Bachelor Project

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Control Engineering

Human-inspired Robot Hands Control

Filip Žemlička

Supervisor: prof. Ing. Jan Faigl, Ph.D.
Field of study: Cybernetics and Robotics
May 2024

BACHELOR‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name:	**Žemlička Filip**	Personal ID number:	**499313**
Faculty / Institute:	**Faculty of Electrical Engineering**		
Department / Institute:	**Department of Control Engineering**		
Study program:	**Cybernetics and Robotics**		

II. Bachelor’s thesis details

Bachelor’s thesis title in English:

Human-inspired Robot Hands Control

Bachelor’s thesis title in Czech:

Ovládání robotických rukou inspirované člověkem

Guidelines:

Familiarize yourself with force control and grasping tasks [1], existing control approaches [2], and particular control of the robotics hands RH6D and RH8D [3] using the Dynamixel Protocols (1.0 and 2.0) with the available software interface [4].

Develop visualization and simulation platform using available Unified Robotics Description Format (URDF) [5] model of the hands and simulation environment [6].

Propose and implement hand control in a demonstration task of signed language for selected symbols [7].

Prospect the developed system in hand gesture recognition [8] and grasping soft objects [9].

Experimentally evaluate the developed solution using RH8D hands.

Bibliography / sources:

[1] K. M. Lynch: Modern robotics: mechanics, planning, and control, Cambridge University Press, 2017.
[2] Y. Li, P. Wang, R. Li, M. Tao, Z. Lium H, Qiao: A Survey of Multifingered Robotic Manipulation: Biological Results, Structural
Evolvements, and Learning Methods, Frontiers in Neurorobotic, Volume 16. 2022.
[3] Human-inspired, Dexterous, Robot Hands and Manipulators
https://www.seedrobotics.com/robot-hands-manipulators-introduction, cited on 2024-02-20.
[4] Seedrobotics repository, https://github.com/seedrobotics/, cited on 2024-02-20.
[5] URDF: Unified Robotics Description Format, https://wiki.ros.org/urdf, cited on 2024-02-20.
[6] CoppeliaSim, https://www.coppeliarobotics.com/, cited on 2023-10-16.
[7] I. A. Adeyanju, S. O. Alabi, A. O. Esan, B. A. Omodunbi, O. O. Bello, and S. Fanijo: Design and prototyping of a robotic hand
for sign language using locally-sourced materials, Scientific African, Volume 19, 2024.
[8] A. Haria, A. Subramanian, N. Asokkumar, S. Poddar, J. S Nayak: Hand Gesture Recognition for Human Computer Interaction,
Procedia Computer Science, 115:36-374, 2017.
[9] Z. Deng, Y. Jonetzko, L. Zhang, J. Zhang: Grasping Force Control of Multi-Fingered Robotic Hands through Tactile Sensing for
Object Stabilization, Sensors, 20(4), 2020.

Name and workplace of bachelor’s thesis supervisor:

prof. Ing. Jan Faigl, Ph.D. Artificial Intelligence Center FEE

Name and workplace of second bachelor’s thesis supervisor or consultant:

Date of bachelor’s thesis assignment: **20.02.2024** Deadline for bachelor thesis submission: **24.05.2024**

Assignment valid until:
by the end of summer semester 2024/2025

prof. Ing. Jan Faigl, Ph.D. Supervisor’s signature	prof. Ing. Michael Šebek, DrSc. Head of department’s signature	prof. Mgr. Petr Páta, Ph.D. Dean’s signature

III. Assignment receipt

The student acknowledges that the bachelor’s thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the bachelor’s thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student’s signature

Acknowledgements

I would like to express my gratitude to my thesis supervisor, Prof. Ing. Jan Faigl, Ph.D., for his excellent guidance throughout my research. I am also grateful to Ing. Jiří Kubík for providing functional code and for his explanations. Additionally, I would like to thank the Department of Computer Science for the access to the laboratory where I was able to work with the robotic hand.

Declaration

I declare that I have completed the submitted work independently and that I have cited all the literature used.

In Prague, 28. May 2024

Abstract

The presented bachelor thesis focuses on the development and evaluation of control methods for the robotic hands RH6D and RH8D by Seed Robotics. The thesis starts with an overview of the force and object grasp control methods. Then, the background of the hands utilized by the Dynamixel communication protocols is described. The developed hand control is demonstrated in the sign language performance of selected symbols. Besides, recognition of hand gestures or grasping of soft objects is discussed. Real experiments using RH8D hands are conducted to evaluate the performance of the proposed solutions in sign language demonstration and grasping tasks. We evaluate the system's performance in grasping tasks and sign language demonstrations.

Keywords: Force control, grasp task, robotic hands, Dynamixel protocols, sign language demonstration, hand gesture recognition, grasping objects

Abstrakt

Prezentovaná bakalářská práce se zaměřuje na vývoj a hodnocení metod řízení pro robotické ruce RH6D a RH8D od společnosti Seed Robotics. Práce začíná přehledem metod řízení síly a uchopení objektů. Následně je popsáno pozadí rukou využívajících komunikační protokoly Dynamixel. Vyvinuté řízení ruky je demonstrováno na výkonu znakového jazyka vybraných symbolů. Dále je diskutováno rozpoznávání gest rukou nebo uchopení měkkých objektů. Skutečné experimenty s rukami RH8D jsou provedeny za účelem vyhodnocení výkonu navrhovaných řešení v demonstraci znakového jazyka a úkolech uchopení. Hodnotíme výkon systému v úkolech uchopení a demonstracích znakového jazyka.

Klíčová slova: Řízení síly, úkol úchopu, robotické ruce, protokoly Dynamixel, demonstrace znakového jazyka, rozpoznávání gest rukou, úchop objektů

Contents

Figures

Tables

Chapter 1

Introduction

This thesis focuses on advanced methods of controlling robotic hands, particularly the RH6D and RH8D models from Seed Robotics [1]. These subjects are selected due to the growing importance of robotic hands in various areas, from industrial manufacturing to medical assistance services, where precise and effective control of these systems plays a key role in successful object manipulation in diverse environments.

The study begins with an in-depth exploration of existing approaches to force control and object manipulation, providing an overview of current technologies in the field. Special attention is paid to the RH6D and RH8D robotic hands, controlled via Dynamixel protocols. Emphasis is placed on understanding their capabilities and limitations, which are crucial for optimizing their application. Additionally, the thesis discusses why creating a visual simulation environment for the RH8D model is impractical. The main argument is that the RH8D hand has 19 degrees of freedom but only 8 actuators with feedback capability, creating an "open system" where each degree of freedom is not equipped with required feedback. The work continues with the importance and implementation of controlling the RH8D hands in sign language tasks, striving to create as many letters of American Sign Language(ASL) as possible. A method is described using an existing program that recognizes these symbols in humans to verify the robotic hand's ability to reproduce sign language symbols. The last section deals with implementing control of the RH8D robotic hand for object gripping. The hand was subsequently subjected to several tests to verify its ability to grasp objects effectively.

Since the goal of the thesis is to control the robotic hands RH8D and RH6D by Seed robotics, an overview of the hands is provided in Section 1.1. Then, we delve into the principles of force and grip control, discussing where and why force control is more beneficial than position control, and explore issues related to grip quality in Chapter 2. We briefly examine various existing approaches, such as learning from human behavior, tactile sensor feedback, and the fusion of tactile and visual data. However, based on the approach review, we opted for tactile sensor feedback in the developed solution. Subsequently, existing control methods are discussed, including reinforcement learning, imitation learning, and hybrid strategies. The methods are crucial for understanding and developing sophisticated control systems that enable precise manipulation in various tasks. Finally, we overview sign languages in Section 1.2

We also include a brief introduction to sign language, which serves as the foundation for demonstrating the implementation of sign language into the RH8D robotic hand. The presented introduction provides context and understanding of the sign language symbols that can be reproduced and how the robotic hand's control system can effectively handle these gestures.

1.1 Robotics Hands RH8D and RH6D

Robot hands are a crucial component in the field of robotics, enabling machines to interact with the physical world in a manner similar to human hands. Seed Robotics specializes in designing and manufacturing advanced robot hands and manipulators that empower interdisciplinary research in areas such as robotics, artificial intelligence, neuroscience, psychology, and cognitive science.

The RH6D robotic hand, depicted in Fig. 1.1, is a compact and technologically advanced robotic arm designed for precision object handling. The RH6D robotic hand has 15 DoF, including an opposable thumb. It provides a wide range of object-handling options with high precision. An integrated sensor system provides position, velocity, and current feedback for each actuator, allowing fine control of movement and estimation of applied forces. It is compatible with ROS [3] and PyPot [4], allowing easy integration into different robotic systems.

The arm has a load capacity of 750 g in vertical thrust and 450 g in 3D space that allows it to manipulate small objects without any problem. The RH6D is designed for easy maintenance and modification thanks to its modular design, which allows easy replacement and modification of individual parts.

Figure 1.1: Visualization of the RH6D robotic hand, adopted from [1].

The RH8D Adult-size Dexterous Robot Hand, depicted in Fig. 1.2, epitomizes the forefront of robotic manipulation technology, representing a culmination of advanced engineering principles tailored for complex manipulation tasks. Developed by Seed Robotics, the RH8D stands as a testament to the convergence of dexterity, mechanical efficiency, and sensory acuity in robotic appendages, catering to the exigencies of contemporary robotics research and industrial applications alike.

Comprising 19 *Degrees of Freedom* (DoF) orchestrated by 8 smart actuators, the RH8D manifests an unparalleled versatility in its ability to articulate and manipulate its appendages with precision and agility. Employing a distributed actuation architecture, all actuators are seamlessly integrated within the hand, affording compactness and facilitating ease of integration into various robotic platforms. Key to its operational efficacy is the incorporation of advanced sensory modalities, enabling real-time, high-fidelity feedback on critical parameters, including positional data, velocity profiles, and current consumption. Such sensory feedback mechanisms empower precise control and adaptive response strategies, which are indispensable for tasks necessitating intricate manipulation and dynamic interaction with the environment.

Regarding the payload capacity, the RH8D boasts a commendable payload capacity of 1 kg in three-dimensional space, augmented by a vertical pull capacity of 2.5 kg. The robust payload capability renders the hand well-suited for a gamut of applications ranging from delicate object manipulation to tasks requiring substantial force exertion. The RH8D is also compatible with ROS [3] and PyPot [4] as the smaller RH6D. Two specific highlights of the RH8D hand are as follows.

