 55.5% success rate in spatial point navigation across 27 trials. Despite proficiency in object selection and detection, limitations emerged in considering object properties for grasp pose generation and grasp execution, marking areas that necessitate further improvement and iterations.
Conclusion: This work represents an initial step towards the advancement of assistive technologies that could significantly improve the quality of life for individuals with tetraplegia. Nonetheless, the proof-of-concept shared control system indicates the need for considerable refinement and additional iterations. Future endeavors should encompass iterative testing and feedback incorporation from individuals with paralysis to ensure the final solution aligns with user needs and delivers tangible benefits.

Preface

This project is a Master's Thesis in Biomedical Engineering and Informatics made by students in group gr10410 from the Biomedical Engineering and Informatics Master's degree at the Department of Health Science and Technology at Aalborg University. The project period was from the 2nd of February 2023 to the 1st of June 2023. Thanks to the supervisors Lotte N. S. Andreasen Struijk and Ásgerður Arna Pálsdóttir for guidance and technical support throughout the project period.

James Nicholas Gardiner

Kenneth Skovgaard Pedersen

Reading guide

This project utilizes the Harvard method of referencing. This method involves referencing sources by presenting the author's last name and the year of publication. When a reference is placed before a period, it pertains to the preceding sentence, while if it is placed after a period, it refers to the preceding section. When there are multiple authors, the reference is made using the last name of the first author, followed by et al. For example, [Safiri et al.,2020] are used when referencing a source with multiple authors.

To reference various elements like tables, figures, listings, and sections, the citation includes the type, chapter, and number of the element. For instance, the first table in chapter one is referred to as Table 1.1, while the first figure in chapter two is referenced as Figure 2.1.

When referring to appendixes, they are organized alphabetically, and the citation should indicate the letter of the appendix.

When presenting an abbreviated term, the full form is written first, followed by the abbreviation in parenthesis. After the abbreviation has been defined, it is used throughout the text. For example, Grasp Pose Detection (GPD) would be used when referencing the abbreviated term.

Contents

1 Introduction

A spinal cord injury at the cervical level results in tetraplegia, or quadriplegia, a condition where there is partial or total paralysis of all four limbs. It is estimated that there are around 0.9 million incident cases, and 20.6 million prevalent cases of spinal cord injuries worldwide, with around 50% classified as tetraplegic. [Ding et al., 2022] In the US, approximately 17,700 new spinal cord injury cases occur each year, with tetraplegia accounting for around half of them. [Bennett et al., 2022] Individuals with tetraplegia experience significant challenges in multiple areas of daily life, including mobility, self-care, communication, and participation in social activities. Individuals with tetraplegia may require extensive assistance from caregivers, and many experience depression, anxiety, and other mental health issues. [Region Midtjylland, 2010; Bennett et al., 2022] Despite advances in medical and rehabilitation interventions, tetraplegia remains a lifelong and debilitating condition, with a significant impact on quality of life.

According to data from the West Danish Center for Spinal Cord Injury (Vestdansk Center for Rygmarvsskade, Regionshospitalet Viborg), tetraplegia caused by spinal cord injuries is a relatively rare condition in Denmark, with an estimated incidence of around 130 new cases per year.[Region Midtjylland, 2010] According to a recent study, a total of 1,751 individuals who suffered from spinal cord injury were admitted to specialized rehabilitation centers in Denmark between 2008 and 2018. [Soendergaard et al., 2022] The most common causes of spinal cord injuries leading to tetraplegia in Denmark are traffic accidents (47%), falls (22%), and sports-related injuries (11%). [Region Midtjylland, 2010] There are approximately 3000 individuals living with spinal cord injury in Denmark, with the majority being male (around 80%) and aged between 18 and 64 years (around 75%). [Region Midtjylland, 2010] The ethnic composition of this population is not readily available, but it is likely to reflect the diversity of Danish society.

Individuals with tetraplegia in Denmark receive comprehensive medical and rehabilitation care through the public healthcare system, which covers the costs of treatment, assistive devices, and personal assistance services. However, despite the availability of high-quality healthcare services, individuals living with tetraplegia still face significant challenges in their daily lives, such as limited mobility, social isolation, and mental health issues.

People who experience tetraplegia come from diverse backgrounds and may have different needs depending on their age, gender, ethnicity, geographic location, and other factors. However, they all share the common experience of living with a severe disability that affects their independence and ability to participate in society.[Region Midtjylland, 2010] Tetraplegic individuals depend on healthcare workers to continue living, as the recent development in the lack of healthcare workers is predicted to increase there is a pressing need to find avenues to aid these patients. [Baes-Jørgensen, 2022]

The utilization of robotic assistance has the potential to enhance the quality of life for individuals with tetraplegia, by providing them with increased autonomy, mobility, and social engagement. However, designing and implementing effective and cost-efficient robotic solutions that cater to the unique needs of this population remains a formidable challenge. Although commercially available

robotic manipulators are currently in use by paralyzed individuals, the rudimentary control systems governing their movements lack the requisite level of generality required for widespread adoption as a substitute for human workers. [Kronhardt et al., 2022; Petrich et al., 2021]

The initial problem statement driving this research project is the investigation of robotics as a means to assist individuals affected by tetraplegia. The focus lies on the development of a prototype system that can be tested and evaluated in real-world settings. By addressing this initial problem statement, we aim to contribute significantly to the advancement of more effective and accessible technologies, ultimately improving the quality of life for individuals with tetraplegia.

Problem Analysis 2

2.1 Tetraplegia

Tetraplegia is characterized by partial or complete paralysis of all four limbs and the torso. Tetraplegia is caused by various conditions that lead to damage or compression of the spinal cord, including traumatic injuries, infections, tumors, and degenerative disorders. [Bennett et al., 2022; Region Midtjylland, 2010]

Tetraplegia can be caused by either a traumatic or non-traumatic injury to the spinal cord. Traumatic spinal cord injuries are often due to direct trauma to the spinal cord or due to compression from fractured vertebrae masses (e.g. epidural hematomas, abscesses). The most common causes of tetraplegia are spinal cord injuries resulting from accidents, such as falls, motor vehicle collisions, violence, and sports injuries. On the other hand, non-traumatic spinal cord injuries typically result from an underlying condition such as degenerative conditions (e.g. spinal stenosis, herniated discs), tumors, vascular problems, infections, or inflammation. [Bennett et al., 2022; Soendergaard et al., 2022]

Tetraplegia can be categorized as complete or incomplete. In cases of complete tetraplegia, all the muscles below the injury site are completely paralyzed, and the sense of touch is lost. People with complete tetraplegia are usually dependent on a wheelchair for mobility. In cases of incomplete tetraplegia, there is some connection to the lower part of the spinal cord, which can result in partial muscle function and/or partial preservation of sensation below the injury site. Individuals with incomplete tetraplegia can have varying levels of function, ranging from minimal sensation to almost normal function. [Region Midtjylland, 2010]

2.1.1 Short explanation of the spinal cord

The spinal cord is a dense bundle of nerves that is responsible for transmitting messages from the brain to the rest of the body. The spinal cord is enclosed by the spinal, or vertebral column, which consists of 33 vertebrae, and is divided into different regions such as the cervical, thoracic, lumbar, and sacral regions. The spinal cord divides into 31 lateral pairs of spinal nerves, eight cervical (C1-C8), 12 thoracic (T1-T12), five lumbar (L1-L5), five sacral (S1-S5), and one coccygeal nerve pair. An illustration of this is presented in Figure 2.1. [Khan and Lui, 2022]

When an injury to the spinal cord happens, it can cause the vertebrae to damage or compress the spinal nerves within the cord and thus disrupt the communication between the brain and body. When such an injury occurs and the spinal cord is damaged, nerve impulses are blocked or disrupted leading to loss of sensory and muscle function below the site of injury. The extent of paralysis can vary depending on the level of injury, for example, if the injury occurs high up in the spinal cord at the cervical level it could result in tetraplegia, as the nerves that control arms, legs, and torso are located in this region. Whereas if the injury occurs at a lower level on the spinal cord (thoracic or lumbar region) it could lead to paraplegia, causing paralysis of the lower extremities as depicted in Figure 2.1. [Bennett et al., 2022; Region Midtjylland, 2010]

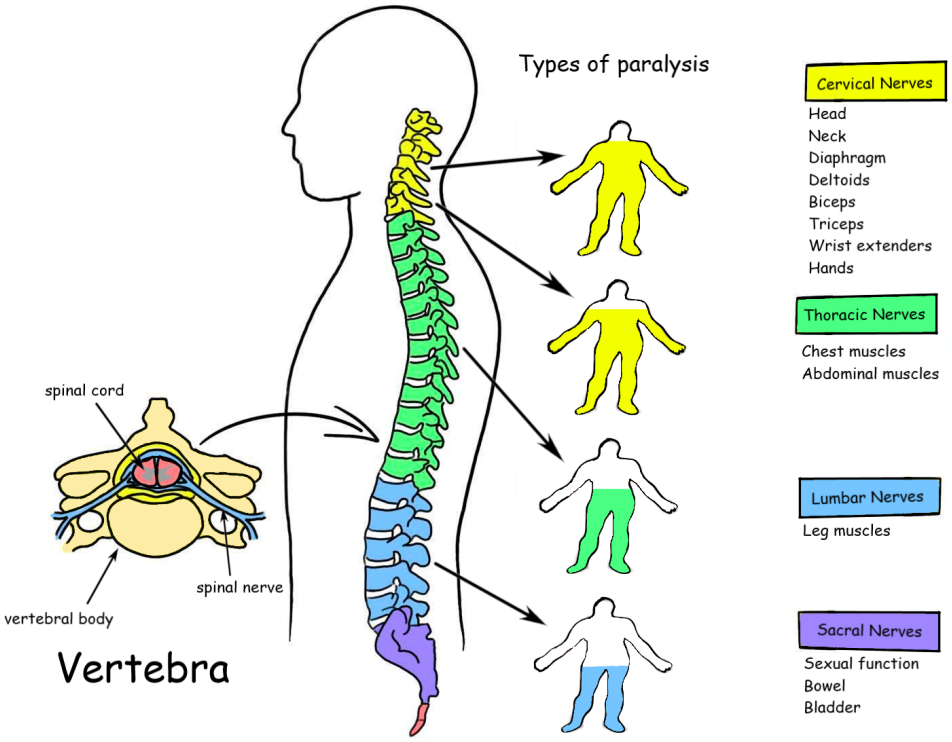


Figure 2.1. Illustration of a vertebra and spinal column highlighting the spinal cord, spinal nerves, and various types of paralysis depending on the level of injury on the spinal column.

Evaluation of individuals with tetraplegia includes a thorough physical examination and clinical assessment for accompanying injuries so as to pinpoint the specific patterns of injury, which help locate where and what type of injury occurred. Conducting a clinical examination with a detailed and accurate evaluation of the nerves related to motor and sensory functions is essential to properly classify the injury. [Bennett et al., 2022]

2.1.2 Challenges for individuals with tetraplegia

Individuals with tetraplegia experience a range of physical and functional impairments, including paralysis or weakness of the limbs, loss of sensation, spasticity, bowel and bladder dysfunction, respiratory problems, and autonomic dysreflexia. These symptoms can significantly impact their quality of life, social participation, and independence. [Bennett et al., 2022; Region Midtjylland, 2010]

The paralysis results in bone decalcification with a 25-fold increased risk of femoral fractures. Likewise, do the sensory disturbances increase the risk of conducting pressure sores, paralysis of the bladder increases the risk of infection and in the long term, damage to the kidneys which eventually can lead to kidney failure.[Region Midtjylland, 2010]

As a wheelchair user, it is usually not possible to return to their previous work if they are even able to work. This has an impact on their economy as well as their home are rarely suitable for a wheelchair. Their sex life is affected by a deficiency of sensation in the genital area, erection, and fertility problems. The family function is also changed as they no longer can perform the tasks they previously had, as they are unable to manage activities of daily living, such as personal hygiene, toileting, eating, and dressing themselves. It is common for these individuals to experience a crisis reaction and depression as a result of their condition. [Region Midtjylland, 2010]

A Danish study conducted by Soendergaard et al. [2022] investigated the socioeconomic consequences of these individuals. The study showed in accordance with previous studies, that healthcare costs (which include costs in relation to healthcare systems, rehabilitation services, medicine, transportation, and personal assistance) are generally highest in the first year after injury and decrease over time, whereas productivity costs (including loss of productivity and other labor market consequences) can have long term consequences and thus exceed the health care costs, especially with younger individuals. Furthermore, it is mentioned that these individuals are at higher risk of change in employment status and divorce.

2.2 Treatment

2.2.1 Acute treatment

The treatment of spinal cord injuries starts at the injury site by stabilizing the patients before transferring them to a hospital. At the site of injury, the patient is immobilized to prevent further damage, and immediate measures are essential to maintain breathing and hemodynamic stability, as hypotension and shock are factors that can worsen the impact of SCIs and likewise worsen the likelihood of neurological recovery. After transfer to a hospital, the SCI patients are treated in specialized neurological intensive care units and trauma units, where high doses of steroids continue to be the primary treatment for acute cases. Although there have been several medications studied for improving the outcomes of spinal cord injury patients, further research is necessary to identify other potential medical treatments for this condition. [Bennett et al., 2022]

2.2.2 Limitations of current treatment and care

The quality of life (QoL) of tetraplegic patients can be improved through conventional treatments such as pharmacological medication, surgery, and physical rehabilitation. However, these treatments still require significant support from family, friends, and healthcare workers for the patient to engage in daily activities. [Region Midtjylland, 2010]

The management of tetraplegia typically involves a multidisciplinary approach, including pharmaceuticals, surgery, assistive devices, physical therapy, and rehabilitation. Medications may be used to manage pain, spasticity, and other symptoms, while surgery may be required to decompress the spinal cord or stabilize the spine. Assistive devices, such as wheelchairs, lifts, and communication aids, can help tetraplegic patients perform daily activities and improve their quality of life. Physical therapy and rehabilitation aim to maximize the patient's functional abilities and prevent secondary complications.[Bennett et al., 2022; Region Midtjylland, 2010]

However, despite these interventions, tetraplegic patients still face significant challenges in their daily lives, including limited mobility, dependence on caregivers, social isolation, and decreased quality of life. Moreover, the cost and availability of these treatments may vary depending on the patient's geographic location and socioeconomic status. [Bennett et al., 2022; Region Midtjylland, 2010]

The shortage of healthcare personnel to aid individuals living with paralysis highlights the pressing need for changes in the healthcare system and only compounds the previously mentioned challenges. The insufficiency of healthcare personnel poses a significant challenge to the provision of essential care to those living with paralysis. Neglecting this issue would result in a significant gap in the provision of care, with adverse consequences for those affected. Therefore, proactive measures are essential to ensure that adequate care is accessible to all individuals, regardless of their physical abilities. For individuals who are paralyzed, daily tasks and functions are often dependent on the assistance of others, making reliable healthcare staff a necessity for their continued well-being.

Without appropriate resources delegated to meet this need, this group will be significantly impacted. In fact, a lack of healthcare staff is a crucial challenge that places additional developmental pressure on the healthcare system. Addressing this challenge will require either the training of more healthcare staff or the exploration of alternative avenues of treatment and care that are not reliant on human resources. [Baes-Jørgensen, 2022]

In recent years, advances in computational power, machine learning, and software capabilities have led to the development of a new field of research dedicated to assisting paralyzed individuals. This field of research involves using technology to help individuals with paralysis perform daily activities. Various technological approaches have been developed, but they all involve utilizing the patient's remaining movement capabilities to control a mechanical device that aids in activities such as drinking, eating, and dressing. The underlying premise of this approach is to enhance the independence and autonomy of paralyzed individuals, enabling them to perform activities without relying on others. [Kronhardt et al., 2022; Andreasen Struijk et al., 2017]

The development of these technologies involves a combination of engineering, computer science, and medical expertise. For example, sensors are used to detect the patient's movements, which are then translated into commands that control the mechanical device. Machine learning algorithms are used to optimize the control system and enable the device to adapt to the patient's specific needs and capabilities.[Nanavati et al., 2023]

2.3 Robotic assistance

2.3.1 Robotics in healthcare

In the field of assistive technologies for paralyzed individuals, there are three main categories that aim to solve the same underlying issue of enabling individuals to perform daily activities independently. These three categories are prosthetics, exoskeletons, and assistive robotic manipulators (ARM). [Rosen et al., 2005; Johansen et al., 2016]

Prosthetics refer to artificial limbs that are designed to replace a missing or non-functioning limb. In the case of paralysis, prosthetics can be used to replace the function of a hand or arm. These prosthetic devices are controlled using signals from the user's remaining limb muscles, which are detected by sensors and translated into movements of the prosthetic. [Lai et al., 2007] Exoskeletons, on the other hand, are wearable devices that provide external support to the user's limbs. These devices are typically worn around the legs and are designed to enable individuals with paralysis to stand and walk. Exoskeletons use sensors to detect the user's movements and provide assistance with the necessary joint movements.[Rosen et al., 2005; Bengtson et al., 2020]

Robotic manipulators are devices that are used to assist with activities of daily living such as eating, drinking, and dressing. These devices are typically mounted on a mobile platform and are controlled using signals from the user's remaining limb muscles or other inputs such as eye tracking or voice commands. Robotic manipulators are designed to provide assistance with tasks that require fine motor skills, such as using utensils or fastening buttons. These manipulators share a lot of the same foundation as prosthetics and exoskeletons in terms of the control systems and hardware, but due to being external to the human body are often easier to implement. [Bjomdahl Mortensen et al., 2021; Hildebrand et al., 2019]

2.3.2 Current adoption of robotic assistance

Assistive technologies such as prosthetics, exoskeletons, and robotic manipulators have demonstrated their effectiveness in improving the quality of life and independence of individuals

with paralysis. Studies have shown that these devices enable individuals to perform activities that were previously difficult or impossible. For instance, robotic exoskeletons have improved mobility and cardiovascular health, increased social participation and psychological well-being for individuals with spinal cord injuries.[Kronhardt et al., 2022; Fattal et al., 2019]

Germanotta et al. [2023] found that an assistive robotic device, the robot named Motore, was able to help rehabilitate stroke patients with similar outcomes as conventional treatment, but with reduced healthcare involvement and reduced financial resources. Maheu et al. [2011] quantified the usage of a JACO ARM, and found that in the context of a controlled environment, the manipulator could reduce the need for caregiving time by 41%.

Upper-limb prosthetics have been shown to significantly improve the quality of life and level of independence for individuals with upper-limb amputations. Robotic manipulators like the JACO ARM have been effective in assisting individuals with daily activities such as eating, drinking, and grooming. Furthermore, ongoing research in the field is aimed at improving the functionality and usability of these devices, as well as developing new technologies that can address specific needs of paralyzed individuals. As technology continues to evolve and improve, these devices will become increasingly accessible and effective in enabling paralyzed individuals to perform a wider range of activities with greater ease and autonomy. [Maheu et al., 2011; Germanotta et al., 2023; Petrich et al., 2021]

In addition to the existing assistive technologies, various control systems have been developed to help paralyzed individuals control these devices. Tongue-controlled systems like the iTongue have shown promising results in enabling individuals with high-level spinal cord injuries to operate a computer, wheelchair, or ARM. These systems utilize a small magnet attached to the tongue and sensors mounted on a headset to detect the movement of the tongue, which is then translated into commands for the device. Similarly, brain-computer interfaces (BCIs) have been developed to allow individuals to control external robotic devices using electrical signaling from the brain. [Hildebrand et al., 2019; Bjomdahl Mortensen et al., 2021]

Currently, robotic manipulators are commercially available for public use in cases where deemed necessary. The availability of such technologies is largely dependent on the geographical location and socioeconomic factors of a given country. [Beaudoin et al., 2018] In Denmark assistive devices meant to aid in eating and similar daily activities are available in most municipalities. A list of welfare technology implementations can be seen on the welfare technology map of Denmark (velfærds landskort) provided by KL. [KL, 2023]

2.3.3 Feasibility of widespread implementation?

Assistive technologies such as prosthetics and exoskeletons offer potential solutions for individuals with paralysis, but their implementation is not without significant technical challenges. Prosthetics, for example, require a high degree of technical expertise to develop and customize for individual users. This involves complex engineering and materials science, as well as a deep understanding of the human anatomy and musculoskeletal system. Calibration and adjustment of prosthetics to fit individual needs can also be a time-consuming and challenging process. [Dellon and Matsuoka, 2007; Hays et al., 2023]

Exoskeletons pose even greater technical challenges, requiring expertise in mechanical engineering, materials science, control systems, and human biomechanics and physiology. These devices must support the weight of the user while also providing precise control over joint movements, which can be difficult to achieve. Furthermore, exoskeletons require a significant amount of power to operate, which can limit their usability in certain settings. [Hays et al., 2023]

Both prosthetics and exoskeletons in their complete implementation pose an almost exact solution to immobility, but also pose technical challenges that are not able to be overcome as of now. This is in large part due to reliance on electrical signaling, whether it be EEG or EMG, which is not only difficult to decode but is also prone to noise and environmental factors. Additionally, the mere fact that prosthetics and exoskeletons are directly attached to the human body also adds a layer of complexity, wherein it is necessary to take into account human physiology, limits, and movement capabilities of the human body. [Beaudoin et al., 2018; Hays et al., 2023]

As a result of the significant technical challenges in developing and implementing prosthetics and exoskeletons, it may be beneficial to further the development of robotic manipulators in the short term. These devices offer a simpler solution that can be more easily integrated into daily life by using the remaining movement capabilities of the individual to control a mechanical device. While they may not offer the same level of mobility as prosthetics or exoskeletons, they can still significantly improve the quality of life for individuals with paralysis. [Beaudoin et al., 2018; Petrich et al., 2021]

2.3.4 Robotic manipulators

Robotic manipulators consist of three main components: a user interface, a control system, and hardware. The user interface is the means by which the individual interacts with the device and provides input to control its movements. This can take the form of physical buttons, touch screens, or voice commands, depending on the specific design and intended use of the device. [Maheu et al., 2011; Hays et al., 2023]

The control system is responsible for interpreting the user input and translating it into the appropriate movement of the robotic manipulator. This involves sophisticated algorithms and programming that must take into account the capabilities and limitations of both the device and the user. The control system also plays a critical role in ensuring the safety and reliability of the device, by monitoring for errors or malfunctions and taking appropriate corrective actions if necessary. [Ding et al., 2022; Law-Kam Cio et al., 2019; Markovic et al., 2015; Xu et al., 2022]

The hardware component of robotic manipulators includes the physical structure and mechanical components of the device, such as motors, joints, and end effectors. These components must be carefully designed and engineered to ensure that they are robust, reliable, and capable of performing the intended tasks. Additionally, the hardware must be lightweight and compact enough to be easily transported and operated by the user, while still providing the necessary strength and durability. [Renfrew, 2004]

Other important considerations when developing robotic manipulators include power consumption, communication protocols, and data storage and analysis capabilities. These factors can have a significant impact on the usability and effectiveness of the device and must be carefully optimized to ensure that the device meets the needs and requirements of its intended users. [Renfrew, 2004]

The development of robotic manipulators represents a promising avenue for improving the quality of life of individuals with paralysis. By focusing on user-friendly interfaces, sophisticated control systems, and well-designed hardware components, researchers and developers can create devices that are both effective and easy to use, ultimately enabling greater independence and autonomy for people with disabilities.[Hays et al., 2023; Renfrew, 2004]

2.3.5 Robotic manipulators in the literature: Challenges and hurdles

Robotic manipulators have come a long way in terms of hardware and mechanical design. The market offers a variety of highly advanced robotic manipulators such as iARM and JACO, that are

capable of complex movements and performing precise tasks with accuracy and speed. However, the development of control systems and interfaces that enable effective communication and decision-making between the robot and the operator remain significant challenges.