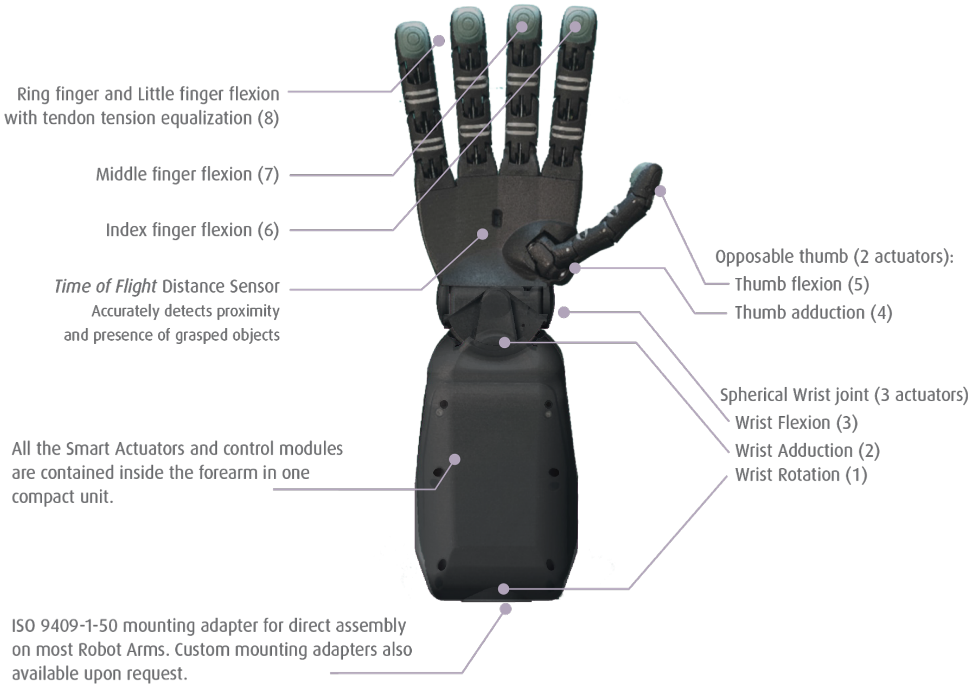


Figure 1.2: Overview of the RH8D robotic hand, adopted from [1].

Advanced Sensory Integration – Equipped with a palm-mounted *Time-of-Flight* (TOF) distance sensor and optional capacitive touch pads, the RH8D engenders rich human-robot interaction paradigms. These sensory modalities facilitate nuanced perception and tactile feedback, fostering intuitive manipulation strategies and enhancing situational awareness.

Compact Form Factor – Despite its formidable capabilities, the RH8D maintains a judicious balance between functionality and form factor, weighing a mere 650 g. The lightweight construction not only augments its portability and maneuverability but also broadens its applicability across diverse robotic architectures and deployment scenarios. The RH8D epitomizes a confluence of cutting-edge engineering principles, operational robustness, and ergonomic design considerations, thereby delineating new benchmarks in the domain of robotic manipulation. From industrial automation and manufacturing workflows to research endeavors in artificial intelligence and human-robot interaction, the RH8D emerges as an exemplar of technological innovation poised to redefine the contours of modern robotics.

1.2 Introduction to Sign Language

Sign language is an intricate and fully-fledged system of communication extensively utilized by the deaf and hard-of-hearing communities. The visual language comprises a diverse array of hand signs, facial expressions, and body postures that express a wide range of subtle linguistic features. Unlike spoken languages, sign languages are not universal and vary widely from one geographic region to another. Each sign language, such as *American Sign Language* (ASL), British Sign Language, or Japanese Sign Language, has its own unique set of rules governing syntax, morphology, and semantics.

In the context of the thesis, we have employed ASL as the medium for demonstrating sign language through robotic hands. ASL is among the most widely used sign languages in the world, particularly in the United States and parts of Canada. It has a grammar system that is distinct and independent of English grammar. ASL includes its own set of rules for the arrangement of sentences, conveying of concepts, and establishment of tone and inflections, much like any spoken language.

The ASL alphabet, depicted in Fig. 1.3, is instrumental in understanding the basic hand configurations used in ASL. Through the utilization of ASL, the thesis project aims to demonstrate the potential of robotic hands to replicate human-like gestures, thus making strides toward more inclusive and effective human-robot interactions.

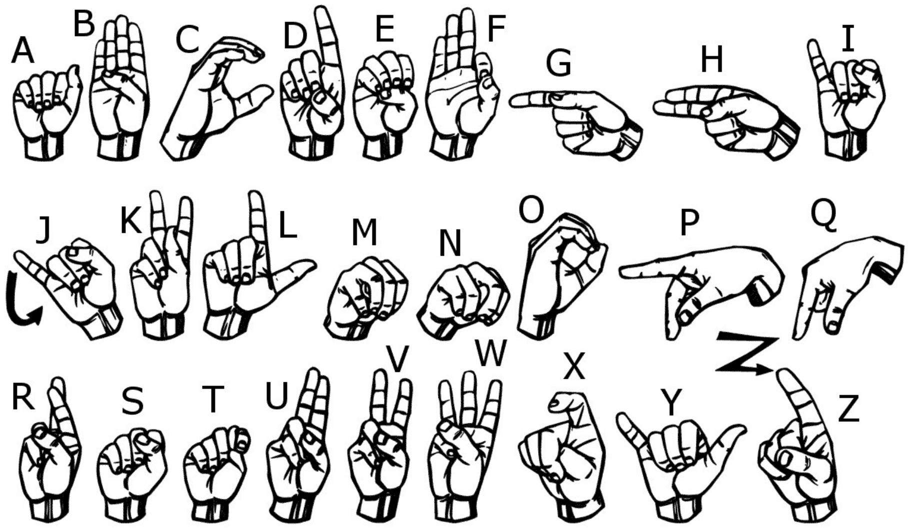


Figure 1.3: ASL Sign alphabet, adopted from [2].

Chapter 2

Robotics Hand Control Methods

2.1 Principles of Force and Grip Control

Effective force and grip control are fundamental for the sophisticated handling capabilities of robotic systems, particularly in fields requiring delicate manipulation such as advanced manufacturing, healthcare, and service industries. The section delves into four primary methodologies employed to manage the force and grip exerted by robotic arms. It highlights the conventional analysis-based approach, the strategy of learning from human examples, the implementation of tactile sensors, and the advanced technique of integrating tactile and visual data to enhance perception and control in robotic systems. An overview of methods as of [5] is summarized in the following paragraphs.

- **Force Control** – Force control is an integral aspect of robotics focusing on how robots apply force to interact with their environment, rather than merely controlling their movement or position. As outlined in [5], it is about manipulating the robot's actuators to produce desired force outcomes essential for tasks that require precise force application like assembly operations, tactile exploration, and surgical interventions. That control paradigm is crucial in scenarios where simple positional commands are insufficient, and the robot must adapt to the physical properties of the environment to perform tasks effectively.

- **Analysis-Based Grip Quality: Formal and Force Closure** – The initial method of controlling grip involves a detailed analysis of hand behaviors and the calculation of various grip quality measures, such as formal and force closure. These metrics are vital for optimizing the robotic grip, thereby stabilizing the objects during manipulation. The approach includes the mathematical modeling of objects and their interactions with the robotic hand, facilitating precise calculations needed for effective force and grip control. Although highly effective, this traditional method generally requires accurate object models, posing significant challenges when the robot encounters unknown or irregularly shaped objects that lack predefined models.

- **Learning from Human Behavior: Data-Driven Grasp Optimization** – Moving beyond traditional models, the second class of methods employs a learning-based approach that teaches robotic hands stabilization strategies directly from human examples. The technique involves gathering extensive data on successful human grasps and the specific contexts in which they occur. The collected data is then used to train sophisticated models that can predict optimal actions for object stabilization. While promising, the approach hinges on vast data requirements and is most effective with objects similar to those in the training set, thereby limiting its utility with novel or unfamiliar items.

- **Tactile Sensor Feedback: Enhancing Real-Time Interaction** – The third method enhances robotic manipulation through the integration of tactile sensors, such as the BioTac sensors discussed in research on multi-fingered robotic hands. These sensors provide invaluable real-time data concerning the interaction between the robotic hand and the objects, including advanced features like slip detection, material identification, and precise contact force estimation. Armed with the data, robotic systems can dynamically adjust grip strength and modify manipulation strategies to stabilize objects effectively in real-time, significantly enhancing the robot's manipulation capabilities. *The method is utilized in our development of a robotic hand.*

- **Fusion of Tactile and Visual Data: Comprehensive Sensory Integration** – The fourth and most advanced method involves the fusion of tactile feedback with visual data to create a comprehensive sensory integration system. Visual sensors add another layer of information about the object's shape, material properties, and the surrounding environment. The data can be synergistically combined with tactile feedback to construct a more detailed understanding of the manipulation task. Such a multimodal approach not only enriches the perception capabilities of the robot but also enhances its control mechanisms, allowing for more robust and adaptable force and grip control across a broader range of objects and tasks. The integration helps mitigate some of the limitations associated with relying solely on tactile or visual data, thus providing a significant advancement in robotic manipulation technologies. *However, regarding the expected time frame for implementing sign language demonstration, the method has not been used in favor of the former one.*

2.2 Adapting to Uncertainty in Object Properties

A significant challenge in robotic manipulation is dealing with objects that have uncertain or unknown properties. The authors of [6] explore innovative methods to enhance force and grip control by dynamically adapting to the uncertainty in an object's characteristics. It discusses how robots can employ advanced sensing strategies to better understand and interact with their environment, especially when conventional models and data are insufficient. The objects can be categorized based on their geometric and physical uncertainties and specific strategies are briefly discussed in the rest of the subsection.

1. **Geometric-Uncertain Objects** have uncertainties in their geometric features such as shape, position, or pose. Handling such objects requires real-time sensing and dynamic updates to the robot's model and strategies to adapt to changes in these geometric parameters during manipulation.
2. **Physical-Uncertain Objects** present uncertainties in their physical properties, such as mass or rigidity, which can significantly affect the grasping strategy. Robotic systems may need to adjust the force and grip dynamically based on real-time physical property estimates to handle such objects effectively.

3. **Unknown Objects** are the most challenging as they have both geometric and physical uncertainties or completely unknown properties. The category requires a more exploratory approach, utilizing a combination of sensing strategies and possibly learning from previous interactions to build a model of the object's properties on-the-fly.

Particularly relevant to our discussion on the fusion of tactile and visual data, also underscored in [6], is the importance of integrating multiple sensory inputs to deal with objects whose physical properties are not well-known. By combining sensors that provide different types of data, such as force/torque sensors with visual sensors, robots can form a more complete picture of their interaction with an object, thus enabling more precise and adaptive manipulation strategies.

A notable research effort in this area is the comprehensive investigation into the development and deployment of a sophisticated system designed for the precise estimation of both static and dynamic physical attributes of objects manipulated by robotic systems [7]. The primary objective is to furnish robots with the capacity to dynamically adjust and finely tune the exerted force and gripping mechanisms on objects, utilizing real-time, advanced sensorial feedback. The RobotScale framework [7] exemplifies that by facilitating the estimation of an object's physical properties through data acquired from tactile sensors positioned on the robot's fingertips and torque sensors integrated within the joint mechanisms. That encompasses the magnitude of force and torque that the robot encounters during its interaction with various objects.

2.3 Existing Control Approaches

In this section, we overview existing control techniques for multi-fingered robotic hands that are integral to contemporary robotic systems. It delves into methods such as reinforcement learning, imitation learning, synergistic control, and hybrid approaches that integrate multiple strategies to enhance the adaptability and dexterity of robotic hands. We reference specific methodologies discussed in [8], which provides a relatively comprehensive overview of the advancements in robotic manipulation.

- **Reinforcement Learning (RL)** is a powerful tool in robotics, used to optimize the actions of robots by leveraging rewards received from interactions with their environment. That optimization enhances the robots' manipulation capabilities significantly. Within RL, there is a distinction between model-based and model-free approaches, each offering unique benefits for various manipulation tasks. Model-based RL utilizes a structured learning framework based on a comprehensive understanding of environmental dynamics, facilitating predictive and strategic planning. In contrast, model-free RL relies on empirical learning, where the robot learns optimal behaviors through trial and error without needing a detailed model of the environment's dynamics. The learning process is governed by the Bellman equation used to mathematically describe the optimal value function for each state, helping to guide the decision-making process effectively

  $$V^*(s) = \max_a \left(R(s,a) + \gamma \sum_{s'} P(s'|s,a) V^*(s') \right), \tag{2.1}$$

  where $V^*(s)$ is the optimal value function for the state s, $R(s,a)$ is the reward for the action a at the state s, γ is the discount factor, and $P(s'|s,a)$ is the probability of transitioning to a new state s'.

- **Imitation Learning** enables robots to acquire complex manipulation skills directly from human operators by observing and replicating human movements and actions. The approach is particularly effective for transferring sophisticated motor skills and cognitive processes from humans to robots, which can be crucial for tasks requiring high levels of dexterity and cognitive decision-making. Imitation learning bridges the gap between human expertise and robotic execution, making it an invaluable tool for enhancing the capabilities of robotic systems in practical applications.

- **Synergistic** control strategies are inspired by the natural coordination observed in human hand movements, where multiple motions or actions are initiated concurrently to perform complex tasks. These methods aim to reduce the complexity of the robots' action spaces and enhance their manipulation efficiency and naturalness. By mimicking the integrated and multifaceted strategies humans use, synergistic methods allow robotic systems to perform more dynamically and adaptively, which is crucial for handling intricate manipulation tasks.

- **Hybrid Methods:** The necessity for hybrid control methods in advanced robotic manipulation is underscored, highlighting their role in combining various control and learning strategies to achieve superior manipulation capabilities. Hybrid approaches might integrate elements of model-based and model-free learning, synergistic techniques, and imitation learning to create a versatile and robust control system. These approaches are designed to enhance the adaptability and universality of robotic hands, allowing them to perform a wide range of tasks more effectively and with greater precision. The integration of diverse learning and control methods enables robots to adapt to new situations and manipulate objects in innovative ways that mimic human-like dexterity.

These control approaches collectively represent the forefront of robotic manipulation technology, driving the development of more capable, flexible, and sophisticated robotic systems. By leveraging the strengths of each method, robotic hands can achieve levels of performance that were previously unattainable.

Chapter 3

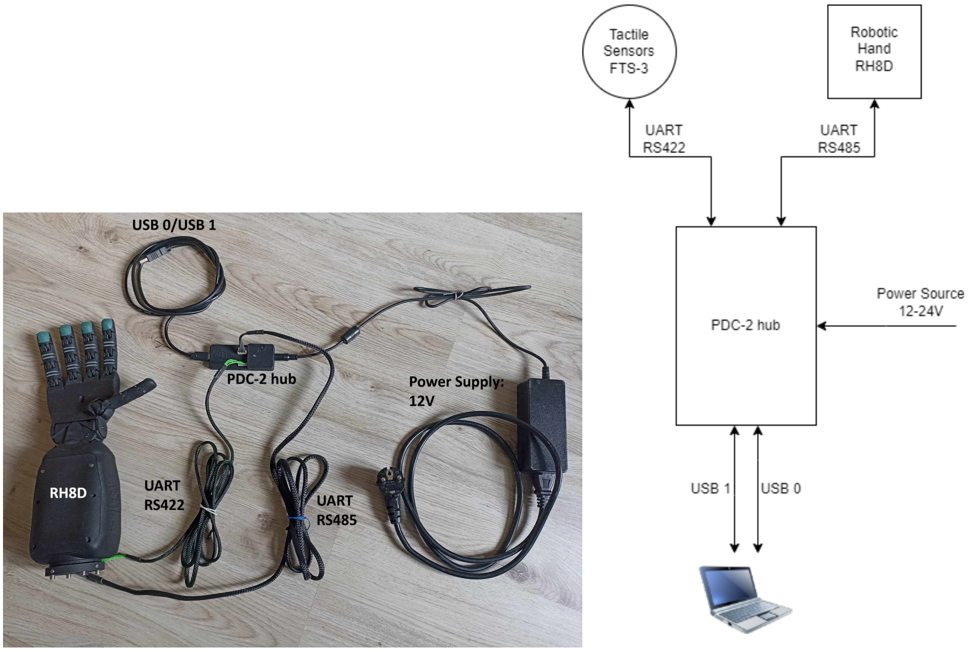
Implementation Background

We outline the comprehensive methodology employed in the development and implementation of the control systems for the RH8D robotic hand, which is selected for implementation of the hand control in sign language demonstration. The RH8D robotic hand was selected for the implementation of sign language due to its anatomically closer resemblance to a human hand, featuring five fingers compared to the three fingers of the RH6D robotic hand. That choice enhances the naturalness and accuracy of sign language gestures, making the RH8D more suitable for our tasks. The focus is on describing the tools and techniques used to enable the robotic hand to interact with the control program to execute sign language interpretation and object manipulation. The description is structured into development tools, including sensors used and communication protocol. It involves detailing the development environment, hardware interfaces, and sensor technologies that collectively form the foundation of our experimental setup.

3.1 Development Tools

Python programming language has been chosen for the development environment to interact with the robotic hand due to its versatility and support for the communication protocols to facilitate communication with the hand. The based implementation was initially developed by Jiří Kubík as a primary example program with the essential functions, such as `set_pose` to set the positions of the fingers and palm. The communication from the control computer with the hand is via *Universal Serial Bus* (USB) connected to the

PDC-2 hub that is a part of the hand, see Fig. 3.1 The hub is linked to the hand through a *Universal Asynchronous Receiver-Transmitter* (UART) RS485 connection, ensuring a direct and stable communication pathway. The communication between the PDC-2 hub and the tactile sensors is facilitated through UART RS422.



(a) Connection of PDC-2 with RH8D.

(b) Block Diagram of Connection PCD-2 with RH8D.

Figure 3.1: RH8D with the PDC-2 hub.

3.2 Tactile Sensors

Tactile sensors are essential components that enable robotic hands to sense and manipulate objects through touch. The utilized RH8D hand is equipped with the FTS3 [9], a three-axis pressure sensor from the Seed Robotics' FTS series. It provides high-resolution measurements across three axes (x, y, and z), enabling robotic hands to adjust their grip dynamically. The sensors can detect slight pressure changes due to their high sensitivity, with a resolution of 1 mN (0.1 g), and can measure forces up to 30 N (3 kg).

These sensors are directly implemented in the RH8D robotic hand, fully integrated into the fingers, see Fig. 3.2. Their ability to detect nuanced forces makes them suitable for precise manipulation and grasping tasks. The integration of these sensors allows for the real-time visualization of force

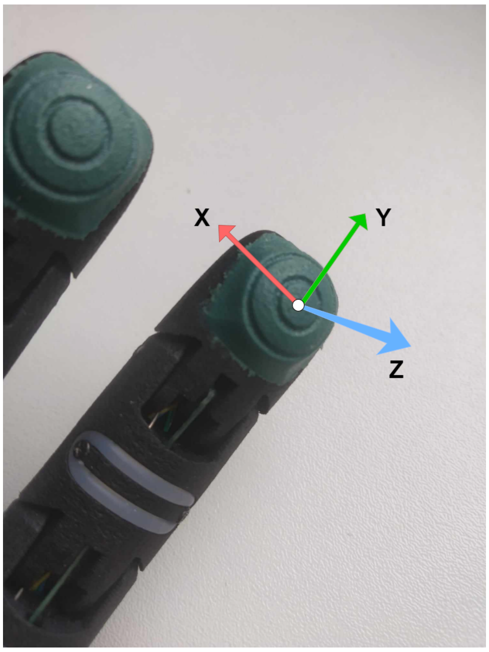


Figure 3.2: A detail of the FTS3 tactile sensors of the RH8D robotic hand.

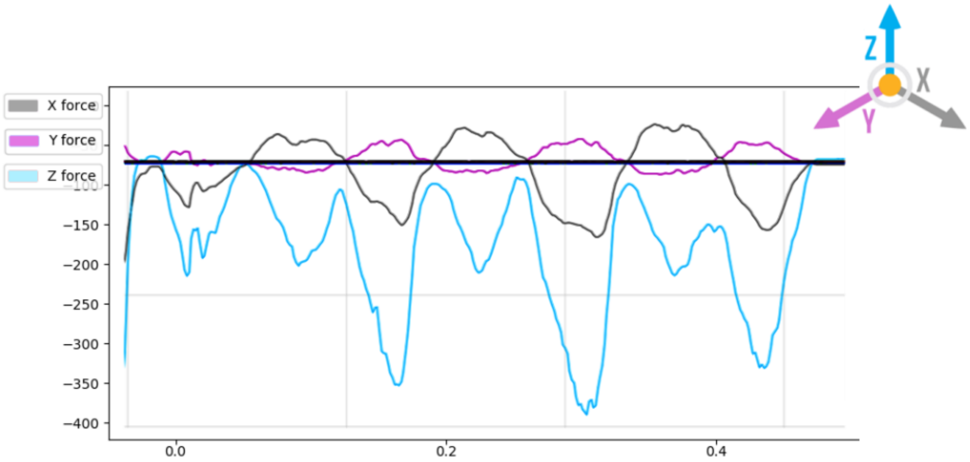
dynamics. For instance, the attached 2D plot in Fig. 3.3a illustrates the force components along the x, y, and z axes over time, providing a detailed insight into the interaction dynamics between the robotic hand and the objects it manipulates. Besides, a more intuitive and comprehensive understanding of force directions and magnitudes can be acquired from 3D plots as in Fig. 3.3b, using `pyplot` and animation from `matplotlib` library [10]. The visualization highlights the spatial dynamics of force application and also enhances the ability to monitor and adjust the robotic hand's actions in real-time.