Although advances in robotics will improve usability, currently deployed models have sufficient capabilities to handle a large part of daily activities. This is in part due to the increase in degrees of freedom (DOF), actuator technology and build quality. The limiting factor is not due to hardware capabilities, but in the control systems that define the decision-making process of the robotic manipulator. The JACO robotic manipulator, for example, is mainly controlled by a joystick mounted to a wheelchair, but any user-dependent signal can be used as an input, such as iTongue, BMI, or EMG-based controllers. This method results in the user controlling the movement of the manipulator, making it challenging to perform routine activities such as eating and drinking. [Ding et al., 2022; Law-Kam Cio et al., 2019; Markovic et al., 2015; Xu et al., 2022]

The manipulator does not use raw user input from the controlling device in a 1:1 relationship, it interprets and remaps inputs to ensure sensible movement. The main limitation, and the underlying reason for why the execution time and workload of using a robotic manipulator is so high is due to the difference in available DOFs of the robot versus the user. A joystick typically has 2 DOFs, having to control a manipulator with 6 or more DOFs. In essence, this means mapping a low dimensional signal to a higher one, while maintaining reasonable control of all DOFs of the manipulator. This issue is compounded when factoring in the variable amount of DOFs of various controllers, and the available mobility of a given individual. Higher spinal injuries lead to less of the body being capable of movement, thus also limiting the DOF of the individual itself. [Kronhardt et al., 2022]

The user is primarily responsible for controlling the ARM, with limited automation of the manipulator. This begs the question if more automation of the controlling aspects of the manipulator might aid in reducing the need for manual control. Making the manipulator completely automatic would reduce the tetraplegic patient's feelings of autonomy and might not be favorable. Moreover, achieving greater autonomy in the manipulator requires sensors and other inputs to develop a "world view" that can interact constructively with the surroundings. This necessitates depth perception, object detection, object characteristics such as shape, weight, possible ways to grasp the object, and optimal paths to the destination. Additionally, a way to determine what the user wants to do, such as drink, eat, or pick up, is necessary.

These hurdles have spurred research into shared control systems. Which aims to split the burden of controlling the manipulator between the individual and the control system itself. This would in theory solve many of the practical issues in daily usage of manipulators, but also pose two new challenges. Firstly, how to automate the control system more so that it can independently execute meaningful tasks without the need for user inputs, and secondly how should the control be split?

Ka et al. [2018] developed a shared control system in which the robot would cease motion when in proximity to an object. The user would then be presented with various manipulation options via audible text output. The user could then decide what the manipulator should do with the object using voice commands. The robot was fitted with a low-cost depth camera to enable object detection and give real-world feedback to the control system. They stated that no benefits were seen with simple tasks, but with more complex tasks execution time was markedly lower, giving credence to the ability of shared control systems to enable more robust usage of manipulators.

Similarly Zhong et al. [2022] developed an algorithm that processes camera input and via the use of Bayseian deep learning would estimate the best possible grasp pose for the giving object. This was implemented in a shared control system and demonstrated the feasibility of such an approach. Such a technology could reduce the need for fine-manipulation of the fingers and wrists for the end-user,

thus enabling more fluent usage of the manipulator. Markovic et al. [2015] similarly created a shared control system in which a prosthetic used a camera, inertial sensors, and bioelectrical signaling to semi-autonomously aid in grasping with positive findings.

Koglin et al. [2019] simulated an ARM and used a machine learning-based approach to let it learn and self-organize sequences of movement into tasks. Although in its early stages of development, it showed the feasibility of letting a robot learn actions and tasks, and self-organize these into higher abstractive sequences of movement.

One of the drawbacks of shared control is the need for an expert programmer to manually outline how the control is shared. Quere et al. [2021] formalized a framework wherein machine learning was used to orient the control system with the user enabling continual reinforcement and task learning.

Xu et al. [2022] created a BMI-based shared control system wherein the user merely had to direct the manipulator towards the task/goal in gross movements, wherein the vision-based semi-autonomous control aspect would incorporate trajectory corrections and grasping. They showed a markedly decreased perceived user workload and a decrease in execution time, giving further evidence to the ability of shared control systems to aid in robotic control. Similarly ten Pas et al. [2021] also created an automatic pose detection algorithm analogous to object detection with promising results. Law-Kam Cio et al. [2019] created a control system using stereo vision, guided by eye-tracking with above 90% task completion rates.

Ding et al. [2022] created an assistive control system that intended to help with the task of drinking. Here the gross movement was made by the user and fine manipulation was delegated to the control system. Primarily results show a decrease in perceived workload and faster execution times, although they state more research is needed to generalize such an approach. Hildebrand et al. [2019] tested a semi-autonomous control of a robotic manipulator with the use of an intraoral tongue control interface. Although the added autonomous control did not give better performance it was noted that it resulted in fewer collisions and displacements of the object during execution. Bjomdahl Mortensen et al. [2021] later tested a very similar setup, with better outcomes. Only two participants were included in the trial, as such it remains to be seen if these results hold up in a broader population.

Kronhardt et al. [2022], tried an alternative approach to shared control by letting the mode switches be automatic based on contextual queues. Findings suggest that the need for mode switching by the user reduced markedly, although completion time and workload were similar to non-adaptive mode switching. They stipulate that more training time might have affected these measurements as well. [Kronhardt et al., 2022].

The primary problem in shared control systems and in general autonomy of robotic agents is the inability to generalize. Bengtson et al. [2020] notes that while many papers show that shared control schemes do indeed show better outcomes, it is only observed within the narrow scope of the tasks that the system was designed to handle. As such there is a pressing need to develop control systems that can generalize and handle arbitrary objects, and more broadly can handle tasks that are tangential to each other, enabling a more robust framework for daily usage.

In conclusion, while robotic control has made significant strides in healthcare, there are still many limitations that need to be addressed. These limitations include the inability to handle arbitrary objects, the need for better collaboration between robots and humans, and the need for more natural and intuitive control systems. Addressing these limitations will be critical to improving the accuracy, safety, and efficacy of robotic systems in healthcare.

Problem statement 3

Tetraplegia is a condition where all four limbs and the torso are partially or completely paralyzed due to spinal cord damage. Traumatic injuries are the most common cause, with falls, motor vehicle collisions, violence, and sports injuries being common causes. Individuals with tetraplegia experience physical and functional impairments that significantly impact their quality of life, social participation, and independence. The shortage of healthcare personnel to aid individuals living with paralysis highlights the need for changes in the healthcare system. The development of new technologies involving computational power, machine learning, and software capabilities has led to the creation of a new field of research dedicated to assisting paralyzed individuals. Various technological approaches have been developed, such as prosthetics, exoskeletons, and robotic manipulators. Robotic manipulators have shown to be effective in assisting individuals with daily activities such as eating, drinking, and grooming. There are still challenges and hurdles that remains as barriers in developing control systems and interfaces that allow effective communication and decision-making between the user and robotic manipulators. While shared control systems have the potential to reduce the need for manual control and improve usage of robotic manipulators, more research is required to develop effective control systems and interfaces that can interact constructively with the surroundings.

The problem analysis lead to the following aim for this project:

"Amplify the capabilities of existing technologies by establishing a shared control system for robotic manipulator systems, thus allowing more advanced task completion and improving the handling of arbitrary objects for individuals with tetraplegia."

Methods 4

The principal objective of this project is to design and implement a semi-autonomous control system intended to mediate interaction between individuals contending with tetraplegia and a robotic manipulator. Tetraplegia, a condition resulting in partial or total loss of use of all four limbs and torso, imposes significant limitations on individuals in executing everyday tasks [Bennett et al., 2022]. This project, in response, strives to construct a system that will aid these individuals in overcoming their physical constraints and augment their capacity to perform Activities of daily living.

This system is envisioned as a shared control platform where the robotic manipulator assists the user in handling objects without usurping complete control from the user. The intention behind such a design is twofold. On the one hand, the system intends to decrease the cognitive load on the user, making the operation of the manipulator less demanding and more efficient. This in turn leads to a reduction in the time spent on tasks, increasing overall productivity. On the other hand, the system maintains a level of user control to ensure users have a sense of agency and freedom in their interactions.

The development of this system is premised on the recognition of specific limitations in existing technologies. Research has shown the viability and potential benefits of shared control systems. These systems incorporate sensors, machine learning, and other inputs to enhance the manipulator's autonomy and enable it to perform more complex tasks, reducing the execution time and easing the operation for the user. However, shared control systems face several challenges. Generalization remains a significant issue, with most systems only demonstrating improved outcomes within the narrow scope of tasks for which they were designed. Consequently, there is a need to develop control systems that can handle a broader range of tasks and arbitrary objects, allowing for a more versatile and robust system for daily use. [Xu et al., 2022; Bengtson et al., 2020]

Improving the quality of life for its intended users forms the bedrock of this project's motivation. The proposed system, by enabling individuals with tetraplegia to conduct daily tasks with diminished reliance on external assistance, seeks to nurture a sense of autonomy. It is hoped that this increased independence will have a positive effect on the overall life satisfaction of these individuals. [Kim et al., 2012; Bengtson et al., 2020]

This document sets out a collection of broad, user-oriented requirements that provide a comprehensive yet non-technical depiction of the proposed system. These requirements focus on key aspects such as the user experience, system robustness, reliability, and safety. Serving as the guiding principles for system design and implementation, these requirements are intended to ensure that the development remains user-focused, combining user needs with technological capabilities within a framework of real-world applicability. By adhering to these principles, the aim is to create a system that not only facilitates task execution but also positively impacts the lives of its users.

4.1 User-Oriented Requirements for the Semi-Autonomous Control System

1. **User Independence and Autonomy:**
    a. The system should empower users with tetraplegia to perform daily tasks independently, reducing their reliance on external assistance.
    b. The system should provide users with a sense of control and involvement in task execution, promoting their autonomy.
    c. The system should ensure transparent communication of control shifts between the user and autonomous control.
2. **Cognitive Load Reduction and Efficiency:**
    a. The system should minimize the cognitive load on the user, making operation less demanding and more efficient.
    b. The system should reduce the time required to complete tasks, increasing overall productivity for the user.
3. **User-Friendly Interaction:**
    a. The system should offer intuitive interaction mechanisms, accommodating the users' capabilities.
    b. The system should provide a user-friendly interface, facilitating easy navigation and control.
    c. The system should offer clear and helpful feedback about user commands and system status.
4. **Versatile and Reliable Task Execution:**
    a. The system should handle a broad range of tasks and arbitrary objects, ensuring versatility for daily use.
    b. The system should accurately and securely execute user commands and autonomous actions, maintaining grip stability and object safety.
    c. The system should demonstrate a high level of reliability in task execution, instilling user confidence in its abilities.
5. **Safety and Control Transition:**
    a. The system should prioritize user safety during operation, implementing measures to prevent harm or accidents.
    b. The system should facilitate smooth transitions between user control and autonomous control, minimizing disruption to the user experience.
    c. The system should provide clear communication of control status and task progress, fostering trust and understanding.
6. **Integration and Compatibility:**
    a. The system should integrate seamlessly into the user's environment and routine, causing minimal disruption and promoting user acceptance.
    b. The system should comply with relevant safety and usability standards, ensuring a safe and user-friendly experience.

4.2 System design

In an effort to address the deficiencies and limitations found in current research on autonomous control of robotics for assisting individuals with tetraplegia, a novel system was designed. This system sought not only to enhance the capabilities of existing technologies but also to tackle specific shortcomings such as inefficiency in handling arbitrary objects and difficulties with fine manipulation.

The primary aim of the system was to strike a balance between user control and autonomous control. Rather than replacing human input entirely, the system was designed to enable individuals with tetraplegia to actively participate in task execution, while also offering the benefits of autonomous control. This approach was intended to empower users, allowing them to handle tasks more efficiently and accurately, particularly those involving arbitrary objects or requiring fine manipulation. An overview of the overarching design can be seen in Figure 4.1. This system design is meant to align and meet the requirements specified in the user requirements.

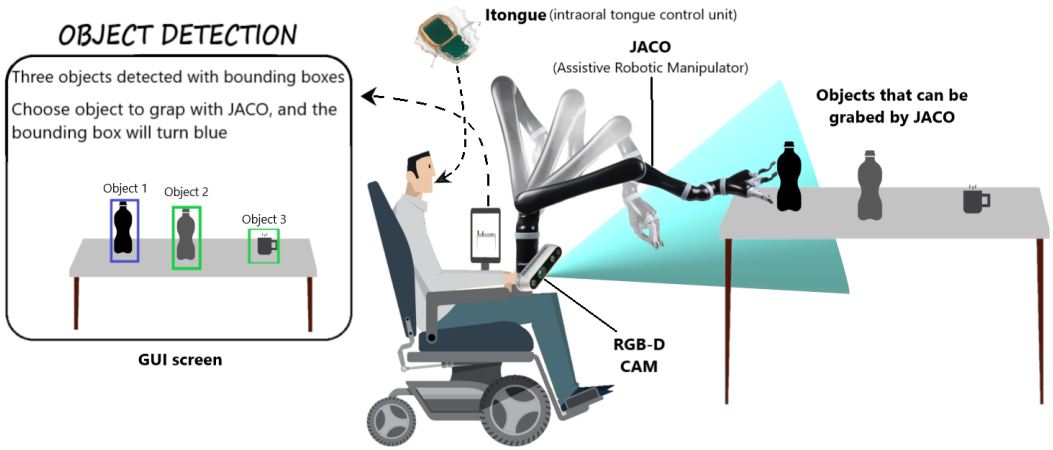


Figure 4.1. The system's illustration depicts the sequence of operations. Initially, the user directs the JACO^2 ARM to move towards a specific object. As the ARM approaches a predetermined distance from the object, the user is provided with a selection menu on the graphical user interface (GUI) screen, enabling them to choose the desired object to be grabbed (depicted on the left side of the GUI screen). Once the user makes a selection, the JACO^2 ARM transitions into autonomous mode, autonomously grasping the chosen object. Subsequently, the ARM returns to the original location where the autonomous action was initiated.

The system operates by incorporating the conventional method of utilizing the ARM for routine tasks. A display, situated on the wheelchair, presents a visual representation of the objects present in the surrounding environment. As the ARM approaches an Object of Interest (OOI) at close proximity, the system becomes activated. Subsequently, users are given the option to interact with the object by means of their input device.

After the user's intention to interact with OOI is conveyed, a transition towards enhanced autonomous control is initiated by the control system. The activation of this transfer of control is contingent upon the user's decision to interact with a particular object. Once a valid object is selected, the control system assumes responsibility for facilitating precise manipulation during the process, thereby alleviating the user from the need to perform multiple mode switches and cope with the cumbersome task of grasping objects.

Upon completion of the autonomous task execution, the control system relinquishes control back to the user, marking a return to the initial shared control state. This restoration of control signals the completion of the task and allows the user to continue with their activity, navigating the ARM to another object or location as needed. This process forms the operational cycle of the system,

with the balance between user and autonomous control being modulated based on the user's intent and the complexity of the task.

4.2.1 System requirements

The system's design was guided by a set of detailed requirements, addressing critical aspects like object selection, autonomous grasp control, object detection and segmentation, user interface design, grasp pose generation, and grasp execution. Each of these requirements was crafted with the aim of ensuring that the system was capable, reliable, and user-friendly, thereby providing a significant improvement over existing systems.

The following requirements describe the envisioned system in its entirety. Despite the constraints of the project timeline, which preclude the implementation of all aspects within this period, they remain crucial. These requirements provide a comprehensive understanding of the complete system and set the direction for future development work. Thus, while full realization may be beyond the scope of the current project, the requirements ensure that further development aligns with the original vision of the system's potential.

1. **Object Selection Requirement:**
    a. The system shall enable the user to select objects in the scene using an input device.
    b. The system shall provide a clear and intuitive interface for object selection.
    c. The system shall visually indicate or highlight the selected objects on the screen.
2. **Autonomous Grasp Control Requirement:**
    a. The system shall autonomously control the robotic manipulator to perform grasp tasks for selected objects.
    b. The system shall seamlessly transition between user control and autonomous control during grasp execution.
    c. The system shall effectively communicate to the user when the control is handed over or returned.
3. **Object Detection and Segmentation Requirement:**
    a. The system shall continuously process the RGB camera feed using an object detection and segmentation algorithm for object detection and segmentation.
    b. The system shall accurately detect and recognize objects in the scene, providing reliable segmentations and labels.
4. **User Interface Requirement:**
    a. The user interface shall allow the user to interact with objects using an input device.
    b. The user interface shall facilitate the intuitive selection of tasks associated with chosen objects.
5. **Grasp Pose Generation Requirement:**
    a. The system shall utilize the depth camera feed to generate feasible and appropriate grasp poses for the detected objects.
    b. The grasp pose generation algorithm shall consider the camera input, object properties, and the robotic manipulator's workspace.

c. The system shall provide accurate grasp pose commands to the robotic manipulator for successful grasping.

6. **Grasp Execution Requirement:**
   a. The robotic manipulator controlled by the system shall execute grasp commands accurately and reliably.
   b. The robotic manipulator shall securely grasp and hold the targeted objects without dropping or losing grip.
   c. The system shall ensure stable holding of grasped objects for a defined duration.
7. **Grasp Complexity Requirement:**
   a. The system shall enable grasping of arbitrary objects.
   b. The system shall enable grasping of objects in cluttered environments.
   c. The system shall handle grasping of partially occluded objects.
   d. The system shall handle grasping from appropriate approaches, such as from above, side and below.

Implementation 5

5.1 System overview

A prototype system was developed in adherence to the framework outlined in the 'System Design' section. Due to the finite duration of the project period, the full breadth of the envisioned system was not realized. The primary objective of the implementation was to fulfill a key subset of the stipulated system requirements. This subset includes the system requirements:

- **1a-c**
- **2a-c**
- **3a-b**
- **4a**
- **5a-c**
- **6a-c**

Figure 5.1 provides a graphical depiction of the system flow, delineating the interplay between various components. The entire system is partitioned into five distinct segments.

The first segment focuses on the detection of objects within the operational scene. Following this, a user-interaction layer is introduced, which allows users to selectively interact with the identified objects.

Next, an interpretation layer is integrated. This layer's purpose is to translate the selected image from the RGB camera stream into a spatially meaningful point cloud representation. Concurrently, this process also involves determining the spatial location of the selected object within this reference frame.

In the fourth segment, grasp poses, centered around the chosen object within the point cloud, are generated. Grasp poses constitute a data structure encoding the robot's movement path, encapsulating both rotational and positional data. This data directs the robot from its current position to the object, setting the stage for a successful object grasp.

In the final stage, the robot moves to the designated object following the calculated grasp pose, effectively grasping the object. This step signifies the culmination of the system flow, resulting in a successful object interaction as per user selection.

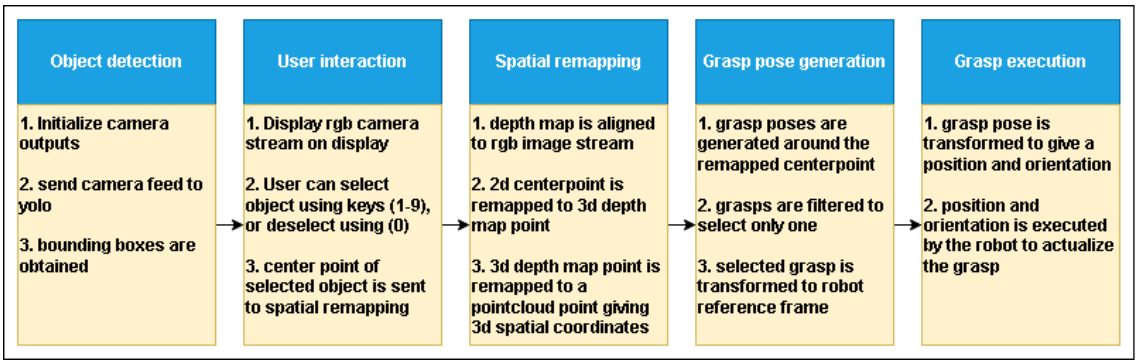


Figure 5.1. The diagram illustrates the overall system flow

The Robot Operating System (ROS) provided the framework for integrating the system's components. The hardware included a JACO^2 robotic manipulator and an Intel® RealSense™ D435 RGB-D camera. [ROS, 2007; Campeau-Lecours et al., 2018; Keselman et al., 2017]

Object recognition was achieved using You Only Look Once (YOLO) V7, an algorithm used for detecting objects in an RGB image stream. Grasp Pose Detection (GPD) was applied to determine grasp poses using a depth camera's data. ROS packages for Intel® RealSense™ and Kinova, along with a proprietary package, were implemented to create a functioning system flow. [ten Pas et al., 2017; Wang et al., 2022; ten Pas, 2021a; Kinova, 2023; Intel, 2023]

The choice of ROS was due to its architecture's suitability for parallel message handling and integration with existing robots. ROS operates on a topic/node system. Topics are data-holding structures and nodes handle decision-making, data processing, and messaging. Nodes publish data to a topic, which other nodes can subscribe to for data access. [ROS, 2007]

Integration of non-ROS conforming packages into ROS requires ROS-wrappers. For example, the Intel® RealSense™ D435 camera uses the Intel® RealSense™ SDK for camera-computer interface communication. The Realsense-ROS package provides a wrapper for this program, enabling the camera stream's use in the ROS framework. [Intel, 2023]

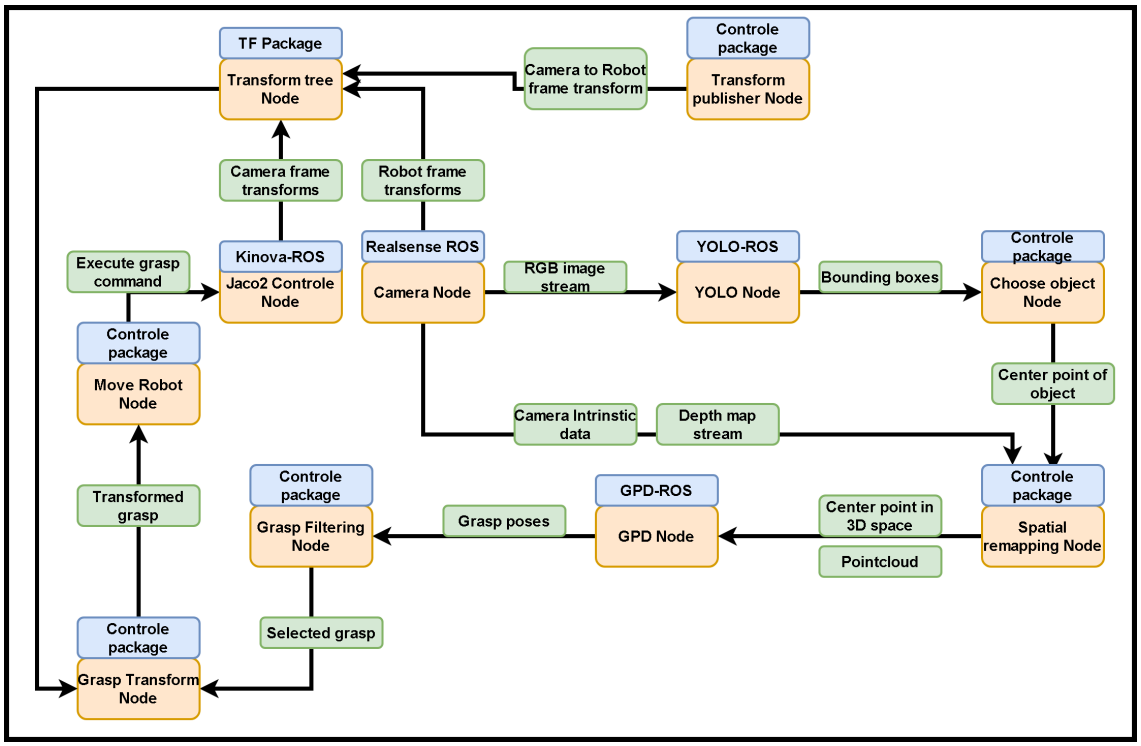


Figure 5.2. Diagram illustrating the connectivity and flow within the ROS framework, detailing the coordinated operation of individual components for object detection, interaction, and grasping. The orange boxes represent nodes, with the blue boxes indicating the source package for each node's execution. Meanwhile, the green boxes demonstrate the content of the messages communicated between them

A comprehensive connectivity and flow diagram of this ROS framework can be seen in figure 5.2. This network of nodes and messaging relays is activated by running a roslaunch file within the control package. Despite the abstraction of certain components, such as the 'choose object' node not sending a center point until user input is received, and the inclusion of error handling functions that can terminate grasp execution when necessary, the diagram illustrates the overall connectivity and messaging flow of the system.