The FTS3 sensors communicate with a computer via a serial connection facilitated by the FTDI chip, bridging to the computer's USB port. Upon connection, a serial device port is created, such as `COMx` on Windows or `/dev/ttyUSBx` on Linux. The sensor array, once powered, transmits data readings in ASCII text format (*Comma-separated Values* – CSV) at a baud rate of $1\,\mathrm{Mbit\,s^{-1}}$ and a frequency of 50 Hz. The data output includes a timestamp and three coordinates, x, y, and z, for each sensor, detailing the force vector in 3D space. [11]

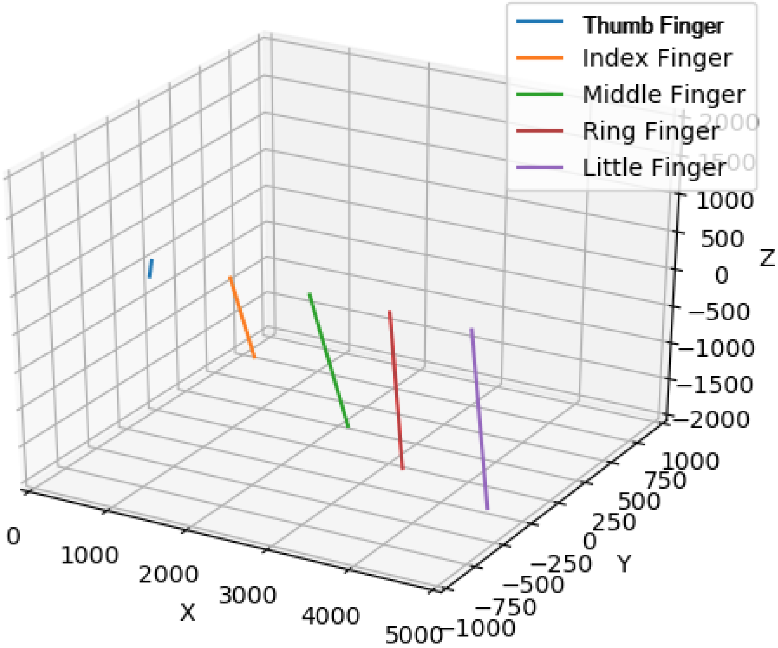
Typical data output looks like

```
@,377,634,−20,15,−943,−44,212,−804,,,,306,−172,−392,−106,−77,−941,
```

where the values are as follows.



(a) 2D force profile of one FTS-3 tactile sensor, adopted from [1]



(b) Five 3D force vectors from FTS-3 tactile sensors in the RH8D robot hand (in Newtons).

Figure 3.3: Example of force evolution displayed in x, y, and z that is used to analyze specific motion commands for grip control.

- The first column identifies the line type (`@` for a reading).
- The second and third columns provide the timestamp in seconds and milliseconds.
- Subsequent columns provide the x, y, and z axis values for each sensor, with empty fields indicating disconnected or undetected sensors.

3.3 Communication Protocols

Since the RH8D is equipped with the servomotors by Robotis, the communication can be realized by the Dynamixel Protocol 1.0 [12], which has been designated to be used with a number of robotic devices, especially servomotors and intelligent actuators manufactured by Robotis. The protocol is designed to provide a flexible yet simple interface for controlling and monitoring a wide range of robotic components. The hand is further equipped with the FTS3 tactile sensors, providing multi-dimensional force feedback essential for handling complex manipulation tasks.

The key features of the Dynamixel Protocol 1.0 are as follows.

- **Unified communication model** – The protocol utilizes a primary-secondary model[1], where the primary control unit communicates with one or more servomotors or actuators that act as secondary units. The model allows for simple and efficient control of complex robotic systems.
- **Half-duplex UART communication** – The protocol uses a half-duplex serial bus for data transmission, enabling communication over a single data cable. It simplifies the hardware and reduces the cost of cabling.
- **Addressability** – Each device on the bus has its unique `ID`, allowing the master to address individual components independently. It is crucial for systems that require individual control of multiple motors or sensors.
- **Control tables** – Each device contains a control table that is essentially a set of registers that can be read or written. These tables hold information about the device's status (e.g., position, speed, current) and allow users to set various device parameters, such as target position.
- **Flexibility and scalability** – The protocol enables the easy addition of new devices to the system without the need for significant changes in configuration or programming. The modularity and scalability make Dynamixel Protocol 1.0 suitable for a diverse range of applications, from educational projects to complex industrial applications.

A structure of the instruction packet and status packet is depicted in Fig. 3.4 and Fig. 3.5, respectively. The instruction packet consists of several fields, starting with a two-byte header (`0xFF`, `0xFF`) indicating the start of

---

[1] Formerly master-slave model.

Header1	**Header2**	**Packet ID**	**Length**	**Instruction**	**Param 1**	**...**	**Param N**	**Checksum**
0xFF	0xFF	Packet ID	Length	Instruction	Param 1	...	Param N	CHKSUM

Figure 3.4: Instruction packet of the Dynamixel Protocol 1.0.

the packet. It is followed by the `ID` byte that specifies the address of the Dynamixel unit targeted by the instruction. The `Length` byte specifies the length of the instruction packet from the `Length` byte to the `Checksum` byte. The `Instruction` field designates the type of operation to be performed on the unit, while `Param 1` to `Param N` fields represent the parameters of that instruction. The final byte `Checksum` is a calculated value based on the previous bytes, and it is used to detect possible transmission errors.

Header1	**Header2**	**Packet ID**	**Length**	**Error**	**Param 1**	**...**	**Param N**	**Checksum**
0xFF	0xFF	ID	Length	Error	Param 1	...	Param N	CHKSUM

Figure 3.5: Status packet of the Dynamixel Protocol 1.0.

The status packet, see Fig. 3.5, serves as a response to the instruction packet. It has a similar structure with the `ID` byte following the header to identify the unit sending the response. The `Length` byte indicates the length of the status packet, and the `Error` byte signals whether an error occurred while executing the instruction. The subsequent bytes `Param 1` to `Param N` represent the data returned by the unit, similar to the instruction packet. The last byte `Checksum` is used to confirm the integrity of the data.

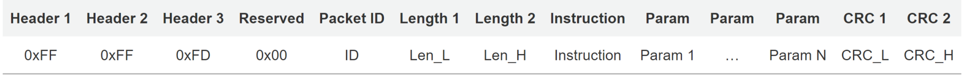
3.3.1 Dynamixel Protocol 2.0

The Dynamixel Protocol 2.0 [13] is the advancement of its predecessor, also developed by the Robotis. It offers enhanced features and capabilities specifically designed to meet the evolving demands of modern robotics. Key features are highlighted in the following paragraphs in comparison with Dynamixel Protocol 1.0.

- **Higher communications speeds** – Dynamixel Protocol 2.0 supports faster communication speeds, scaling up to $3\,\mathrm{Mbit\,s^{-1}}$ compared to the $1\,\mathrm{Mbit\,s^{-1}}$ limit of the Dynamixel Protocol 1.0. The increase allows for faster response times and more efficient data transmission, essential for applications requiring real-time performance.

- **Improved packet handling** – The protocol introduces more sophisticated packet-handling capabilities, including the ability to handle larger packets and more complex error-checking mechanisms. It enhances the reliability of communications, especially in systems where multiple devices are interacting simultaneously.
- **Advanced addressing capabilities** – Dynamixel Protocol 2.0 expands the addressing capabilities, allowing for a larger number of devices to be controlled within a single network. It is particularly useful in large-scale robotic installations, where numerous actuators must operate in coordination.
- **Enhanced instruction set** – The protocol includes an expanded set of instructions, enabling more nuanced control over connected devices. These instructions cover everything from synchronous movement commands to more detailed feedback on device status, providing developers with the tools needed to create more sophisticated behaviors.
- **Backward compatibility** – Despite its advancements, Dynamixel Protocol 2.0 maintains backward compatibility with the Dynamixel Protocol 1.0 devices. It allows users to integrate newer devices into existing systems without the need for a complete overhaul, protecting investment in previously deployed hardware.

The instruction and status packets of the Dynamixel Protocol 2.0 are described to show differences compared to the previous version of the protocol

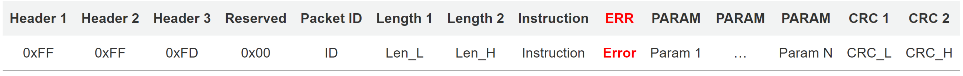


Header 1	Header 2	Header 3	Reserved	Packet ID	Length 1	Length 2	Instruction	Param	Param	Param	CRC 1	CRC 2
0xFF	0xFF	0xFD	0x00	ID	Len_L	Len_H	Instruction	Param 1	...	Param N	CRC_L	CRC_H

Figure 3.6: Instruction packet of the Dynamixel Protocol 2.0.

The structure of the instruction packet is depicted in Fig. 3.6. The packet starts with the extended header comprising three bytes: `0xFF`, `0xFF`, and `0xFD`, which collectively mark the start of the packet. The header is followed by a reserved byte, set to `0x00`, ensuring packet integrity. The `Packet ID` byte identifies the specific Dynamixel unit being addressed. The instruction packet length is denoted by two bytes `Length1` and `Length2`, allowing for an extended range of lengths. The `Instruction` byte indicates the type of action that the unit is expected to perform. It is followed by the parameters `Param 1` to `Param N` that are specific to the particular instruction. To ensure error-free communication, Two *Cyclic Redundancy Check* (CRC) bytes `CRC1` and `CRC2` are included as a more sophisticated variant of the checksum at the end of the packet.

The status packet (depicted in Fig. 3.7), in response to the instruction packet, maintains the header format of the Dynamixel Protocol 2.0. It includes



Header 1	Header 2	Header 3	Reserved	Packet ID	Length 1	Length 2	Instruction	ERR	PARAM	PARAM	PARAM	CRC 1	CRC 2
0xFF	0xFF	0xFD	0x00	ID	Len_L	Len_H	Instruction	Error	Param 1	...	Param N	CRC_L	CRC_H

Figure 3.7: Status packet of the Dynamixel Protocol 2.0.

the `Packet ID` for the responding unit. The length is similarly divided into `Len_L` and `Len_H` bytes. The packet further contains an `Instruction` byte that is mirrored from the instruction packet. The `Error` byte is to indicate if any issues occurred during the execution of the instruction. Parameters `Param 1` to `Param N` convey the data or status response from the unit. The packet concludes with two CRC bytes for data integrity verification.

3.4 EROS Architecture

Seed Robotics uses the EROS architecture to operate robotics hands. It is most likely an evolution of the *Extremely Reliable Operating System* microkernel-based operating system [14]; however, the manufactured does not provide direct information about its origin. Nevertheless, EROS is designed to provide high reliability, flexibility, and scalability for robotic systems. It serves as a central platform for managing various robotic components, such as actuators and sensors, ensuring their efficient cooperation.

The main components of the EROS architecture are as follows:

- **Main Board** is a central unit that connects and controls all other parts of the system. It includes various ports for connecting actuators and sensors that provide the necessary data for precise control.
- **Ports** are physical interfaces on the main board that allow the connection of actuators and sensors. Each port has a unique identifier (ID), enabling the addressing and control of connected devices.
- **Actuators**, like motor units, perform movements based on commands from the main board. EROS supports various types of actuators, increasing the system's flexibility.
- **Sensors** that monitor the robot's status and provide feedback on the current position, speed, temperature, and other parameters.
- **Communication Protocols** supported by EROS include RS485, TTL, USB, and Bluetooth, enabling reliable and fast data exchange between the main board, actuators, and sensors.

- **Firmware** is software running on the main board and actuators that ensure command processing, system status monitoring, and movement control.

The main benefits of EROS architecture are as follows:

- **Flexibility** – It allows the use and combination of different types of actuators and sensors, enhancing application possibilities.
- **scalability** – It supports connecting any number of actuators and sensors, facilitating system expansion.
- **Control Accuracy** – It improves control accuracy and feedback, aiding in more accurate replication of human-like behavior.
- **Voltage Compatibility** – Internal actuators can operate at various voltages.
- **Pin Reduction** – EROS reduces the number of necessary hardware pins, simplifying construction and increasing reliability.
- **Distributed Processing** – Task processing is distributed between the main board and actuators, enhancing system efficiency and performance.
- **Modern Processor Architectures** – EROS supports compatibility with new processor architectures, ensuring long-term relevance and performance.
- **Open-source Collaboration**: Designed with open-source collaboration in mind, facilitating external contributions and innovations.
- **Thermal Management**: Improved thermal behavior due to active cooling increases reliability and device lifespan.

3.5 Communication with the RH8D

The RH8D operates under the EROS architecture using the Dynamixel Protocol, which is compatible with both Dynamixel Protocol 1.0 and Dynamixel Protocol 2.0 versions. It can be integrated into a Dynamixel network and communicates with other devices in the chain using standard settings and designated IDs.

The communication speed is 1 $\mathrm{Mbit\,s^{-1}}$ by default, which is recommended for optimal performance. The speed can be adjusted from 9600 $\mathrm{bit\,s^{-1}}$ to 3 $\mathrm{Mbit\,s^{-1}}$. The default communication setup includes 8 data bits, no parity, and 1 stop bit, which is a communication configuration among Dynamixel devices. The half-duplex communication mode leads to sequential communication following the primary-secondary model of the Dynamixel Protocol.

Each RH8D unit exposes nine IDs on the main bus port, which are assigned to different components of the system. One ID is dedicated to the main control board, allowing access to high-level system functionalities and real-time measurements of the currents and sensor for readings. Eight IDs are dedicated to the actuators, where each actuator has a unique ID with an individual control table. The control table enables one to read various parameters such as joint position, speed, and temperature and also to write commands to control the actuators, like setting target positions or adjusting control parameters such as the *Proportional–Integral–Derivative* (PID) controller settings. The assigned IDs are depicted in Table 3.1.

Table 3.1: IDs of the RH8D Robotic Hand components.

Hand Part	**Default ID**	
	Right hand	**Left hand**
Main board	0	40
Wrist Rotation	31	41
Wrist Adduction	32	42
Wrist Flexion	33	43
Thumb Adduction	34	44
Thumb Flexion	35	45
Index Flexion	36	46
Middle finger Flexion	37	47
4th and 5th finger flexion	38	48

These IDs can be customized as needed, though it is generally convenient to use the default settings for straightforward debugging. A detailed description of the control tables can be found at [15] for servos and [16] for the main board. Selected control table entries for the servomotor are presented in Table 3.2 and Table 3.3 for the main board, which we use in our control system.

Table 3.2: Selected Control Table Entries for Servo Motors

Address	Description	Size(nr of bytes)	Access Mode
`0x1E` (30)	Target Position	WORD (2)	RW
`0x20` (32)	Target Speed	WORD (2)	RW
`0x24` (36)	Present Position	WORD (2)	R

Table 3.3: Selected Control Table Entries for main board

Address	Description	Size (nr of bytes)	Access Mode
`0x04`	Baud Rate	BYTE (1)	RW
`0x2A`	Present Voltage	BYTE (1)	R
`0x2B`	Present Temperature	BYTE (1)	R
`0x6C`	Port 1 Current Reading	WORD (2)	R
`0x6E`	Port 2 Current Reading	WORD (2)	R
`0x70`	Port 3 Current Reading	WORD (2)	R
`0x72`	Port 4 Current Reading	WORD (2)	R
`0x74`	Port 5 Current Reading	WORD (2)	R
`0x76`	Port 6 Current Reading	WORD (2)	R
`0x78`	Port 7 Current Reading	WORD (2)	R
`0x7A`	Port 8 Current Reading	WORD (2)	R

3.6 Visual Simulation Discussion

During the work with the RH8D robotic hand by Seed Robotics, it has been found that creating a simulation environment is deemed inefficient and impractical due to the unique design and characteristics of the robotic hand, especially within the timeframe of the bachelor thesis project. The RH8D robotic hand features 19 (DoF) but only 8 actuators with feedback. That discrepancy between the number of DoF and the actuators makes the system open, presenting several challenges for meaningful simulation. That is further elaborated in the following paragraphs.

Characteristics of the RH8D Robotic Hand

The RH8D robotic hand is designed with 19 DoF, allowing it to perform complex movements similar to a human hand. Specifically, it includes

- Four degrees of freedom for the thumb.

- Three degrees of freedom for each of the other four fingers.
- Three degrees of freedom for palm movement.

However, the hand utilizes only 8 actuators that provide feedback. These actuators are strategically placed and control multiple movements through tendons and other mechanical linkages. While the design is efficient in terms of mechanics and cost, it presents significant challenges for simulation.

Challenges of Simulating an Open System

An open system, like the RH8D, where not every DoF is paired with a dedicated actuator with feedback, complicates accurate modeling and simulation. The primary challenges include the following deficiencies.

1. **Lack of Complete Feedback** – Only eight actuators provide feedback, meaning most DoFs are controlled without direct feedback. Simulation models require precise and complete information on all DoFs to predict the system's behavior accurately.
2. **Complexity of Mechanical Linkages** – The tendons and mechanical linkages connecting actuators to multiple movable parts introduce complex nonlinear dynamics that are difficult to simulate. Creating an accurate model of these linkages would require detailed analysis and modeling, which could be time-consuming and prone to inaccuracies.
3. **Limited Predictive Capabilities** – Without complete feedback and with complex mechanical linkages, the simulation can only roughly approximate the hand's behavior. Hence, it may be insufficient for precise prediction and tuning of performance in real applications.

Practical Implications

Given these challenges, it has been decided that it would be more effective to focus on experimental testing and development directly on the physical device rather than investing time and resources in creating a complex simulation environment that is likely to provide limited and inaccurate results. Any adjustments and optimizations can be made through iterative testing on the actual robotic hand, allowing for more precise and reliable performance data.

Thus, in the context of the RH8D robotic hand, simulation is considered a less effective approach, and development and testing are primarily conducted experimentally with the real device to ensure higher accuracy and reliability of results.

Chapter 4

Proposed Robotic Hand Control for Sign Language Gesture Performance

Due to the first hands-on experience with the robotic hand RH8D, we opted for the manual design of control patterns (commands) for the selected symbols of the ASL. It has been manually handcrafted with visual feedback to avoid gesture commands that would not be sufficiently repeatable.

The robot hand RH8D is designed to automatically detect situations of increased current draw, such as when the movement of fingers is accidentally blocked during the transition to a sign language gesture configuration. In such cases, the system will halt its operation to prevent damage. However, the system cannot currently detect if some fingers are forcibly held in a bent position by external forces, as it does not include sensors to monitor the position of the fingers directly. Instead, feedback is obtained only through the rotation of servomotors connected to the fingers via tendons.

4.1 Design of Sign Language Control System

The task aims to implement control of the RH8D robotic hand for a sign language demonstration task. Our objective is to create a system that allows the user to input a text command, which will then be interpreted and reproduced by the robotic hand in the form of sign language.

The control of the robotic hand is achieved using the Dynamixel protocol 2.0 [17] and suitable libraries available in the seed robotics repository [18]. Programming is done in the Python programming language.

The robotic hand is equipped with a total of 19 DoF, with the thumb featuring four DoF and each of the remaining fingers incorporating three DoF. Additionally, the movement of the palm contributes another three degrees of freedom. Despite the hand's complex structure of 19 DoF, it is controlled by only eight actuators capable of providing feedback. This configuration classifies the system as *open*, implying a lack of comprehensive feedback for each actuator. Consequently, this deficiency complicates the accurate modeling of the hand and the computation of its forward kinematics. The forward kinematics process is essential for determining the precise location of each finger based on the input configurations of the actuators. The open system nature significantly challenges the ability to achieve precise and reliable control over the hand's positioning and movement.

Due to limited mobility, some characters in sign language are inaccessible for our RH8D robotic hand. One of these limitations is the shared actuator for the pinky and ring fingers, preventing independent movement.

For the actual implementation of the individual sign language symbols, we follow the approach used in [19]. The task has been approached by creating arrays of numbers representing the actuator orientations expressed as percentages. Subsequently, the resultant configuration of the robotic hand was compared with that of a human hand displaying the corresponding symbol.

4.2 Sign Language Gesture Performance

This section delves into the practical application of the theoretical concepts discussed earlier, demonstrating the crucial transition from theory to implementation. The focus here is not only on the detailed execution of hand control within a task that demonstrates sign language using the RH8D robotic hand but also on the implementation of a dynamic grip control system that adapts to real-time sensory feedback. It includes configuring the servomotors to precisely replicate sign language symbols, validating the accuracy of these configurations through gesture recognition technology, and ensuring the robotic hand's movements are both accurate and reliable.

We obtained several arrays representing the rotation of servomotors, thereby defining the specific configuration of the fingers and palm of the RH8D robotic hand, see Table 4.1. The description of the servomotors and their corresponding movements is shown in Fig. 4.1.

Table 4.1: Servos configuration for individual sign language symbols for robotics hand RH8D.

Symbol	**Rotation of servos (%)**							
	Servo 1	**Servo 2**	**Servo 3**	**Servo 4**	**Servo 5**	**Servo 6**	**Servo 7**	**Servo 8**
A	50	50	50	0	50	100	100	100
B	50	50	50	50	100	0	0	0
C	50	50	50	70	50	40	40	40
D	50	50	50	70	60	0	60	60
E	50	50	50	100	70	100	100	100
F	50	50	50	70	40	55	0	0
G	50	95	95	40	60	0	95	95
H	50	95	95	30	100	0	0	100
K	50	50	50	95	100	0	0	95
L	50	50	50	0	0	0	95	95
M	50	50	80	80	80	70	70	90
O	50	50	50	80	30	50	50	50
P	50	75	100	30	30	0	40	40
Q	50	0	100	100	50	10	95	95
S	50	50	50	95	40	95	95	95
T	50	50	50	30	95	80	80	95
V	50	50	50	95	90	0	0	95
X	50	50	50	30	95	70	80	80

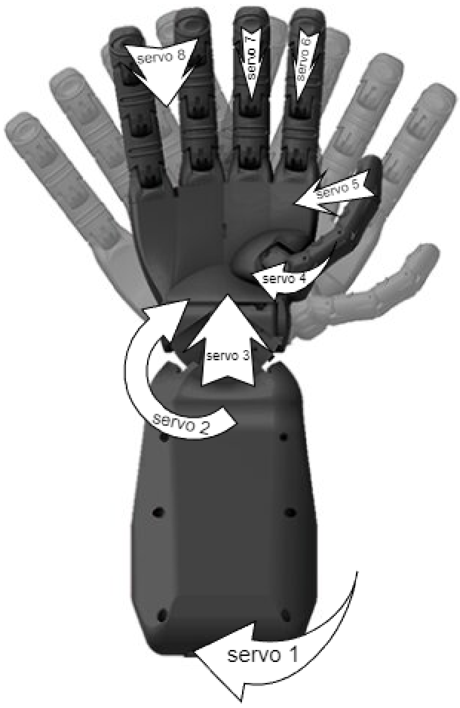


Figure 4.1: Diagram of servomotor assignments for RH8D hand movements.