5.2 Implementation Decisions and reasoning

Subsequent sections delineate the specific aspects of the implementation, detailing each system component and its contribution to the overall functioning of the prototype. This chapter aims to provide an in-depth description of the implementation process and the functionalities of the prototype system.

5.2.1 Hardware selection

Robotic ARM

In the process of selecting a robotic manipulator, it is essential to consider various factors to ensure its suitability and effectiveness. Key considerations include the manipulator's modularity and flexibility, which allow for customization to meet specific requirements. This adaptability enables the ARM to be equipped with different types of end effectors, such as grippers or sensors, broadening its range of applications. Additionally, the compatibility of the manipulator with diverse software development environments is crucial, as it facilitates the development and testing of control systems, thereby expanding potential applications However, different manipulators possess

unique features and strengths, and the choice of development platform ultimately hinges on the specific needs and requirements.

The JACO^2 ARM stands out due to its modular design, which allows for significant customization and adaptability to specific project requirements. With six degrees of freedom, it can be equipped with various types of end effectors to suit different applications, enhancing its versatility. [Keselman et al., 2017; Maheu et al., 2011]

Furthermore, JACO^2's compatibility with various software development environments is a significant advantage. Its widespread use as a research platform has resulted in a wealth of available resources, libraries, and APIs, enabling seamless integration with different software development environments. This compatibility simplifies the development and testing of control systems, potentially broadening the scope of applications. [Maheu et al., 2011; Bjomdahl Mortensen et al., 2021; Hildebrand et al., 2019; Campeau-Lecours et al., 2018; Rakhimkul et al., 2019]

Importantly, the JACO^2 ARM has already been utilized in the context of assisting paralyzed patients, aligning well with the target demographic. This prior application underscores its suitability for projects aimed at this demographic, further validating its selection. [Campeau-Lecours et al., 2018]

Choosing a Camera for a Robotic Autonomous System

When designing an autonomous robotic system intended to assist paralyzed individuals, careful consideration must be given to the choice of the camera, as it plays a pivotal role in the system's ability to interact with the environment. There are several factors to take into account, including the camera's ability to perceive depth, its field of view, and its performance in various lighting conditions. [Bengtson et al., 2020]

Traditionally, robotic grasping is divided into two subproblems: perception and planning. The perceptual component estimates the position and orientation of the object to be grasped, while the planning component reasons about how to move the manipulator into a grasp configuration. However, this approach often assumes that a precise CAD model exists for every object that is to be grasped, which isn't feasible in real-world environments. An alternative approach, grasp detection methods, has been proposed to localize grasp configurations without estimating object pose. These methods use RGBD images or point clouds to detect parts of an image that can be grasped. [ten Pas et al., 2017]

One of the most common sensors used for depth information in robotic systems is some form of stereo vision. The Microsoft Kinect v1 and v2 have been popular choices for many researchers due to their ease of use and acquisition of data. However, these models impose restrictions in terms of possible mounting locations due to their minimum distance of about 0.5 meters. Any object closer than that is unlikely to be captured by the sensor. The Kinect v2, while using a time-of-flight (ToF) camera to acquire depth information, shares the same minimum working distance as the Kinect v1. [Bengtson et al., 2020]

A few studies have used two Kinects, with the second one oriented towards the user of the system for purposes like gesture recognition or detection of the user's face. This approach is designed to move items, like food, to the user's mouth. Those studies that mount their sensor near the end effector primarily rely on customized stereo vision setups. This allows them to control the baseline distance and hence their minimum distance. The only exception is the Carmine camera from PrimeSense, which is marketed as having a minimum distance of about 0.35 m, making it more suitable for such a mounting location than the Kinect. [Bengtson et al., 2020]

Another important consideration in terms of sensor choice is whether active or passive stereo vision is used. Active stereo vision relies on a light source, like infrared light, to actively illuminate

the scene, making it more robust in terms of lacking illumination. On the other hand, passive stereo vision relies solely on the ambient light. However, it can struggle with scenes that lack texture, which can make it harder to recognize the same point in two images. This is necessary for estimating the depth to that point. [Bengtson et al., 2020]

The Intel® RealSense™ D435 camera is a popular choice for such applications due to its wide field of view and global shutter sensor, which are ideal for fast moving applications like robotic navigation and object recognition. The camera offers quality depth perception for a variety of applications, making it suitable for robotics where seeing as much of the scene as possible is vitally important. [Bengtson et al., 2020; Hildebrand et al., 2019; Bjomdahl Mortensen et al., 2021; Rakhimkul et al., 2019]

Furthermore, the D435 is lightweight and comes with highly customizable software, enabling the development of next-generation sensing solutions that can understand and interact with the world in 3D. Different cameras may be more appropriate. However, it's evident that depth-sensing cameras like the Microsoft Kinect and Intel® RealSense™ are popular choices, providing valuable data for robotic perception and planning tasks. One advantage of the Intel® RealSense™ d435 is the ability to have both a RGB image stream along with pointclouds generated from the same camera, this greatly simplifies transforms between the two sensors. The Intel® RealSense™ D435 is chosen as the camera implemented in this system. [Keselman et al., 2017]

5.3 Software components

5.3.1 Operating system choice

In the development of a program designed to control a robotic manipulator semi-autonomously, ROS Noetic on Ubuntu was utilized as the building platform. This decision was influenced by several key factors that made this combination an ideal choice for the project.

ROS Noetic was selected due to its extensive set of software libraries and tools that are specifically designed for building robot applications. It provided a robust framework that facilitated the development of complex robotic behaviors and capabilities. Compatibility with Python 3 was also a significant advantage as Python is widely used in the robotics and AI community due to its simplicity and powerful libraries. ROS has been used in papers within this research field, and has previously been used in this context. [Bjomdahl Mortensen et al., 2021; ten Pas et al., 2017; Rakhimkul et al., 2019]

ROS Noetic allowed for the leveraging of a vast array of pre-existing packages and tools, which significantly accelerated the development process. This included packages for motion planning, sensor integration, and communication protocols, among others. These tools were instrumental in implementing the semi-autonomous control of the robotic manipulator. [ROS, 2007]

The ROS workspace, which served as the repository for all ROS-associated packages and software components, was constructed utilizing Catkin tools. Catkin tools represent a comprehensive suite of utilities that are fundamentally based on the CMake platform. These tools were specifically designed to facilitate the tracking of dependencies, manage version control, and seamlessly integrate disparate packages into a singular workspace. This integration was crucial in maintaining the coherence and functionality of the workspace, ensuring that all components could operate in harmony. The use of Catkin tools, therefore, represented a strategic choice aimed at optimizing the management and operation of the ROS workspace. [ROS, 2007]

Packages

To control the Intel® RealSense™ D435 camera, the RealSense-ROS package was used. This decision was driven by the seamless integration that RealSense-ROS offers with the Intel® RealSense™ D435. The package provides a comprehensive set of drivers that allow for easy access to the camera's depth and color data streams. Additionally, the RealSense-ROS package is fully compatible with the ROS ecosystem, enabling the easy incorporation of the camera data into the broader system. [Intel, 2023]

For controlling the JACO^2 ARM, the Kinova-ROS package was utilized. Kinova-ROS is specifically designed for Kinova's robotic ARMs, including the JACO^2, and provides a robust set of tools for controlling these devices. The package includes capabilities for joint-level control, Cartesian control, and trajectory generation, among others. By using Kinova-ROS, these pre-built functionalities were leveraged, significantly accelerating the development process. [Kinova, 2023]

For object detection, YOLOv7 was used. This decision was based on YOLOv7's impressive performance in terms of speed and accuracy. YOLOv7, or You Only Look Once version 7, is a state-of-the-art object detection system that uses deep learning techniques to identify and classify objects in images. Its ability to process images in real-time made it an ideal choice for the system, where timely detection of objects is critical. [Alexandre, 2023; Wang et al., 2022]

Finally, for grasp detection, the GPD-ROS package was used. GPD is a method for detecting grasp configurations for robotic manipulators in 3D point clouds. The GPD-ROS package integrates this method into the ROS ecosystem, allowing it to be used in conjunction with other tools. GPD's ability to handle cluttered and complex environments made it a valuable addition to the system. [ten Pas, 2021a,b; ten Pas et al., 2017]

These packages were used, along with ROS libraries such as PCL, image transport, and transform-managers were all controlled and initalized from a proprietary control package to handle the flow and communications between nodes in ROS.

5.3.2 Object detection

In the development of semi-autonomous assistive robotic systems, the selection of an appropriate object detection algorithm is crucial. The YOLO (You Only Look Once) algorithm was selected for object detection due to its demonstrated speed and accuracy. The fundamental principle of YOLO, which treats object detection as a regression problem, was found to be advantageous. This approach allows a single neural network to predict bounding boxes and class probabilities directly from full images in one evaluation. This is in contrast to traditional methods that use a two-step process of first selecting regions of interest and then classifying those regions. The ability of YOLO to perform both tasks simultaneously resulted in a significantly faster performance. [Wang et al., 2022]

In the context of robotics vision, the choice of object detection algorithm was found to greatly influence the performance of the system. While YOLO was recognized for its speed and accuracy, it was noted that other algorithms like CenterNet or Faster R-CNN also offered unique advantages. However, the computational expense and slower speed of Faster R-CNN, especially in cases where objects were small or densely packed, led to the preference for YOLO. Similarly, while CenterNet, which detects objects as points rather than bounding boxes, could achieve comparable accuracy to the other methods, its simpler and more flexible architecture did not outweigh the benefits of YOLO. [Rakhimkul et al., 2019]

The YOLO algorithm, which has undergone several iterations, each introducing improvements over the previous ones, was carefully evaluated. The latest version at the time, YOLOv7, was chosen

for its compatibility with ROS, making it particularly suitable for robotics vision applications. YOLOv7 was found to introduce several improvements over its predecessors, including a trainable bag-of-freebies that set a new state-of-the-art for real-time object detectors. [Alexandre, 2023; Wang et al., 2022; Rakhimkul et al., 2019]

The performance of the YOLOv7 model was evaluated, achieving an APtest of 51.4%, an AP50test of 69.7%, and an AP75test of 55.9% on the MS COCO dataset with a test size of 640. It was observed to run at 161 fps with a batch size of 1 and took an average time of 2.8 ms with a batch size of 32. other versions of YOLOv7, such as YOLOv7-X and YOLOv7-W6, were also considered, offering different trade-offs between speed and accuracy. [Wang et al., 2022]

YOLO is optimized for real-time object detection. The algorithm's design allows it to detect objects in a single pass, contributing to its high processing speed. This attribute was found to be particularly beneficial in the context of assistive robotic systems, where rapid identification and response to environmental changes are essential for effective operation. [Wang et al., 2022; Rakhimkul et al., 2019]

The balance between accuracy and speed offered by YOLO was also noted. While higher levels of accuracy might be achieved with other object detection algorithms, these often come at the expense of processing speed. YOLO, however, demonstrated a favorable trade-off between robust object detection performance and speed. This balance is crucial in real-time applications such as the control of an ARM, where delays in object detection could lead to slower response times. [Rakhimkul et al., 2019]

Another significant advantage of YOLO is its ability to detect multiple objects within a single image. This capability is vital for assistive robotic systems, which frequently operate in complex environments with multiple objects. YOLO's ability to identify and differentiate between multiple objects simultaneously enhances the system's decision-making process regarding environmental interactions. [Wang et al., 2022; Rakhimkul et al., 2019]

5.3.3 Grasp pose estimation software

In the field of robotic manipulation, several alternative grasp pose algorithms have been proposed. Some of these include the use of model fitting techniques such as Random Sample Consensus (RANSAC), Speeded-Up Robust Features (SURF), and policy-blending formalisms for shared control. Other approaches include the use of vision-guided autonomous robotics, goal-predictive robotic teleoperation, and affordance detection of tool parts from geometric features. Fast graspability evaluation on single depth maps for bin picking with general grippers has also been proposed. [Rakhimkul et al., 2019; Bengtson et al., 2020]

However, these alternative algorithms were not chosen for several reasons. Firstly, many of these algorithms are based on geometric features and require explicit model fitting, which can be computationally expensive and may not generalize well to a wide variety of objects. Secondly, these algorithms often require a high level of precision in the sensor data, which may not always be achievable in real-world scenarios.

In contrast, a machine learning-based approach like GPD offers several advantages over these standard geometric approaches. One of the key benefits is its ability to handle arbitrary objects, i.e., objects never encountered by the system before. This is achieved by discarding the notion of detecting separate objects and instead relying on detecting good grasping points on an arbitrary object. [ten Pas et al., 2017, 2021]

GPD's methodology is a robust approach for robotic grasping in cluttered environments. It is a

computational package designed to detect 6-DOF grasp poses in 3D point clouds for a 2-finger robot hand, such as a parallel jaw gripper. The GPD process consists of two main steps: sampling a large number of grasp candidates and classifying these candidates as viable grasps or not. [ten Pas et al., 2017, 2021]

The heart of the GPD system is a Convolutional Neural Network (CNN), which is trained to predict the success of a grasp given a local image patch extracted from a depth image of the scene. The architecture of the network comprises three convolutional layers, each succeeded by a max-pooling layer, and two fully connected layers. The input to the network is a 60x60 depth image patch, and the output is a binary label indicating the predicted success or failure of the corresponding grasp candidate. [ten Pas et al., 2017, 2021]

Grasp candidates are generated by sampling a large number of antipodal pairs of points in the point cloud. These candidates are then filtered based on certain heuristics, such as the distance between the points and the curvature of the surface at the points. The remaining candidates are scored using the trained neural network, with the candidate possessing the highest score chosen for execution. [ten Pas et al., 2017, 2021]

Machine learning-based approaches can leverage large amounts of training data to learn complex patterns and relationships, allowing them to generalize well to new objects and scenarios. They are also typically more robust to noise and inaccuracies in the sensor data. [ten Pas et al., 2017, 2021]

For instance, GPD uses a CNN to infer grasping points, which can be significantly faster than traditional geometric approaches. CNNs are highly parallelizable, enabling them to take advantage of modern GPU hardware for accelerated computations [37, 38]. [ten Pas et al., 2017, 2021]

Given the variety of packages available for grasp pose estimation in robotic systems, the selection of GPD and its ROS wrapper, GPD-ROS, can be attributed to several factors:

Versatility: GPD is designed to detect 6-DoF grasp poses for a 2-finger robot hand (e.g., a parallel jaw gripper) in 3D point clouds. This makes it versatile and applicable to a wide range of robotic systems and environments. [ten Pas et al., 2017, 2021]

Integration with ROS: The availability of a ROS wrapper (GPD-ROS) allows for easy integration of GPD into a ROS-based robotic system. This facilitates communication with other components of the system and streamlines the development process. [ten Pas, 2021b]

Real-time Operation: GPD is capable of operating in real-time, which is crucial for assistive robotic systems that need to respond quickly to changes in the environment. [ten Pas et al., 2017, 2021]

5.4 Ros Integration

5.4.1 Realsense-ROS

The RealSense-ROS package was incorporated into the system through the application of the ROS legacy wrapper, which is compatible with ROS Noetic. This wrapper was specifically designed to facilitate the use of the Intel® RealSense™ SDK within the ROS environment. The default configuration of this package was already primed to provide static transforms and facilitate inter-sensor communication, thereby simplifying the integration process. [Intel, 2023]

The default configuration was utilized with minor adjustments to the settings to ensure the publication of the RGB image, depth map, and point cloud. Additionally, the alignment between the RGB image and the depth map was enabled, enhancing the accuracy and consistency of the data.

5.4.2 YOLOv7 ROS

The YOLOv7 ROS package was integrated into the system by leveraging its lightweight implementation in the ROS node style, which allows for easy execution of inference. This ROS package acts as a wrapper around the YOLO implementation. This implementation is based on the original YOLO article publication. It includes various versions of YOLOv7, each with different performance characteristics, such as YOLOv7-X, YOLOv7-W6, YOLOv7-E6, and others. The performance of these models is measured in terms of Average Precision (AP) and frames per second (fps), with YOLOv7 achieving an AP of 51.4% and 161 fps on the MS COCO dataset.[Wang et al., 2022; Alexandre, 2023]

The configuration of the YOLOv7 ROS package involved setting up the weights and classes. The weights for YOLOv7 were downloaded from the official repository. The classes were stored in a simple text file, with each line representing a class. The weights were based on YOLOv7 trained on the COCO dataset. [Wang, 2023; Alexandre, 2023]

The launch file for the YOLOv7 ROS node was configured with several parameters. These included the paths to the weights and class labels, the image topic name to subscribe to, the confidence threshold, the intersection over union threshold, the queue size for publishing, the image size for resizing input images, a flag for visualizing detections, the torch device (cuda or cpu), and the node frequency. Object detection is computationally expensive, and thus such systems often leverage cuda-cores from Nvidia based graphics cards. Unfortunately, a portable computer with such a graphics card was not available, therefore CPU-based processing was used.[Alexandre, 2023]

The ROS package was a lightweight implementation of YOLOv7 in ROS node style, allowing users to run inference easily. The package includes a launch file that can be configured with various parameters, including paths to the weights and class labels, the image topic name to subscribe to, the confidence threshold, the intersection over union threshold, and others. The package also provides the option to visualize detections and select the processing device (CUDA or CPU). [Alexandre, 2023]

The integration of the YOLOv7 ROS package into the system allowed for real-time object detection, enhancing the system's capabilities for autonomous navigation and interaction with the environment. The ROS package, combined with the YOLOv7 implementation, provides a robust and efficient solution for object detection tasks. [Wang, 2023; Alexandre, 2023]

5.4.3 GPD-ROS

GPD and GPD-ROS were used, although various changes were made to facilitate integration with the system. GPD is a standalone program able to run in isolation, where the GPD-ROS package is a part of the catkin workspace and provides a wrapper to interface between GPD and the ROS system. [ten Pas, 2021a,b]

GPD is designed to handle a two-fingered end effector, specifically setup for the baxter robot. As such the config files were changed to better match the hand geometry of the JACO^2 3-fingered end effector which includes detailing finger length, hand width, height, and max aperture. Furthermore, the number of generated grasps was changed from the default 300 to 3000. The color-aligned depth map, transformed to a pointcloud was sent as the input to GPD, along with a sample point corresponding to a point on the object on which to focus the grasps on. The sample point was the center point of the bounding box corresponding to the object the user selected. When GPD-ROS was run, a resulting number of grasp poses were generated containing the position of the grasp along with orientational data from the approach, binormal and axis vectors. Furthermore, a score is attached indicating how good the grasp is based on an internal scoring system within GPD. A

width variable is also sent indication how much the gripper is to be closed to grasp the object. An example of the unfiltered output of GPD can be seen on 5.3. [ten Pas, 2021a,b]

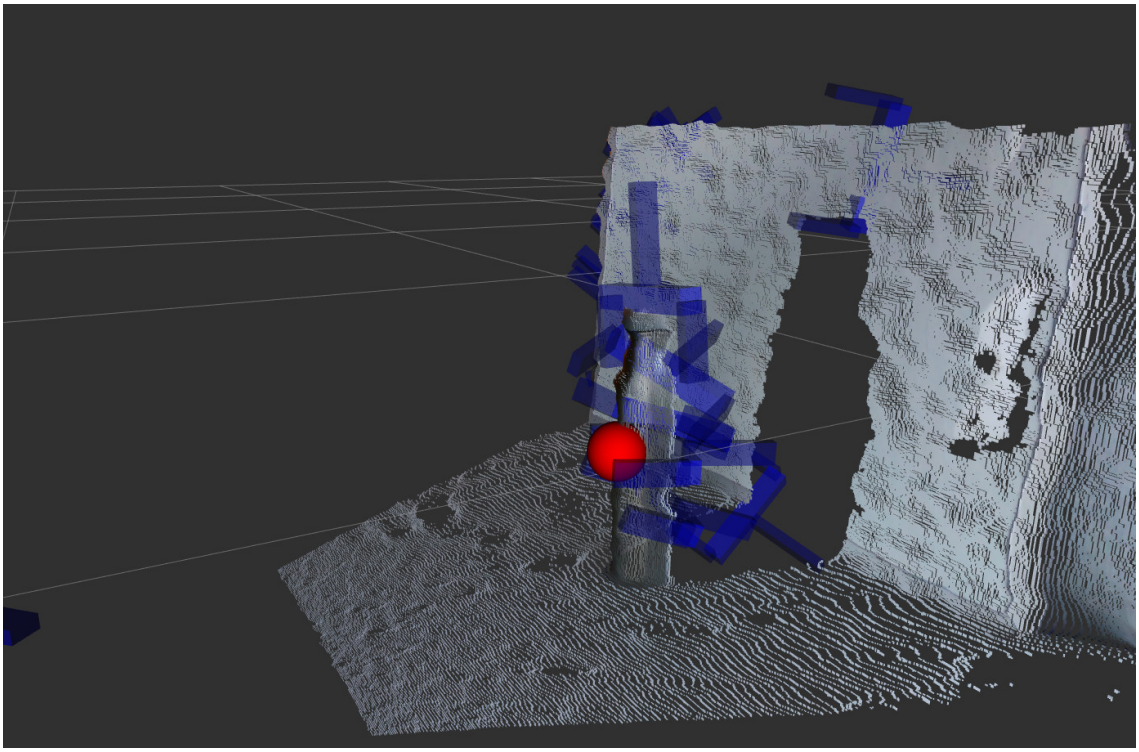


Figure 5.3. Pointcloud visualization with a bottle object and grasp poses. The red marker represents the center point of the bottle, while multiple blue grasp poses are generated by GPD for the object

The package comes with weight files for two different input representations for the neural network: 3 or 15 channels. The default is 15 channels, which include depth values, surface normals, and curvature values, resulting in superior performance compared to using only 3 channels (depth values), at the cost of higher compute times. [ten Pas, 2021a,b]

The GPD system is configured through a set of parameters defined in the configuration files. These parameters include the geometry of the robot hand (finger width, outer diameter, hand depth, hand height, and initial bite), the parameters of the grasp descriptor (volume width, depth, and height, image size, and number of image channels), the preprocessing parameters for the point cloud (voxelization, outlier removal, workspace, camera position, and sampling above the plane), the parameters for grasp candidate generation (number of samples, number of threads, neighborhood search radius, number of orientations, number of finger placements, hand axes, and hand deepening), the parameters for candidate filtering (minimum and maximum aperture, workspace for grasps, filtering based on approach direction, direction, and angle threshold), the parameters for grasp clustering (minimum number of inliers per cluster), and the number of selected grasps. The system can be adjusted to work with different robots and environments by changing these parameters. [ten Pas, 2021a,b]

The GPD system is configured through a set of parameters defined in the configuration files. These parameters include:

1. Geometry of the robot hand:
    - Finger width
    - Outer diameter

- Hand depth
- Hand height
- Initial bite

2. Parameters of the grasp descriptor:
    - Volume width
    - Depth
    - Height
    - Image size
    - Number of image channels
3. Preprocessing parameters for the point cloud:
    - Voxelization
    - Outlier removal
    - Workspace
    - Camera position
    - Sampling above the plane
4. Parameters for grasp candidate generation:
    - Number of samples
    - Number of cpu threads
    - Neighborhood search radius
    - Number of orientations
    - Number of finger placements
    - Hand axes
    - Hand deepening
5. Parameters for candidate filtering:
    - Minimum and maximum aperture
    - Workspace for grasps
    - Filtering based on approach direction
    - Direction
    - Angle threshold
6. Parameters for grasp clustering:
    - Minimum number of inliers per cluster
7. Number of selected grasps

The system can be adjusted to work with different robots and environments by changing these parameters. [ten Pas, 2021a,b]

5.4.4 Kinova-ROS

The integration of the Kinova-ROS package from the Kinova-ROS repository played a vital role in the project, providing essential functionalities for controlling the JACO^2 robotic manipulator. The package offered several features that facilitated the integration and control of the robot within the system. [Kinova, 2023]

One significant advantage of the Kinova-ROS package was the inclusion of pre-configured URDF files and setup for the JACO^2 robotic manipulator. These files described the robot's kinematic structure, joint limits, and other necessary parameters, enabling accurate representation and control of the robot's movements. This feature streamlined the integration process by providing a ready-to-use setup specific to the JACO^2 ARM. [Kinova, 2023]

Furthermore, the Kinova-ROS package is integrated with MoveIt, a widely used motion planning framework in ROS. MoveIt provided advanced motion planning capabilities, such as inverse kinematics and trajectory generation, which were essential for controlling the robot's movements. This integration enabled the implementation of complex motion planning algorithms and facilitated the execution of various motion control tasks. [Kinova, 2023]

In addition to the core functionalities, the Kinova-ROS package also included pre-made demo files. These demo files showcased different control modes, such as Cartesian and joint angle control, allowing users to quickly understand and utilize the capabilities of the JACO^2 ARM. These demos served as valuable starting points for developing custom control strategies and provided a basis for further customization and refinement. [Kinova, 2023]

Overall, the Kinova-ROS package provided a comprehensive set of tools and resources for integrating and controlling the JACO^2 robotic manipulator within the system. The package's pre-configured URDF files, compatibility with RViz, integration with MoveIt, and inclusion of demo files significantly simplified the integration process and enabled efficient control of the robot's movements. [Kinova, 2023]

5.4.5 Control package

To accommodate functionalities that were not directly accessible through the pre-existing packages, several nodes were implemented. These nodes were designed to bridge the gap between the existing capabilities of the included packages and the additional requirements of the system. This strategic implementation ensured that the system could perform tasks beyond the scope of the pre-packaged functionalities, thereby enhancing the overall versatility and effectiveness of the system.