Some of the mentioned finger configurations needed to be divided into two or more steps to achieve the desired finger positioning. For instance, to form the symbol `E`, the index, middle, ring, and pinky fingers must first be bent. Once these fingers are bent, the thumb can then be curled over them. In contrast, the thumb must be bent first to form the symbol `T`, and the other fingers are curled over it afterward. Certain symbols are accomplished by creating two sets of configuration arrays. Once the first configuration is completed, the second one follows to finalize the desired symbol. That ensures the accurate positioning of the fingers and thumb to achieve the intended gestures. It yields an array of arrays representing the individual hand configurations that create the given sign language symbol.

The letter `Z` in sign language represents a dynamic symbol, requiring the RH8D robotic hand to perform the following movement. The motion is executed through several arrays of configurations. By transitioning the hand from the first configuration in the array to the last, the symbol `Z` in sign language is formed. The approach enables the robotic hand to accurately replicate the dynamic motion necessary for the symbol `Z`.

The whole program operates as follows. Initially, the user inputs a word he/she wishes to translate into sign language. The word is then segmented into individual letters. Subsequently, a corresponding array of configurations for the first letter is retrieved, which the robotic hand interprets as a sign language symbol. These configurations are transmitted to the servomotors of the robotic hand. Once all configurations from the array are sent, the robotic hand achieves the pose representing the specified sign language symbol. The hand maintains the position for three seconds. After the period, it returns to a default position with extended fingers and a vertical orientation. The process then repeats for the following letter from the input text, finding its corresponding array of configurations. The block diagram Fig. 4.2 provides the overview of the program's functionality.

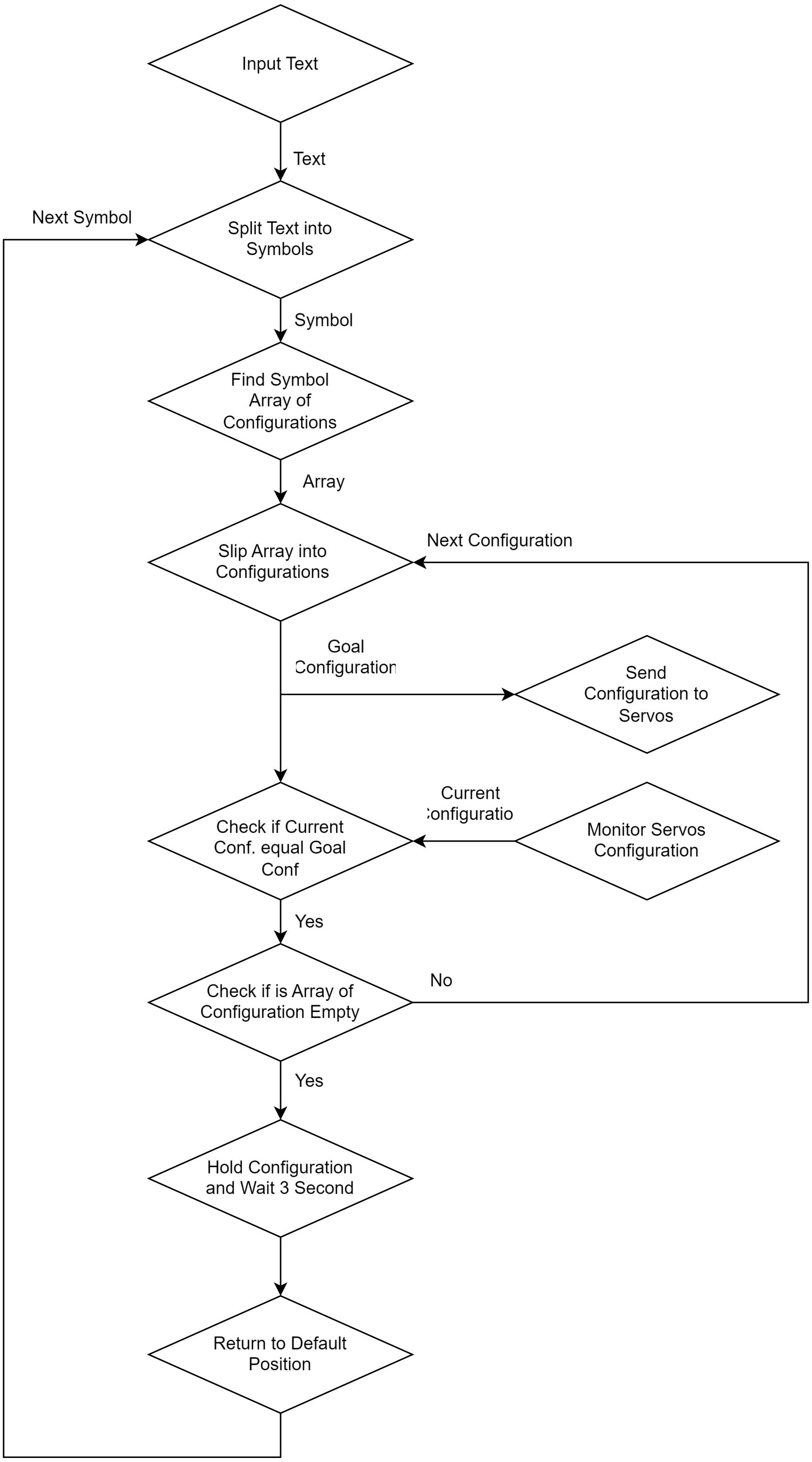


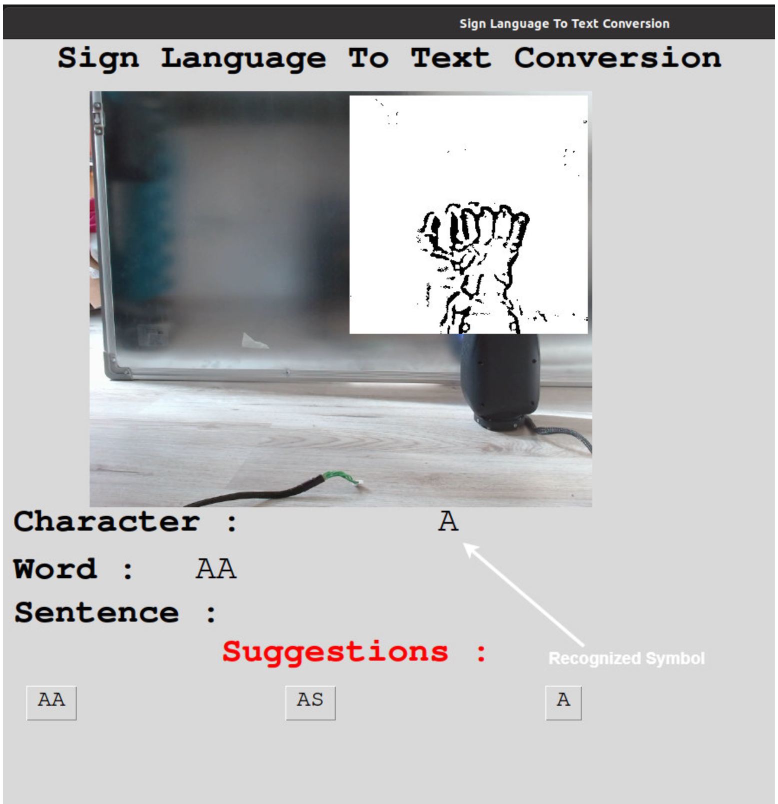
Figure 4.2: Block diagram of the developed program for translating text into sign language.

Chapter 5

Hand Gesture Recognition of Sign Language

An available program for sign language gesture recognition can be utilized to verify whether the configurations of the robotic hand representing sign language symbols are accurate by means of its possible understandability. We require the sign language recognition program to possess the ability to identify individual hand symbols autonomously, without reliance on markers or gloves, akin to the methodologies outlined in the research article concerning hand gesture recognition for enhancing human-computer interaction [20]. If such a gesture recognition program, which has been trained on humans, is capable of recognizing the gestures produced by the RH8D robotic hand, we can demonstrate that the gestures created by the robotic hand are valid and accurate. We consider such a validation suitable despite the fact that the recognition failure might not necessarily cause inaccurate gestures but might be related to the shape of the robotic hand itself.

In the thesis, we utilized an established sign language recognition program [21] instead of developing a bespoke solution. The sign language recognition program uses a webcam Logitech C920s PRO to capture ASL gestures in real-time. The program processes images through trained *Convolutional Neural Network* (CNN) to recognize and classify each gesture into one of the 26 letters of the alphabet with the reported accuracy of up to 98 %. The images undergo preprocessing, including Gaussian blur and adaptive thresholding, to enhance feature extraction. A whiteboard has been used as a background. The final output is displayed as text on a graphical user interface, which is depicted in Fig. 5.1.



Sign Language To Text Conversion

Character : A

Word : AA

Sentence :

Suggestions :

Figure 5.1: Graphical user interface for hand gesture recognition of sign language.

We conducted a series of tests both with and without a white glove to ensure a comprehensive evaluation of the sign language symbols reproduced by the RH8D robotic hand. The glove usage is aimed to increase the resemblance of the robotic hand to a human hand, potentially enhancing the gesture recognition software's accuracy. Additionally, we chose two specific distances from the camera 0.6 m and 0.9 m to guarantee that the entire hand remained within the frame of the recognition software throughout the testing process, see Fig. 5.2.

Each configuration was tested four times to eliminate random variations, resulting in a total of sixteen measurements for each symbol. The test protocol was established to determine how effectively the robotic hand can generate sign language symbols under varied visual conditions, see Table 5.1.

Figure 5.2: RH8D gesture recognition test environment at varied distances and glove usage.