5.4.6 User interactivity

A custom node was implemented to process the bounding boxes generated by YOLOv7-ROS. These bounding boxes are represented by center point (x, y) coordinates and width/height (w, h) values. The node operates by reading the bounding boxes and the RGB camera feed as inputs. Subsequently, the bounding boxes are superimposed onto the image, and each box is assigned a unique number to facilitate object identification. The user is then prompted to select an object by pressing keys 1 to 9 on the keyboard, corresponding to the desired object's assigned number. Alternatively, objects can be deselected by pressing the number 0.

Once an object is selected, the center point coordinates of the chosen bounding box are published to a ROS topic. Simultaneously, the bounding box of the selected object undergoes a color change on the display, indicating that the object has been chosen.

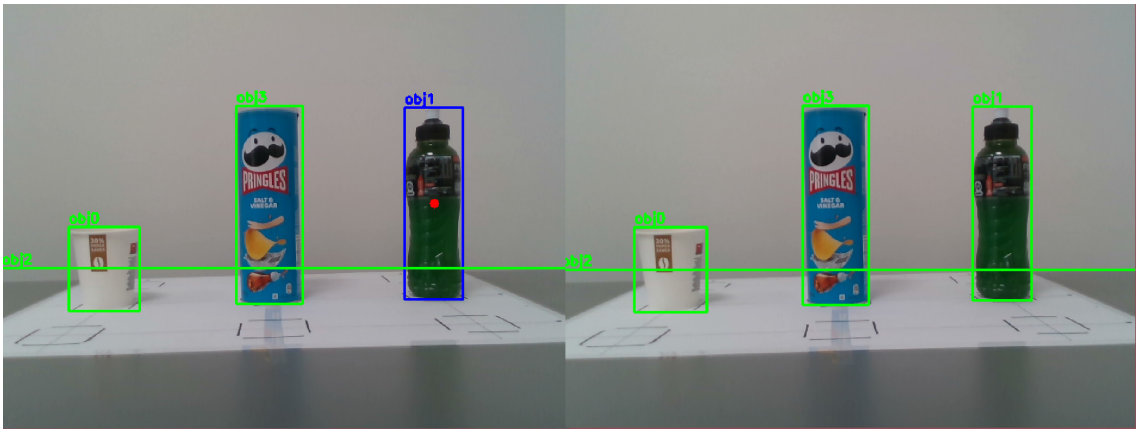


Figure 5.4. In the image on the right, a scene is depicted with three bottles placed on a table. No object has been selected, and the objects are displayed without any distinguishing features. Conversely, the image on the right showcases the same scene, but with a notable difference. Object 1, the bottle, has been specifically chosen. This selection is denoted by a distinct visual indicator, such as a change in the color or highlighting of the bounding box surrounding object 1.

5.4.7 Spatial center point estimation

The obtained center point resulting from the user's selection represents an (x, y) coordinate located at the center of the bounding box surrounding the specified object. This particular point serves as a reference point within a three-dimensional (3D) space, necessitating a series of transformations. Initially, the depth camera stream, which is aligned with the color image, is utilized. The (x, y) coordinates obtained from the center box are applied to the depth image, allowing for the extraction of the corresponding z-value from this two-dimensional (2D) point. Consequently, both the (x, y) coordinates and the distance coordinate (z) are obtained. To further process the data, the intrinsic camera information, in conjunction with the transform tree, is employed to remap the depth map into a point cloud. This is visualized on figure 5.5.

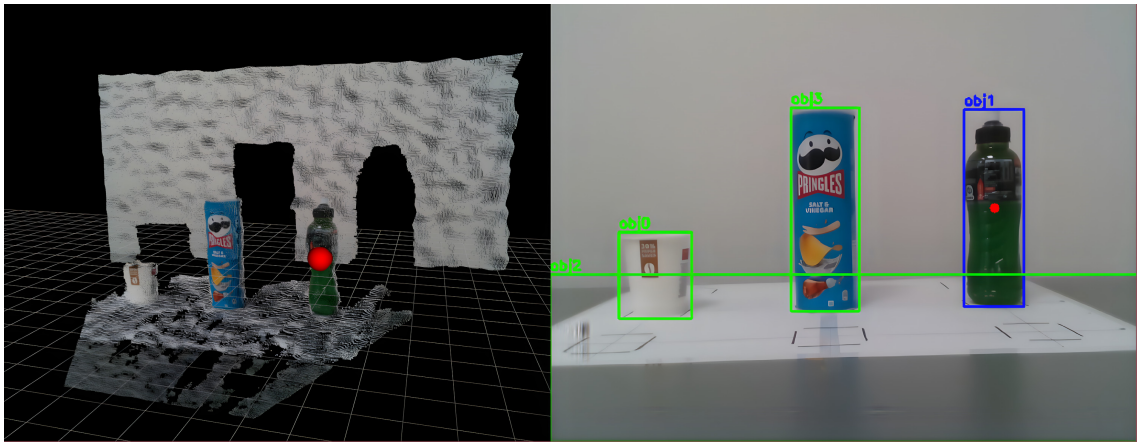


Figure 5.5. On the left side, a 3D point cloud is displayed, consisting of numerous points in space. Among them, a single point is marked in red, representing the resulting center point derived from the user's selection. On the right side, a 2D image is shown, capturing a scene. Similarly, the red marker signifies the same center point as in the 3D point cloud. This center point is obtained by determining the xy coordinates at the center of the bounding box surrounding the object of interest. To enable referencing this point in 3D space, a series of transforms are applied. Initially, the depth camera stream, aligned with the color image, is utilized. The xy coordinates from the center box are applied to the depth image, allowing the extraction of the z value from this 2D point. Consequently, an xy coordinate and a distance coordinate (z) are obtained. By utilizing the intrinsic camera information and the transform tree, the depth map is remapped into a comprehensive point cloud representation.)

The intrinsic camera info, that defines transforms between the different camera reference frames are defined in the transform tree in ROS. A map of the systems transform tree is detailed in Appendix C.

5.4.8 Grasp filtering

The GPD algorithm yielded an output of 3000 grasp poses, arranged in descending order of their respective scores, with the highest scoring pose occupying the first position. It is important to note that GPD does not consider environmental factors such as the proximity of other objects or the surface of the table on which the objects are placed. Moreover, GPD tends to treat all angles as valid, resulting in a significant portion of the poses attempting to grasp the object from behind or from the side.

The point cloud, positioned at the base of the robot, can only form a surface on the facets that face it. Consequently, objects lack spatial information on their sides and rear, as illustrated in Figure 5.3. Although the use of multiple depth cameras or the placement of the camera on the end

effector might mitigate this issue, such solutions were not feasible within the scope of this project. Therefore, an alternative method was implemented to address this concern.

Despite the fact that GPD generates a substantial number of its grasps on the object based on the center point, some poses still attempt to grasp other objects in the scene, such as the table. Consequently, the first filtering constraint was implemented, stipulating that the position of the grasp had to be within 2 cm of the center point. As a result, only grasps in close proximity to the center point were included. The second filtering constraint pertained to the approach.

A unit vector, with its direction from the origin to the center point, was computed. Similarly, the approach vector from the grasp pose was used to derive a unit vector encoding its direction. Grasps were thus filtered to only include those that had an approach within a 45-degree cone, originating at the origin and projecting outwards in the direction of the center point. This resulted in a preference for including grasps that attempted to grasp the object from the front.

Grasps that complied with the position and approach constraints were then included. Subsequently, among all the included grasps, the highest scoring grasp was selected as the optimal grasp candidate and was then used for execution.

The mathematical representation of the unit vector calculation and the filtering process can be represented as follows:

Let $\vec{O}$ be the origin, $\vec{C}$ be the center point, and $\vec{G}$ be the grasp pose. The unit vector $\hat{u}$ from the origin to the center point is calculated as:

$$\hat{u} = \frac{\vec{C} - \vec{O}}{||\vec{C} - \vec{O}||}$$

Similarly, the unit vector $\hat{v}$ encoding the direction of the grasp pose is calculated as:

$$\hat{v} = \frac{\vec{G} - \vec{O}}{||\vec{G} - \vec{O}||}$$

The grasp poses are then filtered based on the angle θ between $\hat{u}$ and $\hat{v}$, which is calculated using the dot product:

$$\theta = \cos^{-1}(\hat{u} \cdot \hat{v})$$

Only those grasp poses for which θ is less than or equal to 45 degrees are included.

The final step in the process involved the selection of the optimal grasp candidate from the filtered set of grasp poses that meet the position and approach angle constraints. This selection was based on the scoring system inherent to the GPD algorithm. The grasp pose with the highest score, indicating the highest probability of a successful grasp, was chosen as the optimal grasp candidate.

The scoring filtering can be mathematically represented as follows:

Let S be the set of scores for each grasp pose, and T be the threshold score. The filtered set of scores S' is given by:

$$S' = \{s \in S : s \geq T\}$$

The optimal grasp candidate is then chosen as the grasp pose associated with the highest score in S':

$$s_{\text{opt}} = \max(S')$$

5.4.9 Axis Orientation and Spatial Transforms

In the domain of spatial manipulation, particularly in the context of robotics, the reference frame associated with a component is of paramount importance. It serves as the contextual basis for interpreting xyz coordinates, thereby providing a spatial understanding of the component's position and orientation. ROS employs a data structure known as a transform tree to establish connections between various reference frames. This is achieved by providing transformations between them, typically expressed as an xyz translation coupled with a quaternion encapsulating rotation. A detailed connectivity map of the transform tree used in this system can be seen in Appendix C.

The RealSense-ROS package provides a comprehensive transform tree that outlines all transformations between the internal sensors. This is instrumental in aligning the optical and infrared sensors, thereby facilitating the alignment of the depth map with the RGB sensor. Within the RealSense-ROS framework, a master reference frame, referred to as "camera-link", is defined. This reference frame is strategically positioned at the left infrared sensor and serves as the origin point relative to which all other sensors are oriented.

In a similar vein, the Kinova-ROS package defines a transform tree originating from its master reference frame, "world". This transform tree establishes a link from the "world" reference frame to the base of the robot, extending all the way to the end effector. However, these packages do not inherently contain a transform that links the "camera-link" to "world" reference frames. To bridge this gap, a transform was manually computed and published, thereby enabling transformations between camera references and robot references.

When employing a JACO^2 ARM in conjunction with the Kinova-ROS package, the "world" reference frame is defined from the perspective of facing the backplate of the JACO^2 as follows:

a. X - left
b. Y - backward
c. Z - up

Conversely, the "camera-link" reference frame, defined from the perspective of standing behind the camera facing the direction of its sensors, is as follows:

a. X - right
b. Y - down
c. Z - forward

Given these definitions, a quaternion was computed and published to establish a link between the camera reference frame and the robot's reference frame. The quaternion is defined as:

a. X - 0.0
b. Y - 0.0
c. Z - -0.70714
d. W - 0.70714

The GPD package utilizes the depth camera as its primary input. The grasp poses, initially in the depth camera's reference frame, are first transformed to the "camera-link" reference frame. Subsequently, these poses are transformed using the manually computed quaternion to realign

them to the world reference frame. An illustrative representation of the orientational relationships between the world frame and the camera-link can be seen in Figure 5.6.

The transformation from the "camera-link" to the world reference frame using the quaternion can be mathematically represented as:

$$\begin{bmatrix} X'' \\ Y'' \\ Z'' \\ 1 \end{bmatrix} = \begin{bmatrix} 1-2Y^2-2Z^2 & 2XY-2ZW & 2XZ+2YW & 0 \\ 2XY+2ZW & 1-2X^2-2Z^2 & 2YZ-2XW & 0 \\ 2XZ-2YW & 2YZ+2XW & 1-2X^2-2Y^2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X' \\ Y' \\ Z' \\ 1 \end{bmatrix}$$

In this equation, X, Y, Z, and W represent the components of the quaternion. The resulting X'', Y'', and Z'' denote the coordinates of the grasp poses in the world reference frame. It's important to note that ROS is capable of reversing this transformation, thereby enabling transformations from the world reference frame back to the "camera-link" reference frame.

This detailed exploration of spatial transforms and axis orientation underscores the intricate interplay between various reference frames in the context of robotic manipulation. The ability to accurately compute and apply these transformations is crucial for the successful execution of tasks such as grasp pose detection. Future work could delve deeper into optimizing these transformations, potentially exploring automated methods for computing necessary transforms.

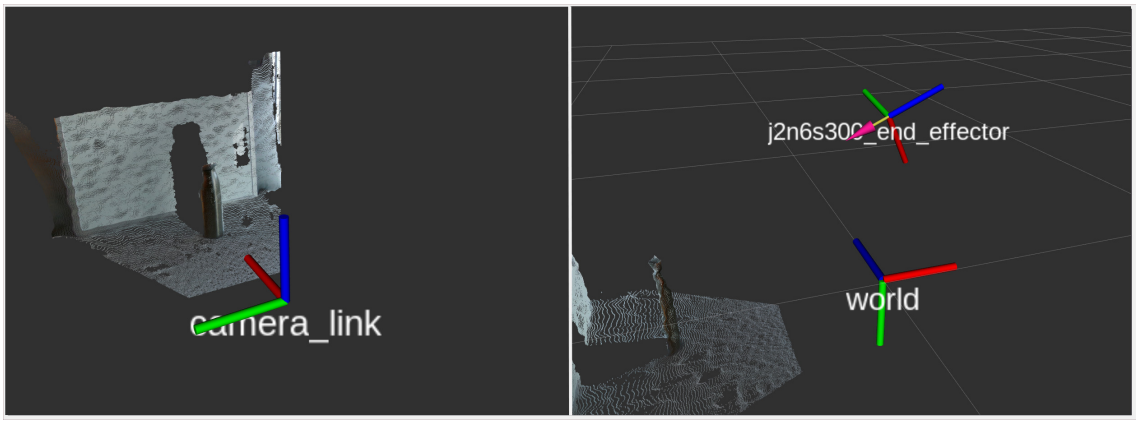


Figure 5.6. Orientation visualization between the 'world' reference frame and the 'camera-link' in the context of spatial manipulation. The manually calculated quaternion is utilized to align the grasp poses from the depth camera to the 'world' reference frame in the JACO^2 ARM. X is the red axis, y is the green axis and z is the blue axis. This defines a standard right-handed coordinate frame.

5.4.10 Interpretation of Grasp Pose and Execution of Grasp

Upon the completion of filtering and transformation of the grasp poses to align with the world reference frame, the grasp pose message is interpreted into a command that the robot can execute.

The magnitude of the position directional vector is calculated based on the grasp pose. This magnitude is a measure of the distance from the origin to the grasp pose position in 3D space. If this magnitude is less than 0.1, indicating that the grasp pose is within a 10cm sphere around the camera, the pose is deemed inaccessible and is not executed.

$$\text{grasp\_point\_magnitude} = \sqrt{\text{position.x}^2 + \text{position.y}^2 + \text{position.z}^2}$$

Subsequently, the position of the grasp pose is adjusted slightly to account for the physical offset between the camera and the robot base frame. The camera offset is applied to the position of the grasp pose to account for the physical offset between the camera and the robot base frame. The offset is defined as a translation vector $\begin{bmatrix} -0.03 \\ -0.1 \\ 0.09 \end{bmatrix}$ in the x, y, and z directions, respectively. This offset is added to the position of the grasp pose as follows:

$$\begin{bmatrix} \text{position.x}' \\ \text{position.y}' \\ \text{position.z}' \end{bmatrix} = \begin{bmatrix} \text{position.x} \\ \text{position.y} \\ \text{position.z} \end{bmatrix} + \begin{bmatrix} -0.03 \\ -0.1 \\ 0.09 \end{bmatrix}$$

A transformation matrix is constructed using the approach, binormal, and axis vectors of the grasp pose. The 4x4 transformation matrix is constructed as follows:

$$\text{transformation\_matrix} = \begin{bmatrix} \text{approach.x} & \text{binormal.x} & \text{axis.x} & 0 \\ \text{approach.y} & \text{binormal.y} & \text{axis.y} & 0 \\ \text{approach.z} & \text{binormal.z} & \text{axis.z} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.1}$$

The grasp pose, now fully defined in terms of position and orientation, is packaged into a message. This message is then passed to a function responsible for commanding the robot to execute the grasp.

The execution function begins by opening the robot's gripper. It then converts the quaternion orientation of the grasp pose into Euler angles, converts these angles from radians to degreesr. These values, along with the position of the grasp pose, are assembled into a command. This command instructs the robot to move its end effector to the desired grasp pose and orientation.

The command to move the robot's end effector to the desired grasp pose and orientation is assembled using the adjusted position of the grasp pose (x', y', z') and the reordered Euler angles $(\theta_x, \theta_y, \theta_z)$. This command can be represented as:

rosrun kinova_demo pose_action_client.py j2n7s300 mdeg – $x'\, y'\, z'\, \theta_x\, \theta_y\, \theta_z$

after the grasp has been executed the end effector closes its grippers to grasp the object.

where:

- x', y', and z' represent the adjusted position of the grasp pose,
- θ_x, θ_y, and θ_z represent the reordered Euler angles.

5.5 Testing

To ensure the system meets the requirements set out during the design phase, testing was conducted. The objective of these tests was to evaluate the system's performance and verify its alignment with the specified requirements. By subjecting the system to testing procedures, we aimed to validate its functionality, reliability, and overall capability.

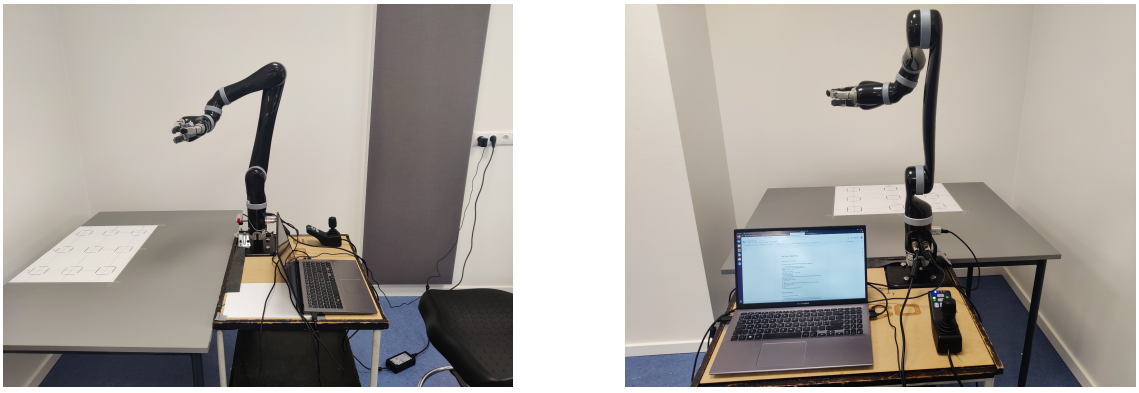
5.5.1 Materials

The following materials and instruments were utilized for the initial setup of the testing environment:

- Intel RealSense D435 RGBD camera.
- JACO^2 ARM with a 3-fingered end effector.
- Power supply for JACO^2.
- Metal base frame for JACO^2.
- Joystick for JACO^2.
- Mobile platform (table) with dimensions: Length: 68 cm, Width: 53 cm, Height: 76 cm.
- Camera stand (a U-shaped metal piece).
- Two USB 3.0 micro cables.
- Table with dimensions: Length: 120 cm, Width: 85 cm, Height: 74 cm.
- Computer: Asus vivobook, Intel® Core™ i5-10210U CPU @ 1.60GHz × 8.
- Rubber band.
- Tape.
- Paper.
- Three objects: one bottle with green water, one Pringles can, and one cup.
- Measuring tape.

5.5.2 Test Setup Procedure

The test setup for the project was conducted following the steps outlined below:



(a) Side view of the experimental setup. **(b)** Rear view of the experimental setup.

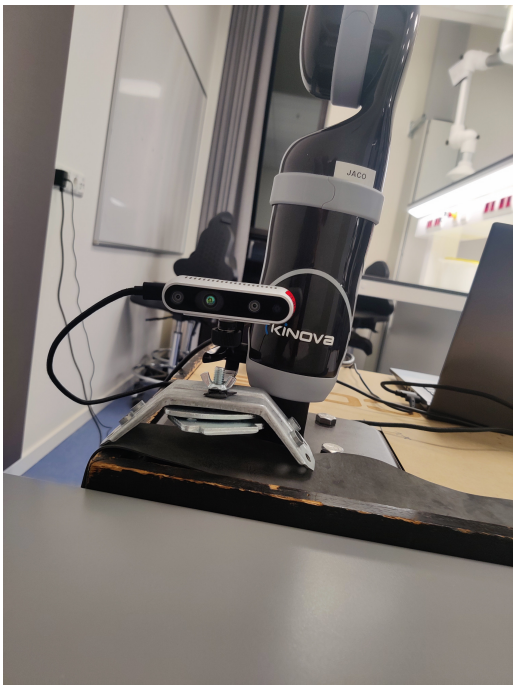
Figure 5.7. Experimental setup for testing the autonomous robotic system. The JACO^2 ARM is mounted on a mobile platform, while the Intel RealSense D435 camera is securely attached to a camera stand. A 3x3 grid is marked on the table surface, providing predefined positions for object manipulation.