Although the authors [21] report high success rates in recognizing gestures generated by American Sign Language, experimental deployment showed only about sixty percent accuracy. The result occurred when the hand was not wearing a white glove and was positioned 0.6 m away from the web camera. The decline in performance could be attributed to several factors, from the limited generalizability of the trained network to inadequate gesture performance by the RH8D robotic hand. Further complications arise from the differences in appearance and limited mobility of the robotic hand compared to a human hand. Although the gestures were created with the utmost effort, they may not be perfect. Moreover, results indicate poorer performance at longer distances and when gloves are used, likely due to an increased number of detected finger bends, which then generate distortions that the application is not designed to handle. Success rates could potentially be improved with training using the robotic hand; however, such an approach was considered unfeasible within the time and scope constraints of the bachelor's thesis project. An example of clearly recognizable sign language symbols is shown in Figure Fig. 5.3, where both the symbol created by a robotic hand and the symbol created by a human hand can be observed.

Table 5.1: Test protocol of sign language character recognition using the RH8D robotic hand.

Symbol	**without Glove**		**with Glove**	
	0.6 m	0.9 m	0.6 m	0.9 m
A	4	4	1	0
B	2	2	0	1
C	4	4	4	3
D	3	2	0	2
E	0	0	0	0
F	2	1	0	1
G	4	3	2	2
H	2	0	1	0
K	2	0	2	1
L	4	4	4	4
M	0	0	0	0
O	4	4	4	4
P	0	0	0	0
Q	1	0	0	0
S	4	3	2	2
T	0	0	0	0
V	4	2	3	0
X	3	3	3	2
Z	4	4	4	3
Success rate [%]	*62*	*47*	*40*	*33*

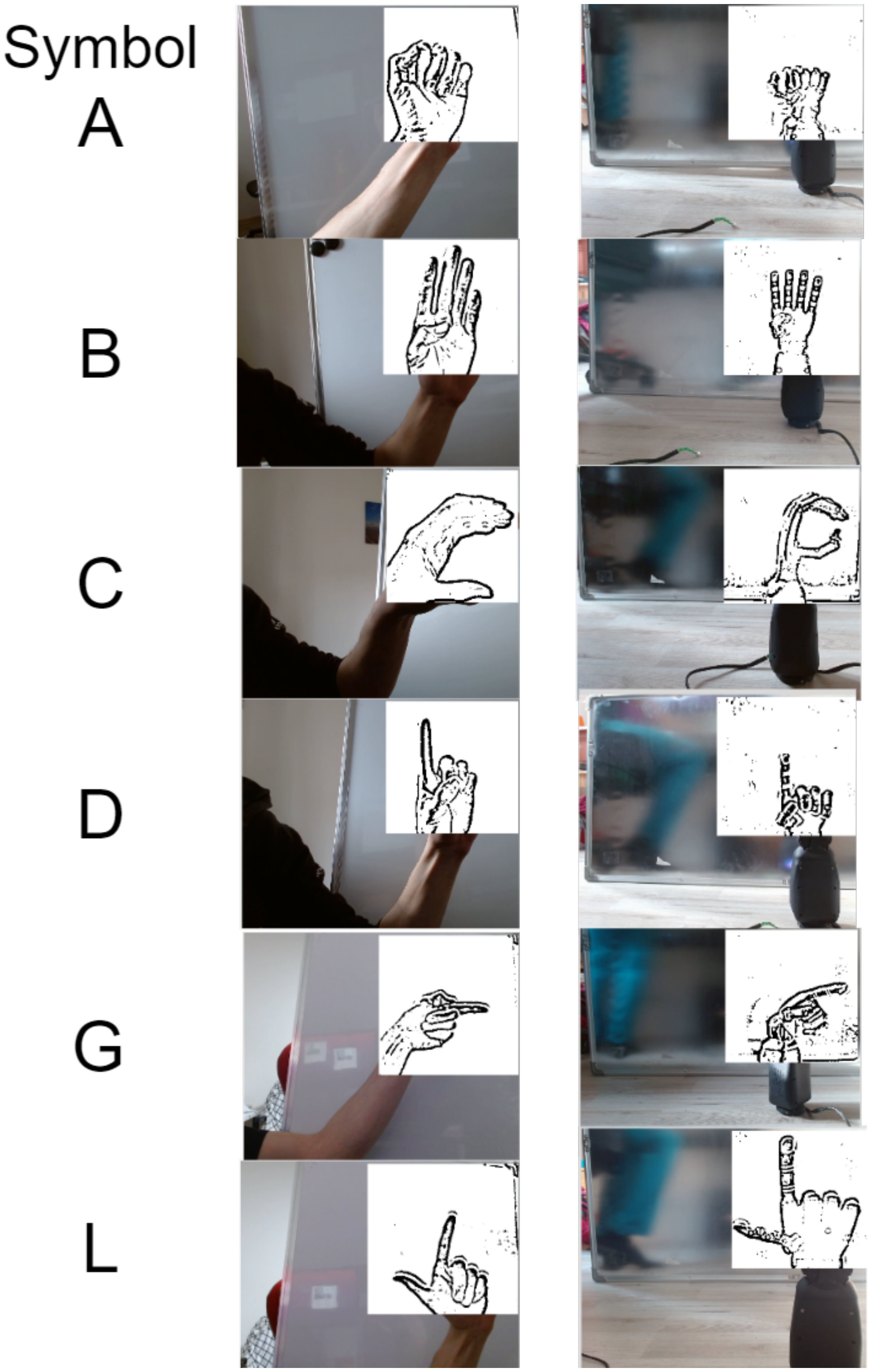


Figure 5.3: Examples of well-recognizable symbols, such as A, B, C, D, G, and L, created by both robotic and human hands.

Chapter 6

Dynamic Grip Control System

6.1 Design

The dynamic grip control system design for the RH8D robotic hand was implemented using tactile sensors located at the fingertips. These sensors are essential for providing feedback on the force exerted on an object being manipulated. The process of the fingers closing is gradual: as soon as the fingers enclose an object, the tactile sensors begin to detect contact and subsequently measure the force applied to the object. The force is compared with a predefined desired grip strength value. Once equilibrium is achieved between the applied pressure and the set value, the finger movement stops. The process ensures that the fingers maintain a stable grip on the object until conditions change or instructions to release the grip are received.

It was initially necessary to efficiently process data obtained from FTS3 tactile sensors located at the fingertips of the robotic hand to implement the grip control system. Data is primarily transmitted via serial communication in the form of text strings. The text is transformed into a structured format to facilitate further processing. Specifically, it is transformed into a multi-dimensional array, where each element represents a corresponding sensor and the forces acting on it in three axes: x, y, and z.

The implementation of the grip control begins with the calibration of the data obtained from the sensors, which is essential for accurately determining the real force values. It is followed by a control mechanism for manipulating the fingers. If data indicate that the force exerted on the object by individual fingers reaches or exceeds a predetermined value, the movement of these fingers stops. The step prevents excessive pressure that could damage the object.

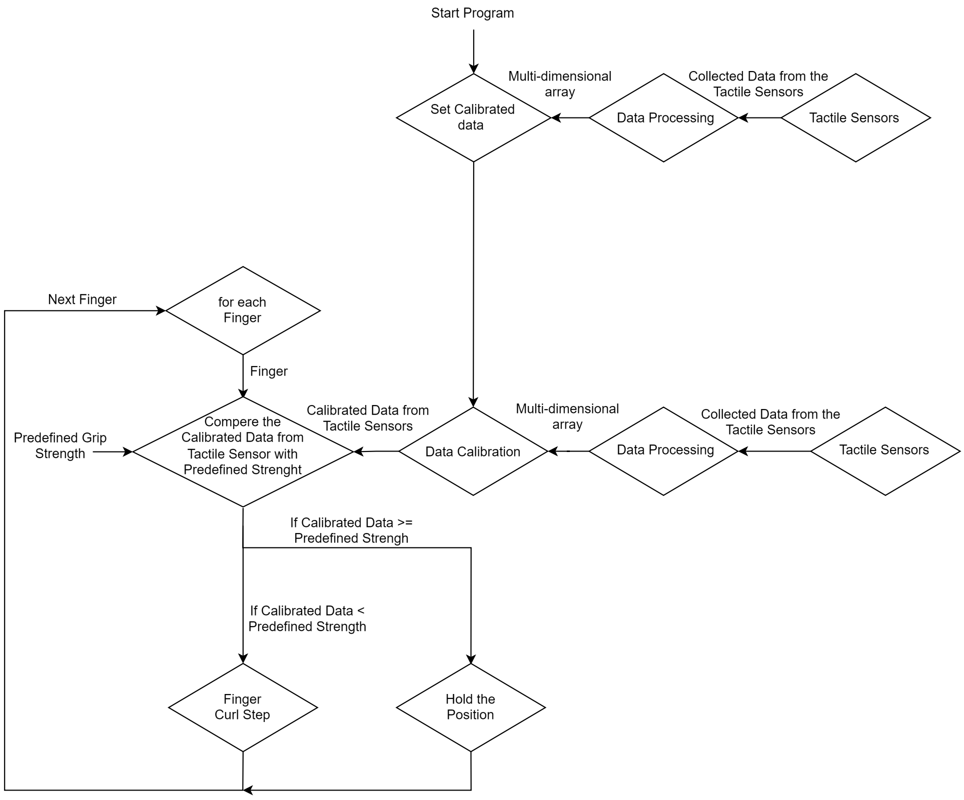


Figure 6.1: Block Diagram of Grip Control for RH8D.

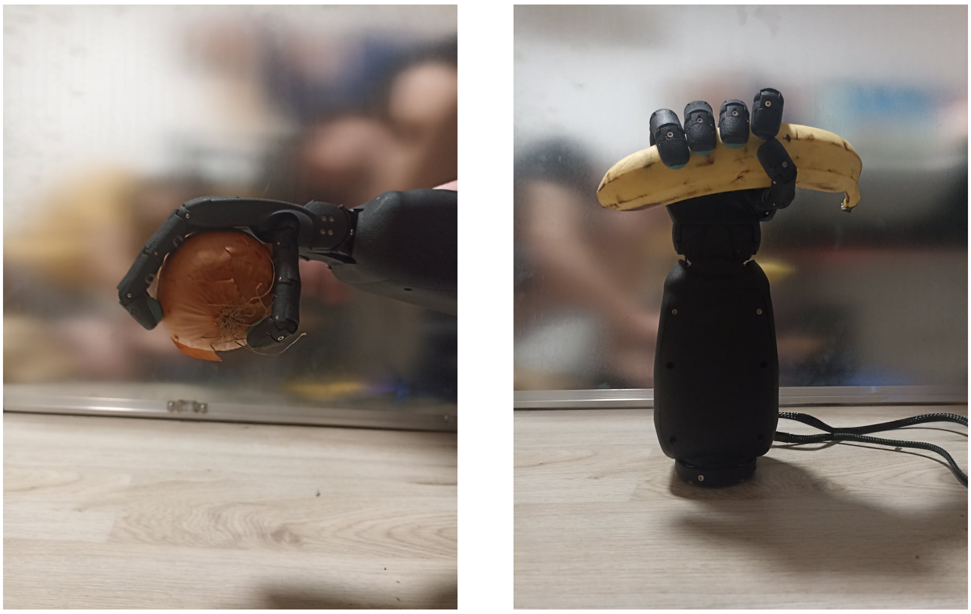
Conversely, fingers that have not reached the required force slightly curl in an attempt to improve grip. The predefined force value must be set to ensure the safe handling of the object without the risk of deformation or object damage. In the event that a decrease in force under the critical value is recorded, automatic corrective curling of the finger occurs to ensure continuous contact with the object. The proposed iterative process allows for dynamic and adaptive grip control. The control design is depicted in the diagram shown in Fig. 6.1.