The JACO^2 ARM, equipped with a 3-fingered end effector, was securely mounted on a mobile platform to simulate it was mounted on a wheelchair as illustrated in Figure 5.7. This provided a stable platform for the ARM during the experiments. The power supply and joystick for the JACO^2 were connected and placed on the table of the mobile platform for convenient access and control throughout the test sessions.

The Intel RealSense D435 camera was securely mounted on a camera stand (a U-shaped metal piece). To ensure a stable attachment, the Intel RealSense D435 camera was connected to the JACO^2 base, and a rubber band was placed under the camera stand to prevent unintended displacement as depicted in Figure 5.8. This setup allowed the camera to capture both RGB and depth information, which were crucial for the system's perception and sensing capabilities.

The camera stand was positioned in a specific configuration relative to the JACO^2. From the robot's perspective, the camera stand was placed 10 cm forward, 8 cm above, and 4 cm to the right from the base origin point of JACO^2 as seen in Figure 5.8. It's important to note that this configuration introduced a non-zero rotation along the XYZ-axis, which was not accounted for in this setup.

(a) Rear view of the camera setup showing the secure attachment of the Intel RealSense D435 camera to the JACO^2 base and the camera stand configuration.



(b) Front view of the camera setup highlighting the position and orientation of the Intel RealSense D435 camera relative to the JACO^2 base.

Figure 5.8. Figure illustrating the camera setup for the experimental setup, showcasing the secure attachment of the Intel RealSense D435 camera to the JACO^2 base and the specific positioning relative to the ARM.

The Intel RealSense D435 camera and the JACO^2 ARM were connected to the computer using two USB 3.0 micro USB cables. These high-speed data transfer cables ensured real-time perception and control capabilities. It was verified that the computer had at least two available USB ports, both compatible with USB 3.0 standards.

The computer, responsible for system monitoring and control, was positioned adjacent to where the JACO^2 ARM was mounted as illustrated in Figure 5.7. This arrangement allowed easy access for monitoring and controlling the system during the experimental procedures.

To enable object manipulation and movement, a 3x3 grid was created by marking it on a piece of paper and securely taping it to the table's surface in front of the mobile platform. The grid was positioned approximately 40 cm away from the baseplate of the JACO^2, as observed from the JACO^2 base panel perspective. This grid served as the designated working area where the robot could engage with objects and perform manipulation tasks. Special attention was given to ensuring

the grid's visibility and alignment, ensuring it was clearly visible and accurately positioned.

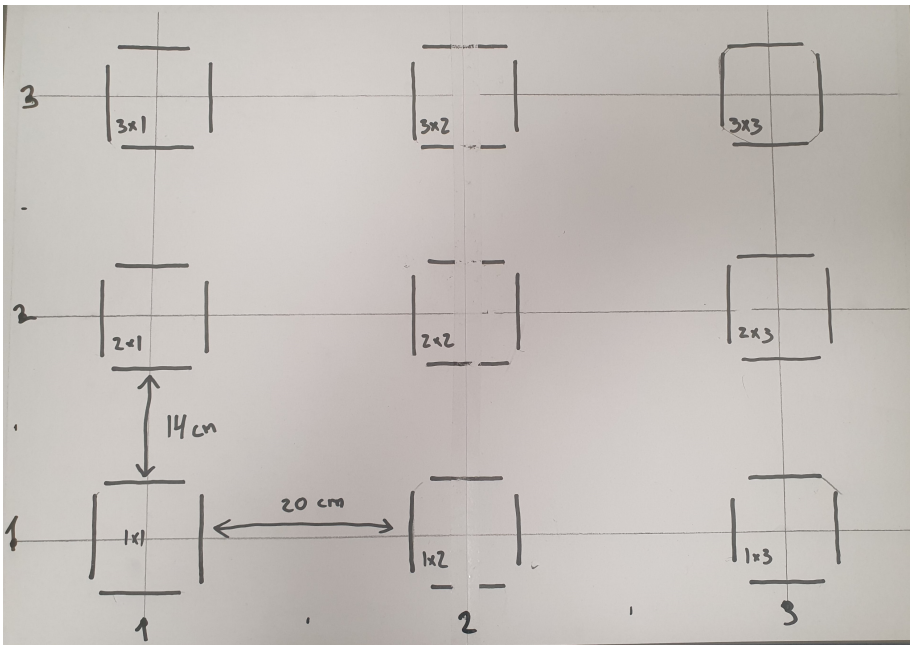


Figure 5.9. Illustration of the 3x3 grid, where the center points of adjacent quadrants are spaced 20 cm apart in the x-direction, and there is a separation of 14 cm between the center points of neighboring quadrants in the y-direction.

The distances from the left infrared sensor of the Intel RealSense D435 camera's reference frame (camera_link_frame) to the 9 grid positions were measured and recorded.

The system underwent testing to assess its spatial accuracy and object selection capabilities. The spatial accuracy test involved estimating the position of a target object within a 3x3 grid using the depth camera. The system processed the camera feed, mapped the estimated points to the depth map, generated a point cloud, and compared the results to real-world measurements.

In order to evaluate the system's compliance with the Object Selection Requirements, a test was conducted. This test focused on assessing the system's ability to enable the user to select objects in the scene, provide a clear and intuitive interface for object selection, and visually indicate or highlight the selected objects on the screen. Through this test, the system's performance in meeting these specific requirements was examined and analyzed. The test was conducted utilizing the keyboard/keypad of the computer. As the Itongue interface was not implemented in the prototype of the proposed system, this alternative input method was employed.

For the Autonomous Grasp Control Requirements, a test was conducted to evaluate the system's capacity to autonomously control the manipulator to perform grasp tasks for selected objects, seamlessly transition between user control and autonomous control during grasp execution, and effectively communicate to the user when the control is handed over or returned. By conducting this test, the system's level of compliance with these requirements was assessed and validated.

To assess the Grasp Pose Generation Requirements, a comprehensive test was executed. This test was designed to test the system's ability to utilize the depth camera feed to generate feasible and appropriate grasp poses for the detected objects, ensure that the grasp pose generation algorithm considers the camera input, object properties, and ARM's workspace, and provide accurate and timely grasp pose commands to the JACO^2 ARM for successful grasping. The results of this test were carefully analyzed to determine the system's adherence to these specific requirements.

By conducting tests for these requirements, a thorough evaluation of the system's performance in meeting these requirements can be obtained, providing valuable insights into its capabilities and functionality. For detailed documentation of the test protocols, the reader is directed to Appendix D.

Results 6

6.1 Results

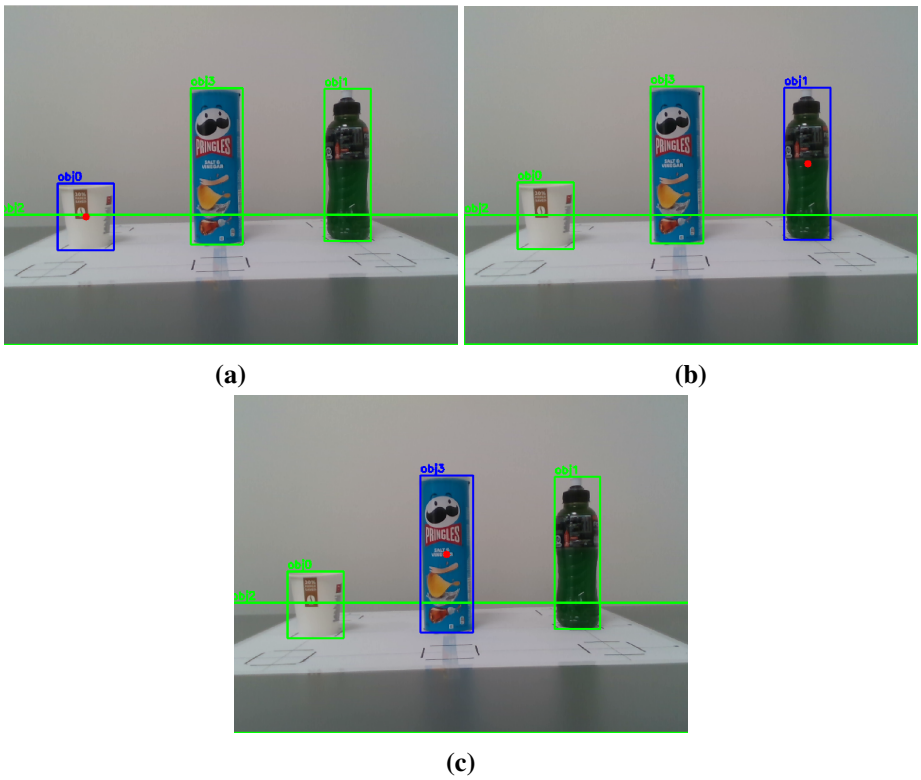
This section presents the outcomes of the project, emphasizing the evaluation of the system's performance concerning the fulfillment of the specified system requirements:

- **1a-c**
- **2a-c**
- **3a-b**
- **4a**
- **5a-c**
- **6a-c**

A variety of tests and assessments were performed to gauge the functionality and effectiveness of each requirement. The subsequent subsections elaborate on the findings pertaining to each requirement.

6.1.1 Object Selection Requirements

The system's ability to facilitate user interaction and provide a clear interface for object selection was evaluated. The evaluation focused on assessing its object selection capability. It was observed that the system successfully achieved requirement 1a. **The system shall enable the user to select objects in the scene using an input device** by allowing a user to select OOIs using the numeric keys on the computer keyboard. The interface design implemented intuitive controls that enhanced the efficiency and accuracy of object selection, aligning with 1b. **The system shall provide a clear and intuitive interface for object selection**. Moreover, the system effectively indicated and highlighted the selected objects by changing the color of the bounding box from green to blue on the screen as illustrated in Figure 6.1, thereby fulfilling requirement 1c. **The system shall visually indicate or highlight the selected objects on the screen.**.



(a) (b)

(c)

Figure 6.1. The correct object is consistently chosen and accurately highlighted with a blue bounding box in all trials, demonstrating the system's precise object identification and selection capabilities

The process of selecting an OOI, as depicted in Figures 6.1a, 6.1b, and 6.1c, was successfully executed without any failures during 10 trials. Each time an OOI was selected by pressing the corresponding key on the keyboard, it was accurately identified and highlighted with a blue bounding box on the computer screen. Consequently, a 100% success rate and accuracy were achieved, aligning with the defined success criteria of 95% or higher, as outlined in the test protocol for this specific test (see AppendixD).

6.1.2 Object Detection and Segmentation Requirements

The evaluation of the system's object detection and segmentation capability was conducted through the continuous processing of the RGB camera feed utilizing YOLOv7, as indicated in requirement **3a**. The system exhibited precise object detection and recognition, effectively delivering dependable segmentations and labels, in adherence to requirement **3b**.

The results of the object-selecting task experiment demonstrate the successful implementation of the system's object detection and segmentation capability. The continuous processing of the RGB camera feed using YOLOv7 showcased the system's ability to accurately identify and segment objects in real-time. This reliable performance highlights the system's effectiveness in efficiently detecting and differentiating various objects within the scene.

6.1.3 Autonomous Grasp Control Requirement

The autonomous grasp control requirement focused on the system's ability to autonomously manipulate the ARM for grasp tasks. Our evaluation revealed that the system seamlessly transitioned between user control and autonomous control during grasp execution, as specified in requirement **2b**. The handover and return of control were clearly communicated to the user,

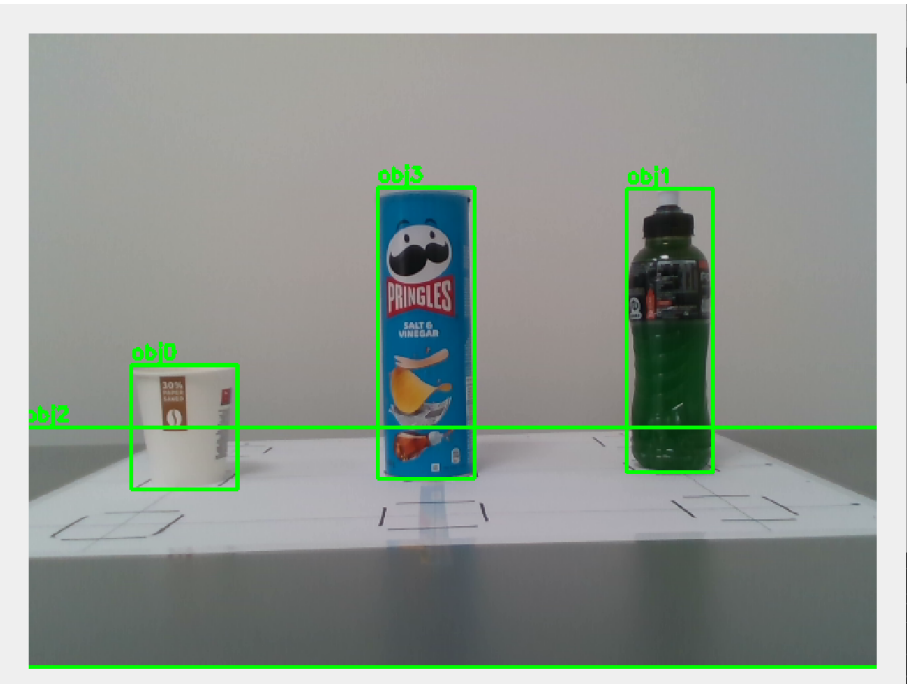


Figure 6.2. Object information is presented to the user on a prominently visible screen, indicated by green bounding boxes surrounding each detected object.

ensuring a smooth and efficient user experience, aligning with requirement **2c**. Furthermore, the system autonomously controlled the manipulator to perform grasp tasks for selected objects, meeting requirement **2a**.

6.1.4 Grasp Pose Generation Requirement

We evaluated the system's grasp pose generation capability by assessing its utilization of the depth camera feed, consideration of object properties and workspace limitations of the JACO^2 ARM, and provision of accurate grasp pose commands to the JACO^2 ARM. Our evaluation confirmed that the system effectively utilized the depth camera feed to generate feasible and appropriate grasp poses for detected objects, aligning with requirement **5a**. The grasp pose generation algorithm took into account relevant factors, including camera input and object properties, but only partially met the criteria outlined in requirement **5b**, falling short of complete fulfillment. Additionally, the system mostly provided accurate and timely grasp pose commands to the JACO^2 ARM, resulting in successful grasping, satisfying requirement **5c** to a certain extent.

The spatial test experiment was conducted from the camera's perspective, utilizing the XYZ-coordinate system where X represented right, Y represented down, and Z represented forward. The obtained results from the depth camera (digital) and manual (real) measurements for each quadrant of the 3x3 grid are presented in Table 6.1 below. According to the test protocol outlined in Appendix D, the success criteria specified a maximum deviation of 0.03 m between the digital and manual measurements. It is important to note that the Y-axis measurements could not be reliably obtained using the measuring tape employed, and thus, they were not included in the manual measurements.

Table 6.1. The spatial test results are presented in a format that includes four columns. The left column indicates the corresponding quadrant, while the next two columns show the digital and real coordinates, respectively. The rightmost column displays the Euclidean distance calculated between the digital and real coordinates. All measurements are presented in meters.

Quadrant	Digital	Real	Euclidean distance
1x1	x: -0.2104 y: -0.0051 z: 0.4900	x: -0.20 y: 0.00 z: 0.50	0.0153
1x2	x: -0.0178 y: -0.0093 z: 0.4979	x: -0.18 y: 0.00 z: 0.50	0.0202
1x3	x: 0.1688 y: -0.0135 z: 0.4990	x: 0.18 y: 0.00 z: 0.50	0.0175
2x1	x: -0.2141 y: -0.0128 z: 0.6290	x: -0.20 y: 0.00 z: 0.62	0.0211
2x2	x: -0.0278 y: -0.0161 z: 0.6340	x: 0.00 y: 0.00 z: 0.62	0.0351
2x3	x: 0.1741 y: -0.0217 z: 0.6470	x: 0.18 y: 0.00 z:0.62	0.0351
3x1	x: -0.2170 y: -0.0206 z: 0.7630	x: -0.20 y: 0.00 z: 0.76	0.02688
3x2	x: -0.0188 y: -0.0235 z: 0.7770	x: 0.00 y: 0.00 z: 0.76	0.0302
3x3	x: 0.1718 y: -0.0288 z: 0.7820	x: 0.18 y: 0.00 z: 0.76	0.0372

Table 6.2 presents the outcomes of the experimental procedure conducted to assess the autonomous robotic system's proficiency in spatial point navigation using the information obtained from its integrated depth and RGB sensors. The table indicates that out of 27 trials, 20 successfully reached the target object, resulting in a 74.07% success rate.

Table 6.2. Results of spatial navigation

	Column 1	Column 2	column 3
Row 3	test 1: fail test 2: fail test 3: fail	test 1: success test 2: fail test 3: success	test 1: fail test 2: fail test 3: fail
Row 3	test 1: success test 2: success test 3: success	test 1: success test 2: success test 3: success	test 1: success test 2: success test 3: success
Row 1	test 1: success test 2: success test 3: success	test 1: success test 2: success test 3: success	test 1: success test 2: success test 3: success

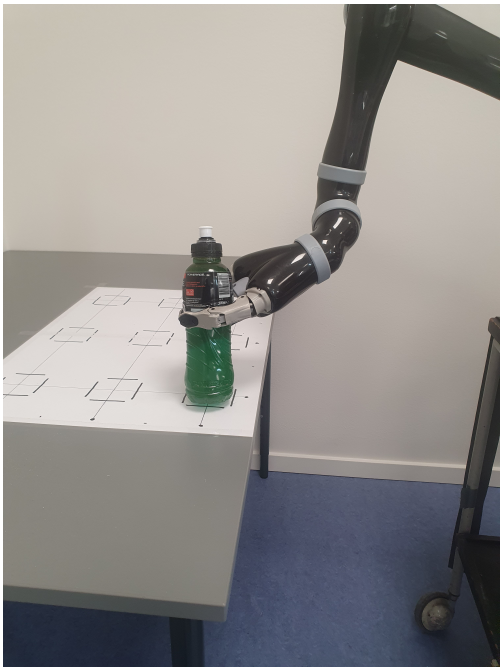
6.1.5 Grasp Execution Requirement

The evaluation of the system's grasp execution capability focused on assessing the ARM's ability to accurately and reliably execute grasp commands, securely hold grasped objects, and maintain stable holding for a defined duration. Our assessment revealed that while the ARM demonstrated successful grasp executions in 15 out of 27 attempts, resulting in a success rate of 55.56%, it did not achieve 100% success in every attempt. However, when successful, the ARM effectively executed grasp commands, securely held the objects without dropping or losing grip, and maintained stable holding for the specified duration, aligning with requirements **6a**, **6b**, and **6c**, respectively.

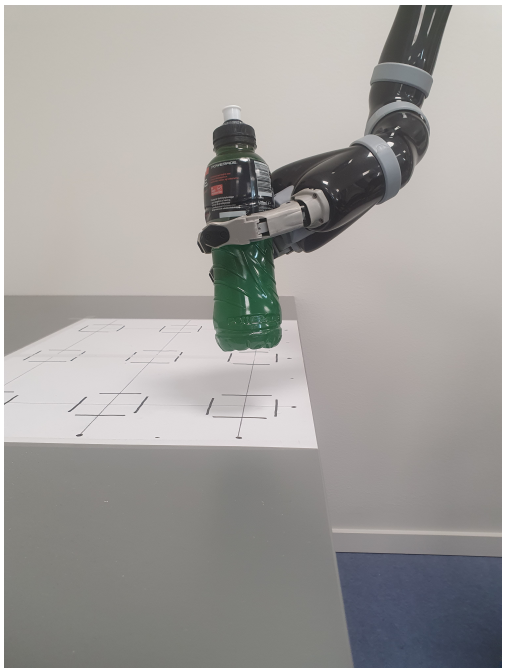
Table 6.3 presented the results of the grasp execution experiment, indicating a total of 15 successful grasps out of the conducted trials. Consequently, the experiment achieved a success rate of 55.56% in effectively grasping and securely holding the object for a duration of 10 seconds as showcased in Figure 6.3). In instances where the grasp attempts were unsuccessful, it was observed that the object was either knocked down due to a last-minute rotation or touched by the fingers during the closing phase (see Figure 6.4).

Table 6.3. Results from grasp test

	Column 1	Column 2	column 3
Row 3	test 1: fail test 2: success test 3: fail	test 1: success test 2: fail test 3: success	test 1: fail test 2: fail test 3: fail
Row 2	test 1: success test 2: fail test 3: fail	test 1: success test 2: success test 3: success	test 1: success test 2: success test 3: success
Row 1	test 1: success test 2: success test 3: success	test 1: success test 2: success test 3: fail	test 1: fail test 2: fail test 3: fail

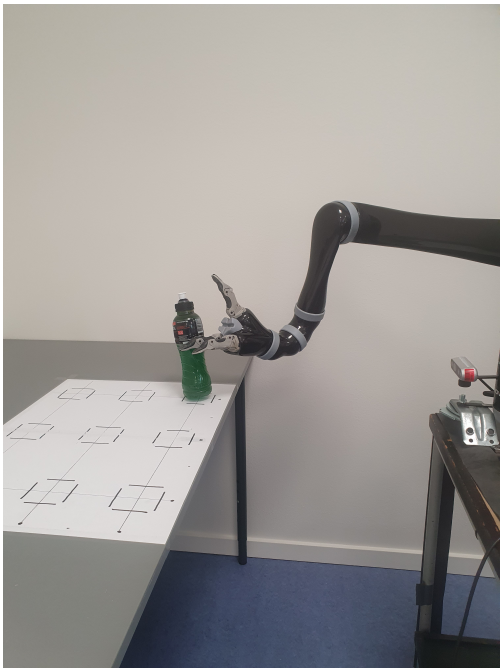


(a) successful grasp

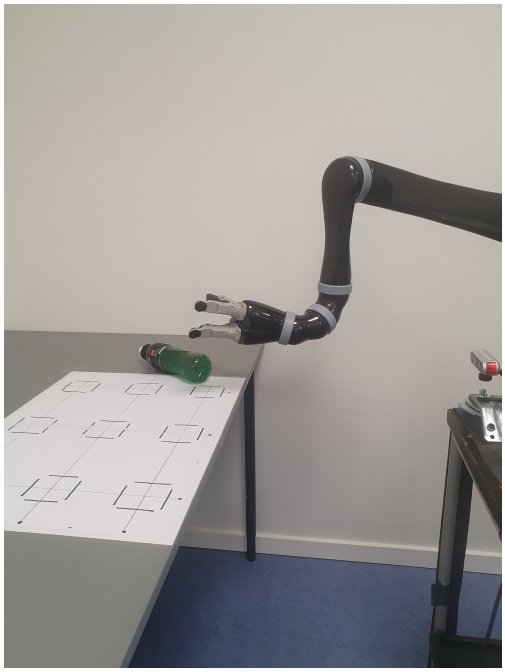


(b) JACO^2 holding the object for 10 seconds

Figure 6.3. showing a successful grasp and holding the object



(a) Moment before the object was knocked down by rotation.



(b) Moment after rotation and object knocked down.

Figure 6.4. Showing unsuccessful grasp.