6.2 Deployment

The system was evaluated using two specific types of grips. The first type is the grasp of a spherical object, with an onion as an example, depicted in

Fig. 6.2a. The second type of grip tested has the grasp of a cylindrical object, such as a banana, as illustrated in Fig. 6.2b.

In Fig. 6.3a and Fig. 6.3b, we can observe the process of grasping an onion and banana, respectively, using the RH8D robotic hand. The depicted plot for the onion grasp in Fig. 6.3a indicates that it was necessary to reduce the motion increment for successful grasping. Conversely, due to the cylindrical shape of the banana, which is more conducive to manipulation, a more significant grasping increment could be maintained, thus allowing for faster handling. Furthermore, the plots reveal that a greater force is exerted on the thumb, which is due to the need to counterbalance the forces applied by the other fingers.

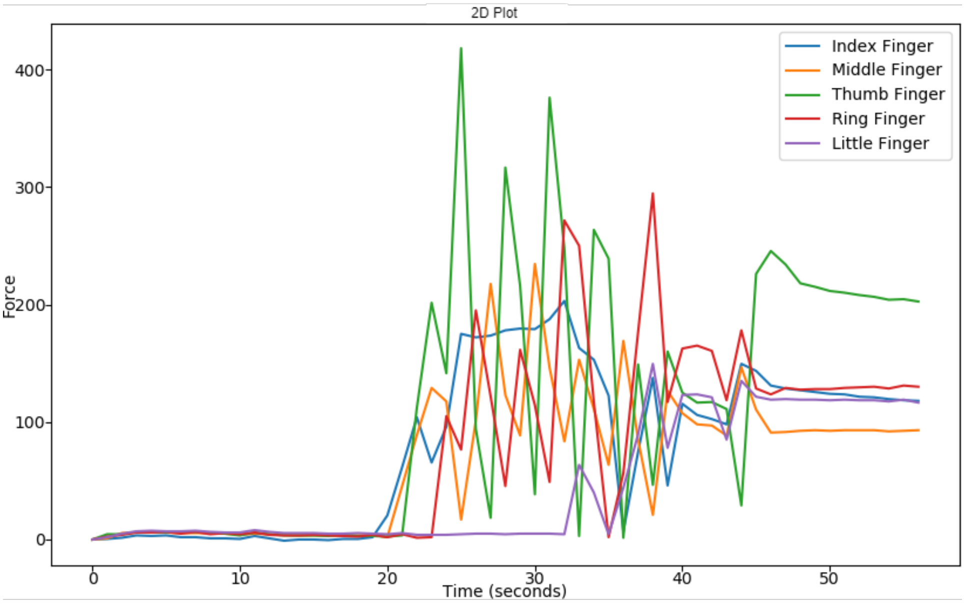


(a) The RH8D robotic hand demonstrates the grasp of a spherical object. The depicted robotic hand holds the onion firmly, showcasing its ability to handle round and smooth surfaces.

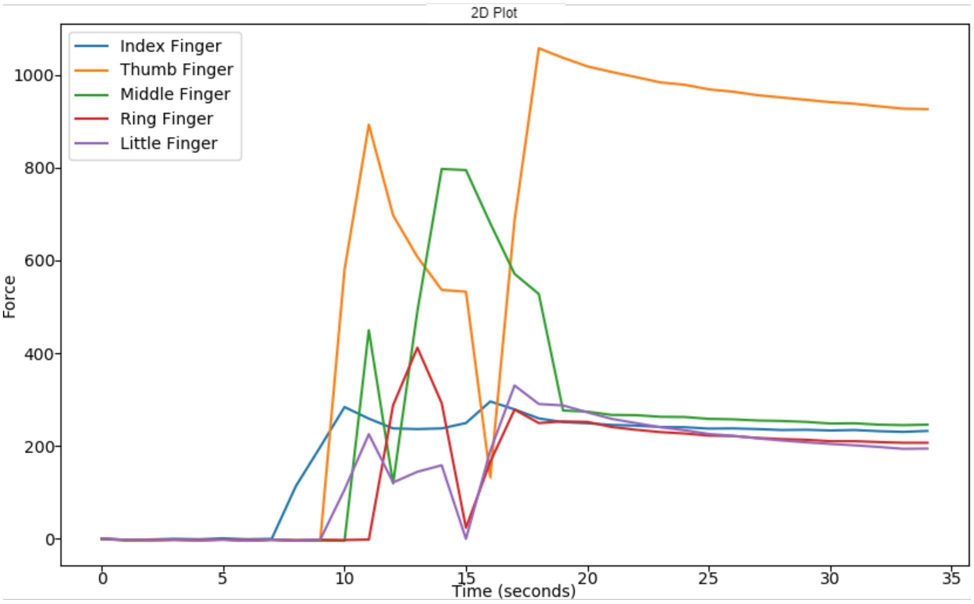
(b) The RH8D robotic hand demonstrates the grasp of a cylindrical object. The image shows the hand securely holding a banana, illustrating the hand's capability to manipulate elongated items.

Figure 6.2: Examples of two deployed types of the grasp.

Given the ability to adjust the force exerted on objects by the RH8D robotic hand, the capability of force modulation enables the safe handling of soft objects.



(a) A plot displaying the forces exerted by the fingers on an onion, at which point the system decided to halt as the target value was achieved.



(b) A plot displaying the forces exerted by the fingers on the banana, at which point the system decided to halt as the target value was achieved.

Figure 6.3: Examples of two plots showing the force applied on the object while the robot hand is grasping.

Chapter 7

Conclusion

This bachelor thesis focused on the development and evaluation of control methods for the RH6D and RH8D robotic hands from Seed Robotics. Various methods of force and grasp control were explored, and a feedback approach using data from tactile sensors was utilized to implement grip control for the RH8D model. The knowledge gained in controlling the RH8D hand was applied in a demonstration project for sign language, where individual symbols were manually created. Besides, grasping semi-soft objects has been demonstrated by grasping an onion and a banana.

The system was tested using existing software for sign language gesture recognition to verify the accuracy of these symbols. The results showed that the system has a limited ability to recognize the created symbols. It might be because the software was initially trained on images of human hands, which differ significantly from robotic hands both visually and in terms of degrees of freedom. Specifically, the RH8D robotic hand is not optimal for mimicking sign language due to its limitations in replicating certain letters like `I`, `J`, `N`, `R`, `U`, `W`, and `Y`. However, grasping of the objects was shown to be firm, which supports the suitability of the RH8D hand for grasping tasks.

Although the presented results are limited, the main contribution of the work is to understand the dynamic aspects of robotic hand control and the expansion of their practical applications. The proposed solution provides a platform for further research and development in the field of robotic hands. A possible future research might be to extend the system's capabilities for improved recognition and adaptation to diverse objects and situations. It should involve advanced machine learning algorithms and deeper integration of sensor data. Additionally, exploring options for miniaturization and incorporating further feedback to improve the manipulative capabilities of robotic hands would be advantageous.

Bibliography

[1] "Human-inspired, dexterous, robot hands and manipulators," access on 2024-03-18. [Online]. Available: https://www.seedrobotics.com/robot-hands-manipulators-introduction

[2] T.-W. Chong and B. G. Lee, "American sign language recognition using leap motion controller with machine learning approach," *Sensors*, vol. 18, p. 3554, 10 2018.

[3] "ROS - Robot Operating System," https://www.ros.org/, access on 2024-03-18.

[4] "Pypot – a python library for dynamixel motor control," https://github.com/poppy-project/pypot, access on 2024-05-15.

[5] K. M. Lynch and F. C. Park, *Modern Robotics: Mechanics, Planning, and Control.* Cambridge University Press, 2017.

[6] C. Wang, X. Zhang, X. Z. Y. L. Guanwen, D. W. Yin, and J. Zhao, "Feature sensing and robotic grasping of objects with uncertain information," *Sensors*, vol. 20, no. 13, 2020.

[7] M. Pavlic, T. Markert, S. Matich, and D. Burschka, "RobotScale: A framework for adaptable estimation of static and dynamic object properties with object-dependent sensitivity tuning," in *IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, 2023, pp. 668–674.

[8] Y. Li, P. Wang, R. Li, M. Tao, Z. Liu, and H. Qiao, "a survey of multi-fingered robotic manipulation: Biological results, structural evolvements, and learning methods," *frontiers in Neurorobotics*, vol. 16, 2022.

[9] “fts-tactile-pressure-sensors,” access on 2024-05-04. [Online]. Available: https://www.seedrobotics.com/fts-tactile-pressure-sensors

[10] “Matplotlib 3.9.0 documentation,” https://matplotlib.org/stable/, access on 2024-05-16.

[11] “Fts3-3d-high-resolution-tactile-(pressure)-sensor,” https://kb.seedrobotics.com/doku.php?id=fts

[12] “Dynamixel protocol 1.0,” access on 2024-04-18. [Online]. Available: https://emanual.robotis.com/docs/en/dxl/protocol1/

[13] “Dynamixel protocol 2.0,” access on 2024-04-18. [Online]. Available: https://emanual.robotis.com/docs/en/dxl/protocol2/

[14] “Eros (microkernel),” access on 2024-05-22. [Online]. Available: https://en.wikipedia.org/w/index.php?title=EROS_(microkernel)

[15] “Control table for the seed robotics smart actuators,” access on 2024-05-16. [Online]. Available: https://kb.seedrobotics.com/doku.php?id=eros:seedactuators_controltable

[16] “Eros main board control table,” access on 2024-05-16. [Online]. Available: https://kb.seedrobotics.com/doku.php?id=eros:mainboard_ctltable

[17] “DYNAMIXEL XM430-W350,” access on 2024-03-18. [Online]. Available: https://emanual.robotis.com/docs/en/dxl/x/xm430-w350/

[18] “Seedrobotics repository,” access on 2024-03-18. [Online]. Available: https://github.com/seedrobotics/

[19] I. A. Adeyanju, S. O. Alabi, A. O. Esan, B. A. Omodunbi, O. O. Bello, and S. Fanijo, “Design and prototyping of a robotic hand for sign language using locally-sourced materials,” *Scientific African*, vol. 19, p. e01533, 2023.

[20] A. Haria, A. Subramanian, N. Asokkumar, S. Poddar, and J. S. Nayak, “Hand gesture recognition for human computer interaction,” *Procedia Computer Science*, vol. 115, pp. 367–374, 2017.

[21] “Sign language to text conversion,” access on 2024-05-09. [Online]. Available: https://github.com/emnikhil/Sign-Language-To-Text-Conversion/