Discussion 7

7.1 Reflections on results

The current implementation of the system has successfully met several of the intended requirements. The display feature and user input functions have been largely successful, allowing users to choose objects effectively. Additionally, the system has been able to translate the chosen object's position in the RGB image stream into a spatial 3D coordinate with an average accuracy below the 3 cm threshold.

Regarding the positional execution test, it has achieved moderate success. However, it is important to note that the system faced challenges as the target object moved further away. Notably, grid sections 3x1 and 3x3 exhibited a high failure rate, as shown in table 6.2. This outcome can be attributed to the fact that the positional test only involved moving the first few links of the JACO^2 ARM, excluding the wrist and end effector. Although the ARM approximated the object's position, it did not make full contact with the objects placed in grid sections 3x1 and 3x3. Thus, it can be concluded that the initial motivation for this test, as well as the resulting protocol, did not effectively demonstrate the system's ability to coordinate the robot to specific coordinates.

The final test aimed to assess the overall quality of the grasp execution. Based on 27 trial runs, the results indicated a 55% success rate. It is worth noting that the failed attempts all involved physical contact with the object. However, none of the trial runs required manual intervention to prevent collisions with the table surface or other critical trajectory failures. Many of the unsuccessful trials did manage to position the ARM favorably for grasping the object. Nonetheless, due to premature physical contact with the object before completion, the object's location had changed. The following sections will discuss the possible reasons and factors that may have influenced these results.

Ding et al. [2022] achieved between 85-100% success rate on a set of more complex tasks, such as opening cabinets, and placing cups on a table. ten Pas et al. [2017], achieved a 75% success rate in the grasp execution test, using the GPD algorithm with a Baxter robotic manipulator. Similarly Bjomdahl Mortensen et al. [2021] achieved a success rate between 60-100%. In light of these papers, the results presented in this rapport do not contend with the performance of similar systems. It is evident that realization of a more competent system is needed.

7.2 Reflections on the System and Improvements

7.2.1 Grasp Pose Detection

Compatabillity with JACO^2 ARM

The utilization of GPD ros facilitated the generation of meaningful grasp poses within the system, although numerous challenges were encountered. The GPD algorithm, originally designed for the Baxter robot primarily employed in industrial settings, presented certain limitations when applied

in a home environment. It was primarily geared towards providing general-purpose grasp poses, necessitating adjustments to its configurations and usage for implementation within a healthcare-based context. The distinctive finger and hand geometry of the JACO^2 robot differed significantly from the Baxter robot. Despite modifying the GPD package configuration to bridge this disparity, it remains evident that GPD's grasp fitter does not adequately represent the JACO^2 end effector. Furthermore, the JACO^2 used in this project was equipped with a 3-fingered end effector, whereas GPD only generates valid grasps for a 2-fingered end effector. Although GPD managed to produce meaningful grasp poses resulting in successful grasping, the additional finger of the JACO^2 would often collide with the object due to the lack of consideration in grasp generation. To mitigate this issue to some extent, the finger width was set significantly wider than the actual JACO^2 finger width, aiming to simulate the dimensions of a 3-fingered robot by enlarging the two fingers from GPD's perspective. However, this crude workaround did not fully resolve the issue. There are several potential approaches that could be explored to address this challenge more effectively.

One possible approach could involve modifying the GPD package's fundamental setup by integrating it with JACO^2's source code from the beginning. This modification would require a significant effort, but it would offer an optimal solution. Another option would be to equip the JACO^2 ARM with a conventional end effector consisting of two fingers, effectively bypassing the entire challenge. While the three-fingered end effector allows for independent control of the additional finger, providing an extra degree of freedom, this feature is rarely necessary when the task involves simply grasping an object.

The third option, aligning with the current system implementation, involves enhancing the grasp filtering approach. Initially, the GPD outputted only around 10 grasps after internal filtering. However, this was modified to generate 3000 grasps. The rationale behind this modification is that a larger pool of grasps increases the likelihood of finding at least one grasp that meets our custom filtering criteria. Lowering this threshold often led to no valid grasps being generated. Implementing additional constraints on the grasp filtering within our system would be a viable approach, although it does not provide a comprehensive solution to the challenge.

Additional fine-tuning of weights

GPD is trained on digitally generated datasets and comes packaged with pre-trained weights. However, these weights can be further fine-tuned or retrained on a new dataset to improve the GPD's performance. The process of training involves feeding the neural network with a large number of examples of different grasp poses and their corresponding outcomes (successful or unsuccessful). Over time, the neural network learns to associate certain grasp poses with success, thereby improving its ability to predict viable grasp poses.

The GPD package provides the capability to generate training data with C++ code, and it also supports different classifier frameworks that exploit different hardware and have different dependencies. This flexibility allows users to choose the most suitable framework for their specific use case and hardware setup.

While GPD demonstrates good performance without additional training, training it on specific objects or scenarios related to the use in the context of home environments could potentially enhance its output. This is because the neural network would have a better understanding of the object's characteristics and the specific requirements of the grasp.

Improved filtering of grasp poses

The process of filtering raw grasps from the GPD effectively limited the range of viable grasps. This was achieved by using both the object's position in space and the direction of approach, as

explained in the implementation section. As a result, during the project's system testing, only grasps that were directly in line with the positional vector in relation to the robot base were used. Additionally, the filter was set to allow only those grasp poses within 2 cm of the object's central point.

By implementing these limitations, grasps that involved lifting objects from above, below, or from the sides were excluded. This restriction was crucial for the proof-of-concept system outlined in this project but also decreased the system's capacity to handle objects in a variety of real-world scenarios.

These stringent selection criteria for the grasp were based on evidence gathered from laboratory experiments. Without filtering, approaches from below often led to the JACO^2 wrist/hand segments colliding with the table, due to a failure to account for environmental collisions. Furthermore, approaches from the sides often failed due to the pointcloud's inability to provide information about the depth of the object.

The filtered grasps derived from GPD usually resulted in a 180-degree rotation around the wrist. Despite the final position and orientation being accurate, this rotation was often unnecessary. In summary, the grasp filtering method discussed in this report could be improved by refining the approach and positional filtering. Also, integrating the axis and binormal could better align the selected grasp with a practical and purposeful grasp.

One possible approach to address the limitations of the depth camera in capturing the full 3D geometry of scenes is through the incorporation of depth completion. Depth completion involves utilizing the RGB image stream in conjunction with the depth camera to combine and estimate the missing spatial information. The underlying concept is to utilize information from a color image to estimate the absent depth information. Specifically, the color image is utilized to estimate surface normals for the entire scene, which are then combined with the sparse set of depth measurements to infer the depth for the complete scene. [Bengtson et al., 2020]

The primary drawback of this approach is the processing time, as indicated by the authors who report a processing time ranging between 0.3 and 1.5 seconds, depending on the hardware utilized. However, it can be argued that the processing time is not a significant concern since it may not be necessary to employ depth completion on every single frame received from the sensor. [Bengtson et al., 2020]

Two-view camera setup

GPD is able to integrate two cameras in its grasp pose estimation. As only one depth camera was available, this project presents a system with only a single camera. It is however quite possible that incorporating two cameras would increase the quality of the system. Although this would add a new degree of complexity when handling reference frame transforms. Nonetheless, it is plausible that incorporating two cameras would enhance the system's performance. Despite the added complexity associated with handling reference frame transforms, the inclusion of an additional depth camera capturing a different angle would augment the system's ability to accurately determine an object's dimensions. Consequently, this would enable the system to perform grasps from above or from the left/right perspective, provided the second camera is suitably angled.

7.2.2 Camera

Field of view, and camera limitations

The system is designed in such a way that objects can only be selected if they were within the field of view (FOV) of the RGB camera sensor. By positioning the camera near the robot base similar to

Ka et al. [2018], this configuration limits object detection to only those directly in front of the robot. Additionally, since the system can only grasp objects within the RGB image frame, the robot's hand, wrist, and fingers would also be visible in the frame when executing a grasp. Therefore, it becomes necessary for the end-user to ensure that the robot hand is not captured in the frame while selecting an object. Consequently, this limitation severely restricts the system's ability to detect and subsequently grasp objects located below, above, or to the side of the camera.

Several potential solutions can potentially alleviate this shortcoming. One approach involves using a two-camera setup as previously described, which would naturally increase the cumulative FOV of the entire scene. The effectiveness of this solution would depend on the specific placement of the second camera. Another possibility is mounting a camera directly to the end effector, enabling selective pointing towards different parts of the environment and enhancing control over autonomous operations. This arrangement would also resolve the issue of occluded objects caused by the ARM since the camera mounted on the end effector remains unaffected by the ARM's position. Determining the optimal camera placement is challenging, as multiple setups are possible, each with its own advantages and disadvantages. [Bengtson et al., 2020; Bjomdahl Mortensen et al., 2021]

In previous studies, a configuration termed 'eye-in-hand', in which a camera is installed on or near the robot's end effector, was utilized to address problems akin to those currently being explored. This particular arrangement was found to address certain constraints that arise when employing a stationary camera. One of the features of this configuration is its flexibility, as it allows for the camera to be reoriented via movement of the ARM, reducing the potential for the robot to block the sensor's view. An additional insight from these studies is that this setup might allow for a more effective interpretation of user intent, given that the end effector - and hence the camera - is likely directed towards the object that the user wishes to interact with. [Bengtson et al., 2020; Bjomdahl Mortensen et al., 2021]

This arrangement also facilitates the automatic repositioning of the end effector, such that the object selected by the user is centrally positioned within the camera's view. This mechanism potentially enhances the quality of interaction with the object. [Bengtson et al., 2020]

In some cases, researchers have taken this idea further by reorienting the sensor to capture information from a variety of angles. One approach that was adopted involved the use of Simultaneous Localization And Mapping (SLAM) techniques. These techniques appeared to enhance the quality of the data collected, represented as a point cloud, and were also reported to have improved the detection of suitable grasp points on the object. [Bengtson et al., 2020]

One alternative method involves employing a fixed positioning of the camera at an elevated location, strategically situated apart from the robot, in order to achieve an improved viewing angle and minimize occlusion caused by the ARM. Rakhimkul et al. [2019] implemented this approach and obtained encouraging outcomes. However, the viability of employing such a placement for a wheelchair mount is subject to debate.

Minimum distance limitations of depth sensors

One of the challenges encountered is the limitation of the point cloud to a range of 30 cm. Objects positioned closer to the camera than 30 cm cannot be accurately detected in terms of their spatial distance by the depth sensors. Consequently, the end-user is unable to interact with objects located within a 30cm range from the camera, including those in proximity to the robot base.

A straightforward solution to this issue would be to incorporate a depth camera with improved capabilities for detecting objects of interest (OOIs) in close proximity. This enhanced depth camera

would enable more reliable detection and measurement of objects within the 30 cm range.

Another approach to address this challenge involves utilizing a two-camera setup, which complements the previously detailed benefits of such a setup. By employing two depth cameras, the region where the depth sensors cannot provide detailed scene information is reduced. The effectiveness of this approach depends on the placement of the two cameras relative to each other. Ensuring that the cameras are positioned at a distance greater than 30 cm effectively eliminates the limitations imposed by the restricted spatial coverage of a single depth camera. [Bengtson et al., 2020; ten Pas et al., 2017]

Implementing either an enhanced depth camera or a two-camera configuration offers potential solutions to mitigate the constraints imposed by the 30 cm range limitation in the point cloud. These approaches provide opportunities for improved detection and interaction with objects that are located closer to the camera. [Bengtson et al., 2020]

7.2.3 You Only Look Once (YOLO)

Additional fine-tuning of the YOLOV7 weights

The weights utilized for the YOLO algorithm in this project were derived from the COCO dataset, comprising 80 distinct labels. While YOLO successfully detected the majority of objects in the environment, it encountered difficulties in recognizing certain objects. Additionally, numerous noisy detections were observed, including oversized bounding boxes encompassing the table surface or background elements, partially obstructing the intended object bounding boxes. Thus, enhancing the implementation of YOLO within this project would be a legitimate objective.

One potential improvement involves fine-tuning the COCO-based weights for YOLO to cater specifically to the objects relevant to this use case. This could be accomplished by assembling a comprehensive list of objects commonly encountered in a home environment, such as bottles, utensils, doors, etc. Subsequently, a dataset of labeled images could be developed, which would serve as the basis for fine-tuning the weights. This approach holds the potential to enhance the stability of the system, mitigate noise detections, and guide YOLO's focus toward objects that hold practical significance for interaction.

Various alternative deep learning models, including VoxNet, Pose CNN, DOPE neural network, and CenterNet, offer diverse techniques for object detection and pose estimation. However, the YOLOv7 system utilized in this project, although efficient, might lack generalizability. Consequently, it would be advantageous to investigate a hybrid approach or construct a cascaded model that leverages the respective strengths of different architectures, thereby improving the overall performance and versatility of the detection system. [ten Pas et al., 2017; Bengtson et al., 2020; Rakhimkul et al., 2019]

Object tracking and object temporal coherence

The labels for the bounding boxes on the display were deduced by merely arranging them in the order received from YOLO. An inherent challenge associated with this technique stemmed from YOLO's potential to alter the sequence of the bounding boxes, which could lead to the inadvertent swapping of object numbers. In an attempt to mitigate this issue, the project incorporated a system that essentially fixed the bounding box's position as soon as an object was chosen. Consequently, even if YOLO modified the numbering, the bounding box relayed to the other components of the system relied on the object's location at the moment of selection.

However, this workaround only partially addressed the issue. It lacked the capacity to realign the bounding box of the chosen object in the event of movement. Therefore, from the moment of object

selection to the act of grasp execution, the object was required to maintain its original position. This limitation obstructed the implementation of real-time grasp estimation. As such, there was a recognized necessity for a more sophisticated approach to object selection and subsequent object tracking. One prospective approach would involve tying YOLO together with an object-tracking algorithm. After the initial choice, the bounding box corresponding to the selected object would then act as a template, or mask which the object tracker would use to find the object in the next frame. Thus the system would retain its user-selecting process, but would update the location of the object continuously.

Following the initial object selection and bounding box determination, an appropriate object tracking algorithm for this context would be essential. Considering compatibility with ROS Noetic, an appropriate choice might be the OpenCV library's tracking algorithms, which provide robust object-tracking capabilities and is known for their broad use in real-time computer vision applications. Among these, the MultiTracker class, with its compatibility with various tracking algorithms like KCF or TLD (Tracking, Learning and Detection), could be a potential solution. The selection of the tracking algorithm would depend on the specific requirements, such as the number and type of objects to be tracked and the processing capacity of the system.

The implementation of object tracking will inevitably introduce a degree of complexity to the system. It necessitates the system's ability to process and analyze additional layers of information in real-time. This not only includes the continual analysis of object positions but also involves the system's ability to correctly re-identify objects when they momentarily disappear from view or when they change in appearance.

Furthermore, integrating an object tracker will demand more processing power. The complexity of object tracking algorithms varies, but all require computational resources to execute. Depending on the chosen algorithm, this could have a significant impact on system performance. For example, algorithms that use deep learning for object tracking are typically more resource-intensive than simpler, feature-based trackers. Therefore, the choice of the tracking algorithm would also depend on the available computational resources and how much can be allocated to the object tracking task without detrimentally affecting overall system performance.

An alternative and considerably simpler approach could involve forgoing the use of YOLO entirely and relying solely on an object tracking algorithm. Under this methodology, a user would select a point on the RGB image, and that point would then be tracked consistently. This approach dramatically streamlines the processing pipeline and presents a viable solution.

However, when considering the practicalities of this method, one significant consideration arises concerning its application for individuals with a high degree of paralysis. The process of selecting a specific location on a display would require careful thought and appropriate technological assistance to ensure it is feasible for these users.

For example, eye-tracking technology could be utilized to allow users to control a cursor with their eye movements and select a point on the display. Other possible solutions might include the use of assistive technologies that translate slight head or facial movements into cursor control. Alternatively, a speech recognition system could be designed to interpret vocal commands to select points on the display.

In summary, while this simpler approach potentially simplifies the pipeline, its implementation must carefully address the needs and capabilities of the intended users, particularly those with significant physical limitations.

Rakhimkul et al. [2019] incorporated a comparable methodology with respect to the overall system

design, employing YOLOv3 for object detection and using a visual display to represent the objects. Similar to the present project, the authors addressed various challenges, including object occlusion and the constrained object detection capabilities of YOLO when utilizing the base COCO weights.

7.3 Robotic Control

Orientational approach: improvements on grasping

While drawing significant conclusions based on the limited sample size of the presented results in this report is challenging, several noteworthy observations can be identified as outliers. Table 6.3 displays the test outcomes of the grasp execution test. A noticeable imbalance in success rates between the left and right sides of the test grid is evident. During the test, it was observed that when executing grasps on the right side, the hand frequently made contact during the reorientation of the wrist, causing the object to be knocked down. This occurrence was primarily attributed to the fact that all tests were performed from the home position of the JACO^2, where the wrist and end effector was oriented towards the left relative to the robot's orientation. Consequently, the robot had to undergo a significant change in the direction of its end effector to align it with the grasp pose. The reorientation often took place when the end effector was already in close proximity to the object, resulting in contact. Although the robot successfully made contact with the object and completed the command in a valid position for grasping, the reorientation displaced the object and thus was considered a failed attempt. This suggests that, overall, the grasp was mostly valid, but the sequencing of the reorientation led to failure. It is worth mentioning that achieving success was often observed when the JACO^2 end effector was reoriented in a manner that directed the fingers toward the object.

The decision to commence all tests from the home position was motivated by the intention to mitigate variations among tests, ensuring that the only difference lied in the placement of the object. However, it is reasonable to question whether this home position is practical for executing grasps in real-world scenarios. In situations where an end-user has already positioned the end effector towards the object, the impact of this reorientation challenge would be negated. Nonetheless, it cannot be assumed that the end user will consistently align the end effector with the object, as this would introduce a cumbersome variable to consider in daily usage. Consequently, a method must be adopted to minimize this effect, enabling the robot to achieve successful grasps regardless of the starting position.

To address this issue, the grasp pose can be considered as two distinct commands. Given that the position of the grasp pose in 3D space is known, a command can be issued to orient the robot's end effector towards this position. Achieving this orientation is relatively straightforward, as a directional vector can be formed from the origin (0,0,0) to the position of the vector. Subsequently, a rotational matrix can be constructed to align the robot with this directional vector. Once this command is executed, the actual grasp pose can be performed. This approach would effectively reduce the necessity for reorientation during the execution of the grasp pose, as the end effector is already aligned with the object's direction. Implementation of such a method would mean that the robot could succeed in grasping regardless of its initial position and orientation in relation to the object at hand.

Bjomdahl Mortensen et al. [2021] tackled this exact challenge by defining a grasp pose and then using this to determine a pre-grasp pose defined as the initial grasp pose with a positional offset, such that the hand was oriented correctly and in close proximity to the object. Subsequently, the ARM would bridge the remaining distance.

Enviromental and contextual awareness

Robotic control encompasses not only the interaction between the robot and objects but also its ability to perceive contextual cues and comprehend spatial information. The current system described in this report largely neglects the consideration of its environment, including factors such as table surfaces and object occlusion. Therefore, it is imperative to enhance the integration of the point cloud, which provides a digital representation of the spatial environment, into the system. This integration should take into account various environmental elements, such as the mounted surface, the user, table surfaces, and objects in close proximity, in order to prevent collisions.

One potential approach to achieve this integration involves incorporating the point cloud data stream into the existing Kinova-ROS package, which is responsible for trajectory planning. By integrating the point cloud data, the system can leverage the spatial information to generate trajectories that are free from collisions. This integration requires the development of algorithms and modules capable of real-time analysis of the point cloud data. These algorithms should be able to detect potential obstacles and modify the trajectory accordingly to avoid collisions. Furthermore, considering the properties of objects and surfaces in the environment, such as their dimensions and positions, is crucial to ensure accurate and safe robot-object interactions.

Fortunately, the Kinova-ROS package and the moveIT inverse kinematic solver already possess the necessary components to incorporate point cloud data. Therefore, the integration of the point cloud data stream would not pose a significant technical obstacle. This integration would enhance the system's ability to navigate its environment while considering spatial information provided by the point cloud, ultimately leading to safer and more efficient robot operations. [Kinova, 2023]

7.3.1 Overall system deliberations

Temporal aspects

The system outlined in this project primarily aims to achieve significant and successful grasps. However, the speed of executing these grasps is equally crucial. Despite the quality and reliability of the grasps, if the system's operation time exceeds manual ARM usage, the benefits diminish. Therefore, it's necessary to enhance the system pipeline's code to reduce execution time, a critical measure often evaluated in similar research in this field. Accordingly, execution time is often a marker used in papers in this research area, and it would thus be relevant to capture this dimension as well to determine the system's performance in this regard. Hildebrand et al. [2019]; Bjomdahl Mortensen et al. [2021]; ten Pas et al. [2017]; Bengtson et al. [2020]

Physical Considerations

As documented in the implementation chapter the JACO^{2}s hand and finger dimensions were manually measured using a measuring tape. The inaccuracies inherent in this measurement approach suggest that the dimensions set in the GPD's configuration may not align with the actual dimensions. Additionally, the GPD package did not provide precise documentation regarding the interpretation of various parameters in the context of orientation. For instance, the hand height parameter was undefined in terms of orientation, leading to ambiguity in its interpretation and setting.

The translatory offset between the camera and the base of the JACO^2 ARM was also measured using a measuring tape, likely introducing similar inaccuracies. More critically, the necessary tools to accurately determine the pitch, yaw, and roll of the camera in relation to the robot were unavailable, and thus, this aspect was disregarded. Any unaccounted rotational or translatory offset between these reference frames would compromise the accuracy of the grasps.

Despite the rudimentary nature of the calibration, the system was still able to execute grasps successfully. However, it is reasonable to infer that a more precise calibration could potentially enhance performance.

7.4 Reflections on Tangential Aspects

Patient group

This report has primarily addressed the topic of spinal cord injury and the subsequent manifestation of paralysis. Additionally, other conditions such as ALS, CP, and other similar disorders present diverse levels of associated paralysis. Consequently, the application of semi-autonomous ARMS (Assistive Robotic Systems) can also hold relevance for this wider range of patients. Exploring the utilization of the system described in this report within this expanded population would be advantageous. This investigation would involve addressing challenges related to different levels of user mobility, muscle tremors, cognitive abilities, and other pertinent factors. By conducting such research, a comprehensive understanding of how the system can be effectively employed in accordance with the needs of this broader patient cohort can be achieved.

User input

The current proof-of-concept implementation has a narrowed scope, excluding the integration of iTongue-based user input. Redesigning the user interaction layer becomes crucial in order to incorporate this input method, while also considering its interface with the existing system. A significant aspect of this redesign would involve mapping the current keyboard-based inputs to the iTongue controls in a meaningful manner, aligning with how individuals typically utilize iTongue in their daily lives. Additionally, conducting tests with iTongue users and individuals with paralysis would be essential to effectively determine the tangible benefits that this system offers to the end user.

Hand-over versus blending of shared control

The semi-autonomous control aspect of the current system design is implemented as a binary switch, where the user either has control or the autonomous control system takes over. The system operates exclusively in one mode or the other. Although it is recommended for future iterations to include a manual override when the system is in autonomous mode, this approach still maintains a distinct separation between the two control schemes. While the current hand-over control scheme is straightforward and effective, it may not be the most suitable approach in terms of providing users with a sense of autonomy. Hence, it would be advantageous for future iterations to explore the possibility of a more integrated control scheme. This integrated scheme would ensure that control is never directly handed over to the autonomous system but seamlessly blended, allowing the user to consistently feel in control while the underlying control system makes real-time corrections and adjustments. Although implementing such a merged control scheme would be a significant endeavor, it aligns better with the overall objective of enabling user autonomy. Bengtson et al. [2020] similarly describes that most of the currently explored shared control systems rely on a binary hand-over approach, thus leading to a gap in the research field for blended control. Rakhimkul et al. [2019]

Bjomdahl Mortensen et al. [2021] is an example of an attempted blended approach. The system defined a likelihood of whether or not a user wanted to grasp an object, and would then guide the ARM to the target based on this parameter. The overall performance of this approach is promising and gives credence to the ability of blended approaches to give a more nuanced interconnection between user and ARM.

7.5 Addressing Key Challenges

System improvements: Prospects for further iterations

Assessing the overall system in its current implementation is challenging when it comes to determining whether it satisfies the subset of requirements that were initially established. The previous discussions indicate that several crucial aspects of the system design and subsequent implementation need to be revised accordingly. At present, the system's core functionality has been implemented, and a framework for future iterations has been established, which should be informed by the reflections presented in this discussion.

Conducting further testing based on these iterations is essential to determine whether the overall system provides a meaningful benefit to the end-user and whether the scope and functionality of the system can meet the entirety of the system requirements. In order to accomplish this, it is necessary to incorporate testing involving individuals with paralysis and gather their feedback.

7.5.1 The generalization problem

The primary challenge encountered in advancing the field of ARM's is primarely based around the control system. This can be divided into several interrelated problems:

1. Robust and Generalizable Object Detection: This pertains to not only detecting objects of interest, but also recognizing their contextual relationship to surrounding entities. For example, it is critical for the system to identify whether a cup is situated on a slippery surface, or whether the house keys are partially obscured by another object. Existing object detectors, while impressive in certain aspects, tend to lack the necessary precision to comprehend such contextual relationships. Furthermore, a near-human-level perceptual network is required to capture the requisite details, given the myriad of edge cases where simple algorithms may falter. Misidentifications, such as perceiving a phone case that has an image of a banana as an actual banana, or categorizing a water bottle with a Big Ben print on it as a large tower due to inadequate visual cues about scale, exemplify these edge cases. Even in the domain of object classification, humans often succeed with minimal input, showcasing a level of predictive power and semantic understanding that contemporary algorithms largely lack.

2. Spatial Awareness: In tandem with object detection, the robot must possess an understanding of the spatial placement of objects relative to its surroundings. This spatial awareness is critical for path planning and avoiding potential harm to surrounding objects. It also necessitates an understanding of the robot's own position relative to other entities.

3. Movement and Adaptability: The last facet pertains to the robot's ability to replicate the inherent human capability of instinctively grasping any object, even if it is a new or unfamiliar item. Though it may seem intuitive for humans, formalizing and encoding this natural grasping ability into a logic-based code for robots poses a significant challenge.

The field of ARMS is ripe for advancements in machine learning and artificial intelligence, as the inherent challenges often require a generalized capability to interact with the surrounding world. In deterministic environments, such as factories, robots akin to JACO^2 are often employed due to the clearly defined, static nature of tasks. However, in healthcare settings, the variability and continuous changes in the environment complicate the application of such systems.

Teaching a robot to perform not just one hard-coded task (an already accomplished feat), but hundreds of tasks with minor variations, remains a daunting challenge. Consider the seemingly simple task of drinking; numerous variables come into play, such as the volume of liquid in the cup, the speed of movement, the method of gripping the cup (based on presence or absence of a

handle), the texture of the cup (whether slippery or rubberized), the temperature of the liquid, and the process of opening a potential bottle cap.

While task recognition can be partially addressed by hard-coding a narrow set of frequent tasks, the ultimate goal is a robot capable of managing arbitrary manipulation tasks and learning new ones. This aspiration necessitates substantial progress in the field of artificial intelligence.

Conclusion 8

The investigation conducted in this project towards shared control systems for robotic manipulators symbolizes a progressive, albeit incremental, enhancement in assistive technologies for individuals affected by tetraplegia. It is important to acknowledge, nonetheless, that the outcomes of this project delineate not only the potentialities but also the present boundaries of such systems.

The assembled prototype, a composition of the JACO² ARM and an Intel® RealSense™ D435 RGB-D camera, encapsulated within the Robot Operating System (ROS) framework, substantiated the viability of a shared control system. It provided empirical evidence that users could successfully identify objects within a visual field, thereby permitting the robotic manipulator to autonomously execute grasping tasks. Despite these promising results, a range of difficulties emerged during the developmental and testing stages, underscoring the inherent complexities of effectively implementing this class of technology.

A salient challenge identified during the project pertained to the management of fine motor control within the robotic manipulator, particularly when targeting objects located at greater distances. Alongside this, the system's overall success rate, amounting to 55.5% across 27 trials, hinted at a degree of inconsistency, demanding further investigation and improvements before contemplating broader application.

Another crucial element this project grappled with was the intricate task of balancing user control and autonomous operation. This exposed the problematic nature of managing discrepancies between the robot's and the user's degree of control. It underscored the essential requirement for control systems that are more intuitive and shared control systems capable of effectively adjusting to a user's intentions and capabilities.

Given these multifaceted complexities and the present restrictions of the system, it is unmistakably apparent that substantial research, fine-tuning, and development are prerequisites for such technologies to reach a stage of widespread implementation. This implies necessitating more exhaustive end-user trials, with a particular emphasis on individuals affected by tetraplegia, in order to gather insightful feedback concerning the system's usability and practicality.

In conclusion, this project makes a significant contribution to the continued evolution of assistive technologies intended for individuals with tetraplegia. Nevertheless, it also brings attention to the formidable obstacles that remain to be surmounted to realize dependable, efficient, and user-centric robotic manipulator systems that can genuinely augment the quality of life for these individuals.

Bibliography

Alexandre, **2 2023**. F. CH Alexandre. *GitHub - alexandrefch/ros-yolov7: ROS package that implement a light yolov7 version in ros node style that allow user to run inference easely.* `https://github.com/alexandrefch/ros-yolov7`, 2023. (Accessed on 05/31/2023).

Andreasen Struijk et al., **2017**. Lotte N.S. Andreasen Struijk, Line Lindhardt Egsgaard, Romulus Lontis, Michael Gaihede and Bo Bentsen. *Wireless intraoral tongue control of an assistive robotic arm for individuals with tetraplegia*. Journal of NeuroEngineering and Rehabilitation, 14(1), 1–8, 2017. ISSN 17430003. doi: 10.1186/s12984-017-0330-2.

Baes-Jørgensen, **jan 2022**. Jens Baes-Jørgensen. *Om bare 8 år kan vi mangle 16.000 SOSU'er*, 2022. URL `https://www.kl.dk/nyheder/momentum/2022/2022-2/om-bare-8-aar-kan-vi-mangle-16000-sosu-er`. [Online; accessed 26. Apr. 2023].

Beaudoin et al., **December 2018**. Maude Beaudoin, Josiane Lettre, François Routhier, Philippe S. Archambault and Isabelle Gélinas. *Impacts of robotic arm use on individuals with upper extremity disabilities: A scoping review*. Can. J. Occup. Ther., 85(5), 397–407, 2018. ISSN 0008-4174. doi: 10.1177/0008417418820878.

Bengtson et al., **2020**. Stefan Hein Bengtson, Thomas Bak, Lotte N.S. Andreasen Struijk and Thomas Baltzer Moeslund. *A review of computer vision for semi-autonomous control of assistive robotic manipulators (ARMs)*. Disability and Rehabilitation: Assistive Technology, 15, 731–745, 2020. ISSN 17483115. doi: 10.1080/17483107.2019.1615998. URL `https://doi.org/10.1080/17483107.2019.1615998`.

Bennett et al., **2022**. J Bennett, J M Das and PD Emmady. *Spinal Cord Injuries*. StatPearls [Internet], 2022. URL `https://www.ncbi.nlm.nih.gov/books/NBK560721/`.

Bjomdahl Mortensen et al., **2021**. Anton Bjomdahl Mortensen, Emil Tribler Pedersen, Jacob Winther Knudsen, Jesper Kjaegaard Mortensen, Marie Hiorth Bogestrand, Lotte N.S. Andreasen Struijk and Asgerour Arna Palsdottir. *Adaptive Semi-automatic Robot Control by Tongue in a Remote Setting for Individuals with Tetraplegia*. BIBE 2021 - 21st IEEE International Conference on BioInformatics and BioEngineering, Proceedings, 2021. doi: 10.1109/BIBE52308.2021.9635539.

Campeau-Lecours et al., **1 2018**. Alexandre Campeau-Lecours, Hugo Lamontagne, Simon Latour, Philippe Fauteux, Véronique Maheu, François Boucher, Charles Deguire and Louis-Joseph Caron L'Ecuyer. *Kinova Modular Robot Arms for Service Robotics Applications*. International Journal of Robotics Applications and Technologies, 5, 49–71, 2018. ISSN 2166-7195. doi: 10.4018/IJRAT.2017070104.

Dellon and Matsuoka, **April 2007**. Brian Dellon and Yoky Matsuoka. *Prosthetics, exoskeletons, and rehabilitation [Grand Challenges of Robotics]*. Robotics & Automation Magazine, IEEE, 14(1), 30–34, 2007. ISSN 1070-9932. doi: 10.1109/MRA.2007.339622.

Ding et al., **6 2022**. Dan Ding, Breelyn Styler, Cheng Shiu Chung and Alexander Houriet. *Development of a Vision-Guided Shared-Control System for Assistive Robotic Manipulators*. Sensors, 22, 2022. ISSN 14248220. doi: 10.3390/s22124351. URL

/pmc/articles/PMC9228253//pmc/articles/PMC9228253/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC9228253/.

Fattal et al., **jan 2019**. Charles Fattal, Violaine Leynaert, Isabelle Laffont, Axelle Baillet, Michel Enjalbert and Christophe Leroux. *SAM, an Assistive Robotic Device Dedicated to Helping Persons with Quadriplegia: Usability Study*. International Journal of Social Robotics, 11(1), 89–103, 2019. ISSN 18754805. doi: 10.1007/s12369-018-0482-7. URL https://link.springer.com/article/10.1007/s12369-018-0482-7.

Germanotta et al., **Mar 2023**. Marco Germanotta, Laura Cortellini, Sabina Insalaco and Irene Aprile. *Effects of upper limb robot-assisted rehabilitation compared with conventional therapy in patients with stroke: Preliminary results on a daily task assessed using motion analysis*, 2023. URL https://www.mdpi.com/1424-8220/23/6/3089.

Hays et al., **April 2023**. Emilly Hays, Jack Slayton, Gary Tejeda-Godinez, Emily Carney, Kobe Cruz, Trevor Exley and Amir Jafari. *A Review of Rehabilitative and Assistive Technologies for Upper-Body Exoskeletal Devices*. Actuators, 12(4), 178, 2023. ISSN 2076-0825. doi: 10.3390/act12040178.

Hildebrand et al., **2019**. Max Hildebrand, Frederik Bonde, Rasmus Vedel Nonboe Kobborg, Christian Andersen, Andreas Flem Norman, Mikkel Thogersen, Stefan Hein Bengtson, Strahinja Dosen and N. S.Lotte Andreasen Struijk. *Semi-autonomous tongue control of an assistive robotic arm for individuals with quadriplegia*. IEEE International Conference on Rehabilitation Robotics, 2019-June, 157–162, 2019. ISSN 19457901. doi: 10.1109/ICORR.2019.8779457.

Intel, **5 2023**. Intel. *GitHub - IntelRealSense/realsense-ros: Intel(R) RealSense(TM) ROS Wrapper for Depth Camera*. https://github.com/IntelRealSense/realsense-ros, 2023. (Accessed on 05/31/2023).

Johansen et al., **7 2016**. Daniel Johansen, Christian Cipriani, Dejan B. Popovic and Lotte N.S.A. Struijk. *Control of a Robotic Hand Using a Tongue Control System-A Prosthesis Application*. IEEE Transactions on Biomedical Engineering, 63, 1368–1376, 2016. ISSN 15582531. doi: 10.1109/TBME.2016.2517742.

Ka et al., **2 2018**. Hyun W. Ka, Cheng Shiu Chung, Dan Ding, Khara James and Rory Cooper. *Performance evaluation of 3D vision-based semi-autonomous control method for assistive robotic manipulator*. Disability and Rehabilitation: Assistive Technology, 13, 140–145, 2018. ISSN 17483115. doi: 10.1080/17483107.2017.1299804. URL https://www.tandfonline.com/doi/abs/10.1080/17483107.2017.1299804.

Keselman et al., **5 2017**. Leonid Keselman, John Iselin Woodfill, Anders Grunnet-Jepsen and Achintya Bhowmik. *Intel RealSense Stereoscopic Depth Cameras*. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2017-July, 1267–1276, 2017. ISSN 21607516. doi: 10.1109/CVPRW.2017.167. URL https://arxiv.org/abs/1705.05548v2.

Khan and Lui, **2022**. Yusuf S. Khan and Forshing Lui. *Neuroanatomy, Spinal Cord*. StatPearls, 2022. URL https://www.ncbi.nlm.nih.gov/books/NBK559056/.

Kim et al., **2012**. Dae Jin Kim, Zhao Wang and Aman Behal. *Motion segmentation and control design for UCF-MANUSAn intelligent assistive robotic manipulator*. IEEE/ASME Transactions on Mechatronics, 17, 936–948, 2012. ISSN 10834435. doi: 10.1109/TMECH.2011.2149730.

Kinova, 2 2023. Kinova. *GitHub - Kinovarobotics/kinova-ros at noetic-devel*. https://github.com/Kinovarobotics/kinova-ros/tree/noetic-devel, 2023. (Accessed on 05/31/2023).

KL, April 2023. KL. *Det velfærdsteknologiske landkort*, 2023. URL https://videncenter.kl.dk/viden-og-vaerktoejer/teknologilandskabet-overblik-og-redskaber/det-velfaerdsteknologiske-landkort/#/?communes=1608. [Online; accessed 26. Apr. 2023].

Koglin et al., jun 2019. Tim Koglin, Bulcsú Sándor and Claudius Gros. *When the goal is to generate a series of activities: A self-organized simulated robot arm*. PLoS ONE, 14(6), e0217004, 2019. ISSN 19326203. doi: 10.1371/journal.pone.0217004. URL https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0217004.

Kronhardt et al., feb 2022. Kirill Kronhardt, Stephan Rübner, Max Pascher, Felix Ferdinand Goldau, Udo Frese and Jens Gerken. *Adapt or Perish? Exploring the Effectiveness of Adaptive DoF Control Interaction Methods for Assistive Robot Arms*. Technologies, 10(1), 30, 2022. ISSN 22277080. doi: 10.3390/technologies10010030. URL https://www.mdpi.com/2227-7080/10/1/30/htmhttps://www.mdpi.com/2227-7080/10/1/30.

Lai et al., 1 2007. J. C.K. Lai, M. P. Schoen, A. Perez Gracia, D. S. Naidu and S. W. Leung. *Prosthetic devices: challenges and implications of robotic implants and biological interfaces*. Proceedings of the Institution of Mechanical Engineers. Part H, Journal of engineering in medicine, 221, 173–183, 2007. ISSN 0954-4119. doi: 10.1243/09544119JEIM210. URL https://pubmed.ncbi.nlm.nih.gov/17385571/.

Law-Kam Cio et al., dec 2019. Yann Seing Law-Kam Cio, Maxime Raison, Cedric Leblond Menard and Sofiane Achiche. *Proof of Concept of an Assistive Robotic Arm Control Using Artificial Stereovision and Eye-Tracking*. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 27(12), 2344–2352, 2019. ISSN 15580210. doi: 10.1109/TNSRE.2019.2950619.

Maheu et al., 2011. Veronique Maheu, Julie Frappier, Philippe S. Archambault and François Routhier. *Evaluation of the JACO robotic arm: clinico-economic study for powered wheelchair users with upper-extremity disabilities*. IEEE Int. Conf. Rehabil. Robot., 2011, 2011.

Markovic et al., nov 2015. Marko Markovic, Strahinja Dosen, Dejan Popovic, Bernhard Graimann and Dario Farina. *Sensor fusion and computer vision for context-aware control of a multi degree-of-freedom prosthesis*. Journal of Neural Engineering, 12(6), 066022, 2015. ISSN 17412552. doi: 10.1088/1741-2560/12/6/066022. URL https://iopscience.iop.org/article/10.1088/1741-2560/12/6/066022https://iopscience.iop.org/article/10.1088/1741-2560/12/6/066022/meta.

Amal Nanavati, Patricia Alves-Oliveira, Tyler Schrenk, Ethan K. Gordon, Maya Cakmak and Siddhartha S. Srinivasa, mar 2023. Amal Nanavati, Patricia Alves-Oliveira, Tyler Schrenk, Ethan K. Gordon, Maya Cakmak and Siddhartha S. Srinivasa. Design Principles for Robot-Assisted Feeding in Social Contexts. In *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, pages 24–33, New York, NY, USA, mar 2023. ACM. ISBN 9781450399647. doi: 10.1145/3568162.3576988. URL https://dl.acm.org/doi/10.1145/3568162.3576988.

Petrich et al., January 2021. Laura Petrich, Jun Jin, Masood Dehghan and Martin Jagersand. *Assistive arm and hand manipulation: How does current research intersect with actual healthcare needs?* arXiv, 2021. doi: 10.48550/arXiv.2101.02750.

Quere et al., 2021. Gabriel Quere, Samuel Bustamante, Annette Hagengruber, Jorn Vogel, Franz Steinmetz and Freek Stulp. *Learning and Interactive Design of Shared Control Templates*. IEEE International Conference on Intelligent Robots and Systems, pages 1887–1894, 2021. ISSN 21530866. doi: 10.1109/IROS51168.2021.9636047.

Rakhimkul et al., 10 2019. Sanzhar Rakhimkul, Anton Kim, Askarbek Pazylbekov and Almas Shintemirov. *Autonomous object detection and grasping using deep learning for design of an intelligent assistive robot manipulation system*. Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics, 2019-October, 3962–3968, 2019. ISSN 1062922X. doi: 10.1109/SMC.2019.8914465.

Region Midtjylland, 2010. Region Midtjylland. *Rygmarvsskadebehandling- og rehabilitering i Vestdanmark*. pages 1–65, 2010.

Renfrew, 2004. Alasdair Renfrew. *Book Review: Introduction to Robotics: Mechanics and Control*. The International Journal of Electrical Engineering & Education, 41(4), 388–388, 2004. ISSN 0020-7209. doi: 10.7227/ijeee.41.4.11.

ROS, 5 2007. ROS. *ROS - Robotic Operating System*. `https://www.ros.org/`, 2007. (Accessed on 05/31/2023).

Rosen et al., 2005. Jacob Rosen, Joel C. Perry, Nathan Manning, Stephen Burns and Blake Hannaford. *The human arm kinematics and dynamics during daily activities - Toward a 7 DOF upper limb powered exoskeleton*. 2005 International Conference on Advanced Robotics, ICAR '05, Proceedings, 2005, 532–539, 2005. doi: 10.1109/ICAR.2005.1507460.

Soendergaard et al., jan 2022. Pernille Langer Soendergaard, Anne Norup, Marie Kruse and Fin Biering-Sørensen. *Socioeconomic consequences of traumatic and non-traumatic spinal cord injuries: a Danish nationwide register-based study*. Spinal Cord 2021 60:7, 60(7), 647–654, 2022. ISSN 1476-5624. doi: 10.1038/s41393-021-00724-3. URL `https://www.nature.com/articles/s41393-021-00724-3`.

ten Pas, 7 2021a. Andreas ten Pas. *GitHub - atenpas/gpd: Detect 6-DOF grasp poses in point clouds*. `https://github.com/atenpas/gpd`, 2021. (Accessed on 05/31/2023).

ten Pas, 7 2021b. Andreas ten Pas. *GitHub - atenpas/gpd_ros: ROS wrapper around GPD*. `https://github.com/atenpas/gpd_ros/`, 2021. (Accessed on 05/31/2023).

ten Pas et al., dec 2017. Andreas ten Pas, Marcus Gualtieri, Kate Saenko and Robert Platt. *Grasp Pose Detection in Point Clouds*. International Journal of Robotics Research, 36(13-14), 1455–1473, 2017. ISSN 17413176. doi: 10.1177/0278364917735594/ASSET/IMAGES/LARGE/10.1177_0278364917735594-FIG16.JPEG. URL `https://journals.sagepub.com/doi/10.1177/0278364917735594`.

ten Pas et al., 2021. Andreas ten Pas, Marcus Gualtieri, Kate Saenko and Robert Platt. *Grasp Pose Detection in Open Worlds*. International Journal of Robotics Research, 36(13-14), 1455–1473, 2021. ISSN 17413176. doi: 10.1177/0278364917735594. URL `https://repository.library.northeastern.edu/files/neu:4f186m86k/fulltext.pdf`.

Wang, 2023. Chien-Yao Wang. *GitHub - WongKinYiu/yolov7: Implementation of paper - YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors*. `https://github.com/WongKinYiu/yolov7`, 2023. (Accessed on 06/01/2023).

Wang et al., 2022. Chien-Yao Wang, Alexey Bochkovskiy and Hong-Yuan Mark Liao. *YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors*. arXiv preprint arXiv:2207.02696, 2022.

Xu et al., 2022. Y. Xu, H. Zhang, L. Cao, X. Shu and D. Zhang. *A Shared Control Strategy for Reach and Grasp of Multiple Objects Using Robot Vision and Noninvasive Brain–Computer Interface*. IEEE Transactions on Automation Science and Engineering, 19(1), 360–372, 2022. ISSN 1558-3783 VO - 19. doi: 10.1109/TASE.2020.3034826.

Zhong et al., mar 2022. Boxuan Zhong, He Huang and Edgar Lobaton. *Reliable Vision-Based Grasping Target Recognition for Upper Limb Prostheses*. IEEE Transactions on Cybernetics, 52 (3), 1750–1762, 2022. ISSN 21682275. doi: 10.1109/TCYB.2020.2996960.

Portfolio A

A.1 Initial planning

The creation of our initial plan depicted in Figure A.1 was rooted in a logical progression and an assessment of the complexity of individual tasks, with the aim of meeting our thesis deadline. We took into consideration the stages of the research process and our understanding of the technologies we were going to employ. Below is a description of our planning and subsequently our reasoning behind partitioning the time in the way we did. Additionally, we include a contemplation on possible enhancements for future endeavors, drawing upon the insights gained from this experience.

The documentation of worksheets, and formulating the rapport was a parallel task running alongside the rest of the project realization. It proved difficult to explicitly incorporate this parallel in the plan and was therefore left as an implicit agreement between us.

At the start of the project, we considered the initial problem foundation and research on iTongue and ARM to be essential. This initial stage was expected to be time-intensive, but we allocated only a few days because we had some prior knowledge about the technologies, which we thought would allow us to delve into the specifics quickly. This stage involved understanding the technologies, investigating prior research, and formulating research questions. It was the foundation of our project, so we assumed getting it right quickly would set us on the right path.

The next stage was the literature review. We allocated a week for each of the subtasks - searching for papers and reading/summarizing them - to ensure that we had a robust understanding of existing work in the field. This is typically a time-consuming process, but crucial for developing a well-informed project.

The subsequent stages were about planning - choosing methods from the literature, understanding hardware and software requirements, and planning for the integration of hardware with ARM. We anticipated that these stages would be less time-intensive than the literature review since they were more about making decisions based on the information we had already gathered.

Once we had our plan and understanding, we moved into the implementation phase, which involved writing algorithms and learning the software structure of the ARM. At this stage, we allocated more time because we knew that writing code and learning new software systems could be time-consuming and unpredictable, especially considering the integration of new hardware and signals.

The final stages were about testing and documentation. We understood the importance of thorough testing and allocated a week for each testing stage. We felt this was ample time to test, debug, and refine our code.

Lastly, we set aside the last few days to format and structure the rapport in terms of formalia, abstract, etc. to finalize the rapport.

Hand in: 01/06/23	February		March				April				May				June
	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14	Week 15	Week 16	Week 17	Week 18	Week 19	Week 20	Week 21	Week 22
Initial problem foundation															
Research previous work on tongue sensor and robot arm control	Sun.	Tue.													
Familiarize ourselves with tongue sensor and robot arm hardware		Wed. -Thu.													
Write down the initial research questions and hypotheses		Fri. - Sun.													
Literature search															
Search databases for relevant papers			Mon. - Sun.												
Read and summarize relevant papers				Mon. - Sun.											
Choosing a suitable set of methods from the literature															
Discuss potential methods and approaches					Mon. - Sun.										
Review and narrow down list of potential methods						Mon. - Sun.									
Research and plan hardware and software															
Research necessary hardware and software for project							Mon.-Tue.								
Select appropriate components							Wed.-Sun.								
Plan integration of hardware with ARM								Mon. - Sun.							
Implementation															
Write code for controlling the robot arm using the tongue sensor and new hardware									Mon. - Sun.						
Research current software structure of the robot arm, as it is now including APIs, coding language, databases, etc										Mon. - Fri.					
Review and understand APIs, coding language, and databases										Sat. - Sun					
Write our software/algorithm for controlling the robot arm using the tongue sensor and new hardware											Mon.	Sun.			
Integrate our software with the robot arm software												Mon.	Sun.		
Testing the code															
Test the software with the robot arm and tongue sensor														Mon. - Sun.	
Testing the arm in a controlled setting															Mon. - Wed.
Writing															
Main part of writing														Fri.	Wed.
Finish Report (Last Corrections)															Thu.

Figure A.1. The initial time management plan.

A.2 Reflections

We started with a solid plan which meticulously detailed our milestones, the subtasks required to achieve them, and the respective start and end dates. This plan was our guideline and we anticipated it to be our road to successful project completion. However, as often occurs in life and in engineering projects, the unforeseen events and technical complexities challenged our ability to strictly adhere to our initial plan.

The first major challenge we faced was personal crises, which significantly impacted both of our abilities to commit time and energy to the project. Such personal circumstances are a reminder of the potential volatilities that life presents, and these external pressures can significantly affect adherence to any plan. In future endeavors, it might be wise to preemptively account for potential personal obstacles when creating an initial time plan, or at least building in some buffer time to cater for unforeseen issues.

The second challenge we encountered was the novelty of the technology we were dealing with. The integration of robot arm control and the use of ROS and the Ubuntu operating system was a unique experience for both of us. Prior to this project, neither member of the group had any previous experience with Ubuntu or robotics in general. Consequently, we spent more time familiarizing ourselves with these systems, writing algorithms, and integrating our software into the processing stream than we initially estimated. This led to delays and an adjustment in our time plan.

From this experience, we have learned that when dealing with unfamiliar technology or a new field of study, we must allocate extra time to allow for a steeper learning curve. We should not underestimate the time it takes to become proficient in a new area, especially when it involves the integration of different technologies and platforms.

The implicit nature of the documentation and writing of the rapport in the plan might have left us with too vague an idea of when and what to write, thus the rapport was primarily written at the end of the project period. In future projects, more explicit and direct planning of this aspect might be a necessary step.

Recognizing the delay in our initial plan, we revised our time schedule near the end of our project. This involved reassessing the time needed to accomplish remaining tasks, re-prioritizing where necessary, and using the lessons we learned from our initial planning to create a more robust and realistic schedule.

Although this revised plan was not part of our original design, it was a critical step that allowed us to get back on track. It allowed us to maintain our focus and complete our thesis in a timely fashion, given the circumstances. This highlighted the importance of flexibility and agility when it comes to project planning, and the necessity to continually review and adapt the plan in response to changing circumstances.

A.3 Revising the plan

When it became evident that our initial plan was not achievable due to delays caused by personal crises and technical complexities, we made the decision to create a revised plan. This second plan aimed to streamline our remaining tasks and allocate time in a way that would allow us to complete the project within the new timeframe. Below is a reflection on how we formulated the second plan and why it was structured the way it was.

The first step in creating the revised plan was to assess the remaining tasks and identify the critical components that needed to be addressed. We recognized the importance of system design and

the need to establish a clear foundation for the project. Therefore, we allocated a significant portion of time to system design, including analyzing requirements, defining the architecture, and leveraging existing technology and resources. This stage was crucial for ensuring a well-structured and functional system.

Once the system design was established, we focused on defining the system itself and its specifications. We emphasized detailed descriptions of functionality, considered alternative solutions, and addressed risks and security aspects. This step aimed to provide a precise and comprehensive understanding of the system's requirements, serving as a reference for the remainder of the project.

To further solidify our understanding of the system, we created a block diagram that visualized its structure and functionality. This diagram encompassed all hardware and software components and their interactions. By including this step in the plan, we ensured a clear and consistent representation of the system, facilitating implementation and testing.

The subsequent step involved implementing the code based on the system design, specifications, and block diagram. We allotted sufficient time for coding, testing, and debugging to ensure the functionality and reliability of the system.

As testing and validation were crucial aspects of our project, we dedicated a separate stage to writing the test protocol and setting up the experimental procedures. This allowed us to define a systematic approach to validate the system, document the results, and ensure the accuracy of our findings.

In the final phase, we allocated time for writing the report. This involved compiling all the documents and materials created throughout the project into a cohesive report. We recognized the importance of structuring the report with a clear introduction, a discussion of results in relation to the specified requirements, and a comprehensive conclusion summarizing the project's achievements.

The revised plan was structured in a way that ensured the completion of essential tasks while accounting for the limited remaining time. By prioritizing system design, specification, and implementation, we aimed to build a solid foundation that would support the subsequent stages of testing and documentation. It was important to optimize the remaining time and make efficient use of available resources to achieve project completion within the new timeframe.

In summary, the second plan was formulated by assessing the remaining tasks, prioritizing critical components, and allocating time accordingly. By focusing on the foundational aspects of the project, we aimed to streamline the implementation and testing processes and ensure the successful completion of the thesis within the revised timeline.

A.4 Conclusion

In conclusion, this experience has been instrumental in teaching us about the realities of time management and planning in the context of a complex engineering project. The challenges we faced not only tested our technical capabilities but also our project management skills. It served as a powerful reminder that effective project management requires not just meticulous planning but also the ability to adapt and adjust to unforeseen challenges. As we move forward, we believe these lessons will guide us in future endeavors, helping us to plan more effectively and manage our time more efficiently.

In retrospect, we now see that some tasks required more time than we anticipated, especially those related to new technology understanding and integration. We also did not account for the potential

of personal crises or other unforeseen events. Despite these oversights, we believe our initial plan was sound, and it was the unpredictable elements that ultimately caused the delay. Moving forward, we would aim to build more flexibility and buffer time into our planning to better accommodate unexpected obstacles and complexities.

Systematic Literature Search

B

In the initial stage of the project, an exploratory literature review was carried out to gain a broader understanding of the topic of tetraplegia and robotic assistance. This review was conducted using various search engines and databases, including Google, Pubmed, Scopus, Embase, and IEEE.

The purpose of this search was to identify relevant keywords for a subsequent systematic literature review, which focused solely on scientific databases. The systematic review aimed to uncover important information from studies related to the methods and techniques utilized by researchers in the context of tetraplegia and robotic assistance, particularly focusing on the robotics aspect.

The quality of the systematic search string was evaluated using specific criteria, and inclusion and exclusion criteria were applied to filter search results, where possible. These criteria, as detailed in Table B.1, were used to assess the relevance of the retrieved information.

Inclusion	Exclusion
Studies that focus on robotic assistance in healthcare for tetraplegic patients, involve the use of the JACO robotic arm, examine semi-autonomous robotic assistance or fine manipulation capabilities, and are published in English	Studies that focus on other patient groups or types of robotic assistance, do not involve the use of the JACO robotic arm, do not examine semi-autonomous robotic assistance or fine manipulation capabilities, and are published in languages other than English

Table B.1. Inclusion and exclusion criteria for the literature search.

B.0.1 Search string creation

Basis for literature search: This literature search is tasked with uncovering studies and articles on "ARM's in the context of healthcare assistance." What were the results? Are the findings of good quality, and does it have a good carry-over to our project? What gaps in knowledge exist in this field right now (state-of-the-art)

Research question: "What is the state of the art in robotic assistance in healthcare for tetraplegic patients and are there any gaps in knowledge or areas that need improvement?"

Different search strings were developed by including various types of operators (such as AND, OR) and MeSH terms were used when possible. Preliminary searches were conducted in the chosen databases to determine the number of results produced by each string. Some strings generated either a low number of results or an excessive number of results, which necessitated a reduction of keywords or trying other combinations of words in order to make it feasible to screen all of the articles within the given time frame of the project. This resulted in the selection of the following search string:

Search Query
(("JACO" AND ("robotic assistance" OR "robotic support" OR "robotic aids")) OR "ARM" OR "autonomous robotic manipulation") AND (("semi-autonomous" OR "manipulation control" OR "manipulation task" OR "grasping") OR ("fine manipulation" OR "dexterous manipulation" OR "precision manipulation" OR "manipulation ability")) AND (("tetraplegia" OR "quadriplegia" OR "spinal cord injuries" OR "neurological disorders" OR "assistive technology" OR "disability" OR "paralysis" OR "motor dysfunction" OR "muscle weakness" OR "upper extremity")) NOT ("brain injury" OR "cognitive impairment" OR "dementia" OR "stroke")

Table B.2. Search query for robotic manipulation in individuals with upper extremity motor impairments

The articles found by the searches in the four databases were collected and then put into Rayyan, a web-based tool used when conducting a systematic literature search. The different steps of the screening process are described in the following protocol.

1. Auto-detect duplicates on the collected literature search (across all databases included)
2. Look for any false positives and false negatives in the duplicate-detection
3. Create a list/table of accepted tags for exclusion
4. Put blinding mode on and exclude articles based on abstract and title, using the accepted tags
    a) After completion, take blinding off and resolve conflicting articles in a group setting and decide on whether to exclude/include
5. Read full text and exclude using the agreed-upon tags and inclusion/exclusion criteria
    a) After completion, resolve conflicting articles in a group setting and decide on whether to exclude/include

The screening process is depicted in the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) diagram shown in Figure B.1.

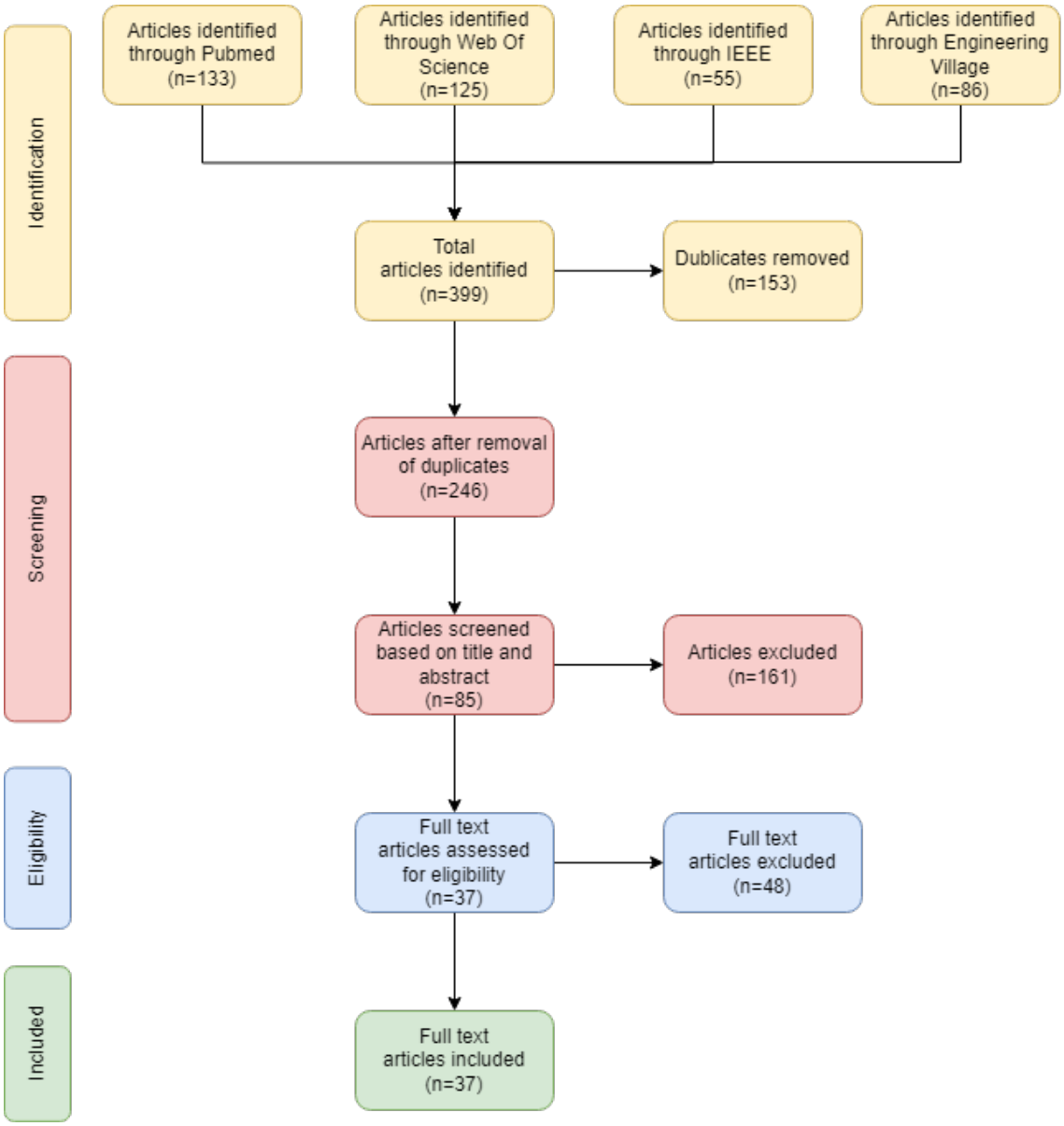


Figure B.1. PRISMA diagram for the conducted systematic literature search, which resulted in the inclusion of 37 articles

The articles and papers identified and included in the systematic literature search were used to gain insight into robotic manipulators and robotics in the context of tetraplegia. This knowledge was in turn used to write the problem analysis, to identify gaps in knowledge, and to establish current state of the art. Secondarily the knowledge was used to make system design decisions, and methodological reasonings during the development cycle.

Implementation documentation C

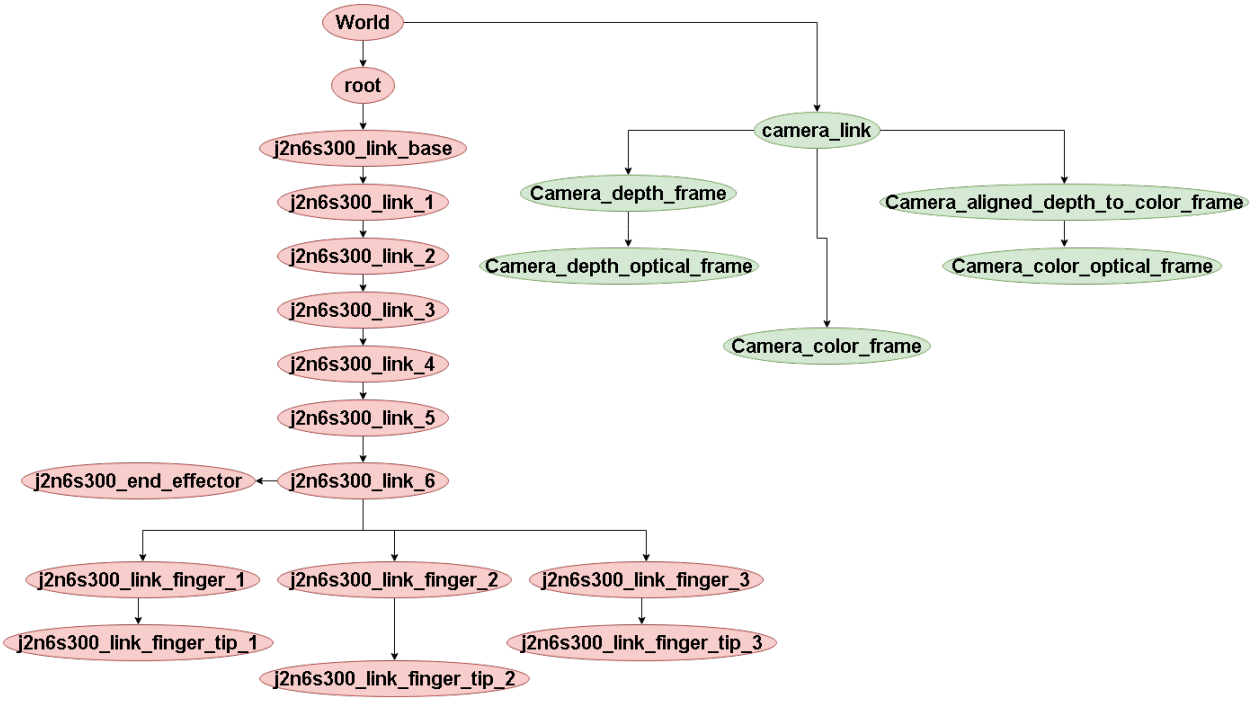


Figure C.1. An overview of the transform tree, used during system operation. The world frame defines the Jaco2s base reference. This is subsequently linked to the robots joints. The camera link defines the base reference for the realsense intel d435 camera. This is subsequently linked to the various sensors available. Camera link is linked to world, such that transforms between camera references and robot references is possible. Each link defines an XYZ translation and quarternion encoding rotation between each frame.

Test Protocols D

The following test protocols were conducted to evaluate the performance of the system in meeting the subset of the systems requirements implemented in this project.

Spatial Accuracy: Distance Estimation

Test Objective: The objective of this test was to verify the accuracy of the depth camera in estimating the spatial position of a target object placed within each of the 9 locations in the 3x3 grid.

Test Setup: The system was set up with an RGBD camera, the JACO2 ARM, and a computer for data processing. The target object was placed within the 3x3 grid.

Test Steps: The test was conducted in the following steps:

1. Object Localization:

   - The camera feed was initialized, and the YOLOv7 algorithm was applied for object detection.
   - The center point of the detected object was estimated.

2. Depth Mapping:

   - The estimated center point was mapped to the depth map obtained from the camera's depth sensors.
   - A 3D depth map coordinate was obtained for the center point, representing its spatial position.

4. Point Cloud Generation:

   - The 3D depth map coordinate was converted into a point cloud representation.
   - The object's surface was represented in three-dimensional space using the generated point cloud.

5. Spatial Reference Transformation:

   - The spatial coordinates of the object's center point were transformed from the camera's reference frame to the robot's spatial reference frame.
   - The object's position relative to the robot was evaluated based on the transformed coordinates.

6. Distance Comparison:

   - The estimated spatial coordinates of the object's center point were compared with the real-world measurements of the corresponding point on the object.
   - The Euclidean distance between these two points was calculated as a measure of the depth camera's accuracy in estimating the object's position.

Expected Results: The depth camera was expected to accurately estimate the spatial position of the target object within each grid location. The Euclidean distance between the estimated and real

spatial points was expected to meet the defined success criteria of 0.03 m or lower.

Object Selection Requirement

Test Objective: The objective of this test was to verify the system's ability to enable users to select objects in the scene using the keyboard input accurately.

Test Setup: The system was set up with the JACO2 robotic manipulator, RGBD camera, and a computer for visualizing and selecting objects. The camera feed was configured to display green bounding boxes around the objects, and the objects of interest were placed within the manipulator's range.

Test Steps: The test was conducted in the following steps:

1. Object Selection Accuracy:

- The user was presented with a series of objects on the computer screen, each marked with a green bounding box.
- The user selected an object by pressing the corresponding numeric key on the keyboard.
- The system was expected to accurately identify and highlight the selected object with a blue bounding box.

2. Success Rate Measurement:

- Multiple rounds of object selection attempts were conducted.
- The number of successful object selections made by the user out of the total attempts was recorded.

3. Selection Accuracy Measurement:

- The accuracy of the system's identification of the selected objects was evaluated.
- The user's intended selection was compared with the system's highlighted object.

Expected Results: The system was expected to accurately identify and highlight the selected object with a blue bounding box. The success rate was expected to meet or exceed the defined success criteria of 95% or higher success rate, and the selection accuracy was expected to meet or exceed the defined success criteria of 95% or higher selection accuracy.

Measures: The following measures were used for evaluation:

- **Euclidean Distance**: The measure of accuracy between the estimated and real spatial points, calculated as the Euclidean distance.
- **Success Rate**: Percentage of successful object selections out of the total number of attempts.
- **Selection Accuracy**: Percentage of correctly identified selected objects.

These two test protocols were conducted to comprehensively evaluate the system's performance in terms of spatial accuracy in distance estimation and object selection capabilities.

Test Protocol for Autonomous Grasp Control Requirement

Test Objective:
Verify the system's ability to autonomously control the manipulator to perform grasp tasks for selected objects.

Test Setup:

- Set up the system with a robotic manipulator, RGB camera, and a computer for visualizing and selecting objects.
- Configure the system to enable autonomous control of the manipulator for grasp tasks.
- Implement a mechanism for smooth transition between user control and autonomous control.

Test Steps:

1. Autonomous Grasp Execution:
    - The user selects an object for grasp using the object selection method from the previous test.
    - Initiate the grasping task and observe the system's autonomous control of the manipulator.
    - Measure the grasp success rate: Percentage of successful grasps out of the total number of attempts.

Expected Results:

- The system autonomously controls the manipulator to perform grasp tasks accurately.

Success Criteria:

- The system should achieve a grasp success rate of at least 90%.

D.1 Manipulation of JACO2 Robot ARM

D.1.1 From Home Position to Object Position

An experimental procedure was designed and executed to evaluate the proficiency of an autonomous robotic system in spatial point navigation. This was achieved by leveraging the information derived from the robot's integrated depth and RGB sensors.

The experimental setup involved the strategic placement of one object in the 3x3 grid system. Consequently, a total of twenty-seven tests were conducted.

The primary objective of these tests was to ascertain the robot's capability to accurately navigate to a specified spatial point, as determined by the processing of data from its depth and RGB sensors. This was deemed a critical competency, given its direct implications on the robot's operational effectiveness in real-world scenarios.

The success criterion for each test was defined as the robot's ability to establish physical contact with the designated object. This criterion was chosen as it provides a tangible and unequivocal measure of the robot's navigational accuracy.

The results of the tests were systematically documented in a 3x3 matrix, with each cell representing a specific grid position. Within each cell, the outcomes for the three tests for each grid position were recorded in a binary success/fail format. This method of presentation facilitated a clear and concise overview of the robot's performance across the different grid positions.

D.1.2 From Home Position to Target with Orientation and Position

An experiment was designed and conducted to assess the ability of the JACO2 robotic arm to generate and execute a grasp pose. The system's integrated depth and RGB sensors, along with the GPD algorithm, were utilized to determine the optimal grasp poses.

The test aimed to evaluate the robot's capability to transition from a home position to a target object, approach the object appropriately, and then grasp it using its end-effector. This sequence of operations is crucial in robotic manipulation tasks.

The experimental setup involved placing an object on the 3x3 grid. As a result, twenty-seven tests were conducted, each corresponding to a unique grid position and three trials on each position.

The success criterion was defined as the robot successfully grasping the object such that it could be lifted without any slippage.

The results of the tests were documented in a 3x3 matrix, with each cell representing a specific grid position. Within each cell, the outcomes for the three trials were recorded as a binary success/fail. This method provided a clear overview of the robot's performance across different grid positions and objects.